Annex D Extended Security Implementation

3.02 Web Services .................................................................6
  Introduction ......................................................................6
  General guidelines ..........................................................6
  Authentication ..................................................................8
  SOAP and XML ...............................................................9
  WSDL .................................................................9
3.04 Secure Shell (SSH) .......................................................11
  Introduction .....................................................................11
  Security requirements ....................................................11
3.06 Web Applications ........................................................21
  Introduction .....................................................................21
  System Hardening ..........................................................21
  System Update ..................................................................22
  Protection of Data and Information ....................................23
  Protection of Availability and Integrity ...............................28
  Authentication and Authorization ........................................41
  Protecting Sessions ..........................................................48
  Authentication Parameter Password ...................................59
  Content Management Systems (CMS) ................................62
  Logging ...........................................................................64
3.08 External Hosting ............................................................65
  Introduction .....................................................................65
  Requirements for external security management ..................65
3.11 Third Parties .................................................................77
  Introduction .....................................................................77
  Organizational requirements ..............................................77
3.12 Proxy Server .................................................................79
  Introduction .....................................................................79
  Definition of terms and application area ............................79
  General requirements for proxy servers ............................80
  (Outbound) proxy servers ...............................................94
  Reverse proxy servers ....................................................97
3.14 Architecture of systems ................................................99
  Introduction .....................................................................99
  Segregation of systems ....................................................99
  Communication .............................................................101
  Layer model ....................................................................105
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Availability</td>
<td>108</td>
</tr>
<tr>
<td>Administration</td>
<td>110</td>
</tr>
<tr>
<td>Adherence to industry standards</td>
<td>111</td>
</tr>
<tr>
<td>3.15 Windows Servers</td>
<td>112</td>
</tr>
<tr>
<td>Introduction</td>
<td>112</td>
</tr>
<tr>
<td>System hardening</td>
<td>112</td>
</tr>
<tr>
<td>Protecting availability and integrity</td>
<td>119</td>
</tr>
<tr>
<td>Protecting availability and integrity</td>
<td>121</td>
</tr>
<tr>
<td>Authentication and authorization</td>
<td>124</td>
</tr>
<tr>
<td>Authentication parameter password</td>
<td>130</td>
</tr>
<tr>
<td>Logging</td>
<td>132</td>
</tr>
<tr>
<td>3.17 JBOSS Application Servers</td>
<td>134</td>
</tr>
<tr>
<td>Introduction</td>
<td>134</td>
</tr>
<tr>
<td>Security requirements</td>
<td>134</td>
</tr>
<tr>
<td>3.20 Third Party Access</td>
<td>139</td>
</tr>
<tr>
<td>Introduction</td>
<td>139</td>
</tr>
<tr>
<td>Technical Requirements</td>
<td>139</td>
</tr>
<tr>
<td>Organisational Requirements</td>
<td>158</td>
</tr>
<tr>
<td>3.21 Unix Servers</td>
<td>159</td>
</tr>
<tr>
<td>Introduction</td>
<td>159</td>
</tr>
<tr>
<td>Unix systems</td>
<td>159</td>
</tr>
<tr>
<td>3.23 Routers and Switches</td>
<td>183</td>
</tr>
<tr>
<td>Introduction</td>
<td>183</td>
</tr>
<tr>
<td>System hardening</td>
<td>183</td>
</tr>
<tr>
<td>System update</td>
<td>187</td>
</tr>
<tr>
<td>Protecting data and information</td>
<td>188</td>
</tr>
<tr>
<td>Protecting availability and integrity</td>
<td>190</td>
</tr>
<tr>
<td>Authentifizierung und Autorisierung</td>
<td>194</td>
</tr>
<tr>
<td>Protecting sessions</td>
<td>198</td>
</tr>
<tr>
<td>Authentication parameter password</td>
<td>200</td>
</tr>
<tr>
<td>Logging</td>
<td>202</td>
</tr>
<tr>
<td>Layer 2 protocols</td>
<td>203</td>
</tr>
<tr>
<td>3.24 MySQL /MariaDB database systems</td>
<td>206</td>
</tr>
<tr>
<td>Introduction</td>
<td>206</td>
</tr>
<tr>
<td>General requirements for MySQL / Maria database systems</td>
<td>206</td>
</tr>
<tr>
<td>Database system hardening</td>
<td>206</td>
</tr>
<tr>
<td>MySQL / Maria -specific requirements</td>
<td>212</td>
</tr>
<tr>
<td>3.29 Oracle Database Systems</td>
<td>216</td>
</tr>
<tr>
<td>Introduction</td>
<td>216</td>
</tr>
<tr>
<td>General Information</td>
<td>216</td>
</tr>
</tbody>
</table>
3.5 Cryptographic Algorithms and IP Security (IPSec)

3.45 SNMP

3.30 Microsoft SQL Server

3.36 Apache Web Servers

3.32 Microsoft IIS
3.55 Network Based Storage Systems ................................................................. 305
  Introduction ........................................................................................................ 305
  Storage Area Network (SAN) ................................................................. 305
  IP based network storage NAS/ iSCSI ......................................................... 309
  System hardening ............................................................................................ 310
  System update ................................................................................................. 312
  Protecting data and information ................................................................... 312
  Protecting availability and integrity ............................................................... 314
  Authentication and authorization ................................................................. 315
  Protecting sessions .......................................................................................... 318
  Authentication parameter password ............................................................. 320
  Logging ............................................................................................................. 321

3.57 Architecture of the access and transport network ........................................ 323
  Introduction ...................................................................................................... 323
  General ........................................................................................................... 323
  Access network ............................................................................................... 326
  Multi Protocol Label Switching (MPLS) ....................................................... 329

3.58 Architecture of datacenter and cloud infrastructures .................................... 333
  Introduction ...................................................................................................... 333
  Infrastructure ................................................................................................... 333
  Virtualization .................................................................................................... 337
  Networks ......................................................................................................... 339
  Firewalls .......................................................................................................... 341
  System management ....................................................................................... 342

3.59 Load Balancer as Web Server ....................................................................... 345
  Introduction ...................................................................................................... 345
  Requirements on web server software .......................................................... 345
  Configuration Requirements .......................................................................... 346
  HTTPS requirements ....................................................................................... 348
  HTTPS requirements ....................................................................................... 349

3.60 PostgreSQL databases ................................................................................. 350
  Introduction ...................................................................................................... 350
  Hardening ......................................................................................................... 359

Asymmetric Encryption Algorithms ...................................................................... 291
Hash Functions .................................................................................................... 292
Message Authentication Codes (MAC) .............................................................. 293
Signature Algorithms ........................................................................................... 293
IP Security (IPSec) ................................................................................................ 295
Key Management .................................................................................................. 300
Table of Comparable Security Levels for Cryptographic Algorithms .................. 304
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>System monitoring</td>
<td>361</td>
</tr>
<tr>
<td>3.80 Nginx</td>
<td>362</td>
</tr>
<tr>
<td>Introduction</td>
<td>362</td>
</tr>
<tr>
<td>Nginx webserver platform requirements</td>
<td>362</td>
</tr>
<tr>
<td>Nginx webserver installation requirements</td>
<td>364</td>
</tr>
<tr>
<td>Nginx webserver configuration requirements</td>
<td>366</td>
</tr>
<tr>
<td>Nginx webserver logging</td>
<td>373</td>
</tr>
<tr>
<td>Configuration requirements</td>
<td>374</td>
</tr>
</tbody>
</table>
3.02 Web Services

Introduction

This security requirement has been prepared based on the provisions of the Group Policy on IT/NT Security. The security requirement is used as a basis for an approval in the PSA process, among other things. It also serves as an implementation standard for provisions of the Group Policy on IT/NT Security in units which do not participate in the PSA process. These requirements shall be taken into account from the very beginning, including during the planning and decision-making processes. When implementing these security requirements, the precedence of national, international and supranational law shall be observed.

General guidelines

Req 1 Basic security functionality for externally accessible web services MUST be implemented in the DMZ and, depending on the criticality, be protected by a Web Service Security Gateway.

DMZ means here the zone, where all external / internet traffic is terminated. Web servers are typically located in the DMZ, whereby databases are located in the MZ, where no direct internet connection is needed. The following functions MUST be realized in the access and validation tier:

• authentication and authorization for the request (i.e., is the apparent sender of the request potentially allowed to access the web service in question)
• basic validation of the request based on the given description such as Web Service Description Language (WSDL)
• pre- and post-processing tasks (such as terminating an SSL/TLS encryption or security relevant logging)
• termination of the tcp traffic and thus preventing all direct access to web service end points in the MZ.

Motivation: The web server in the DMZ must take any reasonable measures to isolate the web services in the MZ from attacks.

Implementation example: A web service which passes back the weather information for a given city name does not need to be protected by a web service security gateway. On the other hand a web service which passes back user information / account data needs to be protected by a web service security gateway.

For this requirement the following threats are relevant:
• Unauthorized access or tapping of data
• Unauthorized modification of data
• Disruption of availability
Req 2 If a web service processes highly sensitive data in the meaning of the payment card standard PCIDSS (such as credit card data, customer data incl. customer passwords), then these data items MUST be individually protected using end-to-end application level mechanisms such as XML encryption.

Motivation: Due to special agreements within the industry, particular data items require confidentiality protection even when communicated over secure networks.

For this requirement the following threats are relevant:
• Unauthorized access or tapping of data
• Unauthorized modification of data

Req 3 A web service request MUST be protected from manipulation / replay attacks while on the public, unprotected communication path.

This can be accomplished through the use of XML signature (XMLsig) on the application layer, SSL/TLS on the transport layer, or other appropriate mechanisms (e.g., VPN). In case of replay attacks, the use of session ids / counters / tokens is a suitable way to protect from this issue. XML Signature can cause interoperability problems in its current implementation state (12 / 2009.)

Motivation: Non-repudiation mechanisms are assumed to become more and more accepted in legal proceedings as well as for accounting and auditing purposes. Therefore, it is highly desirable to not rely on SSL/TLS as ephemeral integrity protection mechanism on the transport layer, but rather to solve this issue more appropriately on the application layer with durable XML signatures. Due to the different time scales and goals pursued with XML signatures, as compared to SSL/TLS integrity protection, the use of a special purpose public/private key pair for signatures becomes inevitable. XML signatures are the only mechanism that allows to prove to an external third party (e.g., a legal entity such as a court) the fact that a message was received from a particular entity. The timestamp and/or message identifier can be used to counter replay attacks. The requirement to sign the entire XML message / SOAP Body is important to counter some sophisticated attacks on the XML signature.

For this requirement the following threats are relevant:
• Unauthorized access or tapping of data
• Unauthorized modification of data
Authentication

Req 4 The sender of a web service request MUST be properly authenticated, if any form of confidential content / information is transferred.

This means, that an access to a weather webservice does not need to be authenticated. An example for a proper authentication mechanism is e.g. certificate based authentication.

Motivation: In order to prevent unwanted usage of resources a proper authentication is necessary.

For this requirement the following threats are relevant:
• Unauthorized access or tapping of data
• Unauthorized modification of data
• Disruption of availability
• Denial of executed activities

Req 5 The mechanism to authenticate a sender MUST rely on strong cryptographic algorithms/ frame-works.

Strong cryptographic frameworks/algorithms in this context are e.g. XML signature or SSL v3/TLS client certificates with adequate algorithms like SHA 1/2 hashes and AES encryption. Additional examples are STS tokens / oAuth tokens, if transferred over ssl / tls.

Motivation: Weak algorithms can be broken by attackers and identities can be faked.

For this requirement the following threats are relevant:
• Unauthorized access or tapping of data
• Unauthorized modification of data
• Denial of executed activities

Req 6 The direct recipient of a web service request MUST be authenticated by the sender of this request.

This requirement is not applicable for a Enterprise Service Bus scenario, as here the bus is here in the request chain. This authentication can be done implicitly by using XML encryption on the request with the appropriate public key from the certificate of the intended recipient, explicitly in the communication setup of a SSL/TLS connection (also tied to a certificate), or another appropriate mechanism (e.g. VPN).

Motivation: In order to prevent man-in-the-middle attacks the sender of a request must ensure that it communicates with the intended recipient of its request.

For this requirement the following threats are relevant:
• Unauthorized access or tapping of data
• Unauthorized modification of data
• Unauthorized use of services or resources
• Disruption of availability
• Denial of executed activities
• Unnoticeable feasible attacks

Req 7 If a web service request is protected with XML signatures, the signature data MUST NOT be removed from the request by an intermediate processor.

Motivation: The XML signature should be validatable over the complete communication path.
For this requirement the following threats are relevant:
• Unauthorized modification of data
• Unauthorized use of services or resources

**SOAP and XML**

Req 8 If a SOAP request or reply is not handled immediately and needs to be buffered in the DMZ the confidentiality and integrity of the buffered message MUST be ensured.

Generally a queuing can happen for various reasons, e.g. if the MZ systems do not react fast enough. Generally a queuing should happen in the MZ, if possible.

*Motivation: Without protecting buffered messages attackers could overtake the buffering server and despite existing ssl point to point connections an attack can take place.*

Implementation example: The messages can be saved AES encoded in the database and additionally there can be created an encrypted hash value for encrypted message body.

For this requirement the following threats are relevant:
• Unauthorized access or tapping of data
• Unauthorized modification of data

Req 9 A new developed web service MUST use XML schemata for content/format validation.

*Motivation: XML schemata is the state-of-the-art technology for formatting web services. Former standards like Document Type Definitions (DTDs) do not offer the same kind of granularity when defining request objects.*

For this requirement the following threats are relevant:
• Unauthorized access or tapping of data
• Unauthorized modification of data
• Unauthorized use of services or resources
• Disruption of availability

**WSDL**

Req 10 Each element and attribute value communicated in a web service request MUST be described with the expected data type, the expected length and/or range, and whenever appropriate a formal specification describing the acceptable data.

*Motivation: The detailed WSDL specification (including regular expressions) allows a much better description of the element and/or attribute values that are expected in an input parameter of the web service request.*

Implementation example: An example for such a formal specification are regular expressions, e.g. “([A-Za-z0-9]+[-._+\&\#\%\/=~])@[A-Za-z0-9]+@([A-Za-z0-9]+[.])+[A-Za-z]{ 2,6}” to specify a valid e-mail address. The base type “int” SHOULD be replaced by an application specific subtype which exactly defines the needed integer range. Equivalently the base type "string" MUST be replaced by an application specific subtype which limits the string's length through "maxLength" and "minLength".

For this requirement the following threats are relevant:
3.02 Web Services

- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources
- Unnoticeable feasible attacks

**Req 11** All Webservice requests MUST be validated at the webservice endpoint against a detailed WSDL specification.

*Motivation:* A validation at an early part enables a very early blocking of malicious codes and therefore also saves resources.

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources
- Disruption of availability

**Req 12** If no such black / white listing can be made, the web service itself MUST be hardened (this means must be able to handle malformed input data) against possible injection attacks.

A formal specification in this context can be the definition of string length and containing characters including the possible sequence. For typical web applications the Open Source tool mod_security represents a good solution.

*Motivation:* Any data that is not expected by subsequent parts of the application may cause unexpected and undesired effects, such as XML injection (in particular CDATA sections), XPath injection, XQuery injection, SQL injection, LDAP injection, code injection, or command injection. If input data is properly sanitized in the DMZ, such attacks are made considerably more difficult.

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources
- Disruption of availability

**Req 13** Only the minimal set of APIs / functions must be made available to the public via Web Services.

In particular, when exposing an internal Web Service to external partners or when extending a legacy application with a web service frontend, care must be taken to only expose required APIs.

*Motivation:* By offering only a minimal set of APIs / calls, the attack window can be reduced.

For this requirement the following threats are relevant:
- Attacks motivated and facilitated by information disclosure or visible security weaknesses
3.04 Secure Shell (SSH)

Introduction
This security requirement has been prepared based on the provisions of the Group Policy on IT/NT Security.

The security requirement is used as a basis for an approval in the PSA process, among other things. It also serves as an implementation standard for provisions of the Group Policy on IT/NT Security in units which do not participate in the PSA process. These requirements shall be taken into account from the very beginning, including during the planning and decision-making processes. When implementing these security requirements, the precedence of national, international and supranational law shall be observed.

Security requirements

General

Req 1 The SSH protocol version 2 must be used exclusively.

Motivation: SSH-2 is the current protocol version, which will be supported in contrast to SSH version 1. SSH protocol version 1 contains design weaknesses. The weaknesses were removed with latest OpenSSH versions, still for secure interaction with other SSH communication partners, version 2 of the protocol must be enforced.

Implementation example: The entry "Protocol 2" in the OpenSSH config file forces the exclusive usage of SSH version 2.

For this requirement the following threats are relevant:
• Unauthorized access to the system
• Unauthorized access or tapping of data
• Unauthorized modification of data
• Unauthorized use of services or resources
• Attacks motivated and facilitated by information disclosure or visible security weaknesses

Req 2 Unnecessary (transport) ciphers must be deactivated - the SSH daemon shall select a cipher or minimal set cipher from the following list: AES128-CTR, AES192-CTR, AES256-CTR and chacha20-poly1305@openssh.com.

To fulfill this and all other requirements OpenSSH >= 6.5 incl. all patches is recommended.

Motivation: The usage of secure algorithms ensures the security of the transferred data. CBC-based ciphers should not used. These ciphers had weakness in SSH-2.

The minimization of the represents a minimization of the attack surface / code base. The reduction of available cipher on server site is a proactive action as shown by the attack on the OpenSSH-implementation of the "AES-GCM cipher". (see http://www.openssh.com/txt/gcmrekey.ad

The usage of the chacha20-poly1305@openssh.com cipher (since OpenSSH 6.5) solves the weakness of block cipher - MAC interplay of OpenSSH (packet length must be transmitted in plaintext). (optional background see https://web.archive.org/web/20140616113024/http://blog.djm.net.au/2013/11/chacha20-and-poly1305-in-
3.04 Secure Shell (SSH)

The algorithm is therefore recommended in deviation to SecReq 3.50-1,2,3,8 (this is an exception for SSH).

OpenSSH supports open source code and has a long development history with very good security support for troubleshooting (security patches).

Since release 6.5. of OpenSSH support for state-of-the-art crypto was added:
- **Transport cipher** “chacha20-poly1305@openssh.com” that combines Daniel Bernstein’s ChaCha20 stream cipher and Poly1305 MAC to build an authenticated encryption mode.
- **Support for key exchange using elliptic-curve Diffie Hellman in Daniel Bernstein’s Curve25519. This key exchange method is the default when both the client and server support it.**
- **Support for Ed25519 as a public key type. Ed25519 is a elliptic curve signature scheme that offers better security than ECDSA and DSA and good performance. It may be used for both M2M-user, user and host keys.**

Implementation example: The entry
Ciphers aes256-ctr,aes192-ctr,aes128-ctr,chacha20-poly1305@openssh.com

in the OpenSSH and OpenSSHD configuration file achieves this requirement.

If possible, the number of ciphers must be reduced to one cipher on server site:
Ciphers aes256-ctr

or since release 6.5
Ciphers chacha20-poly1305@openssh.com

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized modification of data
- Unauthorized use of services or resources
- Disruption of availability
Req 3 The SSH service must be exclusively bound to interfaces and/or IP addresses, which are needed to run/operate the service.

The SSH service on multihomed servers must not listen globally on all interfaces. On network devices this requirement often will be achieved based on ACLs.

*Motivation: Exposing the SSH service to not needed IPs / interfaces increases the attack surface. This requirement achieves for devices with internet interfaces that the SSH service is never accessible from the internet.*

Implementation example: The following entry in the sshd_config file achieves this requirement:

```
ListenAddress 10.40.2.2
```

(Alternatively you can use an alias name instead of the IP address, which will be resolved via /etc/hosts).

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized use of services or resources
- Unnoticeable feasible attacks

Req 4 The SSH daemon must be executed as a stand-alone daemon.

The SSH daemon must not run via inetd, but shall be executed as a stand-alone daemon.

*Motivation: If SSH is launched via inetd, the SSH daemon generates a new server key every time, which ties up re-sources and makes it easier to carry out a DoS attack.*

For this requirement the following threats are relevant:
- Disruption of availability

Req 5 OpenSSH must be used as SSH server software.

OpenSSH >= 6.5 incl. all patches is recommended.

It should recommend to vendors, to use OpenSSH in future. Oracle SSH on Solaris already uses a patched version of OpenSSH (fork).

*Motivation: OpenSSH supports open source code and has a long development history with very good security support for troubleshooting (security patches). Oracle SSH on Solaris uses a patched version of OpenSSH (fork).*

Since release 6.5. of OpenSSH support for state-of-the-art crypto was added:
- Transport cipher “chacha20-poly1305@openssh.com” that combines Daniel Bernstein’s ChaCha20 stream cipher and Poly1305 MAC to build an authenticated encryption mode.
- Support for key exchange using elliptic-curve Diffie Hellman in Daniel Bernstein’s Curve25519. This key exchange method is the default when both the client and server support it.
- Support for Ed25519 as a public key type. Ed25519 is a elliptic curve signature scheme that offers better security than ECDSA and DSA and good performance. It may be used for both M2M-user, user and host keys. Currently the MyCard/TKS 2000 smartcards does not yet support Ed22519 keys (=user keys).
For this requirement the following threats are relevant:
- Unauthorized access to the system
- Disruption of availability

---

**Req 6**  
TCP/IP port forwarding must be used in a controlled manner.

TCP/IP port forwarding must be deactivated if it is not necessary for the server to operate.

TCP/IP port forwarding should be used to protect applications which offer no means of securely transmitting authentication information.

TCP/IP ports of applications that do not necessarily have to be accessible in general on the network, but for instance are only used for administrative purposes, should be linked to the local loopback address and should then be forwarded via SSH TCP/IP port forwarding. The advantage is that the computer offers fewer network services and is thus less vulnerable to worms which spread over networks.

*Motivation: Applications can be protected by using TCP/IP port forwarding.*

Implementation example: The entry

```plaintext
AllowTcpForwarding no
```

in the OpenSSH config file disables the usage of port forwarding within the SSH protocol. **WARNUNG:** Users with shell-access can install their own forwarders.

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data
- Unauthorized use of services or resources
- Unnoticeable feasible attacks

---

**SSH Options**

---

**Req 7**  
Gateway ports must not be activated.

If TCP/IP port forwarding is used, it must be not possible that remote hosts connect to forwarded ports.

*Motivation: The system is exposed to uncontrollable risks, if locally forwarded ports can accessed from remote hosts.*

Implementation example: The entry

```plaintext
GatewayPorts no
```

in the OpenSSH config file achieves this requirement on server side.

**Hint:** Additionally the same config must be applied on the client side, see appropriate requirement.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized use of services or resources
- Unnoticeable feasible attacks
3.04 Secure Shell (SSH)

Req 8  X11 forwarding must be used in a controlled manner ("X11-forwarding").

X11 forwarding must be deactivated if it is not necessary for the server to operate. If X11 forwarding is used, the X11 forwarding server service must be linked to the local loopback address.

**Motivation:** X11 forwarding basically provides access to the remote X11 server (user PC) and exposes this to unnecessary threats.

Implementation example: The entries

```
X11Forwarding no
X11UseLocalhost yes
```

in the OpenSSH config file achieve this requirement. If the first setting cannot be applied, at least the last setting must be applied. If `X11Forwarding` cannot be set to `no`, make sure that `X11UseLocalhost` is set to `yes`.

For this requirement the following threats are relevant:

- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unnoticeable feasible attacks

Req 9  SSH ‘agent forwarding’ must be prohibited on the server side.

This requirement is not relevant for jump hosts. SSH agent forwarding may be prohibited here. Instead, SSH port forwarding can be used, if other requirements do not prohibit it (see SecReq 3.20 3rd party access).

**Motivation:** The server side deactivation blocks the creation of a server side agent forwarding socket, this socket consequently cannot be misused. Notwithstanding the primary control is only on user / client site (ForwardAgent no). Furthermore an attacker could enable the server-side setting in a misuse scenario. However, his deviation should be detected by the Security Monitoring (Assurance process).

Implementation example: The following entry in `sshd_config`:

```
AllowAgentForwarding no
```

achieves the requirement.

For this requirement the following threats are relevant:

- Unauthorized access to the system
- Unauthorized use of services or resources

Req 10  Tunnel devices must not be used (‘tunnel device forwarding’).

The tunnel device feature (=VPN-feature) of SSH is not needed within data center environments. Firewall systems must not be circumvented or links from protected environments exported.

**Motivation:** Network tunneling renders security devices such as firewalls, access control lists (ACL), IDS tools, etc. use-less and is therefore not permitted.

Implementation example: The entry

```
PermitTunnel no
```

in the OpenSSH config file achieves this requirement.

For this requirement the following threats are relevant:

- Unauthorized use of services or resources
The usage of the SSH service must be restricted to dedicated groups (resp. users).

To obtain additional security, the usage the SSH service must only be used by dedicated groups and/or users.

*Motivation: Defense in depth: The explicit allowance/restriction prohibits, that the SSH service is used be users like bin, sys, uucp.*

Implementation example: Assign all users using the SSH service to a special group, e.g. ssh and allow only this group in the SSH configuration:

```
AllowGroups ssh
AllowUsers root,webadmin
```

For this requirement the following threats are relevant:
• Unauthorized access to the system

---

**SSH logins must be logged.**

Logins must be logged in a log file. The OpenSSH log level shall be 'verbose'.

*Motivation: Only in the verbose-Loglevel the fingerprint of the used SSH-public key is recorded. To facilitate forensic analyses after a security incident and for system auditing, log data detailing accesses must be available.*

Implementation example: The entry

```
LogLevel VERBOSE
```

in the OpenSSHD config file achieves this requirement.

For this requirement the following threats are relevant:
• Denial of executed activities
• Unnoticeable feasible attacks

### Authentication and Accounts

**The authentication on server systems must be based on asymmetric cryptographic algorithms (Publickey, OTP).**

Regarding non server-systems such as routers the user authentication via asymmetric cryptographic algorithms is strongly recommended. Server authentication (Hostkey verification) is always done by asymmetric cryptography. It is recommended to use Ed25519 as a public key type for hostkeys. Ed25519 is a elliptic curve signature scheme that offers better security than ECDSA and DSA and good performance.

*Motivation: Passwords are usually attackable via Phishing, Keylogger and Brute Force attacks. Additionally passwords can be easily shared and a non-repudiation therefore is reasonable hard.*
Implementation example: The entries

- PasswordAuthentication no
- PermitEmptyPasswords no
- ChallengeResponseAuthentication no
- RSAAuthentication yes
- PubkeyAuthentication yes
- PermitRootLogin without-password

(or no, e.g. if root access is needed only via console) in the OpenSSHD configuration file achieve this requirement.

in the OpenSSHD configuration file achieve this requirement.

For this requirement the following threats are relevant:

- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources
- Disruption of availability

---

**Req 14** The SSH service must not offer password authentication on server systems.

Even if users use only keys in practice, the server processes passwords for authentication, if it's not forbidden in the configuration. On NON server systems such as routers the offer of password authentication by the SSH service is not recommended.

Motivation: If the server does not offer password authentication, it could not be exploited.

Implementation example: The entries

- PasswordAuthentication no
- PermitEmptyPasswords no
- ChallengeResponseAuthentication no
- RSAAuthentication yes
- PubkeyAuthentication yes

in the OpenSSHD configuration file achieve this requirement.

For this requirement the following threats are relevant:

- Unauthorized access or tapping of data
- Unauthorized modification of data
- Disruption of availability

---

**Req 15** The authentication information of SSH users (public SSH keys in the AuthorizedKeysFile) must be monitored and protected.

There must be a regular check/reconciliation of the access privileges (target vs. actual state) of all users. This targets operational processes. Authentication information should be controlled by an account management system and related operational procedures should be established. Independently of the account management system the public keys should be stored in a file system area protected from user access (no write access), e.g.

AuthorizedKeysFile /etc/keys/%u/authorized_keys
**3.04 Secure Shell (SSH)**

**Motivation:** Changes on access privileges must be only performed, if they are authorised and traceable.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources
- Disruption of availability

---

**Req 16** The privileges and services provided by the SSH server to the user must be kept to the necessary minimum ('least privilege').

If only secure file transfer or TCP port forwarding is to be offered, users do not need to be granted full shell access. Instead, access to SFTP (see solution concept Secure File Transfer using SFTP (former Security requirement 3.05)) or to TCP port forwarding must be limited.

**Motivation:** This reduces the attack surface of the system

Implementation example: Implementation example (TCP port forwarding for Oracle database port 1521):

```
Add
   from="Client-IP",command="sleep 1d",permitopen="127.0.0.1:1521",no-agent-forwarding,no-pty,no-user-rc,no-X11-forwarding <ssh public key>
```
to the file authorized_keys.

Implementation example (Transfer of a file):

```
Add
   from="Client-IP",command="/usr/bin/cat /var/log/auth.0.gz",no-agent-forwarding,no-pty,no-user-rc,no-X11-forwarding,no-port-forwarding <ssh public key>
```
to the file authorized_keys.

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Disruption of availability

---

**Req 17** Host-based authentication (via rhosts/shosts) must not be used.

**Motivation:** To ensure that access cannot be gained to other systems without authentication once a system has been compromised, a machine-to-machine trust setting must not be used.

Implementation example: The entries

```
RhostsRSAAuthentication no
HostbasedAuthentication no
IgnoreRhosts yes
```
in the OpenSSHD config file achieve this requirement.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources
- Disruption of availability
3.04 Secure Shell (SSH)

Req 18  The host key fingerprint or host key must be provided.

Without the fingerprint the user cannot check the host key. The user then has to rely on the security of the first login (no man in the middle attack, trust on first use).

Hint: This is a requirement for the operations department.

*Motivation: To increase security all users should be able to check the host keys on their own.*

Implementation example: All host key fingerprints or host keys (public part) could be downloadable on a trusted side, so that all users can check the keys on their one.

Another option is to verify the host keys via DNS SSHFP resource records. For this purpose you have to publish the host key fingerprints to DNS.

When accepting an unknown host key, the user is informed that an matching host key fingerprint was found in DNS.

```
$ ssh -o "VerifyHostKeyDNS ask" argus.ada.t-online.de
[...]
Matching host key fingerprint found in DNS.
Are you sure you want to continue connecting (yes/no)?
```

Another option is to make all known host keys available for all users on every server (filename `/etc/ssh/ssh_known_hosts`).

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data
- Disruption of availability
- Denial of executed activities

Req 19  For SSH client applies, the SSH server key (host key) must be verified during login.

The login process is only allowed to be continued, if the verification of the SSH-server-hostkey was successful. This requirement is applicable for M2M-clients on server as well as for users of workstations.

*Motivation: A verification of the host key is necessary to detect attacks based on DNS- or IP spoofing.*

Implementation example: In OpenSSH (ssh_config) set

```
StrictHostKeyChecking to ASK or Yes (if /etc/ssh/ssh_known_hosts is maintained) VerifyHostKeyDNS to Ask
```

Additional measure to support this requirement: The security monitoring detects changes on the host keys and in-forms the incident management / alarm system.

Hint: This requirement must be supported by an operational procedure:
A user has to be advised properly to this requirement (e.g., in writing when requesting an account). Additionally the fin-gerprints of the hosts or host keys itself should be provided to the user (see appropriate requirement), otherwise the user has to blindly trust the server on first use.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized use of services or resources
- Denial of executed activities
- Unnoticeable feasible attacks
Requirements on the SSH client

**Req 20** For SSH client applies, Private keys must be protected.

The storage of private keys on a smart card including a PIN or hardware security module (HSM) is recommended.

In all other cases, the private key must be protected with a passphrase. The passphrase must be stronger than an or-dinary password. The passphrase must be at least 12 characters long and must follow regular password complexity rules (e.g. at least 80 bit entropy). If not realisable e.g. for technical accounts, the usage of these keys must be restric-ted based on the source IP of the connecting system and/or the issued commands (OpenSSH: „from=“ resp. „command=“-directive).

Hint: This requirement must be supported by an operational procedure: A user has to be advised properly and verifiably to this requirement (e.g., in writing when requesting an account).

**Motivation:** A private key without passphrase is not protected and misuse cannot be limited in case of loss or theft.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Denial of executed activities

---

**Req 21** For SSH client applies, the usage of private keys when using an SSH authentication agent (on work-stations, terminals) must be allowed based on keyboard interaction. Alternatively the client must not forward the authentication agent.

When using the authentication agent of OpenSSH, the private key has to be loaded with „ssh-add –c ...“ or a special version of putty-agent must be used (see below).

Hint: This requirement must be supported by an operational procedure: A user has to be advised properly to this re-quirement (e.g. in writing when requesting an account).

**Motivation:** Without control on the client side, an attacker can misuse the SSH Authentication agent via the forwarded sockets of the authentication agent.

The following entry in the sshd_config

```
AllowAgentForwarding no
```

supports this requirement, but does not fully achieve it, as control takes place on the client side. For this requirement the following threats are relevant:
- Unauthorized access or tapping of data
- Unauthorized use of services or resources
- Denial of executed activities

---

**Req 22** For SSH client applies, Gateway ports must not be activated on client side.

On the client side “GatewayPorts no” must be set in ssh_config (default) and is not allowed to be reactivated via the –g switch using SSH.

Hint: This requirement must be supported by an operational procedure: A user has to be advised properly to this re-quirement (e.g. in writing when requesting an account).

**Motivation:** The system is exposed to uncontrollable risks, if locally forwarded ports can accessed from remote hosts.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unnoticeable feasible attacks
3.06 Web Applications

Introduction

This security requirement has been prepared based on the provisions of the Group Policy on IT/NT Security.

The security requirement is used as a basis for an approval in the PSA process, among other things. It also serves as an implementation standard for provisions of the Group Policy on IT/NT Security in units which do not participate in the PSA process. These requirements shall be taken into account from the very beginning, including during the plan-ning and decision-making processes. When implementing these security requirements, the precedence of national, international and supranational law shall be observed.

System Hardening

<table>
<thead>
<tr>
<th>Req 1</th>
<th>Unused software must not be installed or must be uninstalled.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>During installation of a system often software components will be installed or parts of software will be activated which are not needed for the operation or functionality of the system. This includes also parts of a software, which will be in-stalled as examples but typically not be used (e.g. default web pages, example databases, test data). Such compon-ents should not be installed or must be deleted after installation.</td>
</tr>
</tbody>
</table>

*Motivation:* Vulnerabilities in software of a system offer an attack window for attackers to infiltrate the system. Unin-stalling components that are not required can therefore reduce the possibility of a successful compromise of the sys-tem.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Disruption of availability

<table>
<thead>
<tr>
<th>Req 2</th>
<th>Unused functions of the operated software and hardware must be deactivated.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>During installation of software and hardware often functions will be activated that are not necessarily needed for operation or function of the system. Functions of software are currently inherent part which could not be deleted or dein-stalled individually. Such functions must be deactivated in the configuration of the system permanently.</td>
</tr>
</tbody>
</table>

Beside the functions of the software also hardware functions are active which are not necessary for a system. Func-tions like unused interfaces must permanently deactivated. Permanent means that they must not be reactivated again after system reboot.

*Motivation:* The hardware or software of a system often contains functions which are not used and so will be a risk for system security. Such functions give an attacker the possibility to manipulate the system. Furthermore it is possible to get unauthorized access other areas or data of the system. An example is a debugging function in software which can be used for troubleshooting but must not be activated during normal operation. Or a hardware interface that will not be used and so is unsecured an allows possibly unauthorized access to the system.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Disruption of availability
System Update

Req 3  Software and hardware components that are no longer supported by vendor, producer or developer must not be used.

Components that have reached end-of-life or end-of-support must not be used. Excluded are components that have a special support contract. This contract must guarantee the correction of vulnerabilities over components lifetime.

Motivation: Hardware and software components that have reached end of life or end of support represent a risk for a system. This means that a vendor does not supply remedial updates or patches for a component should errors or vulnerabilities occur. This means that vulnerabilities cannot be fixed when they occur and could be exploited to compromise the system or to impair its availability.

For this requirement the following threats are relevant:

• Unauthorized access to the system
• Unauthorized access or tapping of data
• Unauthorized modification of data
• Disruption of availability

Req 4  Known vulnerabilities in software and hardware of the system must be fixed or protected.

Prior to installation of a software or hardware component, users must check whether any vulnerability has been discovered and published for the version they are installing. Any component that proves to have a vulnerability must not be installed or used. Excepted from this rule are components for which the vendor has already provided a measure to remedy the vulnerability, e.g. a patch, update or workaround. In this case, the additional measure must be implemented on the system. Furthermore it is a ongoing process during the complete life cycle of the system to fix upcoming vulnerabilities promptly.

Motivation: Publication of vulnerabilities increases the risk of successful exploitation by an attacker, especially since the published information usually includes details on how to exploit the vulnerability and tools that make exploitation possible. One example of this is when Web servers are compromised due to not patched vulnerabilities.

For this requirement the following threats are relevant:

• Unauthorized access to the system
• Unauthorized access or tapping of data
• Unauthorized modification of data
• Disruption of availability
Protection of Data and Information

Req 5 Data with need of protection must be secured against unauthorized viewing and manipulation during transmission and storage.

Adequate security measures for transmission and storage must be implemented for data with a need for protection that are classified as internal, confidential or strictly confidential. The chosen measure depends on the classification for the data and other factors such as the type of network used during transmission, the storage location for data, etc. Furthermore must be guaranteed that confidential and strictly confidential data will not be unprotected during temporary storage (e.g. in web cache, temporary folders).

All authentication data such as user names, passwords, PINs, etc. must be protected against unauthorized viewing and manipulation. This applies equally to permanent storage and transmission. Typical measures taken to protect authentication data or combinations of these are:

For storage:
- Client systems: encryption or obfuscation of authentication data, no persistent storage, limitation of access rights.
- Server systems: hashing of authentication data with PBKDF like scrypt or bcrypt or when this is not possible hashes with salt.

For transmission:
- Usage of cryptographically protected network protocols.
- Challenge response method (Disadvantage of this solution is that passwords needed in clear-text on server. If this solution is feasible depends on the individual threat scenario)

Files of a system that are needed for the functionality must also be protected against manipulation. This is necessary because system's integrity can be damaged when the system access this kind of files. An example is the use of check-sum or cryptographic methods to validate if e.g. firmware images, patches, drivers or kernel modules are free of manipulations.

For transmission of data with a need of protection network protocols that are insecure due to insufficient security measures shall not be used. Examples are: SSLv3, SSHv1, FTP, Telnet, SNMPv1 and 2c. In case of these protocols a newer version without vulnerabilities or a secure alternative must be used.

Motivation: If data with a need of protection will not be secured an attacker could record or manipulate the data during transmission over a network. An example is the recording of user names and passwords during system administration with the telnet clear-text protocol. Storing data on a system without adequate protection may mean that unauthorized users can copy or modify it. One example is when passwords can be read out when they stored in an inadequate secured way (e.g. usage of unsecure hashing algorithms like MD5 or SHA-1) or even without encryption on a system or manipulation of firmware to affected the system integrity.

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data
- Unauthorized modification of data
Req 6 TLS with server authentication and encryption must be used for web applications with authenticated sessions.

Motivation: TLS is the usual mechanism for web applications in order to ensure the confidentiality of communications and the authenticity of the application or server. Otherwise, data with need of protection would be transmitted or stored as plain text (e.g., in caches or log files).

For this requirement the following threats are relevant:
• Unauthorized access or tapping of data
• Unauthorized modification of data

Req 7 If TLS is used, all components of the web application must be transmitted in encrypted form.

This includes even possibly non-critical data such as JavaScript, style sheets, images, etc.

We recommend that the web application also uses HTTP Strict Transport Security to force all future connections from the browser to be established in encrypted form only. For this purpose, the web application needs to set the “Strict-Transport-Security” HTTP response header.

Motivation: The behavior of the various browsers is extremely inconsistent when it comes to processing mixed HTTP and HTTPS resources. The resulting vulnerabilities, for example "man-in-the-middle" or "surf-jacking" attacks, are prevented by full encryption or also by setting the "secure" cookie attribute.

In addition, only in this way is it possible to reliably ensure that the user is shown the relevant symbol or message regarding the encryption of this page. By contrast, browser warning messages that state that parts of the web application are not protected unsettle users and dissuade them from using the web application.

Full encryption also prevents a session cookie from being sent unprotected if the "secure" option has not been set correctly.

If a browser that supports HTTP Strict Transport Security (HSTS) receives the HSTS header from a web application, the browser will communicate with this web application only in encrypted form for the length of time specified in the header. If the user subsequently calls up the web application with an HTTP address, the browser automatically changes this to the corresponding HTTPS address. If the encrypted transmission leads to an error code of any kind, the browser stops the connection with an error message. This includes certificate errors that a user can then no longer ignore. First and foremost, HSTS addresses the general problem that many users often do not type in, for example, https://www.anwendung.de as an address, but instead anwendung.de or www.anwendung.de or call up the application from their bookmarks in unencrypted form. Usually, a redirect from HTTP to HTTPS follows. However, this initial unencrypted request including the redirect already allows attackers to perform man-in-the-middle attacks. If a web application sets the HSTS header, this initial unencrypted request is avoided. Furthermore, the browser does not call up any other unencrypted resources if, for example, mixed HTTP and HTTPS resources were accidentally set in the web application.

For this requirement the following threats are relevant:
• Unauthorized access or tapping of data
• Unauthorized modification of data
3.06 Web Applications

Req 8  The web application must not use URL parameters to transmit data with need of protection.

Therefore, the web application must use the POST method if the user's browser is to transmit data with need of protection.

Accordingly, the GET method may be used only to request content from the server.

Motivation: When using the POST method, parameters are transmitted in the request body. This ensures that no parameters are logged by web servers and clients, or can be easily read within the URL. In the case of the GET method, parameters are transmitted via the header; the parameters are visible in the URL and can be logged by web servers and clients. If the web application contains links to other web applications, it is also possible for the URL parameter to be transmitted to these other web applications via the referrer field in the HTTP header and then appear in log files there.

For this requirement the following threats are relevant:
• Unauthorized modification of data

Req 9  The web application must not save data with need of protection on the client side, neither temporarily nor persistently.

This applies both for cookies and for other kinds of local storage of the browser.

Instead data with need of protection must be stored on the server side. As an exception to this, the web application must transmit session identifiers in non-persistent cookies. As a further exception, the web application may store data for identifying the user, such as the name or user name, in persistent cookies or other local storages. Other data with need of protection may be stored in cookies or local storage in encrypted form only.

The web application must also prevent data with need of protection that the user enters in an HTML form from being saved by the browser and automatically entered the next time such a form is used. For this purpose, the web application must set the “autocomplete = off” attribute. For example, this applies in particular to fields for credit card data and other account or payment information. However, we recommend implementing this in general for data with need of protection.

Motivation: A web application can set an expiry date for a cookie via the expires or max-age attribute. This is known as a “persistent cookie”. The browser saves persistent cookies on the hard drive until the expiry date. Every person with access to the computer can read the persistent cookies. This can, for example, happen in an Internet café or through a successful attack.

Cookies without an expiry date are not persistent, meaning they are not permanently saved. The browser deletes these cookies as soon as it is closed. Until then, attackers can also access this data. Therefore, cookies that are not persistent must not contain data with need of protection either.

The various technologies that are used for Rich Internet applications (RIA) or HTML5 applications provide different mechanisms for saving data locally to the client (similar to persistent cookies). Local storages (e.g., flash local shared objects), however, can be analyzed or tampered with on the client, too.

Deactivating autocomplete prevents data with need of protection from being saved locally regardless of the browser configuration. Otherwise other users of the browser could discover this data. In this context it should be noted, that most current browser versions ignore the autocomplete attribute for password fields.

For this requirement the following threats are relevant:
• Unauthorized access or tapping of data
Req 10  The web application must prevent caching of data with need of protection.

For this purpose, the web application must set the following HTTP headers:

Pragma: no-cache  
Cache control: no-cache, no-store  
Expires: <Current server date>  
Date: <Current server date>

Content that does not include data with need of protection, e.g., images, style sheets or public information, may be cached.

With some older versions of Internet Explorer, setting the “Cache Control” header can lead to problems when users need to open or download files. If this functionality is required but is prevented by the header settings for relevant browser versions, this header may be omitted.

Motivation: The web application must prevent data with need of protection from being revealed by means of caching, for example due to other users who use the same computer. The appropriate HTTP headers instruct a browser or a proxy as to how to handle caching. Even if HTTPS prevents the data with need of protection from being saved in proxies, relevant provisions have to be specified. For one thing, this is an additional security measure, and for another, the browser needs to be prevented from caching the data anyway. As browsers and proxies do not act consistently with regard to caching, the above headers need to be set in order to prevent caching as reliably as possible.

- “Pragma: no-cache” issues the instruction not to allow caching;  
  (This HTTP 1.0 directive must still be used, because certain legacy proxies do not support the HTTP 1.1 directive "Cache-Control". Although this is originally a request directive only, many proxies accept it as a response directive, too.)
- “Cache-Control: no-cache” instructs the browser to always request a new page;
- “Cache-Control: no-store” instructs browsers and proxies not to cache;
- “Expires” specifies the date from which content should be treated as expired;
- “Date” specifies the date on which the content was generated.

The usage of meta-tags such as `<META HTTP-EQUIV="Pragma" CONTENT="no-cache">` is not an ideal solution since, for example, proxies do not evaluate any content of HTML documents.

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data

Req 11  Information with need of protection must not be contained in files, outputs or messages that are accessible by unauthorized users.

Information with need of protection must not be accessible in files, outputs or messages of the system by unauthorized users. This includes information relating to the operating system, used middleware or applications such as vendor, product name, product identifier, installed software versions, installed service packs, patches, hot fixes and serial numbers. Examples for system messages which must be free of sensitive data are:

- Comments in downloadable files
- Error and system messages
- Stack traces
- Network protocols
- Login windows and dialogs
Furthermore, details of implementation and information relating, e.g., to backend software/systems, function calls, SQL instructions or structure of database, must not be contained in error messages. Excluded from this are displays and outputs that can be viewed and retrieved by authorized users who are logged in. In addition, an internal transfer of system internal information for error analysis is allowed in an adequate dimension. In this case the continuative regulations or guidelines (e.g. of data privacy) must be noticed.

Motivation: The information named above can be used by an attacker to prepare specific attacks on a system. In this way an attacker could, for example, use the precise software version to identify vulnerabilities in the product and, in a second step, exploit them.

For this requirement the following threats are relevant:
• Unauthorized access or tapping of data
• Attacks motivated and facilitated by information disclosure or visible security weaknesses

---

**Req 12** The web application must restrict access from external search engines and internal search functions to the files and content that are permitted to be shown.

If the web application offers a search function, hidden files, files with need of protection, configuration files, etc. must not be included in the search function index.

It is recommended that the web application provides a defensively specified robots.txt.

Motivation: If hidden files or files with need of protection are included in a search function index, an attacker can obtain and display information about such files via the search function.

For this requirement the following threats are relevant:
• Unauthorized access or tapping of data
Protection of Availability and Integrity

Req 13  The web application must validate on the server side all data that is transferred by a client.

In this process, the data must be checked with regard to the data type, length, characters, format, value range, encoding and content. “Rich input” (e.g., XML, HTML, etc.) must be checked for both correct structure and validity of the data contained in the structure.

This applies to all data that a user has entered, for instance URL and POST parameters with such data. However, this also applies expressly for data that the user does not influence during normal use, e.g., cookies, data from local storages, hidden fields, variable names, HTTP header fields, etc.

Before the data is validated, it must be decoded and canonicalized so that it is available in a standardized (canonical) form. Generally, the utilized frameworks ensure this.

Client-side validations that are, for example, performed using JavaScript are not considered trustworthy from the viewpoint of a server-side application. However, they may additionally be performed to improve the user-friendliness of the web application.

Motivation: All data that is transmitted by the client is not considered trustworthy since an attacker would have no difficulty manipulating it. Input validation ensures that the application uses data in the expected format and that therefore no unexpected side effects occur.

This data can be available in different encodings and notations. Depending on the encoding scheme used, the same value can accordingly be interpreted differently. If a web application validates data without taking account of its encoding and notation, it may not recognize harmful data. An attacker will then be in a position to circumvent the data validation by using a specific encoding or even multiple encodings.

For this requirement the following threats are relevant:
• Unauthorized access to the system
• Unauthorized access or tapping of data
• Unauthorized modification of data

Req 14 If input data is already processed by web application logic on the client side without any possibility for it to be validated on the server side first, the data must be validated on the client side before processing.

This concerns user input data, data (subsequently) loaded from untrustworthy servers or any data loaded from local media.

Motivation: If input data is processed on the client side, this processing also needs to be protected through sufficient validation. If, for example, JSON is used, return parameters must be analyzed for manipulations; after that they may be forwarded to the interpreter. Another area of use is a Flash file which subsequently loads data (via a LocalConnection, from a server or from another Flash file). Here, too, validation should take place on the client side so as not to be put at risk by malicious data. Local media can also be easily manipulated by an attacker, e.g., in order to carry out overflow or injection attacks.

For this requirement the following threats are relevant:
• Unauthorized access to the system
• Unauthorized access or tapping of data
• Unauthorized modification of data
Req 15 The web application must initially consider all data that is transmitted from another system to be un-trustworthy.

Depending on the criticality of the web application and the trustworthiness of the other system, the web application must validate this data, too.

Motivation: Data that is transmitted by a client and can therefore be directly influenced by a user is not the only type of data that can contain malicious character strings. This can also be the case with data that comes from other systems, for example if these systems do not perform sufficient data validation themselves. Therefore, it is important to check whether validation needs to be performed for this data as well. Ultimately, it is important to ensure that the web application uses data in the expected format only, thus preventing any unexpected side effects.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data

Req 16 The input validation for the web application must be performed by means of white listing.

White listing involves checking whether the data contains only characters or character strings that were previously defined as valid characters/character strings. All other characters/character strings are rejected.

With blacklisting, on the other hand, a complete list of all characters/character strings that could be dangerous for the application and possibly other systems is defined. These characters/character strings are rejected. All other characters/character strings are accepted.

Motivation: The selection of the validation principle is a fundamental decision. In the case of blacklisting, problematic patterns are defined and filtered out of the data. Everything else is let through. With white listing, the reverse is the case: anything not expressly allowed is blocked. Input validation by means of blacklisting is problematic because there is a danger of patterns that may cause problems being forgotten at the design stage or during implementation. Due to new attack methods, for example, further problematic patterns may arise during live operation. With the whitelisting approach, these patterns are generally blocked automatically. With a blacklist approach they are let through until the rules are modified. Another problem with blacklisting is that characters or character strings may be filtered out that are explicitly desired in special use cases.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data

Req 17 If the web application discovers during data validation that the data deviates from the expected content or formats, the web application must reject this data and stop the requested action.

Depending on the criticality of the web application, it may also be advisable to terminate the current session. This applies in particular if conscious data manipulation can be assumed. This can, for example, be the case if the length of the transmitted data exceeds the length that was specified in the form field, if additional parameters are transmitted or if the transmission method for the parameters does not match the specifications of the web application (GET vs. POST, cookies vs. hidden field).
In some scenarios, web applications automatically correct incorrect entries ("sanitizing"). This enables a web application to, for example, allow user-friendly entry of data with various spellings. However, this also leads to new risks due to the additional complexity and new attack vectors. Therefore, sanitizing must be avoided wherever possible. Sanitizing should be used only in those scenarios in which misuse of the sanitizing or conscious manipulation of the data can not occur. With sanitizing, it is also important to pay attention to the nested input of attack vectors (such as "&lt;script&gt;ript"), which an attacker can, for example, use to circumvent filters.

**Motivation:** Unexpected input data and suspected data manipulation indicates an attempted attack. A web application must respond to this in a suitable manner in order to prevent this attack or at least delay it.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data

---

**Req 18** The web application must not use any user input data for direct access to files or to other server-side resources.

Example: A web application shows the Deutsch.txt help file if a user enters “Deutsch”. Also, the web application does not sufficiently validate the user input. This means the attacker can also cause the web application to show the user file, for example by entering “../../etc/passwd%00” (Path Traversal/Local File Inclusion). Hazardous characters in this context are the characters for switching the directory and the null byte. In this example, the null byte character ends the string, thus preventing “.txt” from being added. The attacker can also attempt to use these malicious characters in various encodings. Apart from input validation and, where applicable, verification of the permitted files and file types by means of white listing, the most effective form of protection against such attacks is configuration settings, e.g. for php. Furthermore, if there is a corresponding security gap, an attacker can even incorporate files into a script and thus cause code to be executed (remote file inclusion).

**Motivation:** If a web application uses input data for direct access to files, an attacker may be able to gain access to files that he or she would not usually be able to access.

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data

---

**Req 19** If files can be uploaded by a client or transferred from other systems, the web application must check these files and process them in a such a way that no risks arise for the application or for users.

Suitable measures for this include:
- Limiting the size of transmitted files.
- Limiting the number of files that a user can transmit.
- Storing files as database BLOB instead of in the file system; if the files are stored in the file system, the system shall be hardened with regard to authorizations.
- Examining files for malware or, alternatively, transcoding (in the case of images, video and audio files).
- Analyzing archive files for malware and ZIP bombs.
- Validating the file names for correct file extensions, integrated path information and active code, in particular JavaScript in file names.
- If meta data of files (e.g. Exif data) is processed or displayed, the meta data is also validated for active code.
• Validating the format of uploaded files. The format must correspond to the file extension and be valid, i.e. the format meets the specification; and only certain file formats are accepted, e.g. only image formats in the case of a photo gallery.

• Setting the content disposition header to “attachment” and explicitly setting the “filename” parameter if users open untrustworthy files. This must be taken into account particularly when users open files that were provided by other users.

Motivation: Active code or malware in uploaded files jeopardize the application and/or other users who open or download these files. Suitable transcoding or analyses must be used to remove malware from the file. In addition, other risks may arise as a result of uploaded files, e.g., denial-of-service attacks. The “attachment” value in the content disposition header in turn prevents files from being automatically displayed in-line in the browser. Instead, the browser outputs a download message. In this way, for example, the execution of XSS in pdf files can be prevented.

For this requirement the following threats are relevant:
• Unauthorized access or tapping of data
• Unauthorized modification of data
• Disruption of availability

Req 20 If the web application transfers data to another system or to a client (browser), the web application must ensure that this data cannot trigger any undesired actions there or have any other undesired effect.

Based on the input data, web applications often create data structures or command lines that are then forwarded to a sub-system or client. Input data is usually user input, but may also be data that is transmitted from another system. In the sub-system or client, an interpreter evaluates the data structure or executes the command. However, for every sub-system, there are characters that have a special meaning for this system. These are known as meta characters. Using these meta characters and appropriate input data, an attacker could gain access to the interpreter or manipulate the interpreter's behavior. For example, data that is transmitted to an SQL database may contain database commands that the attacker inserted via his or her input data (SQL injection).

Such an attack is generally known as an interpreter injection. Common types of interpreter injection include HTML injection (better known as Cross-Site Scripting, XSS), SQL injection and NoSQL injection, XML injection, Xpath injection, LDAP injection, code injection, and so on.

Preventing interpreter injections requires consistent separation of input data and commands.

The preferred method for this is to use a safe API, which provides a parameterized interface so that the interpreter is not called up by means of input data. If no parameterized API is available, the meta characters must be explicitly cleaned for the receiving system or client. They must be escaped, encoded or filtered out before the output is transmitted.

Motivation: Correct and complete validation of the transmitted data is always the basis for successful defense against interpreter injections. For certain interpreters, additional measures can be taken, in particular in the form of safe APIs, which, in combination with data validation, provide the best possible protection.

To prevent SQL injections, for example, a safe API (in this case prepared statements) can be used and/or the transmitted data can be cleaned. This ensures that no unwanted database commands are triggered when the database is processing this data.

On the other hand, JavaScript is usually active in the browsers, meaning sections of an HTML page marked with appropriate tags (<script>…</script>) are executed in the browser as program code. The web application must, therefore, ensure that HTML pages only contain...
script code where this is explicitly provided for. If the content of the HTML page is based on user input, unwanted program code may be run in the browser (Cross-Site Scripting, XSS) unless appropriate output validation is performed.

For this requirement the following threats are relevant:
• Unauthorized access to the system
• Unauthorized access or tapping of data
• Unauthorized modification of data

---

Req 21 Each output of the web application must be in correct and defined encoding and must specify the correct MIME type.

In the case of HTML output, this must be in an encoding that complies with ISO 8859-1 (Latin-1), ISO 8859-15 (includes the € symbol) or UTF-8.

The type and encoding of the output must be defined, generally by using the relevant HTTP header: “Content-Type: text/html; charset = ISO-8859-1”.

In the case of HTML documents for which the application scenarios lead to the assumption that users will download the documents, the content type and character set must additionally be defined using a corresponding <meta> directive.

In addition, the HTTP header “X-Content-Type-Options” must be set with the value “nosniff”.

For content that is embedded (usually by means of <embed> or <object>), the MIME type must be specified using the “type” parameter.

For other output, suitable encoding tags must also be used (e.g., via a suitable encoding attribute in the XML declaration for XML documents).

**Motivation:** These measures prevent clients from interpreting the data that a web application supplies incorrectly or decoding it incorrectly.

The nosniff header instructs the browser to trust the supplied content type and not to attempt to determine the content type itself.

Incorrect determination of the content type or faulty decoding can lead to security problems. Frequently attackers can insert meta characters and thus perform successful attacks because these meta characters are not recognized due to decoding errors.

For this requirement the following threats are relevant:
• Unauthorized access or tapping of data
• Unauthorized modification of data

---

Req 22 The web application must validate all input data that is returned to the browser and it must HTML encode all meta characters to prevent Cross-Site-Scripting (XSS).

At least the following HTML meta characters must be converted into normal plain-text characters:
• & in: &amp;
• " in: &quot;
• < in: &lt;
• > in: &gt;
• ' in: &#39;

If an encoding other than ISO 8859-1 (Latin-1) is used (e.g. UTF), the corresponding meta-character encoding must be taken into account.

However, the correct encoding of HTML meta characters is not only relevant for web applications that output HTML content to browsers. This also always applies when outputting
relevant content to other web clients. An example of this is e-mail clients if e-mails can be initiated by means of a web application and users can specify the content of these e-mails.

XSS refers to the planting of malicious HTML and JavaScript code on a web page. There are three basic types of XSS attacks:

- **Reflected XSS**: With this variant, the server returns a user's input directly to the browser. If this input contains script code that the user's browser then interprets, malicious code can be run in the browser. This type is known as non-persistent because the malicious code is only temporarily introduced. The malicious code is not permanently saved. If you call up the page again without the manipulated URL or the manipulated form, the malicious code is no longer contained. Example: a search function is to allow all documents on a website to be searched. The search term is transferred via URL parameters: http://example.com/?search=search_term. On the results page, the search term is shown again ("reflected"). However, the web application outputs the search term without sufficiently validating the user input: <p>You searched for: search_term</p>. If the following search term is used: <script type="text/javascript"> alert("XSS") </script>, the following HTML code is generated:<p>You searched for: <script type="text/javascript"> alert("XSS")</p>. In order to launch an attack by means of reflected XSS, the victim is usually tricked into clicking on a malicious link or submitting a specially crafted form.

- **DOM-based XSS**: In contrast to the above XSS variants, the web application on the server is not involved here. This means even static HTML pages with JavaScript support are vulnerable to this attack. The malicious code is transmitted for execution directly to a script on the client side. Such an attack requires that a user specifically access a manipulated URL. Example: http://example.com/foobar.html#arg=argument_value. JavaScript code on the client side reads out the URL parameters and inserts them into the HTML document without verification:<p>You have attached this string to the URL: argument_value</p>. If the URL is now manipulated as follows:

http://example.com/foobar.html#arg=<script type="text/javascript"> alert("XSS")</script>, the HTML code that the client-side script outputs will be as follows: <p>You have added this string to the URL: <script type="text/javascript"> alert("XSS")</script></p>. The script is thus run in the context of the page being accessed.

The JavaScript code used in these examples only generates a harmless warning message with the text "XSS". However, the same method can be used to insert other JavaScript code with much worse consequences.
3.06 Web Applications

Motivation: XSS is the most common vulnerability in web applications. Such an attack is possible if the web application accepts non-trustworthy data and sends this to a web browser without sufficient validation or encoding. XSS thus enables an attacker to run script code in a victim’s browser in order to, for example, take over sessions, change page content or redirect the user to malicious websites. To prevent XSS, the various meta characters need to be escaped or encoded. However, this must always be performed depending on the specific context in which the meta characters are output (HTML, JavaScript, CSS, URL, etc.). HTML encoding is the basic protective mechanism against these attacks if user input is simply output in the website HTML. In the case of output in other contexts, protection against XSS is much more complicated. Developers must implement this in their web applications if this is not already ensured through a correctly used framework.

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data
- Unauthorized modification of data

Req 23 If text formatting is permitted in user input, a whitelist input filter for supported and permitted HTML tags must be implemented in the web application to prevent Cross-Site-Scripting (XSS).

Attributes that can be used to execute scripts, e.g., `<style>` or `<on*>`, must not be allowed.

Motivation: If the use cases of the web application do not allow HTML encoding, white listing is the most secure alternative for preventing XSS. The permitted HTML tags must be selected carefully and restrictively, otherwise dangerous attack variants can easily be overlooked.

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data
- Unauthorized modification of data

Req 27 If input data is used when forming database commands, the web application must prevent inter-preter injections (SQL injection or NoSQL injection) through suitable countermeasures.

This applies both to SQL injection in the case of SQL databases and to, for example, CQL or JavaScript injection in the case of NoSQL databases. For SQL and CQL requests, all variables must be bound in prepared statements. If stored procedures are used, these must also be used with bind variables. Dynamically combined requests must not be used.

If no secure access method, such as prepared statements, is available, the data must be cleansed. This must be performed depending on the specific interpreter language that is used. All meta characters in the interpreter language must be removed from the input or escaped. Here, meta characters are all characters that are used as string delimiters or for syntactical formatting of the code (" "; { } $ % # | etc.).

Example: SQL injection (NoSQL injection) flaws occur when a web application uses untrusted data in the construction of database commands:

```java
String query = "SELECT * FROM accounts WHERE custID='" + request.getParameter("id") + "';
```

The attacker modifies the 'id' parameter in the browser to send ' or '1'=1:

```java
http://example.com/accountView?id=' or '1'=1
```

This changes the meaning of the query to return all the records from the table accounts.

```java
SELECT * FROM accounts WHERE custID=' ' or '1'=1
```

In the worst case, the attacker uses this weakness to invoke special stored procedures in the database, allowing a complete takeover of the database or of the database host.

Motivation: Structured Query Language (SQL) is a query language for relational databases. Predefined sections are combined with user inputs to create a complete query. If user input is
not sufficiently checked or preprocessed, an attacker can plant any SQL commands in the query (SQL injection). The use of prepared statements prevents user inputs from being processed by the SQL interpreter, since these are precompiled and parameterized for processing.

**CQL (Cassandra Query Language)** is an interpreter language for Cassandra databases. CQL is therefore generally vulnerable to injection attacks, too, although it is a NoSQL database.

Other NoSQL databases, such as MongoDB, use other interpreter languages, such as JavaScript. In these cases, the respective syntax can be used to perform interpreter injections. This means data that is used to form the database queries must be validated with reference to these meta characters.

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data
- Unauthorized modification of data

---

**Req 28** If the web application creates XML data structures, XSLT style sheets or XPath queries and also uses user input data for this, the web application must prevent XML injections or XPath injections with suitable countermeasures.

For this, the user input must be checked against a whitelist and rejected if necessary. If white listing is not possible, the characters ( ) [ ] < > / " : , * and all white space characters (spaces and tabs) must be removed or escaped.

Example: An XPath query that returns the account whose username is "gandalf" and the password is "ic3" would be the following:

```xml
string(//user[username/text()='gandalf' and password/text()='ic3']/account/text())
```

If the web application does not properly validate user input, the attacker will be able to inject XPath code and interfere with the query result. For instance, the attacker could input the following values:
- Username: ' or '1'='1'
- Password: ' or '1'='1'

Using these parameters, the query becomes:

```xml
string(//user[username/text()=' or '1'='1' and password/text()=' or '1'='1']/account/text())
```

As in a common SQL Injection attack, the attacker has created a query that always evaluates to true, which means that the web application will authenticate the user even if no valid combination of username and password has been provided.

Another type of XML injection is where CDATA elements are used to insert malicious content that is ignored by the XML parser.

```html
<! [ C D A T A [ < I M G S R C = h t t p : / / w w w . e x a m p l e . c o m / l o g o . g i f o n - mouseover=javascript:alert('Attack');>;]]>
```

In this example XML injection could be used for an XSS attack against the web application.

**Motivation:** XPath is a language for querying XML data. XPath injection is the XML counterpart to SQL injection. If an attacker plants XML meta characters, this can lead to malformed XML. In addition, it is possible to introduce entire tags or attributes.

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data
- Unauthorized modification of data
Req 29 To prevent code or command injections, the web application must not use any user input data to create shell commands or commands for the active program (e.g. via eval()). This applies on both the server side and the client side.

Example with PHP:
```php
$myvar = 'somevalue';
$x = $_GET['arg'];
eval('$myvar = ' . $x . ';');
```
If in this case the request parameter "arg" is assigned "10; system('/bin/echo uh-oh')" processing the above with eval() will execute injected code.

Example with JavaScript (via JavaScript Object Notation, JSON):
```javascript
eval({"menu":{"address":{"line1":addressLine1,"line2":"","line3":""}}});
```
If in this case the addressLine1 variable from the request is assigned the value ",arbitrary:alert('executed!'),continue:" injected code will be executed here, too.

Example with JSON array
```javascript
mydata=['1', '2', '3']
```
If '3' is changed to '];alert(xss);//, the evaluation of "mydata," e.g., var myArray=eval(mydata), executes the contained JavaScript.

**Motivation:** With code injection, an attacker plants executable code. This can be server based (e.g. via PHP) or client based (e.g. via JavaScript). If the attacker is successful, this code is executed and can cause major damage. However, an attacker can also use code injections to insert Cross-Site Scripting or SQL injection attacks.

All user inputs represent a potential attack surface for this. Direct execution via eval() is highly problematic as the data is interpreted directly without further checks with regard to security or validity.

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources
- Disruption of availability
Req 30 If the web application uses user input data to form HTTP headers, this data must be sufficiently validated. Dangerous characters, in particular CR and LF (%0d and %0a), must be filtered out before the input data is inserted into the header of an HTTP response.

HTTP header injection can occur if headers (e.g. redirection URL, set-cookie instruction) are generated dynamically and input data is included that was not verified sufficiently. Header injections can be used for HTTP response splitting attacks. To do this, the attacker places a CR/LF character in the input data followed by a complete HTTP header. He or she thus triggers a response with an HTTP request, which the victim interprets as two HTTP responses. The second of these responses is entirely under the attacker’s control.

Example: The name of an author, e.g. of a weblog entry, is read from an HTTP request.
```
String author = request.getParameter(AUTHOR_PARAM);
The name is set in a cookie header of an HTTP response.
```

However, in case the user input is not validated sufficiently, the value of the cookie will only maintain its correct form, if the value submitted for AUTHOR_PARAM does not contain any malicious characters. If an attacker submits a string, such as "Wiley Hacker\r\nHTTP/1.1 200 OK\r\n", then the HTTP response would be split into two responses of the following form:
```
HTTP/1.1 200 OK
...
Set-Cookie: author=Wiley Hacker
HTTP/1.1 200 OK
...
```

Clearly, the second response is completely controlled by the attacker and can be constructed with any header and body content desired.

Motivation: The ability of the attacker to construct arbitrary HTTP responses permits a variety of resulting attacks, including: Defacement, Cache Poisoning, Cross-site Scripting (XSS) and Page Hijacking. In order to prevent this, input data must be sufficiently validated and dangerous characters must be filtered out, before the data is inserted into the header of an HTTP response.

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data
- Unauthorized modification of data

Req 31 The web application must not use any user input data to form URLs for redirects or other URLs.

Motivation: Attackers can use such insecure redirects, for example, to implement more effective phishing attacks or to lure users to pages with malicious code. The victim clicks on apparently serious links that he or she received from the attacker, but is then redirected to a phishing site. The attacker simply manipulates the input data that is used for a re-direct, thus using the original website as a trustworthy springboard.

By manipulating a URL an attacker may also be able to perform injection attacks or cause other side effects.

For this requirement the following attacker threats are relevant:
- Unauthorized access or tapping of data
- Attacks motivated and facilitated by information disclosure or visible security weaknesses
Req 32 Steps must be taken to prevent e-mail commands from being injected through a web application into a downstream mail server (e-mail injection).

Relevant e-mail commands in this context are SMTP commands, IMAP commands, attributes of Internet messages and MIME boundaries.

Therefore, a secure e-mail library must be used that does not permit this kind of injections.

If no secure e-mail library is available, all input data must be validated and cleansed before being forwarded to an e-mail server:

- For ARPA Internet messages: Input data must not contain “<CRLF>” (apart from in the body).
- For IMAP data: Input data must not contain “<CRLF>”.
- For SMTP data: Input data must not contain “<CRLF>”. Exception: In the case of the SMTP command DATA, every occurrence of “<CRLF>” must be replaced by “<CRLF>..”.
- For MIME e-mails: Input data that is sent as part of the e-mail body must not contain “<CRLF>-”. Input data that contains “<CRLF>--” must be cleansed, e.g., by inserting a space between “<CRLF>” and “--” (which however cannot be reversed by the recipient).

Example SMTP injection: The injection is performed against a webmail parameter associated with sending an e-mail. Commonly, the webmail application presents to the users a form where they must provide the required information. The corresponding input data is then used to subsequently create SMTP commands.

A typical part of a HTTP request for e-mail sending would look like this:

```plaintext
POST http://<webmail>/compose.php HTTP/1.1
...
Content-Disposition: form-data; name="subject"
SMTP Example
...
```

Which would generate the next sequence of SMTP commands:

```plaintext
MAIL FROM: <mailfrom>
RCPT TO: <rcptto>
DATA
Subject: SMTP Example
...
```

If the application doesn't correctly validate the value in the parameter "subject", an attacker could inject additional SMTP commands into it:

```plaintext
POST http://<webmail>/compose.php HTTP/1.1
...
Content-Disposition: form-data; name="subject"
SMTP Injection Example
.
MAIL FROM:
notexist@external.com RCPT TO:
user@domain.com
DATA
Email
data
```
The commands injected above would produce a SMTP command sequence that would be sent to the mail server, including the MAIL FROM, RCPT TO and DATA commands as shown here:

```
MAIL FROM: <mailfrom>
RCPT TO: <rcptto>
DATA
Subject: SMTP Injection Example .
MAIL FROM: notexist@external.com RCPT TO: user@domain.com
DATA
Email
data .
```

And an additional email will be sent.

**Motivation:** Validating user input prevents attacks via e-mail injection in which the attacker can execute various undesirable actions, e.g., sending anonymous e-mails, spamming, relaying, circumvention of Captchas and other restrictions in the application as well as the exploitation of protocol vulnerabilities.

Secure e-mail libraries exist which execute corresponding validation measures and per se already prevent e-mail injection. The most secure method is therefore the use of such an e-mail library.

For this requirement the following threats are relevant:

- Unauthorized use of services or resources
- Denial of executed activities

---

**Req 33** If it is possible to initiate an e-mail or SMS from within the web application without prior authentication (e.g. in case of a password reset or for recommendations) or to send data to the web application without prior authentication (e.g. contact forms or page ratings), the web application must prevent misuse of this function (in particular for spam or denial of service) through suitable security measures.

If this can be implemented in the specific case, it is recommended that the web application requires a predefined security question to be answered prior to that. In particular, this measure is suitable for sending a registered user a new initial password if the user has forgotten his or her password (password reset).

Alternatively, the number of e-mails, text messages or forms that can be initiated using this function within a certain time period can be restricted. A disadvantage of this measure is that it can easily be misused to perform a denial-of-service attack for this function, thus blocking legitimate users.

Captcha can also be used for this. Either the web application requests an answer to a captcha for each action, or the web application displays a captcha only in case a defined threshold value is exceeded or in case of other irregularities.

A Captcha is a test to distinguish between machines and people. The most frequently used variety is image Captchas, consisting of an image with distorted numbers and letters. If the content of a Captcha is successfully entered into a text field, the web application can assume that the input was not performed automatically. However, you should note that these solutions are generally not barrier-free. This can be a problem in particular in the case of web applications that employees use. There are also low-barrier versions of Captchas. These could be text tasks that are also read out by a screen reader or can easily be scaled by increasing the
3.06 Web Applications

text size. Examples of this are text Captchas that present easily solvable arithmetic problems, logic puzzles or knowledge questions (e.g. “the number 12 plus the number 6 equals?”). Alternatively, an audio Captcha can be provided in addition to the image Captcha. So that this audio Captcha cannot be solved via voice recognition, a disguised voice or acoustic backdrop with background noise is usually used.

In case of a function for sending recommendations, the following measures must be implemented:

- The specification of recipient addresses must be limited to a sensible and minimal amount.
- The web application must specify the subject line of an e-mail recommendation. Either the application automatically inserts the subject line or the user selects the subject line from a predefined list.
- A note must be included in the e-mail recommendation stating that the sender’s e-mail address has not been verified.
- If the text body of the e-mail recommendation contains a text field that the user can fill out, the e-mail must contain information regarding the purpose of the e-mail. The text the user enters must be identified accordingly as user-defined text. Only letters, digits and certain carefully selected special characters must be permitted in the text field. If an HTML e-mail is sent, special characters must be HTML encoded (e.g. “&lt;” instead of “<”).

Motivation: An attacker sending a large number of SMS text messages or e-mails to one or more users leads to dissatisfied users, image problems and possibly to a large number of inquiries forwarded to customer service. If users can send contact forms without prior authentication, an attacker can anonymously send a great many of these forms in an easy way. When these forms are evaluated by customer service, an overload might occur, so other legitimate requests will not be answered (on time). By means of mass transmission or systematic transmission of page ratings, an attacker can arbitrarily manipulate any rating.

For this requirement the following threats are relevant:

- Unauthorized modification of data
- Unauthorized use of services or resources
- Disruption of availability
3.06 Web Applications

Authenticated and Authorization

Req 34 For every user action, the web application must verify on the server side that this user is authorized to access the data/content or execute the function.

In particular, access protection must not rely solely on data, content or functions being kept secret. For example, access protection which is merely based on the fact that such resources are not linked is not sufficient.

Motivation: A web application must recognize and prevent unauthorized access attempts. Otherwise an attacker can, for example, manipulate parameters and thus gain access to data or execute functions for which he or she has no authorization. All parameters that are exchanged between the browser and web application can be manipulated – form fields, URL parameters, hidden fields, HTTP headers, etc. An attacker can access data and functions that are kept secret by determining the relevant addresses through fuzzing, finding them out via social engineering or because he or she knows them thanks to previous activities.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources

Req 35 The web application must monitor the correct transaction flow on the server side to ensure that the application logic cannot be circumvented.

When a step is executed in a transaction, it must be checked on the server side that all necessary previous steps have been carried out (e.g., delivery of goods only after billing). To this end, the current transaction status must be stored in the application at all times. The web application must recognize invalid steps and prevent them by rejecting the transaction and, if necessary, also invalidating the session.

This also applies for forms that have to be completed in a specific order. The web application must ensure by means of verification on the server side that these forms have been completed and/or sent in the correct order.

Motivation: A correct transaction flow must be ensured as otherwise an attacker can circumvent the application logic. An invalid transaction flow indicates a possible attack that needs to be prevented.

For this requirement the following threats are relevant:
- Unauthorized modification of data
- Unauthorized use of services or resources

Req 36 Accounts must be used that allow unambiguous identification of the user.

Users must be identified unambiguously by the system. This can typically be reached by using a unique account per user. So named group accounts, i.e. the use of one account for several persons, must not be used. On exception of this requirement are so named machine accounts. These will be used for authentication and authorization from sys-tem to each other or for applications on a system and can’t be assigned to a single person. Such accounts must be assigned on a per system or per application basis. In this connection, it has to be guaranteed that this account can’t be misused. Possibilities to protect these accounts are:

- Configuring of a Password that fulfils the security requirements and is known by less than possible circle of ad-ministrators.
- Configuring the account that only a local use is possible and a interactive login isn’t possible.
• Use of a technique for authentication of the specific account with public and private key or certificates.
• Limiting the access over the network for legitimised systems.

Additional solution must be checked on their usability per individual case.

Motivation: Unambiguous user identification is a prerequisite for assigning a user the rights that he requires to perform his tasks on the system. This is the only way to adequately control access to system data and services and to prevent misuse. Furthermore, it makes it possible to log activities and actions on a system and to assign them to individual users.

For this requirement the following threats are relevant:
• Unauthorized access to the system
• Unauthorized use of services or resources
• Denial of executed activities

Req 37 If the web application can be used via the Internet, the registration process for the web application must ensure that the registration cannot be performed automatically.

We recommend using a double opt-in procedure for this – the web application sends a confirmation link (or an initial password / activation token) to the new user's e-mail address. The web application completes the registration only after the user has confirmed the registration via this link.

Captcha can also be used for this. A Captcha is a test to distinguish between machines and people. The most frequently used variety is image Captchas, consisting of an image with distorted numbers and letters. If the content of a Captcha is successfully entered into a text field during a registration, the web application can assume that the registration was not performed automatically. However, you should note that these solutions are generally not barrier-free. This can be a problem in particular in the case of web applications that employees use. There are also low-barrier versions of Captchas. These could be text tasks that are also read out by a screen reader or can easily be scaled by increasing the text size. Examples of this are text Captchas that present easily solvable arithmetic problems, logic puzzles or knowledge questions (e.g. “the number 12 plus the number 6 equals?”). Alternatively, an audio Captcha can be provided in addition to the image Captcha. So that this audio Captcha cannot be solved via voice recognition, a disguised voice or acoustic backdrop with background noise is usually used.

Motivation: Attackers create accounts en mass in order to abuse these. Spammers automate the registration of e-mail addresses, for example, and cause the operator major damage when using these accounts.

For this requirement the following threats are relevant:
• Unauthorized access to the system
• Unauthorized use of services or resources

Req 38 Accounts must be protected against unauthorized use by at least one authentication attribute.

The various user and machine accounts on a system must be protected from misuse. To this end, an authentication attribute is typically used, which, when combined with the user name, enables unambiguous authentication and identification of the authorized user. Authentication attributes include:
• Cryptographic keys
• Token
• Passwords
• PINs
This means that authentication based on a parameter that can be spoofed (e.g. phone numbers, public IP addresses or VPN membership) is not permitted. Exceptions are attributes that cannot be faked or spoofed by an attacker. Several of the above options can be combined (dual-factor authentication) to achieve a higher level of security. Whether or not this is suitable and necessary depends on the protection needs of the individual system and its data and must be evaluated for individual cases.

Motivation: Accounts that are not protected with a secret authentication attribute can be used by an attacker to gain unauthorized access to a system and the data and applications stored on it.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized use of services or resources
- Denial of executed activities

 Req 39 Accounts with extensive rights must be protected with two authentication attributes.

Accounts, for example used for administration, maintenance and troubleshooting, have extensive rights. Therefore a single protection (e.g. a password), as for normal user accounts with less rights, is not suitable. To get a higher protection level it is necessary to use more than one authentication attribute. For this a combination of an attribute that the user knows and an attribute that the user owns will be used often. This kind of authentication will be named as 2-factor authentication. Examples for 2-factor authentication are:
- Smartcard with PIN
- Private key with Passphrase
- Secure-ID Token with Password

Motivation: Accounts with extensive rights as used for system administration have a higher risk for system’s security. An attacker can get extensive rights by compromising such an account to get access to wide parts of the system and stored data.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized use of services or resources
- Denial of executed activities

Req 40 The web application must request a new authentication of the user, if the user wants to change critical data or trigger critical actions. In particular for a password change the web application must request that the user enters the current password correctly (and enters the new password twice).

Critical actions are for example triggered, if the user wants to change the email address used for the password reminder, to close his account, to change his address for dispatch oder to change to a role with additional rights.

Motivation: Asking for the password again prevents an attacker who has taken over a user’s session from changing his/her password or any other critical data as well. Entering the new password twice prevents a one-off mistake from causing a user to be locked out and having to request a new password.

For this requirement the following threats are relevant:
- Unauthorized modification of data
- Unauthorized use of services or resources
Req 41 If a single sign on procedure based on authentication tickets (e.g., cryptotickets) is implemented for the web application, replay attacks and unauthorized use of tickets must be prevented.

For this purpose, an authentication ticket must meet the following requirements:

- A ticket must be generated as soon as the web application is accessed. It must not be generated when the SSO portal authenticates the user otherwise a short ticket validity duration may lead to problems.
- A ticket must be generated specifically for the corresponding web application.
- The web application must check that the origin of a ticket is correct by verifying the digital signature (or possibly implicitly by decrypting an encrypted ticket).
- Each ticket must not be accepted by the web application more than once.
- The web application must check the time stamp of the ticket and the validity period must be as short as possible, usually a few minutes.

Both the transmission of the authentication ticket from the SSO portal to the user and from the user to the web application must be protected by TLS.

Motivation: With a single sign on (SSO) procedure that is based on authentication tickets, the user logs in to the SSO portal once. He or she can then use other web applications that are connected to this SSO procedure without having to provide any other authentication to these applications. When the user wishes to access another web application, the SSO portal generates a ticket for this. The ticket contains all the necessary information regarding the user’s identity. The SSO portal also digitally signs it (or encrypts it). The user is forwarded to the web application with this ticket. The web application checks the validity of the ticket on the basis of the signature and the information contained in the ticket. If this check is successful, the user does not have to provide authentication again.

However, measures must be taken to ensure that attackers cannot misuse these tickets. It is therefore necessary that the tickets are securely transmitted and restrictively generated and verified.

Implementation example: SAP login tickets, Cryptotickets.

For this requirement the following threats are relevant:
- Unauthorized access to the system

Req 42 If the web application accepts network-based authentication of mobile devices via HTTP headers (e.g., x-forwarded-for), it must be ensured that this header has been sent from internal systems (e.g., MIG, Mobile Internet Gateway) and has not been manipulated.

This check can be done on the basis of cryptographic procedures or by means of IP whitelisting of trusted network areas. For HTTP requests for which this check fails, the authentication must be rejected. Manipulation of the connection between the system that carries out this header enrichment and the target system must also be prevented.

By means of the network-based authentication the SIM card of the customer is authenticated. But even if implemented correctly, it is not possible to determine, how a HTTP request of the mobile device was initially triggered. It may as well have been triggered, for example, by a
malicious app. This must be considered subject to the specific use case, when an authentication procedure is chosen.

**Motivation:** Attackers must be prevented from simulating another identity by adding the header information for authentication directly to HTTP requests.

For this requirement the following threats are relevant:
- Unauthorized access to the system

---

**Req 43** Predefined accounts must be deleted or disabled.

Many systems have default accounts (e.g. guest, ctxsys), some of which are preconfigured with or without known pass-words. These standard users must be deleted or disabled. Should this measure not be possible the accounts must be locked for remote login. In any case disabled or locked accounts must configured with a complex password (12 character and more, use of upper/lower case, numbers and special characters). This is necessary to prevent unauthorised use of such a account in case of misconfiguration.

Exceptions to this requirement to delete or disable accounts are accounts that are used only internal on the system involved and that are required for one or more applications on the system to function. Also for this accounts remote access or local login must be forbidden to prevent a abusive use by users of the system.

**Motivation:** Standard users are typically generally known and can be used by an attacker for targeted brute force and dictionary attacks. Standard user accounts represent a special risk if they do not use a password or only use a stand-ard password that is generally known. Such standard user accounts can easily be exploited by an attacker in order to gain access to the system involved without being authorized to do so.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized use of services or resources
- Denial of executed activities

---

**Req 44** The authorizations for accounts and applications must be reduced to the minimum required for the tasks they have to perform.

Authorizations to a system must be restricted to a level in which a user can only access data and use functions that he needs in the course of his work. Suitable authorizations must also be assigned for access to files that are components of the operating system or of applications or that are generated by the same (e.g. configuration and logging files).

Alongside access to data, execution of applications and components must also take place with rights that are as low as possible. Applications should not be executed with administrator or system rights.

**Motivation:** If the rights granted to a user on a system are too broad, it could be possible for him to access data and applications which he is not permitted to view or use. This would give him the opportunity to disclose or modify confidential data and to manipulate system files. Applications with rights that are too broad can be used by a user to extend his own authorizations and thus to gain access to files and system components to which he would not have had access with his authorizations under normal circumstances.

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources
Req 45 If the web application allows a user to access various customer data records, it must be detected when a defined threshold number of accesses to customer data records has been exceeded. In this case, appropriate countermeasures must be taken to prevent further access.

Typical use cases for this requirement are for example sales or customer care portals that allow employees to access data records of a lot of different customers.

The threshold value specifies how many different customer data records can be accessed within a certain period of time. This is to be set depending on the application in question.

This applies to both individual users and user groups. For user groups, the accesses of all group members are added and compared with the threshold value. A user group may consist, for example, of all users in a specific call center.

If a threshold value is exceeded

- The user account must be blocked for a defined period of time or the relevant access methods must be blocked for a defined period of time; in the case of a user group, this applies to all user accounts that are assigned to the group.
- For users with extended access rights, e.g., system administrators, and users with normal access rights, an e-mail or an alert must be generated and sent to a defined group of people and a corresponding log entry must be generated.

If a user account is blocked, any active session for this account must be terminated without delay. This block must be in place for at least one hour.

Motivation: If a user requests an unusually large number of customer data records within a defined period of time, this indicates that someone is trying to obtain customer data for misuse. To make it possible to take countermeasures, the application needs to be able to detect possible misuse as early as possible. If there is suspicion of misuse of customer data records, appropriate countermeasures must be taken.

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data

Req 46 If the application includes a search function for customer data records, the search options must be restricted as much as possible.

The search function must force the user to enter at least one customer-specific attribute (last name, phone number, date of birth, etc.). It must not be possible to carry out a search based solely on a city name, a zip code, the contract type or the contract term, for example.

Motivation: A search with any search options can easily be misused. For example, all customers with a certain contract can be found in order to send them a competitor’s offer.

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data
Req 47 If the application includes a search function for customer data records, the number of search results that the search function returns must be restricted.

Typically, the number of search results must be restricted to a maximum of 50 customer data records.

Motivation: Using a search function, a user can obtain general information on all customers with a specific characteristic. This is prevented or at least made more difficult if the maximum number of results is restricted.

For this requirement the following threats are relevant:
• Unauthorized access or tapping of data

Req 48 If the web application provides a function for the mass export of customer data records, this function must be restricted to individual and documented users. In addition, the web application must log when a user makes use of this export function.

Mass data export must be permissible only if it is absolutely necessary for the user.

Motivation: A mass data export function offers very high misuse potential. Therefore, the use of this function must be restricted to well-known users who absolutely require this function for their work.

For this requirement the following threats are relevant:
• Unauthorized access or tapping of data

Req 49 If the web application allows a user to access customer data records, the customer’s consent must be verified before calling up or displaying detailed customer data.

If there is direct contact with the customer, the customer’s consent must be verified via the web application, e.g., by means of
• the user of the web application having to enter a TAN which was sent to the customer, e.g., by SMS,
• the user of the web application having to enter at least two values which are typically only known to the customer. These values could be the customer’s date of birth, his phone number or his customer account number, for instance.

If there is no direct contact with the customer, e.g., as part of backoffice or support activities, the customer’s declaration of consent must be available in writing.

Motivation: In order to prevent misuse, detailed customer data should be displayed only if the customer has consented to this.

For this requirement the following threats are relevant:
• Unauthorized access or tapping of data
Protecting Sessions

Req 50 The length of the web application's session identifier must be at least 120 bit.

Character strings of more than 36 digits [0...9] or character strings consisting of more than 20 upper case letters [A...Z], lower case letters [a...z] and numbers [0...9], for example, meet the requirements. All the relevant characters must be generated at random.

Motivation: An attacker can attempt to determine valid session identifiers by means of statistical analyses or brute force attacks. If this is successful, the attacker can take over the victim's session. This can be prevented by using random, complex session identifiers for the web application.

For this requirement the following threats are relevant:
• Unauthorized access to the system

Req 51 Cookies must be used to transmit the session identifier ("session cookies").

Motivation: Session cookies are the most secure mechanism for transmitting session identifiers. For cookies, the web application can define important, security-relevant characteristics (no permanent saving, secure transmission, no script access, restricted area of validity). In addition, other problems are avoided that can occur if the web application uses URL parameters to transmit session identifiers – for example, it is not possible to prevent users from copying and sharing a URL containing a valid session identifier. Furthermore, there is a risk of the session identifier being sent via the referer header to another web application.

For this requirement the following threats are relevant:
• Unauthorized access to the system

Req 52 A session cookie must not be persistent.

Motivation: A web application can set an expiry date for a cookie via the expires or max-age attribute. This is known as a "persistent cookie". The browser saves persistent cookies on the hard drive until the expiry date. Every person with access to the computer can read the persistent cookies. This can, for example, happen in an Internet café or through a successful attack.

Cookies without an expiry date are not persistent, meaning they are not permanently saved. The browser deletes these cookies as soon as it is closed.

For this requirement the following threats are relevant:
• Unauthorized access to the system

Req 53 If the session is secured by means of TLS, the web application must set the "secure" attribute in the session cookie.

Motivation: The "secure" attribute prevents the browser from sending cookies without encryption. This happens, for example, if part of the content of a web application is not encrypted. However, it can also occur through an active attack in which the attacker injects or presents unencrypted links or references.

For this requirement the following threats are relevant:
• Unauthorized access to the system
Req 54 The web application must set the “httpOnly” attribute in the session cookie.

A web application must not be designed so that the session cookie has to be accessed by JavaScript. This is the case, for example, if the session cookie has to be sent with XMLHttpRequests. This functionality prevents the web application from setting this attribute.

**Motivation:** The “httpOnly” attribute prevents client side script code from accessing the cookie. This prevents session cookies from being picked up by means of Cross-Site Scripting (XSS).

For this requirement the following threats are relevant:
- Unauthorized access to the system

Req 55 The web application must not set the “domain” attribute in the session cookie.

The “domain” attribute specifies the domain names for which the cookie is valid. The browser sends the cookie with all requests that the browser sends to this domain and its sub-domains.

Not setting this attribute is the most restrictive setting – in this case, the host name of the server that set the cookie is used as the default value.

The “domain” attribute might be set however, if no other web applications run on the specified domain and on all its sub-domains, so the session cookie is prevented from being sent to third party web applications, too.

Additionally, in this connection it is generally recommended that web applications with different levels of criticality or security are also run under different domains. Otherwise vulnerabilities in one web application may also endanger the security of the other web applications.

**Motivation:** Restrictive use of the “domain” attribute prevents the session cookie from being sent to other web applications. This also minimizes the risk of vulnerabilities in other web applications endangering your own web application.

For this requirement the following threats are relevant:
- Unauthorized access to the system

Req 56 The web application must set the “path” attribute in the session cookie so restrictively that the cookie is not sent to other web applications on the same host.

Web applications can have the same host name but be in different directories. It is therefore important to ensure that different web applications on the same host do not receive the cookies for the respective other application. If the “path” attribute is specified when setting a cookie, it is only valid in this directory and all subdirectories. The “path” attribute must therefore be set so that no other web application receives the session cookie.

**Motivation:** Restrictive use of the “path” attribute prevents the session cookie from being sent to another web application.

For this requirement the following threats are relevant:
- Unauthorized access to the system
3.06 Web Applications

Req 57 Following successful authentication of a user, the web application must generate a new session identifier and send it to the user's browser.

Motivation: This prevents a web application from being attacked by means of session fixation. For an attack of this type, the attacker first establishes a session, and then foists the session identifier in question on a victim. This is usually done by means of a link that contains the session identifier as a URL parameter. If the victim uses this identifier and then logs on into the web application, the attacker can take on this identity in the established session.

For this requirement the following threats are relevant:
• Unauthorized access to the system

Req 58 Only one session must be active for a user account at any one time.

A second login with the same user account must be prevented. Alternatively a second login can be permitted; in this case, the first session must be terminated. This variant can be sensible to prevent a user account from being temporarily blocked, for example due to the browser being closed or crashing without the user first logging out.

It is recommended that the web application shows the user a warning message when he or she logs in but a session for this user account is already in progress. This increases the probability that attacks on accounts will be detected.

However, there are web applications that are explicitly designed for access via various channels (web, mobile, TV) or have to permit multiple logins for other obligatory reasons. In such cases, it must be possible to open multiple parallel sessions, if required.

Motivation: If several sessions are active simultaneously for a user account, this may mean that different users are using the account at the same time or that an attempt is being made to attack this account.

For this requirement the following threats are relevant:
• Unnoticeable feasible attacks

Req 59 The web application must have a function that allows a signed in user to logout at any time.

Motivation: A user must have the possibility to protect a session and therefore his/her data against unauthorized access. A specific logout can be used to end a session in order to ensure that this session can not be continued by an unauthorized person.

For this requirement the following threats are relevant:
• Unauthorized access to the system

Req 60 If a user accessed the web application by means of a single sign on procedure and then logs out of the web application again, both sessions must be terminated – the session with the web application and the session with the original SSO portal.

A static logout URL however must not be used for this purpose, as it would be possible to misuse it for denial of service attacks on dedicated users (e.g. by placing requests to this URL on a web site controlled by an attacker).

If the session with the original SSO portal cannot be ended automatically, as an alternative the original SSO session must be shown (again) in the user's browser on logging out of the web application. This can be implemented, for example, by means of a redirect to the SSO portal after logging out.
Motivation: Single sign on (SSO) means that a user is able to use additional applications following one-off authentication on an SSO portal without having to re-authenticate himself to these applications. If, on logging out of an application, only the session with the application is invalidated and a session with the SSO portal that continues to be valid is not displayed, in most cases the user will not log out of the SSO portal. The session with the SSO portal remains valid and an attacker can take it over without being noticed.

For this requirement the following threats are relevant:
• Unauthorized access to the system

Req 61 If a user accessed the web application by means of a single sign on procedure and then logs out of the SSO portal, the session with the web application must also automatically be terminated.

For this purpose, the SSO portal must initiate a termination of the web application session as well.

A static logout URL must not be used for this purpose, as it would be possible to misuse it for denial of service attacks on dedicated users (e.g. by placing requests to this URL on a web site controlled by an attacker).

Motivation: Single sign on (SSO) means that a user is able to use additional applications following one-off authentication on an SSO portal without having to re-authenticate himself to these applications. When the user logs out of the SSO portal, he or she may not realize that the other applications are still in use. If the sessions with these applications are not terminated automatically, an attacker can take them over without being noticed.

For this requirement the following threats are relevant:
• Unauthorized access to the system

Req 62 After a user is inactive for a specified amount of time, a session timeout must occur in the web application.

The precise amount of time after which a session times out must be specified individually for each application. It depends on the application's sensitivity and purpose. It is recommended that the timeout occurs after a maximum of 60 minutes. It is also recommended that this time period be a configurable system parameter.

Motivation: The timeout protects the user if he or she forgets to log out. Inactive sessions can be taken over by an attacker without being noticed. A 60-minute timeout does not generally inconvenience users but it does significantly reduce the risk of session stealing. The reasons for choosing the specific timeout period may change during operation, in some cases at short notice. This situation can only be sorted out with a configurable parameter for the timeout.

For this requirement the following threats are relevant:
• Unauthorized access to the system

Req 63 When a user logs out or a session times out, the web application must make the session and the corresponding session identifier invalid on the server side.

Motivation: If a session is not completely invalidated on the server side, an attacker can continue an open session if he or she gains access to this session. This is possible, for example, if the attacker uses the same computer as the victim or if he or she has determined the session ID using a different attack.

For this requirement the following threats are relevant:
• Unauthorized access to the system
Req 64  The web application must use a mechanism against attacks by means of cross-site request forgery (CSRF). This mechanism must prevent an attacker from placing faked requests on a website that he controls, which would trigger valid actions for the web application, if users (that are authentic-ated to the web application) visit this website.

In particular, requests in the web application that can cause a data change or status change must be protected ac-cordingly. GET requests that merely request information do not need to be protected against CSRF.

Generally, “CSRF tokens” are incorporated as a hidden field for this. A CSRF token must be unpredictable and must be individual at least for every session. The web application must check the token before the requested action is per-formed.

If implementation as a hidden-field is not possible or reasonable, e.g., for AJAX requests, the CSRF token can be trans-mitted in a specific HTTP header (e.g., “X-CR SF token”). However, information that the browser sends automatically, such as cookies, does not provide any protection.

Example CSRF attack: A web application allows a user to transfer money:

http://example.com/app/transferFunds?amount=1500&destinationAccount=4673243243

As this state changing request does not include a secret token, an attacker can construct a request that will transfer money from the victim’s account to the attacker's account. The attacker embeds this request in an image or iframe and stores the request on a site under his control.

If the victim visits this prepared site, while already authenticated to example.com, the forged request will include the user's session information, inadvertently authorizing the request, which will therefore be executed.

Motivation: Requests that are not protected in this way are susceptible to cross-site request forgery (XSRF or CSRF, also known as session riding).

With this, a victim is made to unknowingly send a prepared HTTP request. This request then triggers an action in the name of the victim within an active session. This occurs, for example, through a visit to a malicious website that con-tains a relevant link to another web application (e.g. as an img, script or iframe tag). However, the victim cannot recog-nize this link. The browser follows this link in the background, so to speak. This successfully triggers an action as long as the user has an active session for this other web application, as the browser sends the session cookie automatic-ally.

However, if every HTTP request includes an unpredictable token, an attacker cannot prepare a valid request. A CSRF attack is then no longer possible.

The incorporation of the token as a hidden field results from the requirement that POST requests are to be used for re-quests that cause a data modification.

To ensure comprehensive protection against CSRF, a different CSRF token is to be used for each individual request in a web application that causes data to be modified. This prevents intercepted tokens from being used for a CSRF at-tack. In particular if a web application were to always use the same token for a session and also transmit this as URL parameters, it would still be susceptible to CSRF by reading out the browser history or via “history hacks”. However, most frameworks do not support CSRF tokens that are individual for each request. This variant may also lead to prob-lems using the web application, for example when working in several tabs or windows in parallel.

For this requirement the following threats are relevant:

- Unauthorized modification of data
Req 65 The web application must use a mechanism against clickjacking attacks which prevents the web application from being represented by other unauthorized web applications within an iFrame.

This is usually achieved by outputting the HTTP header “X-FRAME-OPTIONS” in the web application's responses. The header is set to “DENY” or “SAMEORIGIN”. If it is not possible to use this header, JavaScript validations must be used to ensure that the web application is not displayed in a frame.

In particular, forms und functions in the web application that can cause a data change or status change must be protected accordingly. Pages that merely present information do not need to be protected against Clickjacking.
Motivation: Clickjacking (a contracted form of click hijacking, also known as UI redressing) is an attack technique in which an attacker tricks a victim into clicking on elements he/she never intended to click. Like in a CSRF attack, the victim has to have an authenticated session with a “legitimate” web application and visit the attacker’s malicious page in parallel. The attacker uses an IFrame to insert parts of the legitimate application into his/her malicious page, which are masked or hidden by other elements on the malicious page. The victim is then led to click on content that is sup-posedly on the attacker’s page. However, in reality this executes a click and an action on the legitimate web application.

The “X-FRAME-OPTIONS” HTTP header prevents a page being displayed in a frame. If the element is set to “DENY”, the content is generally prevented from being displayed in a frame. The “SAMEORIGIN” value restricts the display in frames to pages within the same domain in which the web application itself is located.

JavaScript code can be used to check whether a page containing the code is actually the top page in the frame hier-archy. Unfortunately, various techniques are now known which get past the various common variants of these JavaS-cipt mechanisms.

Implementation example:

```<script>
if(top.location != location) top.location = self.location;
</script>
```

If the page is not the top page in the frame hierarchy, this JavaScript code simply makes the current page the top page in the hierarchy again. However, if the attacker’s page places itself in the top position again when it comes to display-ing the content of the iframe, this mechanism is circumvented.

```<style>
body {display: none}
</style>
```

```<script>
if(self == top) {
    document.getElementById('body')[0].style.display = 'block';
} else {
    top.location = self.location;
}
</script>
```

The style instruction causes a blank page to be displayed at first. The code then checks to see whether the page is dis-played as the top page in the frame hierarchy. If this is the case, the code sets the style property of the body element to an invalid value, which means the browser ignores it. As a result, the page is displayed normally. If the page is now dis-played inside a frame, it simply stays blank and tries again to get to the top of the hierarchy. If this does not succeed, the browser continues to display a blank page. If an attacker does succeed in getting his/her page into the uppermost frame, he/she can only display a blank page. This foils the attack. At the present time, this JavaScript code has resisted all attacks. However, the disadvantage is that, if JavaScript is deactivated, a white page is always displayed. Further-more, websites need to be rearranged if they use the style attribute of the body element.

For this requirement the following threats are relevant:
• Unauthorized modification of data
Req 66 If JSON is used to transmit data with need of protection, the web application must employ a suitable measure against JSON hijacking.

One of the following measures is sufficient to prevent JSON hijacking:

- A protective mechanism against cross-site request forgery is also implemented for requests for JSON documents. For this, an unpredictable token is added to every relevant request and checked by the web application.
- The direct execution of the response is prevented by a prefix which is added on the server side and removed on the client side.
- The content-type of the request must be expressly set to "Content-Type: application/json" to detect requests by harmful scripts. Requests with a different content type (such as "Content-Type: text/javascript") are rejected.
- The JSON data is commented out to prevent execution of this data. In this case, however, input validation must also be used to prevent the planting of comment characters in the JSON data.

Motivation: JSON hijacking circumvents the same origin policy (SOP) and enables unauthorized access to data with need of protection in a vulnerable web application. AJAX data requests are usually performed via an XMLHttpRequest object. But due to the SOP, a request of this kind does not work if the web page displayed in a browser tries to access a server in another domain. However, the SOP does not affect the src attribute of a <script> element.

In a similar manner to an CSRF attack, with JSON hijacking, the attacker lures the victim to a malicious website. There, a script tag is used to request the JSON data from the vulnerable web application on which the victim has a parallel active session. The data can then be evaluated and, for example, sent to a server belonging to the attacker. However, this only works because, as a rule, every valid JSON document is already valid JavaScript (unlike, for example, XML documents) and is thus executed directly by the JavaScript interpreter.

If JSON is used, the aforementioned measures help to prevent JSON hijacking:

- The attacker does not know the current value of this token, and therefore cannot create a valid request.
- While the engine of the attacked browser can only evaluate the received JSON data directly, the actual page can transform them before evaluation. Hence the data can be given a prefix to prevent execution by the at-tacked browser. It is recommended that an endless loop be used, preferably "for(;;);". The added prefix can be removed from the actual code on the client side.
- The malicious code is unable to set the content type. The default value is "Content-Type: text/javascript", if the JSON document is called up via <script src= ...>.
- The attacked browser’s engine can only evaluate the received JSON data directly, but cannot remove the comment characters. Direct execution is thus prevented.

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data
### Req 67

Authorization for cross-domain access to the web application must be defined restrictively.

The web application may allow solely such hosts cross-domain access, for which this is explicitly provided and absolutely required. Cross-domain access to content of the web application with need of protection must not be permitted. In particular, cross-domain access to the content of authenticated sessions must not be permitted. It is possible to prevent access to content with need of protection, for example, by placing the public methods of the application in a separate domain or possibly in a dedicated virtual directory. A separate cross-domain policy file is then defined for this domain or directory. Intranet applications must not grant authorizations for cross-domain access to domains that do not belong to the in-tranet.

Adobe Flash and ActionScript:

- Authorizations for additional hosts must be defined restrictively with complete, qualified host names and without using wild cards. This is usually done in the “crossdomain.xml” policy file.
- Authorizations to access dedicated APIs for other Flash files must be defined restrictively via the LocalConnection.allowDomain() ActionScript method.
- Authorizations for full access through other Flash files via the Security.allowDomain() ActionScript method must not be granted.
- The LocalConnection.allowInsecureDomain() and Security.allowInsecureDomain() methods must not be used.

The allowScriptAccess parameter, which can be used to specify whether an SWF file is permitted to call up scripts on the browser, must not be set. Thus, the default "sameDomain" applies, meaning the SWF file and the surrounding HTML page can also only communicate with one another if they originate from the same domain.

Microsoft Silverlight:

- Authorizations for additional hosts must be defined restrictively with complete, qualified host names and without using wild cards. This is usually done in the “crossdomain.xml” or “clientaccesspolicy.xml” policy file.
- The possibilities to access a Silverlight file using JavaScript methods or to modify the HTML file must not be extended via the EnableHTMLAccess property. By default, this is only permitted within the same domain.

In reverse, the possibilities for JavaScript to call up the Silverlight file methods via EXTERNALCallersFromCrossDomain must not be extended. By default, this is also only permitted within the same domain.

Local Messaging must not be permitted across domains.

Cross Origin Resource Sharing (CORS) defines a mechanism that allows the browser to initiate cross-origin requests. These requests are made, for example by means of the XMLHttpRequest (XHR) used by AJAX. This mechanism is based on the exchange of header information. Browsers send the ORIGIN header, which specifies the domain of the application that sends the request. Servers, in turn, return the Access-Control-Allow origin header to show which domains are permitted to access their resources. The browsers then perform this accordingly.

Permitted domains must therefore be defined by whitelisting without the use of wild cards. Furthermore, the server may send these headers only for selected URLs / resources which require cross-domain access. These must not be sent for the whole application. These requirements apply in the same way for the XDomainRequest implemented in Internet Explorer.

JavaScript:

Using the document.domain property, web applications can allow different sub-domains reciprocal access by means of JavaScript. The same origin policy applies by default. Using this
property, a shared domain can now be specified. For example, login.example.de and payment.example.de can each set the property to example.de.

This kind of release must not be provided via document.domain as it is not possible to prevent other sub-domains setting this value accordingly and thereby gaining unauthorized access.

Workarounds must not be used to enable cross-domain access. This applies, for example, to fragment identifiers (the part of a URL after the hash symbol) and to JSONP (JSON with padding).

The Cross-Origin PostMessage-API expands every window and every frame with a new method that allows text messages to be sent from one window to another. This is then possible independently of any existing cross-domain policies.

The “recipient” of the data must be explicitly defined in order to avoid the data being sent to an incorrect destination. In particular, the recipient must not be set to ".

The “sender” (“origin” attribute) must be verified. The data (“data” attribute) must be validated.

The exchanged data must be evaluated as data, never as code (e.g., using eval). To avoid DOM-based XSS attacks, the DOM of a page must not be modified on the basis of the exchanged data (e.g. by means of “innerHTML”).

All requirements regarding cross-domain interaction apply equally to windows, dialog boxes, framesets, frames and iframes, etc.

Motivation: The “Same-Origin-Policy” (SOP) is a security concept implemented in the browsers. It aims to ensure that each resource of a website may only communicate with the server from which it was loaded and may only access objects which were loaded from the same server. Cross-domain access is thus prevented.

However, there are configuration options for individual web technologies that relax the SOP and allow cross-domain access. This is dangerous because it allows other web applications to execute code in the context of the domain of our own web application. By default, a Flash application executed in a browser, for example, is not authorized to load data from anything but the original host. A restrictive definition of authorizations which may be necessary can reduce the associated risk.

If cross-domain access is permitted too freely, possibly even data with need of protection that is actually protected by means of authentication can be accessed.

If, for example, cross-domain requests are made by the Flash player, browsers send these requests with the cookies for the domain of the site to which these requests are sent. A malicious Flash file on the server of one domain can therefore send requests to another web-application on another domain from the browser's flash player. Cookies from any session running in parallel with this other web-application are then also sent and the other web application accepts these. The Flash file can in return evaluate the responses and gain access to the content with need of protection. However, this only works if a correspondingly permissive crossdomain.xml file was defined on the server of the other domain and the browser was therefore permitted to make the relevant cross-domain requests.

The basic protection against access from the Internet is overturned if an application that can, in fact, only be reached from the intranet allows cross-domain access from other domains. Flash files that are hosted on servers in these other domains would then, for example, be able to access this intranet application if they are accessed by a browser in the internal network.

For this requirement the following threats are relevant:

• Unauthorized access or tapping of data
• Unauthorized modification of data
Req 68 Non-trustworthy Flash or Silverlight files that the web application delivers must be placed in a separate domain for non-trustworthy content. They must not receive any authorizations for cross-domain access.

Files shall be classified as trustworthy if they were validated with regard to possible security problems.
Files that were not validated and were developed by external companies are not to be classified as trustworthy.

The web application must not deliver files of an unknown or external origin without any contractual relationships to the developer.

Motivation: It is necessary to prevent damage being incurred by customers due to these files from external sources. If such files are nevertheless supplied, the risk for the web application must be minimized through restrictive configuration.

For this requirement the following threats are relevant:
• Unauthorized access or tapping of data
• Unauthorized modification of data

Req 69 Flash or Silverlight files that are supplied by the web application must not access external, non-trustworthy data or load correspondent content.

Motivation: Such external, subsequently loaded content would otherwise be able to access the data of the file that loads them, allowing wide-ranging attack and manipulation opportunities.

For this requirement the following threats are relevant:
• Unauthorized access or tapping of data
• Unauthorized modification of data

Req 70 The web application must not use any RIA services that circumvent the “Same-Origin-Policy”.

This includes AJAX service bridges, HTML bridges, AJAX proxies and aggregate sites.

Motivation: The “Same-Origin-Policy” (SOP) is a security concept implemented in the browsers. It aims to ensure that each resource of a website may only communicate with the server from which it was loaded and may only access objects which were loaded from the same server. Cross-domain access is thus prevented.

AJAX service bridges / AJAX proxies serve, for example, as proxies for forwarding data traffic between JavaScript on the client and an external server. This circumvents the SOP. In addition, there is the risk that access restrictions on a third-party system will also be circumvented in this way. Another potential way of misusing services of this kind is to conceal the origin of an attack on the third-party system. In the case of aggregate sites, which combine widgets for various domains on one site and therefore also circumvent the SOP, malicious widgets would have full access to all functions of the other widgets.

For this requirement the following threats are relevant:
• Unauthorized access or tapping of data
• Unauthorized modification of data
Authentication Parameter Password

Req 71 If a password is used as an authentication attribute, it must have at least 8 characters and contain three of the following categories: upper cases, lower case, numbers and special characters.

A system may only accept passwords that comply with the following complexity:
- Minimum length of 8 characters.
- Comprising at least three of the following categories: upper/lower case letters, numbers and special characters

When a password is assigned, the system must ensure that the password meets these requirements. If a central system will be used for user authentication this function can be forwarded or delegated to this system.

Motivation: Passwords with the above complexity offer high robustness against attacks coupled with acceptable user friendliness. Passwords with this level of complexity have proved their efficiency in practice. Trivial passwords that are too short are susceptible to brute force and dictionary attacks and are therefore easy for attackers to determine. Once a password has been ascertained it can be used by an attacker for unauthorized access to the system and the data on it.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized use of services or resources
- Denial of executed activities

Req 72 If a password is used as an authentication attribute, the registration process for the web application must ensure that a user chooses his password on his own:
- Either the user sets his password during registration; or he receives an individual initial password (e.g. via email), that he must change immediately after logging in for the first time.

We recommend that the web application informs users about the criteria for passwords. In addition, the web application may display an indicator of the password strength ("password meter"). Alternatively, the web application may visualize, which criteria of the password policy (above all length and categories) are met by the password.

Motivation: User accounts without passwords or with predefined passwords that cannot be changed are particularly at risk of being taken over by attackers. In addition, pre-defined, unchangeable passwords are frequently not treated with care by the users (e.g., noted down), as they are usually not both securely selected and easy to remember.
Initial passwords are frequently transmitted in unencrypted format or set by third parties within the framework of support processes. To reduce the risk of misuse, new users must change these passwords immediately.
Transparent criteria for selecting strong passwords increase users' acceptance and awareness of security.

For this requirement the following threats are relevant:
- Unauthorized access to the system
Req 73 If initial passwords (or activation tokens) are used in the registration process for the web application, these must lose their validity after a suitable period of time.

The validity period must be set individually for each application based on its sensitivity and the password assignment procedure.

If initial passwords are sent electronically, they must lose their validity 24 hours at the latest after they are sent.

Motivation: If a new user has not used the initial password after a long time, it can be assumed that there has been an error or the user does not want to complete the registration process. Furthermore, the initial password is often still accessible (for example, in e-mail inboxes). Unauthorized persons can therefore determine the password and misuse it.

Therefore, it must lose its validity after a short time.

For this requirement the following threats are relevant:
- Unauthorized access to the system

---

Req 74 If a password is used as an authentication attribute, users must be enable to independently change the password anytime.

The system must offer a function that enables a user to change his password at any time.

When an external centralized system for user authentication will be used it is possible to redirect or implement this function on this system.

Motivation: The fact that a user can change his authentication attribute himself at any time enables him to change it promptly if he suspects that it could have been accessed by a third party.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized use of services or resources
- Denial of executed activities

---

Req 75 If a password is used as an authentication attribute, the web application must have a process that is protected against misuse and by means of which a user can reset his or her password.

For this, we recommend that users have to select a “security” question and enter the correct answer. The web application must request this data as part of the registration process or after the user's initial login. In addition, an e-mail address for the user, which he or she entered at the time of registration, for example, must be stored in the web application. If the user has forgotten his or her password, there is an appropriate function to select when logging in. Once the user has entered the relevant “security question” correctly, the password is reset. The web application creates a new initial password, which is valid only temporarily, and sends it to the stored e-mail address.

In the case that a user has also forgotten the answer to his or her security question, a fallback mechanism can also be provided, if required. For this, the user is offered a further process, but which can generally not be performed automatically, for example calling a hotline. Even for such a fallback mechanism, it is important to ensure that it cannot be mis-used.
Motivation: If a user has forgotten his or her password, it must be possible to reset the password. However, this must not lead to a reduction in the existing security level of the web application. Attackers must be prevented from finding a possibility to determine the (initial) passwords of other users and taking over their user accounts. It also should not be possible for a password reset process to be (automatically) misused, for example by resetting passwords en masse and bothering other users.

For this requirement the following threats are relevant:
- Unauthorized access to the system

Req 76 If a password is used as an authentication attribute, a protection against brute force and dictionary attacks that hinder password guessing must be implemented.

Brute force and dictionary attacks aim to use automated guessing to ascertain passwords for user and machine accounts. Various measures or a combination of this measures can be taken to prevent this.

The most commonly used protection measures are:
- Increasing the delay (e.g. doubling wait times at each attempt) for each renewed password input following an incorrect entry ("tar pit").
- Blocking an account following a specified number of incorrect attempts (typically 5). However has to take in account that this solution needs a process for unlocking and an attacker can force this to deactivate accounts and make them unusable.
- Using CAPTCHA to prevent automated attempts (often used for Web applications).

In order to achieve higher security, it is often meaningful to combine two or more of the measures named here. This must be evaluated in individual cases and implemented accordingly.

Motivation: Implementation of one or more of the above measures can prevent successful execution of brute force and dictionary attacks to ascertain passwords.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized use of services or resources
- Denial of executed activities

Req 77 If a user's attempt to log in fails, the web application must not give any information regarding which of the login information entered was incorrect.

Motivation: A more general error message makes it more difficult for attackers to find valid user names.

For this requirement the following threats are relevant:
- Attacks motivated and facilitated by information disclosure or visible security weaknesses

Req 78 If a password is used as an authentication attribute, they must be hidden when displayed on screen.

When a user enters a password or the password is otherwise displayed on screen, it must be made unreadable. Typically, the individual characters of the password are replaced by a character such as "*". Under certain circumstances it may be permissible for an individual character to be displayed briefly during input. Such a function is used, for example, on Smartphone's to make input easier. However, the entire password is never output to the display in plaintext.
Motivation: To prevent another person reading a password by chance or intentionally on screen during input, the pass-word must be rendered unreadable during input or display.

For this requirement the following threats are relevant:
• Unauthorized access to the system
• Unauthorized use of services or resources
• Denial of executed activities

Content Management Systems (CMS)

Req 79 Access to the editing environment of the content management system must not be via the Internet.

The editing environment must either be accessible from internal networks only or access restriction must be implemented by means of a technical solution, e.g. VPNs.

If protection via access restriction cannot be implemented, the access to the editing environment must be protected by means of another additional security level, e.g. by using two-factor authentication.

Motivation: This minimizes the risk of attackers gaining unauthorized access to the CMS (e.g., through CMS vulnerabilities they discover) and modifying content or reading information that has not been published.

For this requirement the following threats are relevant:
• Unauthorized access to the system

Req 80 The content management system must allow the authorizations for different content management activities to be assigned to different users/user groups so that a multi-stage publication process can be implemented.

Possible different user groups are Reader, Editor, Chief Editor, Administrator, Approver, and Publisher. Publication of content must take place according to at least a dual-control principle by an Editor and an Approver or Publisher.

Motivation: An appropriate publication process can minimize the risk of undesirable or incorrect content being published (through malicious intent or due to an error).

For this requirement the following threats are relevant:
• Attacks motivated and facilitated by information disclosure or visible security weaknesses

Req 81 The content management system must allow certain content to be assigned exclusively to particular editors or a group of editors.

It must then not be possible for other, non-authorized editors or groups to view or modify this content. This can be achieved by allowing the CMS to serve several clients, or by means of a suitable role concept.

Motivation: This enables unpublished (confidential) content to be recorded without it being revealed to all editor groups. It also minimizes the risk of content being modified without authorization.

For this requirement the following threats are relevant:
• Unauthorized access or tapping of data
• Unauthorized modification of data
Req 82 If the content management system provides functions for creating content that is to be published at a later date, it must not be possible to view or find this content in the web application before the publication date.

Likewise, it must not be possible to find the unpublished content using the web application’s search functions, secret URLs or through active manipulation.

Motivation: This enables unpublished (confidential) content to be recorded without being viewed in advance by web application users.

For this requirement the following threats are relevant:
• Unauthorized access or tapping of data

---

Req 83 The content management system must provide functions that make it possible to restrict the use of active content and scripting within the created content (to specific content or specific user groups) and, if necessary, to prevent it altogether.

Motivation: Content that could potentially pose a risk for users of the portal should be used only when necessary and only be created by editors who are explicitly responsible for this. For example, if script languages are used, content with cross-site-scripting attacks could be created.

For this requirement the following threats are relevant:
• Attacks motivated and facilitated by information disclosure or visible security weaknesses

---

Req 84 Preview functionalities of the content management system must be protected from unauthorized access – only successfully authenticated CMS users should be able to view the preview content and, in doing so, they may have access only to the content for which they have as a minimum read rights.

Motivation: This enables unpublished or incomplete content to be entered and verified without it being viewed in advance by web application users or other non-authorized editors.

For this requirement the following threats are relevant:
• Unauthorized access or tapping of data
Logging

Req 85 Security relevant events must be logged with a precise timestamp and a unique system reference.

Systems must log the occurrence of security-relevant incidents. So that these events can be evaluated and classified, they must be logged together with a unique system reference (e.g., host name, IP or MAC address) and the exact time the incident occurred. Furthermore, legal and data privacy regulations (e.g. time of storage of logging data) must be proved and followed. The following table lists events that are relevant to security and the corresponding data that typically has to be logged by a system. Exceptions are systems for which no or only restricted logging applies. Examples of such systems are customer devices such as Smartphone’s or IADs/Homegateways (e.g. Speedport).

Logging must be done considering the currently valid legal, wage and company regulations. This regulations state among others that logging of events can be done only earmarked. Logging of events for doing a work control of employees is not allowed.

Typical event that reasonable should be logged in many cases are:

<table>
<thead>
<tr>
<th>Event</th>
<th>Event data to be logged</th>
</tr>
</thead>
<tbody>
<tr>
<td>Incorrect login attempts</td>
<td>• Account,</td>
</tr>
<tr>
<td></td>
<td>• No. of failed attempts,</td>
</tr>
<tr>
<td></td>
<td>• Source (IP address) of remote access</td>
</tr>
<tr>
<td>System access with accounts with administrator rights</td>
<td>• Account,</td>
</tr>
<tr>
<td></td>
<td>• Access timestamp,</td>
</tr>
<tr>
<td></td>
<td>• Length of session,</td>
</tr>
<tr>
<td></td>
<td>• Source (IP address) of remote access</td>
</tr>
<tr>
<td>Account administration</td>
<td>• Administrator account,</td>
</tr>
<tr>
<td></td>
<td>• Administered account,</td>
</tr>
<tr>
<td></td>
<td>• Activity performed (configure, delete, enable and disable)</td>
</tr>
<tr>
<td>Change of group membership for accounts</td>
<td>• Administrator account,</td>
</tr>
<tr>
<td></td>
<td>• Administered account,</td>
</tr>
<tr>
<td></td>
<td>• Activity performed (group added or removed)</td>
</tr>
<tr>
<td>Critical rise in system values such as disk space, CPU load over a</td>
<td>• Value exceeded,</td>
</tr>
<tr>
<td>longer period</td>
<td>• Value reached</td>
</tr>
<tr>
<td></td>
<td>(Here suitable threshold values must be defined depending on the individual system.)</td>
</tr>
</tbody>
</table>

Logging of additional security-relevant events may be meaningful. This must be verified in individual cases and implemented accordingly where required.

Motivation: **Logging security-relevant events is a basic requirement for detecting ongoing attacks as well as attacks that have already occurred. This is the only way in which suitable measures can be taken to maintain or restore system security. Furthermore, the logging data is used as evidence so that legal steps can be taken against attackers.**

For this requirement the following threats are relevant:
• Denial of executed activities
• Unnoticeable feasible attacks
3.08 External Hosting

Introduction

This security requirement has been prepared based on the provisions of the Group Policy on IT/NT Security. The security requirement is used as a basis for an approval in the PSA process, among other things. It also serves as an implementation standard for provisions of the Group Policy on IT/NT Security in units which do not participate in the PSA process. These requirements shall be taken into account from the very beginning, including during the planning and decision-making processes. When implementing these security requirements, the precedence of national, international and supranational law shall be observed.

Requirements for external security management

<table>
<thead>
<tr>
<th>Req 1</th>
<th>The hosting provider must provide a policy-level contact for all security matters. The contact shall be available for technical/non-technical aspects.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Motivation: In the event of a security incident, it must be possible to deal with the matter quickly and competently. In order to prevent avoidable delays, a contact person is essential.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Req 2</th>
<th>The hosting provider / SaaS provider must provide a telephone line with 7x24 availability to accept all security matters. The hosting provider / SaaS provider must ensure the availability during regular business hours, if not stated otherwise in the SLA.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Motivation: In the event of a security incident, it must be possible to deal with the matter immediately. Availability at all times is therefore essential.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Req 3</th>
<th>The hosting provider / SaaS provider must have an annually updated security framework, implement it and submit it to the purchaser or a person nominated by the purchaser on request.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Motivation: The implementation of a security framework guarantees a structured and transparent approach with regard to security matters.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Req 4</th>
<th>The hosting provider/SaaS provider must establish processes and process documentation to respond quickly and efficiently to vulnerabilities and security incidents and to present them to the purchaser or a person nominated by the purchaser on request.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Motivation: In the event of a security incident, it must be possible to deal with the matter quickly and competently. A process description including training on the processes is essential to prevent avoidable delays.</td>
<td></td>
</tr>
</tbody>
</table>
Req 5 The hosting provider / SaaS provider must ensure that the purchaser and possibly also the Security Management responsible must be informed without delay of a security incident / security vulnerability.

Req 6 The hosting provider/SaaS provider must technically ensure that data of the purchaser based on his request or the request of the security management will be made unavailable within one hour.

In case of dedicated hosted systems, a complete shutdown / separation from the network is a suitable solution for this requirement. Who may be the requestor of such a measure shall be determined for each system individually.

Motivation: In the event of a security incident, it must be possible to quickly isolate corrupted systems for forensics and to avoid further damage.

Req 7 The hosting provider / SaaS provider must grant the right to perform its own audits of the hosting infrastructure used by it (supplier audits).

The hosting provider shall support the customer units named by it with the implementation. In detail, support means that

- test accounts are made available
- relevant security documentation / information is visible
- accesses (technical/physical) are made available.

Motivation: The principle of double checking shall be ensured by assigning different auditors.

Req 8 The hosting provider / SaaS provider must perform a security review of its hosting environment every 12 months and provide the results to the customer.

The scope of the audit should include at least the network infrastructure and the basic applications such as Apache or Oracle.

Apart from checks on the basis of port scanners and version scanners (communication shall be checked for anomalies.

Motivation: In the event of an undiscovered security incident, it can be detected by means of a review. Furthermore, cyclic checking of the systems enables adjustment to new developments and problem areas that have not been recognized up to now.
Req 9 To ensure the sustainability of the security review/audit measures, subsequent to the audit the hosting provider/SaaS provider must submit a appropriate time schedule for the near future elimination of the vulnerabilities to the purchaser or a person nominated by the purchaser on request.

At least network infrastructure and basis applications such as Apache or Oracle must be in scope of the Audit. Besides portscans also analysis of logfile anomalies and network traffic anomalies should take place.

Motivation: Tracking of events is possible on the basis of an action plan including time lines. This ensures that temporarily tolerated system states are resolved within agreed times.

Req 10 The hosting provider/SaaS provider must sign a Non Disclosure Agreement (NDA) with the customer, if the provider can gain business related internal information.

An NDA SHALL be concluded between the customer and the hosting provider to ensure that any information that has been passed on or made accessible remains confidential.

Motivation: By signing an NDA, information of the Group requiring protection can be forwarded on the basis of the “need-to-know” principle.
3.08 External Hosting

Req 11  The hosting provider/SaaS provider must sign a commissioned data processing contract with the customer provided this is relevant for the respective system/application for data protection reasons and depending on the applicable legal provisions.

Motivation: An agreement for commissioned data processing is, for example, necessary in Germany if a hosting provider/external partner processes data. This is due to national data privacy law.

Req 12  The hosting provider/SaaS provider must ensure that internal information made available is only accessible (subject to the “need-to-know” principle) to a small group of persons who are essential for executing the order.

Motivation: Narrowing down data to the smallest possible group of addressees minimizes the risk of data being passed on.

Req 13  In case of subcontracting the hosting provider/SaaS provider, if he receives access to confidential data (including personal related data), must ask for permission to use dedicated subcontractors.

Motivation: A high level of security can be ensured only if all participants in the process work in compliance with the same standards and this is documented by means of a contract.

Req 14  The hosting provider/SaaS provider must always be able to provide up-to-date meaningful information about hardware, the operating system, software, the architecture, contacts, hardening and escalation possibilities to the customer, free of charge.

Motivation: A high level of security can be ensured only if all participants in the process work in compliance with the same standards and these are known.

Req 15  The hosting provider/SaaS provider must ensure that both office spaces, technical rooms and the data centers used are protected against unauthorized technical/physical access.

Motivation: The risk of theft of information carriers with confidential content is limited by safeguarding the data centers and office spaces.

Req 16  The hosting provider/SaaS Provider must ensure that redundant systems are accommodated in separate fire sections.

Motivation: Operating redundant systems in separate fire sections offers protection against physical damage such as e.g. water and fire.
Req 17 The hosting provider/SaaS provider must ensure that additional security measures such as lock-able computer cabinets for secure hosting operation are provided for applications (systems) designated beforehand as relevant.

Motivation: Breaking down the security zones makes it possible to establish a higher security level for systems requiring special protection without all systems having to be protected generally at high cost.

Req 18 The hosting provider/SaaS provider must ensure that customer applications / data are operated in a logically and, if possible, even physically separate location from other customers’ applications/ data (if the type of application allows physical separation at all).

The minimum information that should be stored is which individual/ID accessed a building when. Storage can take place, for example, in the form of a separate database solution.

Motivation: Separation of systems according to customer minimizes the risk of systems influencing one another and the risk of a lower security level of a system of one customer of the hosting provider jeopardizing a system.

Req 19 The hosting provider/SaaS provider must ensure that MZ systems are not directly accessible from the Internet.

MZ stands for “militarized zone”, i.e., the zone containing business-critical back-end systems, which may only be ad-dressed by external sources via the detour of systems in the “demilitarized zone” (DMZ).

Motivation: Termination of all traffic in the DMZ enables more in-depth protection of all MZ systems.

Req 20 The hosting provider/SaaS provider must ensure that the actual data management of an application always takes place in zones not reachable from the internet.

Motivation: Placing the data management systems in the MZ minimizes the direct risk of attack.

Req 21 The hosting provider/SaaS provider must technically ensure that the exchange of data between a data center and the hosting provider’s data center can take place securely. The possibility of terminating encrypted (e.g., IPSec) traffic must be provided for this pur-pose.

It is also to notice, that the interfaces on both side must be secure, e.g. validate input and restrict access.

Motivation: Establishing a secure communication channel must be regarded as a basic measure so that individual effort for individual communication lines can be minimized.
Req 22  The solution architecture must be in accordance with industry best practices including separation from functional blocks.

Besides the architecture itself the building components such as database and web server must not be hosted on the same logical system.

Motivation: By using industry proven approaches, the costs for implementing security, reliability and availability can be reduced and the attack vectors can be limited.

Req 23  The hosting provider/SaaS provider must provide measures/tools for detecting attacks and use them.

Suitable measures at this point can be:

- Log file analysis tools (e.g., access/error logs), not meaning simple texteditors
- Firewall systems
- Intrusion detection systems
- Network monitors

Intrusion prevention systems are not mandatory from the current technology perspective.

Motivation: Damage can be averted be detecting attacks early.

Req 24  The hosting provider/SaaS provider must provide, implement and document suitable measures and processes for warding off attacks from the Internet on the systems/platforms.

Suitable measures at this point can be:

- Routers with traffic shaping
- Firewall mechanisms
- Multiple peering points

Motivation: Only with a timely response and proactive protection measures, a suitable availability of infrastructure can be guaranteed.

Implementation example: One example of suitable segmentation is the use of a separate VLAN.
Req 25 The hosting provider/SaaS provider must ensure the creation of a DMZ/MZ by using suitable active network elements (at least packet filters).

A demilitarized zone (DMZ) designates a computer network with security-controlled possibilities of access to the server connected to it. Only DMZ systems may be accessed from the external network/Internet.

Motivation: Establishing DMZ/MZ zones without using active network elements does not offer any additional protection.

Req 26 The hosting provider/SaaS provider must ensure that administrative access to hosting systems containing applications/data can take place only via approved and reliable encrypted protocols/entry points such as SSH.

Motivation: The risk of eavesdropping on administrative identifiers is clearly reduced by the use of encrypted protocols.

Req 27 The administrative access from the internet must be realized via certificate or OTP based authentication.

Motivation: The risk of eavesdropping on administrative identifiers is clearly reduced by the use of encrypted protocols.

Req 28 Depending on the information being worked with in the hosted service, the hosting provider/SaaS provider must ensure that its employees are obligated to comply with country specific telecommunications privacy and data protection laws of the respective legal entity.

In case of hosting personal data according to the national privacy law, this requirement must be fulfilled by a commissioned data processing agreement.

Motivation: To ensure that the data forwarded or accessible to the hosting provider is protected, the employees must be made aware in a suitable manner that the data requires protection and they must be obligated to observe this data protection requirement (where legally required).

For customer details in Germany this means, for example, the conclusion of a contract for commissioned data processing and the obligation of the employees to comply with telecommunications privacy and the German Federal Data Protection Act (Bundesdatenschutzgesetz).

Req 29 The hosting provider/SaaS provider must ensure that only authorized administrators/processes may set up, modify, delete, activate, block and view users subject to the “need-to-know” principle.

Motivation: Narrowing down certain critical functions to individual employees reduces the general risk of misuse.
Req 30 The hosting provider/SaaS provider must ensure that an administrator or user is never allowed to issue more rights/permissions than are granted to him.

Motivation: Narrowing down certain critical functions to individual employees reduces the general risk of misuse.

Req 31 The hosting provider/SaaS provider must ensure that systems are kept free from harmful software e.g. by the use of up-to-date virus scanners or other technical options.

Motivation: Harmful software represents a considerable risk for operation because access through such software to other systems cannot be ruled out, for example.

Req 32 By means of regular checks (i.e., at least once a month), the hosting provider/SaaS provider must ensure that only absolutely necessary software is installed/activated on the systems. The definition for the needed software usually is delivered from the SaaS/hosting partner.

This requirement must be considered in the context of the commissioned service. This means that this requirement does not apply to managed services where the service provider is responsible for the service.

Motivation: Regular checks can contribute towards ensuring that the security level stays at a known level and unnecessary software can be ruled out as an additional source of danger.

Req 33 By means of regular checks (i.e., at least once a month), the hosting provider/SaaS provider must ensure that only necessary services defined by the hosting/SaaS partner run on the systems.

This requirement is highly context sensitive, e.g. it is not valid for managed services as the service offerer hereby is responsible for the service.

Motivation: Regular checking can contribute towards ensuring that services needed for administrative or support purposes are ruled out as an unnecessary source of danger.

Req 34 The hosting provider/SaaS provider must ensure that systems/applications/middleware operated are hardened based on "best practice" approaches.

This requirement depends on the scope of the actual hosting agreement and shall be implemented on the basis of a complete hosting above and beyond a housing model. A good set of best practices can be found in the www.csisecurity.org webpage.

Motivation: Each hardened system boosts safeguarding of the entire data center.
Req 35 The hosting provider/SaaS provider must verifiable ensure a documented, implemented process for patch management.

Motivation: Current patch status reduces the risk of intrusions.

Req 36 The hosting provider/SaaS provider must verifiable ensure that applications/operating systems al-ways have an up-to-date patch level, which enables a flawless operation.

Motivation: Systems constantly kept at an up-to-date patch level are less vulnerable to attack from external/internal hackers. An up-to-date patch level therefore represents an elementary component of a secure system.
Req 37 The hosting provider/SaaS provider must establish, actively practice and test a backup and recovery process and present it on request.

Motivation: In the event of damage, backups and recovery processes enable live operation to be resumed quickly and therefore contribute towards quality of operation.

Req 38 The hosting provider/SaaS provider must ensure that only actively supported software is utilized by the supplier/manufacturer/developer.

Motivation: If an error should occur in a software component, a guarantee that patches can be provided within a short period of time is only possible if the software list is actively maintained.

Req 39 All input to the application must be validated in accordance with industry best practices.

Motivation: By using industry proven approaches, the costs for implementing security, reliability and availability can be reduced and the attack vectors can be limited.

Req 40 Secure configuration of database systems must be according to industry best practice.

Motivation: By using industry proven approaches, the costs for implementing security, reliability and availability can be reduced and the attack vectors can be limited.

Req 41 The hosting provider/SaaS provider must ensure that access to internal systems and applications of the provider is logged.

Logging should at least cover the last seven days and should include access to account management/system administration systems.

Motivation: Logging the access to internal systems enables effective clarification of all manner of incidents.

Req 42 The hosting provider/SaaS provider must ensure that only personalized accounts are used to perform internal work.

Internal work within the scope of this requirement includes:
- Setting up new accounts
- Extending firewall rules
- Any form of network administration measures

Motivation: In the event of damage, the use of group accounts does not permit neat investigation.

Req 43 The hosting provider must ensure a documented, implemented process for account management.
Motivation: Only by means of an established process is it possible to ensure that both newly hired individuals and individuals leaving the company are not provided with unauthorized access.

Req 44 Access control and authentication mechanisms must be in accordance with industry best practices.

Motivation: By using industry proven approaches, the costs for implementing security, reliability and availability can be reduced and the attack vectors can be limited.

Req 45 The hosting provider/SaaS provider must ensure that confidential information such as access data is stored only in an encrypted form based on current encryption standards.

Encryption of access and usage data on the hard disk should correspond to a strength of AES 256 bits.

Motivation: Coding of the data stored on data carriers makes it difficult for hackers to gain access to critical data.

Req 46 The hosting provider/SaaS provider must back up/protect data in such a way (technically, physically and organizationally) that unauthorized parties cannot gain access to the data.

Motivation: As backups may contain confidential information and generally enable conclusions to be drawn as to defensive measures, safeguarding of backups (by locking them away in a steel cabinet) is imperative to back up the information without any gaps.

Req 47 The hosting provider/SaaS provider must provide a process for secure destruction of data carriers / secure deletion of data, have put this process into practice and be able to produce evidence of the destruction of data carriers.

Motivation: Without secure deletion of data carriers, there is an immanent risk of data loss which must be explicitly kept to a minimum.

Req 48 All data handled by the application must be confidentiality and integrity protected in accordance with industry best practices.

Motivation: By using industry proven approaches, the costs for implementing security, reliability and availability can be reduced and the attack vectors can be limited.
Req 49 The hosting provider/SaaS provider must ensure, that only defined log files, which only contain data regarding accounts and/or the service usage by end customers in the role as business partner, will be securely transmitted request within 24 hours.

Data can be transferred on the basis of encrypted e-mail communication, for example.

*Motivation: In the event of an operational disruption, it must be possible to respond at short notice to maintain operation.*

Req 50 The hosting provider/SaaS provider must protect log files against loss and not approved access in an adequate and tamper-proof manner.

*Motivation: As log files generally allow conclusions to be drawn as to defensive measures, safeguarding of log files is imperative to avoid pointing out additional attack vectors to hackers.*

Req 51 Fraud prevention must be implemented as appropriate based on industry best practices.

*Motivation: By using industry proven approaches, the costs for implementing security, reliability and availability can be reduced and the attack vectors can be limited.*
3.11 Third Parties

Introduction

This security requirement has been prepared based on the provisions of the Group Policy on IT/NT Security. The security requirement is used as a basis for an approval in the PSA process, among other things. It also serves as an implementation standard for provisions of the Group Policy on IT/NT Security in units which do not participate in the PSA process. These requirements shall be taken into account from the very beginning, including during the planning and decision-making processes. When implementing these security requirements, the precedence of national, international and supranational law shall be observed. This document is only applicable if the contractual (frame-)agreement with the 3rd party does not contain the standardised security annex.

Organizational requirements

Req 1 In the event that services are provided by third parties, internal customers must specify and manage the necessary data protection and security requirements for the IT/NT systems by entering into contractual agreements with the third-party providers.

The internal customer is the single point of contact for security and therefore accountable for the management of the third party in this context.

Motivation: Entering into a binding contractual agreement is the only way to ensure that third-party providers fulfill the data protection and security requirements.

Req 2 Every third-party provider must have a technical and organizational identity management process in place, which is used for administration of the identities of users who have access to systems.

The identity management process must be introduced and documented transparently. The third-party provider must be able to provide information about this process on request. Assignment of identities must follow the authorization concept.

Motivation: An appropriate identity management process reduces the risk of deniability of actions or misuse.

Req 3 Every third-party provider must be in a position to provide detailed information (including authorizations) about the assigned users at any time.

Motivation: This makes it possible to verify and monitor appropriate application of the need-to-know and need-to-do principles.

Req 4 The third-party provider must implement a state-of-the-art security framework and present this on request.

The term framework covers at least the following blocks:

• Definition of security processes and
3.11 Third Parties

- Definition of security requirements

Motivation: The implementation of a security framework guarantees a structured and transparent approach with regard to security matters.

---

Req 5 Every third-party provider must provide its employees with appropriate regular training in conjunction with the company’s security framework.

Third-party providers must ensure that authorized users who are granted access to IT systems and IT equipment in conjunction with the commissioning of a specific order are given appropriate training to enable them to use these facilities efficiently. 

Motivation: Training in security awareness is a further prerequisite for successful implementation of the security framework.

---

Req 6 Third-party providers must name a security officer to act as contact in all matters relating to information security.

Motivation: A central contact and clearly defined responsibilities are prerequisites for prompt and efficient problem resolution if and when an incident occurs.

---

Req 7 Every third-party provider must ensure that security incidents are reported to the security officer. This applies to all incidents that affect the service provided.

---

Req 9 If third-party providers plan to contract out services or parts of services to a subcontractor, they must obtain prior written consent.

---

Req 10 Every third-party provider must ensure that all security requirements which apply to the services it provides are also valid for any subcontractors.

---

Req 11 The third-party provider must issue and comply with a physical security policy.

The policy must govern aspects such as building security, perimeter security and physical access control. In particular, client equipment which a third party provider uses to connect must provide the Company with suitable protection through access controls.

Motivation: Third-party providers shall control threats relating to physical access with an appropriate policy.

---

Req 12 The third-party provider must not store confidential data on local systems without additional protection mechanisms.

The service model may sometimes make it necessary to store data locally. In such cases, third-party providers must encrypt the information, or use another method to protect it suitably from unauthorized access or misuse. Third-party providers must draw up a data protection and security concept.
3.12 Proxy Server

Introduction
This security requirement has been prepared based on the provisions of the Group Policy on IT/NT Security.
The security requirement is used as a basis for an approval in the PSA process, among other things. It also serves as an implementation standard for provisions of the Group Policy on IT/NT Security in units which do not participate in the PSA process. These requirements shall be taken into account from the very beginning, including during the planning and decision-making processes. When implementing these security requirements, the precedence of national, international and supranational law shall be observed.

Definition of terms and application area
A proxy server is a communication interface in a network, which is itself capable of handling and exerting influence on communications. Proxy servers normally operate at a protocol level between OSI layers 5 and 7. These include, for example, HTTP and FTP. Proxy servers act as servers for clients and as clients for servers. Many proxy servers also act as caches. Identical queries by different clients on a server result in just one single query on the server. The proxy server responds to all subsequent instances of the query from the cache.
A basic difference is made between outbound and inbound (reverse) proxy servers. Inbound and outbound proxy servers do not vary in the way they operate but in their function. An outbound proxy server enables a (relatively) small, defined number of clients to access a large (not usually defined) number of servers. In contrast, an inbound proxy server enables a large (not usually defined) number of clients to access a few, precisely defined servers.
In a network, an outbound proxy server will normally be found “close to” the client, an inbound proxy server “close to” one or more servers.
A typical example of an outbound proxy server is a proxy server in a corporate intranet via which workstation computers connect to the Internet.
An inbound proxy server can be connected, for example, upstream of an Internet portal. It manages accesses to the portal and if necessary can even serve them from a cache to relieve the portal. With the appropriate configuration, an inbound proxy server can also assume the role of a web application firewall.
A proxy server can be implemented as a service on a server computer (Linux, Windows, or similar) or as an appliance. This is a combination of special hardware and corresponding firmware (operating system).
In terms of security, special aspects must be taken into account for each of the above described versions of proxy servers. They are presented in this document.
This document does not apply in particular to proxy servers that are used to provide Internet access via a customer’s mobile equipment.
Nor does this document apply to proxy servers that are used purely for routing purposes on the basis of a protocol above OSI layer 4. In particular, the text does not deal with SOCKS proxy servers.
A proxy server is not a substitute for network security components such as a network firewall (OSI layer 3). Such components shall be considered in addition to use of a proxy server.
General requirements for proxy servers

System hardening

Req 1: Unused services and protocols must be deactivated.

After installation of systems and software products typically local or remote reachable services and protocols are active, which are not necessarily needed for operation and functionality of the system. Such services and protocols must be completely disabled on the system. Additionally it is important that a deactivation survives a system reboot.

This kind of system hardening must be done before the system is reachable from the network. Otherwise an attacker has the possibility to attack and maybe compromise the unsecured system.

Motivation: Services and protocols that are not required for system operation increase the potential attack surface and thus the risk of the system being compromised. This risk is further increased by the fact that a security inspection and an appropriate optimisation of the configuration for unused services and protocols will not be done.

For this requirement the following threats are relevant:

- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized use of services or resources
- Disruption of availability
- Attacks motivated and facilitated by information disclosure or visible security weaknesses

Req 2: The reachability of services must be restricted.

Typically services that are enabled in the basic configuration are accessible over all interfaces of the system and can be reached from systems in connected networks. This availability is often not needed or meaningful for system functioning. For this reason, services should only be enabled on interfaces where their usage is required. On interfaces where services are active, the reachability must be limited to legitimate communication peers. This limitation must be realized on the system itself (without measures (e.g. firewall) at network side).

Motivation: Disabling services on interfaces which do not require system accessibility or by limiting the reachability can greatly reduce the potential vulnerabilities offered to an attacker. For example, access to a system via SSH from the Internet is not necessary. If this service could be accessed from the interface connected to the Internet, this would greatly increase the risk of attacks on the service.

For this requirement the following threats are relevant:

- Unauthorized access to the system
- Unauthorized use of services or resources
- Disruption of availability
- Attacks motivated and facilitated by information disclosure or visible security weaknesses

Req 3: Access to a proxy server’s administrative interfaces from the Internet must be prevented.

A proxy server’s administrative interfaces, e.g., an SSH port or web interface, must not be accessible from the Internet.

Motivation: All the configuration settings of a proxy server can be changed via the proxy server’s administrative interfaces. Making a change to a configuration could enable attackers to gain access to other systems, to override security regulations or even gain a direct insight
Proxy Server

3.12 Proxy Server

into data traffic passing over the proxy server. It is therefore vital to do everything possible to restrict access to administrative interfaces.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Disruption of availability

**Req 4** Unused software must not be installed or must be uninstalled.

During installation of a system often software components will be installed or parts of software will be activated which are not needed for the operation or functionality of the system. This includes also parts of a software, which will be in-stalled as examples but typically not be used (e.g. default web pages, example databases, test data). Such components should not be installed or must be deleted after installation.

Motivation: Vulnerabilities in software of a system offer an attack window for attackers to infiltrate the system. Uninstalling components that are not required can therefore reduce the possibility of a successful compromise of the system.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Disruption of availability

**Req 5** Unused functions of the operated software and hardware must be deactivated.

During installation of software and hardware often functions will be activated that are not necessarily needed for operation or function of the system. Functions of software are currently inherent part which could not be deleted or de-installed individually. Such functions must be deactivated in the configuration of the system permanently.

Beside the functions of the software also hardware functions are active which are not necessary for a system. Functions like unused interfaces must permanently deactivated. Permanent means that they must not be reactivated again after system reboot.

Motivation: The hardware or software of a system often contains functions which are not used and so will be a risk for system security. Such functions give an attacker the possibility to manipulate the system. Furthermore it is possible to get unauthorized access other areas or data of the system. An example is a debugging function in software which can be used for troubleshooting but must not be activated during normal operation. Or a hardware interface that will not be used and so is unsecured an allows possibly unauthorized access to the system.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Disruption of availability

**System update**

**Req 6** Software and hardware components that are no longer supported by vendor, producer or developer must not be used.

Components that have reached end-of-life or end-of-support must not be used. Excluded are components that have a special support contract. This contract must guarantee the correction of vulnerabilities over components life-time.

Motivation: Hardware and software components that have reached end of life or end of support represent a risk for a system. This means that a vendor does not supply remedial updates or patches for a component should errors or vulnerabilities occur. This means that vulnerabilities
3.12 Proxy Server

cannot be fixed when they occur and could be exploited to compromise the system or to impair its availability.

For this requirement the following threats are relevant:
• Unauthorized access to the system
• Unauthorized access or tapping of data
• Unauthorized modification of data
• Disruption of availability

---

**Req 7** Known vulnerabilities in software and hardware of the system must be fixed or protected.

Prior to installation of a software or hardware component, users must check whether any vulnerability has been discovered and published for the version they are installing. Any component that proves to have a vulnerability must not be installed or used. Excepted from this rule are components for which the vendor has already provided a measure to remedy the vulnerability, e.g. a patch, update or workaround. In this case, the additional measure must be implemented on the system. Furthermore it is an ongoing process during the complete life cycle of the system to fix upcoming vulnerabilities promptly.

**Motivation:** Publication of vulnerabilities increases the risk of successful exploitation by an attacker, especially since the published information usually includes details on how to exploit the vulnerability and tools that make exploitation possible. One example of this is when Web servers are compromised due to not patched vulnerabilities.

For this requirement the following threats are relevant:
• Unauthorized access to the system
• Unauthorized access or tapping of data
• Unauthorized modification of data
• Disruption of availability

---

**Protecting data and information**

**Req 8** Data with need of protection must be secured against unauthorized viewing and manipulation during transmission and storage.

Adequate security measures for transmission and storage must be implemented of data with a need for protection that are classified as internal, confidential or strictly confidential. The chosen measure depends on the classification for the data and other factors such as the type of network used during transmission, the storage location for data, etc. Furthermore must be guaranteed that confidential and strictly confidential data will not be unprotected during temporary storage (e.g. in web cache, temporary folders).

All authentication data such as user names, passwords, PINs, etc. must be protected against unauthorized viewing and manipulation. This applies equally to permanent storage and transmission. Typical measures taken to protect authentication data or combinations of these are:

For storage:
• Client systems: encryption or obfuscation of authentication data, no persistent storage, limitation of access rights.
• Server systems: hashing of authentication data with PBKDF like scrypt or bcrypt or when this is not possible hashes with salt.
3.12 Proxy Server

For transmission:

- Usage of cryptographically protected network protocols.
- Challenge response method (Disadvantage of this solution is that passwords needed in
clear-text on server. If this solution is feasible depends on the individual threat scenario)

Files of a system that are needed for the functionality must also be protected against
manipulation. This is necessary because system’s integrity can be damaged when the system
access this kind of files. An example is the use of check-sum or cryptographic methods to
validate if e.g. firmware images, patches, drivers or kernel modules are free of ma-nipulations.

For transmission of data with a need of protection network protocols that are insecure due to
insufficient security measures shall not be used. Examples are: SSLv3, SSHv1, FTP, Telnet,
SNMPv1 and 2c. In case of these protocols a newer version without vulnerabilities or a secure
alternative must be used.

**Motivation:** If data with a need of protection will not be secured an attacker could record or
manipulate the data during transmission over a network. An example is the recording of user
names and passwords during system administration with the telnet clear-text protocol. Storing
data on a system without adequate protection may mean that unauthorized users can copy or
modify it. One example is when passwords can be read out when they stored in an inadequate
se-cured way (e.g. usage of unsecure hashing algorithms like MD5 or SHA-1) or even without
encryption on a system or manipulation of firmware to affected the system integrity.

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data
- Unauthorized modification of data

---

**Req 9** HTTP proxy servers must comply with the caching behavior required in a request
or response.

An HTTP proxy server must not be configured in such a way that data is stored longer than
specified in the request or response. It must not, in particular, be stored with an existing
“cache-control: no-store” HTTP header.

**Motivation:** Sensitive data can be protected from undesired storage betting a cache control
header. It is therefore im-portant that all components involved in data exchange observe the
specified caching behavior.

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data

---

**Req 10** Information with need of protection must not be contained in files, outputs or
messages that are ac-cessible by unauthorized users.

Information with need of protection must not be accessible in files, outputs or messages of the
system by unauthorized users. This includes information relating to the operating system, used
middleware or applications such as vendor, product name, product identifier, installed software
versions, installed service packs, patches, hot fixes and serial num-bers. Examples for system
messages which must be free of sensitive data are:
- Comments in downloadable files
- Error and system messages
- Stack traces
- Network protocols
- Login windows and dialogs
Furthermore, details of implementation and information relating, e.g., to backend software/systems, function calls, SQL instructions or structure of database, must not be contained in error messages. Excluded from this are displays and outputs that can be viewed and retrieved by authorized users who are logged in. In addition, an internal transfer of system internal information for error analysis is allowed in an adequate dimension. In this case the continuative regulations or guidelines (e.g. of data privacy) must be noticed.

Motivation: The information named above can be used by an attacker to prepare specific attacks on a system. In this way an attacker could, for example, use the precise software version to identify vulnerabilities in the product and, in a second step, exploit them.

For this requirement the following threats are relevant:

- Unauthorized access or tapping of data
- Attacks motivated and facilitated by information disclosure or visible security weaknesses

Protecting availability and integrity

Req 11 The system must be robust against overload situations.

A system must provide security measures to deal with overload situations. In particular, partial or complete impairment of system availability must be avoided. Potential protective measures include:

- Restricting of available RAM per application
- Restricting of maximum sessions for a Web application
- Defining the maximum size of a dataset
- Restricting CPU resources per process
- Prioritizing processes
- Limiting of amount or size of transactions of an user or from an IP address in a specific time range

Motivation: An attacker can effect systems availability through targeted exploitation of vulnerabilities in resources ad-ministration. An attacker can, for example, send bulk queries to a Web server and, by doing so, compromise its avail-ability if the maximum number of permissible Web sessions is not restricted.

For this requirement the following threats are relevant:

- Disruption of availability

Req 12 If an overload situation cannot be prevented, the system must act in a predictable way.

A system must be built in this way that it can react on a overload situation in a controlled way. However it is possible that a situation happens where the security measures are not longer sufficient.

In such case it must be ensured that the system cannot reach an undefined and thus potentially insecure state. In an extreme case this means that a controlled system shutdown is preferable to uncontrolled failure of the security func-tions and thus loss of system protection.

Motivation: With denial-of-service attacks can an attacker try to overload a system to effect its availability or integrity. Unpredictable acting of the system is then a risk for functionality and data and possibly also for other systems.
For this requirement the following threats are relevant:
• Disruption of availability

### 3.12 Proxy Server

**Req 13** The system must be robust against unexpected input.

During transmission of data to a system it is necessary to validate this before processing. This includes all data which are send to the system. Examples for this are user input, values in arrays and content in protocols. The following typical implementation mistakes must not be done:

- No validation on the lengths of transferred data
- Incorrect assumptions about data formats
- No validation that received data complies with the specification
- Insufficient handling of protocol errors in received data
- Insufficient restriction on recursion when parsing complex data formats
- White listing or escaping for inputs outside the values margin

**Motivation:** An attacker can try to put a system in an unsecure state through targeted manipulation of transmitted data. The object of such an attack is to compromise the usability, availability or integrity of individual services or of the entire system. For instance a unclean memory handling can lead to a buffer overflow that allows an attacker to execute arbitrary code on the effected system.

For this requirement the following threats are relevant:
• Unauthorized access to the system
• Unauthorized use of services or resources
• Disruption of availability

### Authentication and authorization

**Req 14** System functions with a need of protection must not be used or accessed without successful authentication and authorization.

The usage of a system functions or access of data classified as internal, confidential or strictly confidential must only be possible unambiguous user identification and successful authentication on basis of the user name and at least one authentication attribute. Excepcted from this are functions for public use such as those for a Web server on the Internet, via which information is made available to the public. Examples for functions which require a prior authentication are network services (like SSH, SFTP, Web services), local access via a management console, local usage of operating systems and applications. The following examples are possibilities that could be used for authentication.

- Query user name and password
- Use of cryptographic keys and certificates (e.g. as Smartcard)

This requirement must also be applied to accounts that are only used for communication between systems (M2M).

**Motivation:** The authentication is necessary to doubtless identify a user because the allocated authorization, and therefore the access on data and services of the system depends on that.

For this requirement the following threats are relevant:
• Unauthorized access to the system
• Unauthorized use of services or resources
• Denial of executed activities
3.12 Proxy Server

Req 15 Accounts must be used that allow unambiguous identification of the user.

Users must be identified unambiguously by the system. This can typically be reached by using a unique account per user. So named group accounts, i.e. the use of one account for several persons, must not be used. On exception of this requirement are so named machine accounts. These will be used for authentication and authorization from system to each other or for applications on a system and can’t be assigned to a single person. Such accounts must be assigned on a per system or per application basis. In this connection, it has to be guaranteed that this account can’t be misused. Possibilities to protect these accounts are:

- Configuring of a Password that fulfils the security requirements and is known by less than possible circle of administrators.
- Configuring the account that only a local use is possible and a interactive login isn’t possible.
- Use of a technique for authentication of the specific account with public and private key or certificates.
- Limiting the access over the network for legitimised systems.

Additional solution must be checked on their usability per individual case.

Motivation: Unambiguous user identification is a prerequisite for assigning a user the rights that he requires to perform his tasks on the system. This is the only way to adequately control access to system data and services and to prevent misuse. Furthermore, it makes it possible to log activities and actions on a system and to assign them to individual users.

For this requirement the following threats are relevant:

- Unauthorized access to the system
- Unauthorized use of services or resources
- Denial of executed activities

Req 16 Accounts must be protected against unauthorized use by at least one authentication attribute.

The various user and machine accounts on a system must be protected from misuse. To this end, an authentication attribute is typically used, which, when combined with the user name, enables unambiguous authentication and identification of the authorized user. Authentication attributes include:

- Cryptographic keys
- Token
- Passwords
- PINs

This means that authentication based on a parameter that can be spoofed (e.g. phone numbers, public IP addresses or VPN membership) is not permitted. Exceptions are attributes that cannot be faked or spoofed by an attacker. Several of the above options can be combined (dual-factor authentication) to achieve a higher level of security. Whether or not this is suitable and necessary depends on the protection needs of the individual system and its data and must be evaluated for individual cases.

Motivation: Accounts that are not protected with a secret authentication attribute can be used by an attacker to gain unauthorized access to a system and the data and applications stored on it.

For this requirement the following threats are relevant:

- Unauthorized access to the system
- Unauthorized use of services or resources
- Denial of executed activities
Accounts with extensive rights must be protected with two authentication attributes.

Accounts, for example used for administration, maintenance and troubleshooting, have extensive rights. Therefore a single protection (e.g. a password), as for normal user accounts with less rights, is not suitable. To get a higher protection level it is necessary to use more than one authentication attribute. For this a combination of an attribute that the user knows and an attribute that the user owns will be used often. This kind of authentication will be named as 2-factor authentication. Examples for 2-factor authentication are:

- Smartcard with PIN
- Private key with Passphrase
- Secure-ID Token with Password

Motivation: Accounts with extensive rights as used for system administration have a higher risk for system’s security. An attacker can get extensive rights by compromising such an account to get access to wide parts of the system and stored data.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized use of services or resources
- Denial of executed activities

Predefined accounts must be deleted or disabled.

Many systems have default accounts (e.g. guest, ctxsys), some of which are preconfigured with or without known pass-words. These standard users must be deleted or disabled. Should this measure not be possible the accounts must be locked for remote login. In any case disabled or locked accounts must configured with a complex password (12 character and more, use of upper/lower case, numbers and special characters). This is necessary to prevent unauthorised use of such a account in case of misconfiguration.

Exceptions to this requirement to delete or disable accounts are accounts that are used only internal on the system in-volved and that are required for one or more applications on the system to function. Also for this accounts remote access or local login must be forbidden to prevent a abusive use by users of the system.

Motivation: Standard users are typically generally known and can be used by an attacker for targeted brute force and dictionary attacks. Standard user accounts represent a special risk if they do not use a password or only use a standard password that is generally known. Such standard user accounts can easily be exploited by an attacker in order to gain access to the system involved without being authorized to do so.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized use of services or resources
- Denial of executed activities

Predefined authentication attributes must be deleted or disabled.

Normally, authentication attributes such as password or cryptographic keys will be preconfigured from producer, vendor or developer of a system. Such authentication attributes must be changed an own attribute not known by other parties.

Motivation: Authentication attributes like password or cryptographic keys preconfigured from third parties are not trustable. Such authentication attributes can be used to compromise systems or their data.
For this requirement the following threats are relevant:

- Unauthorized access to the system
- Unauthorized use of services or resources
- Denial of executed activities

Req 20 The authorizations for accounts and applications must be reduced to the minimum required for the tasks they have to perform.

Authorizations to a system must be restricted to a level in which a user can only access data and use functions that he needs in the course of his work. Suitable authorizations must also be assigned for access to files that are components of the operating system or of applications or that are generated by the same (e.g. configuration and logging files).

Alongside access to data, execution of applications and components must also take place with rights that are as low as possible. Applications should not be executed with administrator or system rights.

Motivation: If the rights granted to a user on a system are too broad, it could be possible for him to access data and applications which he is not permitted to view or use. This would give him the opportunity to disclose or modify confidential data and to manipulate system files. Applications with rights that are too broad can be used by a user to extend his own authorizations and thus to gain access to files and system components to which he would not have had access with his authorizations under normal circumstances.

For this requirement the following threats are relevant:

- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources

Req 21 The system must be connected to a central system for user administration.

Accounts and their rights will be administrated on central identity management systems. For provisioning of these authorization information must the system provide a central interface (e.g. LDAPs for authorization, Kerberos for authentication, locking information for certificates) or decentralized mechanisms (e.g. public-key authentication) . A central solution for identity management must be preferred.

In areas where a central identity management system is not available a central system such as LDAP, TACACS+ or RADIUS server for the administration of accounts and their authentication and authorization must be used.

Motivation: Central administration of identity of accounts and their rights means that they only have to be maintained once instead of separately on each system. From the aspect of security, the advantage is that an account and its rights only known on a single central side. This information can be transmitted from a central side to systems (provisioning), central administrated (reconciliation) and central deleted (deprovisioning). This reduces the risk of accounts being forgotten during changing or deletion since they are configured on multiple systems. This could give a user wrong system rights or continued access to a system.

For this requirement the following threats are relevant:

- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources
Protecting sessions

Req 22 Sessions must be protected against high jacking.

A function must be implemented for all user sessions to prevent other users assuming or pursuing control of the sessions of an authorized user. Such protection can be achieved by implementing the following measures or a combination of these measures:

- Using an appropriate cryptographic algorithm
- On network level: Use of TCP protocol (with sequence number) and filter lists.
- On transport level: SSL/TLS
- On application level: Negotiation of a random, secret value between sender and recipient (e.g. session ID, sequence number, time stamp)

Motivation: An attacker who gains knowledge about a session which is not protected against assumption of control can continue to participate in such a session and thus gain unauthorized access to the system involved. One example is an attacker who records the session cookie for a Web application during non-encrypted communication. The attacker can then view and take control of the session of the original user.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources
- Denial of executed activities

Req 23 It must be possible that users can logoff their session.

The system must have a function that allows a signed in user to logout at any time.

Motivation: A user of a system must have the possibility to protect a session and therefore its data against unauthorized access. Therefor a logout function must be available that can be used to end a session. This prevents that the current session will stay open and can be reactivated and used by an unauthorized person.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources
- Denial of executed activities

Req 24 Session must be terminated after an adjusted period of inactivity for the purpose of use.

It is necessary that a session will be terminated automatically after a specified time of inactivity. For this reason a time-out must be set for sessions. The selected period depends on use and if applicable the physical environment. This means for example that a time-out for an application in an unsecure environment must be shorter (less minutes) than the time out for an application used of operational staff for monitoring tasks used in an protected area (60 minutes and longer).

Motivation: In case of an open or unused session exists the possibility that an unauthorized user can high-jack and use this session to get access to the effected system and its data. Furthermore open session will allocate system resources.
For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources
- Denial of executed activities

**Authentication parameter password**

Req 25 If a password is used as an authentication attribute, it must have at least 8 characters and contain three of the following categories: upper cases, lower case, numbers and special characters.

A system may only accept passwords that comply with the following complexity:
- Minimum length of 8 characters.
- Comprising at least three of the following categories: upper/lower case letters, numbers and special characters

When a password is assigned, the system must ensure that the password meets these requirements. If a central system will be used for user authentication this function can be forwarded or delegated to this system.

Motivation: Passwords with the above complexity offer high robustness against attacks coupled with acceptable user friendliness. Passwords with this level of complexity have proved their efficiency in practice. Trivial passwords that are too short are susceptible to brute force and dictionary attacks and are therefore easy for attackers to determine. Once a password has been ascertained it can be used by an attacker for unauthorized access to the system and the data on it.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized use of services or resources
- Denial of executed activities
3.12 Proxy Server

Req 26 If a password is used as an authentication attribute, users must be enabled to independently change the password anytime.

The system must offer a function that enables a user to change his password at any time. When an external centralized system for user authentication will be used it is possible to redirect or implement this function on this system.

Motivation: The fact that a user can change his authentication attribute himself at any time enables him to change it promptly if he suspects that it could have been accessed by a third party.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized use of services or resources
- Denial of executed activities

Req 27 If a password is used as an authentication attribute, a protection against brute force and dictionary attacks that hinder password guessing must be implemented.

Brute force and dictionary attacks aim to use automated guessing to ascertain passwords for user and machine accounts. Various measures or a combination of this measures can be taken to prevent this.

The most commonly used protection measures are:
- Increasing the delay (e.g. doubling wait times at each attempt) for each renewed password input following an incorrect entry (“tar pit”).
- Blocking an account following a specified number of incorrect attempts (typically 5). However has to take in account that this solution needs a process for unlocking and an attacker can force this to deactivate accounts and make them unusable.
- Using CAPTCHA to prevent automated attempts (often used for Web applications).

In order to achieve higher security, it is often meaningful to combine two or more of the measures named here. This must be evaluated in individual cases and implemented accordingly.

Motivation: Implementation of one or more of the above measures can prevent successful execution of brute force and dictionary attacks to ascertain passwords.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized use of services or resources
- Denial of executed activities

Req 28 If a password is used as an authentication attribute, they must be hidden when displayed on screen.

When a user enters a password or the password is otherwise displayed on screen, it must be made unreadable.Typically, the individual characters of the password are replaced by a character such as “*”. Under certain circumstances it may be permissible for an individual character to be displayed briefly during input. Such a function is used, for example, on Smartphone’s to make input easier. However, the entire password is never output to the display in plaintext.
Motivation: To prevent another person reading a password by chance or intentionally on screen during input, the pass-word must be rendered unreadable during input or display.

For this requirement the following threats are relevant:

- Unauthorized access to the system
- Unauthorized use of services or resources
- Denial of executed activities
Logging

Req 29 Security relevant events must be logged with a precise timestamp and a unique system reference.

Systems must log the occurrence of security-relevant incidents. So that these events can be evaluated and classified, they must be logged together with a unique system reference (e.g., host name, IP or MAC address) and the exact time the incident occurred. Furthermore, legal and data privacy regulations (e.g. time of storage of logging data) must be proved and followed. The following table lists events that are relevant to security and the corresponding data that typically has to be logged by a system. Exceptions are systems for which no or only restricted logging applies. Examples of such systems are customer devices such as Smartphone’s or IADs/Homegateways (e.g. Speedport).

Logging must be done considering the currently valid legal, wage and company regulations. This regulations state among others that logging of events can be done only earmarked. Logging of events for doing a work control of employees is not allowed.

Typical event that reasonable should be logged in many cases are:

<table>
<thead>
<tr>
<th>Event</th>
<th>Event data to be logged</th>
</tr>
</thead>
<tbody>
<tr>
<td>Incorrect login attempts</td>
<td>• Account,</td>
</tr>
<tr>
<td></td>
<td>• No. of failed attempts,</td>
</tr>
<tr>
<td></td>
<td>• Source (IP address) of remote access</td>
</tr>
<tr>
<td>System access with accounts with administrator rights</td>
<td>• Account,</td>
</tr>
<tr>
<td></td>
<td>• Access timestamp,</td>
</tr>
<tr>
<td></td>
<td>• Length of session,</td>
</tr>
<tr>
<td></td>
<td>• Source (IP address) of remote access</td>
</tr>
<tr>
<td>Account administration</td>
<td>• Administrator account,</td>
</tr>
<tr>
<td></td>
<td>• Administered account,</td>
</tr>
<tr>
<td></td>
<td>• Activity performed (configure, delete, enable and disable)</td>
</tr>
<tr>
<td>Change of group membership for accounts</td>
<td>• Administrator account,</td>
</tr>
<tr>
<td></td>
<td>• Administered account,</td>
</tr>
<tr>
<td></td>
<td>• Activity performed (group added or removed)</td>
</tr>
<tr>
<td>Critical rise in system values such as disk space, CPU load over a</td>
<td>• Value exceeded,</td>
</tr>
<tr>
<td>longer period</td>
<td>• Value reached</td>
</tr>
<tr>
<td></td>
<td>(Here suitable threshold values must be defined depending on the</td>
</tr>
<tr>
<td></td>
<td>individual system.)</td>
</tr>
</tbody>
</table>

Logging of additional security-relevant events may be meaningful. This must be verified in individual cases and implemented accordingly where required.

Motivation: Logging security-relevant events is a basic requirement for detecting ongoing attacks as well as attacks that have already occurred. This is the only way in which suitable measures can be taken to maintain or restore system security. Furthermore, the logging data is used as evidence so that legal steps can be taken against attackers.

For this requirement the following threats are relevant:
• Denial of executed activities
• Unnoticeable feasible attacks
3.12 Proxy Server

Req 30 Security relevant logging data must be send to an external system direct after their creation.

Logging data must be forwarded to an external system in appropriate logging files as well as being stored locally. Standard protocols like Syslog, SNMPv3 must be preferred.

Motivation: If logging data is only stored locally it can be manipulated by an attacker who succeeds in compromising the system in order to conceal his attack and any manipulation he has performed on the system. This is the reason why the forwarding must be done immediately after the event occurred.

For this requirement the following threats are relevant:
- Denial of executed activities
- Unnoticeable feasible attacks

(Outbound) proxy servers

Req 31 Connections over outbound proxy servers must be authenticated and authorized.

The identity of an user or a machine must be determined beyond doubt. In this case, authentication of a machine may take place via the network address (IP), a directory service or a certificate.

In addition, a check must be carried out to establish whether the user or the machine is authorized to access the target resource.

Authentication and authorization may be switched off for a very limited and precisely defined target area (e.g., the corporate intranet) after consulting with and gaining approval from the data protection and/or security management unit responsible in each case.

Motivation: The authentication and authorization of connections greatly limits the probability of proxy server misuse. If a security incident occurs in which a connection is made via the proxy server, the authentication information as logged can be used to trace the offender.

For this requirement the following threats are relevant:
- Unauthorized use of services or resources
- Denial of executed activities

Req 32 A procedure must be configured on outbound proxy servers which can be used to block access to undesired target addresses or undesired content (filtering).

Filtering must be coordinated with the data protection and/or security management unit responsible in each case. An external (i.e., non-inhouse) service can be used for filtering, which could classify the target addresses into categories.

Motivation: Not all content is suited for delivery to a client computer by the system. Attention must be paid not only to content that contains malicious program code that would harm the client or the company but also in particular to unlawful content.

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Disruption of availability
Req 33 In the case of M2M connections via outbound proxy servers, access to non-required URLs must be blocked.

For example, in order to perform updates, a Windows server system may require access to the Microsoft update servers via the proxy server. However, the server does not require other connections to carry out the updates. The proxy server must therefore prohibit them.

*Motivation: Limiting callable URLs reduces the risk of downloading malware via this connection and reduces the overall attack surface.*

For this requirement the following threats are relevant:

- Unauthorized use of services or resources

Req 34 It is recommended, that an outbound proxy server reviews objects downloaded from non-secure networks for viruses or malware.

Objects for which the proxy server has identified a virus or malware to clients must not be delivered to clients.

*Motivation: Proxy servers represent gateways to a corporate intranet. As far as possible, the system should intercept harmful objects (files) at this point. The scan on a proxy server shall be seen as an extension of other scans, e.g., on a workstation computer or file server, complying with the concept of a graded defense system.*
**Req 35** In some cases, connections are opened via outbound HTTP proxy servers using the HTTP CONNECT method. In such cases, the protocols which can be tunneled via the HTTP connection must be limited.

*Motivation:* The HTTP CONNECT method allows all protocols to be tunneled to a proxy server via an HTTP connection. This makes it generally possible to set up unwanted connections such as a VPN connection from the intranet to an external network. Furthermore, a tunneled connection can only be monitored and reviewed by a proxy server to a limited extent.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized use of services or resources

**Req 36** Outbound proxy servers must not disclose the client’s network address outside the intranet.

In the case of HTTP proxy servers, this involves in particular disclosure of the network address (IP address) in an HTTP header (e.g., “X-Forwarded-For” or “client-IP”).

*Motivation:* The target computer does not normally require the client’s network address. If the system discloses a client’s network address, it gives an attacker the chance to obtain valuable information about the intranet structure.

For this requirement the following threats are relevant:
- Attacks motivated and facilitated by information disclosure or visible security weaknesses

**Req 37** The proxy server must record information on each connection in a log file.

In particular, the proxy server must file the following information:
- Date and time
- Source IP address or unique computer name
- Target IP address or FQDN
- Protocol used

For HTTP connections, the proxy server must also file:
- Target URL
- HTTP method
- HTTP status code

*Note:* When implementing measures resulting from this requirement, the applicable participation rights of the responsible employee representatives/trade unions as well as the works and collective agreements shall be observed.

*Motivation:* Without the logging process it would not be possible to trace actions relevant to security (e.g., an attack on a web application via a proxy server) and to prevent these actions in the future.

For this requirement the following threats are relevant:
- Denial of executed activities
- Unnoticeable feasible attacks
Reverse proxy servers

Req 38 All applications that are addressed via a reverse proxy server must be assigned an individual network address (IP address) or an individual port on the proxy server.

If an application has several complete domain names (FQDN), these may also be linked to the same IP address (where appropriate using an alias).

*Motivation:* To prevent unauthorized access to an application (or part of an application), it shall be (logically) separated from other applications.

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources

Req 39 An inbound network address (IP address) and a port may only allow access to the relevant applications.

Rules must be explicitly configured on the proxy server that only permit access to the application to which a network address is assigned. It must be ensured that the proxy server cannot be misused as an open proxy.

*Motivation:* It shall be ensured that the proxy server routes inbound connections to the correct application. If this is not the case, an attacker could gain access to other applications via a proxy server.

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources

Req 40 A reverse HTTP proxy server must reject requests with non-required HTTP methods.

In particular, a reverse HTTP proxy server must not support the HTTP CONNECT method. Normally, only the GET and POST methods are required. It is recommended, that the proxy server responds to requests containing non-required methods with HTTP status code 405 (Method Not Allowed). It declares that the method of the query is not permitted for the desired target.

*Motivation:* The use of non-required HTTP methods can give an attacker unauthorized access to a downstream web or application server.

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources

Req 41 All inbound network connections must be terminated on the reverse proxy server.

A reverse proxy server must reject inbound connections with network protocols that cannot be terminated on the proxy server.

*Motivation:* Any connection that is not terminated on the proxy server enables direct access to a downstream system via the proxy server. Hence, the proxy server provides no more than a...
minimum security standard. A downstream system may also be situated in an area of the network that is not suitably protected, so that an attacker is offered various attack vectors.

For this requirement the following threats are relevant:
• Unauthorized access to the system

---

Req 42 Reverse proxy servers must reject requests that do not comply with appropriate protocol specifications.

HTTP requests must comply with RFC2616. If, for example, an incorrect content length header is sent, the proxy server must reject the request. It must also respond to the request using HTTP status code 403 without a detailed error description. FTP requests must satisfy RFC 959, other specific RFCs apply to other protocols.

Motivation: Any request that does not satisfy the RFC specification could indicate an attempted attack.

For this requirement the following threats are relevant:
• Unauthorized access or tapping of data
• Unauthorized modification of data
• Unauthorized use of services or resources

---

Req 43 It is recommended that reverse proxy servers validate the URL in inbound requests for compliance with a specified pattern.

The pattern depends on the application assigned via the reverse proxy server. As such, it is necessary to define this pattern individually. The proxy server should reject requests in which the URL does not comply with the pattern. The proxy server must in particular reject requests in which the URL (for the application) contains invalid characters.

Motivation: URLs that do not correspond to a pattern defined by a downstream application and expected by the application indicate an attack. They could cause unexpected behavior in the application or enable an attacker to gain unauthorized access to the application.

For this requirement the following threats are relevant:
• Unauthorized access or tapping of data
• Unauthorized modification of data
• Unauthorized use of services or resources
3.14 Architecture of systems

Introduction
This document has been prepared based on the provisions of the Group Policy on IT/NT Security.

The security requirement is used as a basis for an approval in the PSA process, among other things. It also serves as an implementation standard for provisions of the Group Policy on IT/NT Security in units which do not participate in the PSA process. These requirements shall be taken into account from the very beginning, including during the planning and decision-making processes. When implementing these security requirements, the precedence of national, international and supranational law shall be observed.

Segregation of systems
The rule for the interplay between systems is to not rely on single security measures. So the independence of network based protection is essential.

Req 1 Systems must be separated from each other appropriately in line with their protection requirements.

The systems' need of protection results from different factors like, e.g., processed and stored data, services exposed and used applications. Similarly, systems on which personal data is processed must be protected against unwanted access or data flows from the same network or other networks by using state-of-the-art measures (such as firewalls). If a system is configured in layers, these must be appropriately separated. A critical system (i.e., system or component with high protection requirements as regards confidentiality, availability, integrity or data protection) has to be separated from other systems, also from other critical systems, at least logically. Requirements demanding a physical separation (e.g., in the case of very high worthy of protection) have to be reviewed and implemented.

Examples of network segmentation: VLAN, VxLAN, private VLAN or otherwise securely configured network segregation

Motivation: It is more likely for a less protected system to be compromised. This must not result in other systems with higher protection requirements being more easily attacked by this compromised system in the network. Breaking down the network makes it possible to establish a higher security level for systems requiring special protection without all systems overall having to be protected at a high cost.

The risk of compromise caused by other systems through the use of shared networks and resources should be minim-ized, where possible.

Implementation example: A critical customer management system is separated by VLANs from other systems, e.g., a webserver delivering banner ads. In addition, the customer management system contains a huge database cluster that is implemented in an own network segment to isolate it from log and statistic servers belonging to the same overall customer management system.

For this requirement the following threats are relevant:
• Unauthorized access to the system
• Unauthorized access or tapping of data
• Disruption of availability
• Unnoticeable feasible attacks
• Attacks motivated and facilitated by information disclosure or visible security weaknesses
3.14 Architecture of systems

Req 2 Production systems must be completely separated from test and development systems.

Apart from servers, infrastructure elements such as the network and storage systems must be separated as well. A separation that is merely logical in parts is only permitted if it is not possible to bypass this logical separation and, on the other hand, it is ensured that there is no security impact on production systems when performing tests and/or development activities. A physical separation is advised.

If communication between these different system types is absolutely necessary, it can only be accomplished by employing security systems that control the entire communication. Apart from that jump hosts between these network areas should be used.

Testing activities related to security like penetration tests and network-based security scans must be possible at all times and without any impact on the production systems.

It is recommended to use clearly distinguishable IP address ranges for these different kinds of network areas.

Motivation: A sufficiently secure system status cannot be assumed in the case of test and development systems, in particular, because these are, by their nature, exposed to permanent changes. When linking system types to each other or when using the same platform, there is a risk that unauthorized parties will be able to access production systems and live data from within the test/development environment or that the stability of the production systems and thus the availability of the associated services will be put at risk. A test and development infrastructure that is completely independent from the production makes it possible to implement changes for test and development systems quickly and also during a frozen zone as they do not have an impact on the production systems. This way security updates for the production can faster be tested and accepted.

Implementation example: For an acceptance test environment, designated for security scans and penetration tests, a dedicated test environment has been chosen. Another example is a performance test environment that is physically separated from the production, so that performance test activities do not have any negative impact on the production environment as it could be possible in the case of a shared environment, e.g., shared storage or network components.

For this requirement the following threats are relevant:
• Unauthorized access to the system
• Unauthorized access or tapping of data
• Disruption of availability
• Attacks motivated and facilitated by information disclosure or visible security weaknesses

Req 3 Segregations must not be bypassed.

Systems and system components must be separated with an identical effect at all points via which communication is possible. A network-based separation of systems (e.g., through VLANs, private VLANs or other layer-2 techniques) must take place in all connected network areas (e.g., production network, administration network, storage/backup network) according to the same logic. The same applies to other network technologies (e.g., Fiber Channel (FC)). Please note that it must not be possible to communicate between systems or have one system compromise another one using the management network.
Motivation: A separation with the intention to prevent other systems from being attacked following a compromise only generates genuine added value if it is executed on all interfaces in the same way.

Implementation example: From a “customer perspective”, presentation and database layer are separated on the network side. However, the administrative management interfaces would be connected in the same network and communication among systems would be possible. If the presentation layer was compromised, an attacker could access the systems of the database layer on layer-2 by circumventing a possible firewall.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Disruption of availability
- Unnoticeable feasible attacks
- Attacks motivated and facilitated by information disclosure or visible security weaknesses

Communication

The classification of data and the protection requirements during transmission can be found in the Group Policies on “IT/NT Security” and “Information Security and Data Protection”.

Usually, confidentiality and integrity are two different objectives (integrity protection identifies changes to the data, encryption prevents unauthorized reading). Modern communication protocols provide both. Furthermore, an authentication of the communication partners is needed.

The following table gives an overview of the minimal requirements by encryption:

<table>
<thead>
<tr>
<th></th>
<th>[size]Local transmission owned data center across own systems, networks and lines</th>
<th>[size]Network</th>
<th>[size]Other, e.g., public network</th>
</tr>
</thead>
<tbody>
<tr>
<td>Strictly confidential</td>
<td>E2E-encrypted</td>
<td>E2E-encrypted</td>
<td>E2E-encrypted</td>
</tr>
<tr>
<td>Confidential</td>
<td>encrypted or non-encrypted*</td>
<td>encrypted</td>
<td>encrypted</td>
</tr>
<tr>
<td>Internal</td>
<td>non-encrypted</td>
<td>non-encrypted</td>
<td>encrypted</td>
</tr>
<tr>
<td>Open</td>
<td>non-encrypted</td>
<td>non-encrypted</td>
<td>non-encrypted</td>
</tr>
</tbody>
</table>

"E2E-encrypted": Encrypted and transmitted “end-to-end” on the application level *) non-encrypted only if protection is ensured by different, equivalent controls

Req 4 Network-based filter elements must be set up which ensure that only the necessary services of a system and its components are reachable.

Each system must be protected by a protection mechanism in the network, e.g., a firewall at layer-3 or above. Daemons might have started temporarily on a system which are not required in regular operation (e.g., through a system update) and which should not be available at all.

Motivation: An independent, network-based protection mechanism reduces the likelihood of the system becoming compromised.

Implementation example: Servers within a (sub-) network communicate with each other. However, incoming connections from outside this network are not required. The router or the firewall must prevent such communication attempts.
For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized use of services or resources
- Disruption of availability
- Attacks motivated and facilitated by information disclosure or visible security weaknesses

Req 5 Possible communication paths between systems must be reduced by filter elements to the minimum needed to fulfill the purpose of the systems.

The default policy must be: deny any any. In case of an communication activation, a suitable trade-off must be found between rules for single ip addresses and rules for ip ranges covering more than one system. The use of ranges avoids frequent changes of the ruleset.

To avoid activations for huge TCP/UDP port ranges, necessary services/protocols with dynamic port assignment should be investigated for alternatives, as far as a solution by using an appropriate firewall system with application-aware protocol support is not available. Otherwise, services might be accessible via the enabled ports which should actually be blocked.

Motivation: Communication channels provides remote access to systems. Less communication possibilities provides more security to the target systems.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized use of services or resources
Req 6 The initiation of a communication channel must follow the principle of "least privilege".

Initiation of a communication link with an user account, not system administrator. Login at the remote site also with un-privileged rights.

Motivation: Implementation of the principle least privilege.

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data
- Unauthorized modification of data

Req 7 If data with a need of protection is transmitted, the involved communication partners must be authenticated appropriately.

Information in need of protection is in this case all data classed as “internal” or higher with regard to the protection goal of confidentiality or data with an increased need on integrity. For "confidential" or higher classified data, an authentication based on IP addresses is normally not sufficient. Usage of certificates is the preferred solution.

Motivation: Information with need of protection must not end up in the wrong hands.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources

Req 8 Information classified as “strictly confidential” must be transmitted “end-to-end” encrypted on the application level.

Data classed this way may only be processed and viewed by the application itself so that end-to-end encryption including an appropriate authentication of the communication partners is required directly in the application (e.g., through XML encryption or PGP). This has the side effect that the content cannot be filtered by intermediate systems.

If SSL/TLS is used, it must be ensured that the communication link is established from end system to end system without interception by an intermediate system like a loadbalancer.

Motivation: Encryption ensures that the data cannot be manipulated or read on its way from sender to recipient. If this takes place at the application level, only the target system application intended for this will be able to further process this data.

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data

Req 9 Information classified as “confidential” must be transmitted in encrypted form.
Generally, confidential data must be transmitted encrypted. Encryption on the application level ("end-to-end") is to be preferred to encryption by the system.

If communication involves the interplay of such a trusted network with other networks, it is possible to use tunnel mechanisms with encryption (e.g., IPSec) to bridge non-trusted networks instead of protecting each connection individually. Such tunnels have to be considered as a part of the infrastructure (for further readings on this refer to the requirement about terminating IPSec tunnels).

Virtual network constructions such as VPNs or MPLS networks which work without encryption and only separate traffic from each other are generally not sufficient for the transmission of data classed as confidential. Additional devices can be used here which encrypt data at line level independently from any protocols that are transported over these lines.

Motivation: Encryption ensures that the data cannot be manipulated or read on its way from sender to recipient. Only authorized individuals are permitted to access this data.

For this requirement the following threats are relevant:
• Unauthorized access or tapping of data

---

Req 10 Content classified as "internal" or not classed at all must be encrypted when being transmitted over unsecure or public networks (e.g., Internet or wireless networks).

The confidentiality of internal information cannot be ensured in networks that can be viewed by third parties.

Motivation: Internal data must be protected against public viewing.

Implementation example: Usage of SSH, SSL/TLS and IPSec.

For this requirement the following threats are relevant:
• Unauthorized access or tapping of data
Layer model

Req 11 Systems which provide information in need of protection must be divided into at least two layers which are represented by different networks.

Information in need of protection is all data classed as “internal” or higher with regard to the protection goal of confidentiality or data with a high need on integrity or availability. A layer model mostly consists of a presentation layer with direct access by the users, a layer with application servers and a third layer comprising the databases. Where this is simplified, it is recommended that the application servers also run on the database servers. In some cases a loadbalancer can be considered as the presentation layer if the loadbalancer validates all input requests to the subsequent layers.

The layers correspond to individual networks, usually VLANs, which are connected through appropriate means, e.g., firewalls.

If access to data sources of multiple systems is required, this must take place from within an application layer.

Motivation: A layered model ensures that the user is never given direct access to layers with data that is to be protected. The aim is to render as few system components as possible visible to the user – internal structures must not be visible from the outside.

Implementation example: Databases are normally not addressed by users directly but via upstream application servers instead. This makes it possible to set up the databases in such a way that they are not directly accessible to the user.

For this requirement the following threats are relevant:
- Disruption of availability
- Attacks motivated and facilitated by information disclosure or visible security weaknesses

Req 12 The possible communication paths between the layers (networks) of a system must be reduced to the necessary connections through a filtering element.

This communication constraint has to take place via filter elements which have to be independent of the system components that have to be secured. Depending on the situation, firewalls, routers or loadbalancers with installed packet filters are suitable.

Motivation: The use of active filter elements between the networks offers additional, network-based protection for the systems.

Implementation example: If a system is divided into a presentation layer (e.g., web server) and another layer (e.g., databases) which are located in separate networks, communication from the web server to the database must only be possible via a firewall that is independent of these systems. This firewall is configured in such a way that it only allows for necessary connections to be established.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized use of services or resources

Req 13 Information in need of protection must not be stored permanently in the presentation layer.

The presentation layer can be accessed directly (internally or externally) making it the first target of attacks. This is why information with need of protection (i.e., the protection
3.14 Architecture of systems

requirement regarding the protection goal of confidentiality is “intern” or higher) must not be stored in this layer longer as absolutely needed.

Motivation: Storing this data in downstream systems reduces the risk of attack on this data.

Implementation example: When customers register at a web portal, the customer data to be protected must not be kept on the web server. It must be passed on to the next layer (e.g., an application server) for storing and further processing.

For this requirement the following threats are relevant:
• Unauthorized access or tapping of data
Req 14 If systems or system components are accessible from external networks, they must be physically separated from internal systems.

All components of the externally available presentation layer have to be physically separated from internal systems. Normally, this is already done through divided infrastructures.

Motivation: IT/NT systems that can be accessed from public networks, for example, are exposed to considerably higher risk of attacks than internal systems. All requests and data forwarded to downstream systems such as application servers and databases must be validated. On the basis of this requirement, the risk of a system being compromised downstream can be significantly minimized.

Implementation example: A reverse proxy terminates HTTPS connections in the presentation layer. The data packets whose content is validated there (possibly using re-encryption) are forwarded to the target systems.

For this requirement the following threats are relevant:
• Unauthorized access to the system
• Unnoticeable feasible attacks

Req 15 Connections from/to external systems must be secured by an appropriate application layer gateway such as a webserver or reverse proxy.

Incoming network connections, also SSL/TLS, are terminated at this point. Internal systems communicate only via this presentation layer to external systems; SSL/TLS is not intercepted by the outgoing proxy.

Motivation: First, internal network structures are hidden to outside systems, second, a proxy can filter the traffic and incoming content.

For this requirement the following threats are relevant:
• Unauthorized use of services or resources
• Attacks motivated and facilitated by information disclosure or visible security weaknesses
Availability

Req 16 On top of a operating system instance only those services and applications may be operated together when they have the same administrators, data owners and have similar, low, protection requirements.

Systems (services or applications) with different protection requirements are generally separated from each other. With this in mind, on top of a operating system instance only those services and applications may be operated

• which have the same low protection requirements and
• which are operated by the same team and
• whose data are subject to a common liability.

Motivation: Mixed operation of different protection needs results in higher risks for applications with higher needs of protection, otherwise too expensive measure are in place for systems with lower needs.

For this requirement the following threats are relevant:
• Unauthorized access or tapping of data
• Disruption of availability
• Attacks motivated and facilitated by information disclosure or visible security weaknesses

Req 17 If a system/system component is accessible from external networks, only one application service may run on the corresponding operating system instance.

For example, a web server or a mail server is allowed, but not both on the same operating system instance at the same time. Multiple services of the same kind (e.g., web server) are allowed as long as they have a common, low, need of protection. Special purpose loadbalancers are permitted to provide multiple services if their integrated security functions are considered as sufficient.

A mixture of internally and externally accessible services must be avoided in general.

Motivation: If the application / the system is being compromised, other services must not be involved.

Implementation example: Encapsulating of the application in a container oder small virtual machine. These virtual machines can run together with others on a common physical host.

For this requirement the following threats are relevant:
• Unauthorized access or tapping of data
• Disruption of availability

Req 18 Systems and components which are only accessed internally must use IP addresses which cannot be accessed from external networks.

This can be implemented in various ways:

• Use of special “private” or “unique local” IP addresses which are not routed on the Internet
• Use of corresponding routing and firewall rules when using other IP addresses

Motivation: This measure already ensures on the network level that systems are not addressable and thus not accessible by attackers.

For this requirement the following threats are relevant:
3.14 Architecture of systems

• Unauthorized access to the system
• Attacks motivated and facilitated by information disclosure or visible security weaknesses

---

Req 19 If the system provides UDP based services, measures must be in place to compensate the missing security features of the stateless transport protocol.

Measures must be in place
  • against processing spoofed IP packets as well as
  • against abuse as an amplifier in an UDP-based attack.

*Motivation: Defend Denial-of-Service attacks against the system and avoidance of amplification attacks based on the datagram character of UDP.*

Implementation example:
  • Use of a monitoring solution to early detect suspicious requests and initiate countermeasures.
  • Defense measures within the application, to adapt application behavior.
  • No processing of packets from spoofed IP sources by dropping such packets in the network.

For this requirement the following threats are relevant:
• Disruption of availability
• Unnoticeable feasible attacks
• Attacks motivated and facilitated by information disclosure or visible security weaknesses

---

Req 20 System and data must be recoverable after a system outage.

According to availability needs measures have to be taken to recover a system in case of an incident. If an automatic restart is required, the system must not rely on dependencies on itself.

*Motivation: The system and its data must be within its agreed availability.*

Implementation example: Classical Backup and Recovery strategy, Snapshots of file systems or whole virtual ma-chines, use of centrally managed storage with own backup, re-creation of systems according to templates and config-uration data in configuration database.

For this requirement the following threats are relevant:
• Disruption of availability
Administration

This always refers to a network based login in contrast to direct console or ILO access to the hardware.

Req 21 Administrative access (system administrator) must take place via a management network that is in-dependent from the actual production network.

Administrative access may only be offered on the interface to this management network. Independent physical inter-faces must be used if available or installable, due to the separate infrastructure of the network. These interfaces can still be accessible even if others are overloaded or disrupted. In individual cases, virtual or logical interfaces are suffi-cient. Naturally, for virtual systems only virtual interfaces can be provided.

The service that is used to login into the system (e.g., SSH or RDP) must only be linked to this interface that is intended for administrative activities. In case of systems which need administrative access on more than one interface, this has to be limited to the minimal number.

Motivation: Administrative access to systems generally takes place using high privileges. This separation is required in order to separate different kind of access and data flows from each other.

Implementation example: A webserver has one customer facing interface and one for internal backend communica-tion. For system administration another interface is needed to provide administrative access (and only there).

For this requirement the following threats are relevant:
• Unauthorized access to the system
• Attacks motivated and facilitated by information disclosure or visible security weaknesses

Req 22 Administrative access (system administrator) must take place only from secure workstations and systems.

Administrative access (system administrator) must not occur from workstations/systems which offer a higher risk of being compromised. Such risky systems
• are located in a standard office building without additional security measures,
• have direct access (also via a NAT-gateway) to the internet,
• use an internet browser or other client with access to content within the public internet,
• use an E-Mail client where it is possible that malicious attachments are executed,
• have no current patch level.

Besides dedicated, administrative workstations, jumphosts, terminal server or an Admin-VPN could be sufficient to avoid direct access from a potentially unsecure workstation to the production systems and to better control the traffic flows.

If a system is only externally manageable (e.g., via internet) the administrative access to the system and applications must be restricted to specific static IP addresses.

Motivation: Avoidance of compromising systems by malware or unauthorized access.

For this requirement the following threats are relevant:
• Unauthorized access to the system
• Denial of executed activities
• Unnoticeable feasible attacks
3.14 Architecture of systems

Req 23 The interactive administrative access to systems with high protection requirements must happen via a dedicated jumphost.

A jumphost provides a managed channel for access to systems. A jumphost is assigned to a small number of systems of one supplier and technology. Systems within the same application context which are operated by the same group of people, can use a common jumphost. Depending on the criticality of a system, jumphosts may be needed also for access from users.

A jumphost must intercept the connection to the target system only in the case when the device of the accessing people is untrusted or a logging of all activity is needed.

Motivation: Ensure technically that only a small group of authorized people have access to these systems.

For this requirement the following threats are relevant:
• Unauthorized access to the system
• Unnoticeable feasible attacks

Req 24 If a manipulation-resistant logging of privileged access is needed, it must take place on a special configured jumphost.

If manipulation-resistant logging of system administration tasks is required, a jumphost (hopping, terminal server) must be mandatory through technical restrictions for login into the target system. Users of the jumphost must not have administrative rights on this server. An appropriate logging has to be configured on the jumphost.

Motivation: This is the only way to ensure a reliable audit trail.

For this requirement the following threats are relevant:
• Denial of executed activities

Adherence to industry standards

Req 25 If a system processes or stores payment data of credit cards, the rules and requirements of PCI DSS must be fulfilled.

The PCI DSS (Payment Card Industry Data Security Standards) requirements and rules as well as other security standards of the payments industry must be complied with by systems that process, forward or store payment data of credit cards. Such service providers must sign a declaration of PCI DSS compliance.
3.15 Windows Servers

Introduction
This security requirement has been prepared based on the provisions of the Group Policy on IT/NT Security.

The security requirement is used as a basis for an approval in the PSA process, among other things. It also serves as an implementation standard for provisions of the Group Policy on IT/NT Security in units which do not participate in the PSA process. These requirements shall be taken into account from the very beginning, including during the planning and decision-making processes. When implementing these security requirements, the precedence of national, international and supranational law shall be observed.

System hardening
Installation

<table>
<thead>
<tr>
<th>Req 1</th>
<th>Unused software must not be installed or must be uninstalled.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>During installation of a system often software components will</td>
</tr>
<tr>
<td></td>
<td>be installed or parts of software will be activated which are</td>
</tr>
<tr>
<td></td>
<td>not needed for the operation or functionality of the system.</td>
</tr>
<tr>
<td></td>
<td>This includes also parts of a software, which will be in-</td>
</tr>
<tr>
<td></td>
<td>stalled as examples but typically not be used (e.g. default</td>
</tr>
<tr>
<td></td>
<td>web pages, example databases, test data). Such components</td>
</tr>
<tr>
<td></td>
<td>should not be installed or must be deleted after installation.</td>
</tr>
</tbody>
</table>

Motivation: Vulnerabilities in software of a system offer an attack window for attackers to infiltrate the system. Un-installing components that are not required can therefore reduce the possibility of a successful compromise of the system.

For this requirement the following threats are relevant:
• Unauthorized access to the system
• Disruption of availability

<table>
<thead>
<tr>
<th>Req 2</th>
<th>The Windows Server Core installation must be used.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Windows Server Core supports various server roles</td>
</tr>
<tr>
<td></td>
<td>that can be viewed at following links. If techni-</td>
</tr>
<tr>
<td></td>
<td>cally possible, Windows Server should be installed</td>
</tr>
<tr>
<td></td>
<td>with the listed server roles as a Windows Server Core:</td>
</tr>
</tbody>
</table>

Note: The Windows Server Core mode is to be selected prior to the installation of Windows Server 2008 or Windows Server 2008 R2. A subsequent change is not possible after installation. With the release of Windows Server 2012 or higher, a change between core and full installation after installation is possible.

Alternatively, the installation of Windows Server 2012 or higher in the mode "Minimal Server Interface" is to be performed. After completing the configuration, you should change to the core mode.

Motivation: Windows Server Core is an installation option in Windows Server, which significantly reduces attack surface, since it installs no graphic user interface and only the most urgently required system files and services.
For this requirement the following threats are relevant:
• Unauthorized access to the system
• Unauthorized modification of data
• Unauthorized use of services or resources
• Attacks motivated and facilitated by information disclosure or visible security weaknesses

---

Req 3 The Windows Server operating system must be installed with the minimum of components, server roles and features required for operation.

With regard to Windows Server 2008 or higher, this refers to the functions and roles which can be managed through the server manager. Server roles provide basic services, and features expand the server installation to include additional functions. In many cases, server roles depend on features or optionally supplement them with additional attributes.


Motivation: Every Windows Server component offers an attack surface – components that are not used and therefore not installed minimize this attack surface. This increases security because only the applications that are actually needed are installed on the server.

For this requirement the following threats are relevant:
• Unauthorized access to the system
• Unauthorized access or tapping of data
• Unauthorized use of services or resources
• Disruption of availability

---

Configuration

Req 4 Unused services and protocols must be deactivated.

After installation of systems and software products typically local or remote reachable services and protocols are active, which are not necessarily needed for operation and functionality of the system. Such services and protocols must be completely disabled on the system. Additionally it is important that a deactivation survives a system reboot.

This kind of system hardening must be done before the system is reachable from the network. Otherwise an attacker has the possibility to attack and maybe compromise the unsecured system.

Motivation: Services and protocols that are not required for system operation increase the potential attack surface and thus the risk of the system being compromised. This risk is further increased by the fact that a security inspection and an appropriate optimisation of the configuration for unused services and protocols will not be done.

For this requirement the following threats are relevant:
• Unauthorized access to the system
• Unauthorized access or tapping of data
• Unauthorized use of services or resources
• Disruption of availability
• Attacks motivated and facilitated by information disclosure or visible security weaknesses
3.15 Windows Servers

Req 5 Unused functions of the operated software and hardware must be deactivated.

During installation of software and hardware often functions will be activated that are not necessarily needed for operation or function of the system. Functions of software are currently inherent part which could not be deleted or deinstalled individually. Such functions must be deactivated in the configuration of the system permanently.

Beside the functions of the software also hardware functions are active which are not necessary for a system. Functions like unused interfaces must permanently deactivated. Permanent means that they must not be reactivated again after system reboot.

Motivation: The hardware or software of a system often contains functions which are not used and so will be a risk for system security. Such functions give an attacker the possibility to manipulate the system. Furthermore it is possible to get unauthorized access other areas or data of the system. An example is a debugging function in software which can be used for troubleshooting but must not be activated during normal operation. Or a hardware interface that will not be used and so is unsecured an allows possibly unauthorized access to the system.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Disruption of availability

Req 6 The automatic launch of applications on removable media must be deactivated.

Removable media such as CD-, DVD-, USB-Sticks or USB-Storage drives shall not automatically start any applications they contain.

Motivation: Automatic application launch could inadvertently launch malware.

For this requirement the following threats are relevant:
- Unnoticeable feasible attacks

Req 7 The Windows server system must be configured in accordance with the Baseline Server Hardening by Microsoft or "Security Best Practices".


Motivation: The application of Baseline Server Hardening by the manufacturer as well as of "security best practices" to reduce vulnerabilities by restrictive configuration settings.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources
- Disruption of availability
- Unnoticeable feasible attacks
Network

Req 8 The administration of the operating system must be done via a network interface which is independent from the production network.

Administrative access to a server must not be done via an interface which provides productive services. Access must be limited to legitimate systems. The administration of applications can also be done using this network interface.

The restriction can be done with, e.g., filter mechanisms, local access lists or a packet filter. This limitation has to be done as restrictive as possible, i.e., limit to single IP addresses or at least small IP ranges.

Motivation: In the event of a successful attack, an attacker may gain access to confidential information or even to the entire system. By restricting the accessibility to legitimate systems, the group of potential attackers can be reduced, and thus also the likeliness of a successful attack. Furthermore, systems must be manageable even in the case the customer network is down.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unnoticeable feasible attacks
- Attacks motivated and facilitated by information disclosure or visible security weaknesses

Req 9 Services for administration must be bind to exact one interface.

The administrative services (e.g., SSH, HTTPS, RDP) must be bind to exact one interface. Hence the separation of management traffic from user traffic, this is the IP address in the management network. If the system - or parts of it - is managed by more than one interface, the management services have to be bind to the lowest possible number.

Motivation: This ensures that it can be clearly foreseen under which address these management services are reachable. In addition, a unique address is important for implementing filters and firewall rules and for checking the authenticity of keys and certificates when using cryptographic procedures to secure management traffic.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unnoticeable feasible attacks
- Attacks motivated and facilitated by information disclosure or visible security weaknesses
Req 10 Network based access used for operating system administration must have integrity, be encrypted and securely authenticated.

Access is only permitted by using secure protocols (e.g., SSHv2, HTTPS, SNMPv3). The administrator must ensure that any network connection between his workstation or a management system and the operating system to be administered is securely authenticated, encrypted and protected against tampering.

*Motivation: If the administrator transmits changes to the configuration settings via unencrypted or unsecure connections, there is a risk that unauthorized parties exploit vulnerabilities. Information could be gained (configuration settings, access IDs, etc.) to exploit additional security vulnerabilities.*

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Denial of executed activities

Req 11 The reachability of services must be restricted.

Typically services that are enabled in the basic configuration are accessible over all interfaces of the system and can be reached from systems in connected networks. This availability is often not needed or meaningful for system functioning. For this reason, services should only be enabled on interfaces where their usage is required. On interfaces where services are active, the reachability must be limited to legitimate communication peers. This limitation must be realized on the system itself (without measures (e.g. firewall) at network side).

*Motivation: Disabling services on interfaces which do not require system accessibility or by limiting the reachability can greatly reduce the potential vulnerabilities offered to an attacker. For example, access to a system via SSH from the Internet is not necessary. If this service could be accessed from the interface connected to the Internet, this would greatly increase the risk of attacks on the service.*

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized use of services or resources
- Disruption of availability
- Attacks motivated and facilitated by information disclosure or visible security weaknesses

Req 12 If services cannot be bind to the minimal required interfaces by configuration, a local packet filter must regulate the accessibility of the service.

Sometimes, software cannot be bind to dedicated interfaces. A local packet filter or TCP-wrapper can ensure this.

*Motivation: Packet filters offer effective protection in order to prevent misuse services from other networks.*

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized use of services or resources
- Attacks motivated and facilitated by information disclosure or visible security weaknesses
3.15 Windows Servers

Req 13 The Windows Server Firewall must be configured to filter out ports that are not required provided there is no alternative firewall for the system.

The Windows Firewall filters TCP/IP data traffic on the basis of various TCP/IP attributes (e.g., TCP port, source and destination address), thus preventing unwanted data traffic. The Windows Server Firewall does not need to be used if the Windows Server is operated in an Internet Protocol (IP) subnet that is protected by a firewall anyway or if a third-party manufacturer’s software firewall is used which applies the same filter rules.

Motivation: A component that is jointly used by the subnet servers can filter TCP/IP data traffic. This is possible, for example, if multiple file and print servers of an Internet Protocol (IP) subnet are united in a network which is available for the same workstation system services. It is also permissible for the Windows Firewall to filter TCP/UDP ports and an additional component to filter subnets that are permitted to set up connections.

The use of filter rules effectively prevents unwanted data traffic and thus makes it more difficult for any malicious software to spread in an uncontrolled manner.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources
- Disruption of availability

Req 14 The IPv4 and IPv6 addresses of all interfaces of a server must be configured statically.

IP addresses providing services must not be changed on external influence, even in the case of an enforced reboot. An automatic assignment of IP addresses, e.g., using DHCPv4/v6 or IPv6 auto-configuration is permitted only in the case when deactivated after initial allocation or secured otherwise. IPv6 router advertisements must be ignored.

It is recommended to form the host share of the IPv6 addresses randomly. Due to the very large address space of IPv6, this way it is very time-consuming for an attacker to use scans to discover systems.

Motivation: Avoidance of bypassing authorized systems by man-in-the-middle attacks: The automatic configuration of IP addresses enables an attacker to divert traffic or impair communication.

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data
- Disruption of availability

Req 15 Kernel based network functions not needed for the operation as a server must be deactivated.

Routing functions are not needed on a server; consequently the routing function must be disabled. Additionally the answering routine for broadcast ICMP packages must be disabled. Usually this and other network features are already configured correctly out-of-the-box.

Motivation: The routing functions enable misuse scenarios, meaning that an attack can route malicious packets through the server to connected networks.

For this requirement the following threats are relevant:
- Disruption of availability
- Unnoticeable feasible attacks
3.15 Windows Servers

Req 16  The TCP/IP protocol must be installed and used to the exclusion of all others.

Administrators should uninstall or deactivate alternative network protocols.

Motivation: Alternative network protocols are less common and are usually not developed further. Some of them also represent a risk, e.g., NetBIOS facilitates the search for potential vulnerabilities and can be used for attacks. Also, a NetBIOS that is activated over TCP/IP, in particular in distributed environments, can lead to a marked reduction in response times. TCP/IP is the standard network protocol of the Windows Server operating system. Web applications do not require the NetBIOS enhancement of TCP/IP.

For this requirement the following threats are relevant:
• Unauthorized access or tapping of data
• Unauthorized modification of data
• Unauthorized use of services or resources
• Disruption of availability
• Unnoticeable feasible attacks
• Attacks motivated and facilitated by information disclosure or visible security weaknesses

Req 17 Software and hardware components that are no longer supported by vendor, producer or developer must not be used.

Components that have reached end-of-life or end-of-support must not be used. Excluded are components that have a special support contract. This contract must guarantee the correction of vulnerabilities over components life-time.

Motivation: Hardware and software components that have reached end of life or end of support represent a risk for a system. This means that a vendor does not supply remedial updates or patches for a component should errors or vulnerabilities occur. This means that vulnerabilities cannot be fixed when they occur and could be exploited to compromise the system or to impair its availability.

For this requirement the following threats are relevant:
• Unauthorized access to the system
• Unauthorized access or tapping of data
• Unauthorized modification of data
• Disruption of availability

Req 18  Known vulnerabilities in software and hardware of the system must be fixed or protected.

Prior to installation of a software or hardware component, users must check whether any vulnerability has been discovered and published for the version they are installing. Any component that proves to have a vulnerability must not be installed or used. Excepted from this rule are components for which the vendor has already provided a measure to remedy the vulnerability, e.g. a patch, update or workaround. In this case, the additional measure must be implemented on the system. Furthermore it is a ongoing process during the complete life cycle of the system to fix upcoming vulnerabilities promptly.

Motivation: Publication of vulnerabilities increases the risk of successful exploitation by an attacker, especially since the published information usually includes details on how to exploit the vulnerability and tools that make exploitation possible. One example of this is when Web servers are compromised due to not patched vulnerabilities.

For this requirement the following threats are relevant:
• Unauthorized access to the system
• Unauthorized access or tapping of data
3.15 Windows Servers

- Unauthorized modification of data
- Disruption of availability

Req 19 A Windows server operating system must be migrated before the end of the Extended Support on an operating system in Mainstream Support.

Motivation: Windows Server operating systems contain design errors, critical errors and security gaps. Such gaps in older Windows Server operating systems are known and can be used with the intention of planting viruses or similar malicious software into the server or to misuse the server.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources
- Disruption of availability
- Attacks motivated and facilitated by information disclosure or visible security weaknesses

Protecting availability and integrity

Req 20 Data with need of protection must be secured against unauthorized viewing and manipulation during transmission and storage.

Adequate security measures for transmission and storage must be implemented of data with a need for protection that are classified as internal, confidential or strictly confidential. The chosen measure depends on the classification for the data and other factors such as the type of network used during transmission, the storage location for data, etc. Furthermore must be guaranteed that confidential and strictly confidential data will not be unprotected during temporary storage (e.g. in web cache, temporary folders).

All authentication data such as user names, passwords, PINs, etc. must be protected against unauthorized viewing and manipulation. This applies equally to permanent storage and transmission. Typical measures taken to protect authentication data or combinations of these are:

For storage:

- Client systems: encryption or obfuscation of authentication data, no persistent storage, limitation of access rights.
- Server systems: hashing of authentication data with PBKDF like scrypt or bcrypt or when this is not possible hashes with salt.

For transmission:

- Usage of cryptographically protected network protocols.
- Challenge response method (Disadvantage of this solution is that passwords needed in clear-text on server. If this solution is feasible depends on the individual threat scenario)

Files of a system that are needed for the functionality must also be protected against manipulation. This is necessary because system’s integrity can be damaged when the system access this kind of files. An example is the use of check-sum or cryptographic methods to validate if e.g. firmware images, patches, drivers or kernel modules are free of manipulations.

For transmission of data with a need of protection network protocols that are insecure due to insufficient security measures shall not be used. Examples are: SSLv3, SSHv1, FTP, Telnet, SNMPv1 and 2c. In case of these protocols a newer version without vulnerabilities or a secure alternative must be used.
Motivation: If data with a need of protection will not be secured an attacker could record or manipulate the data during transmission over a network. An example is the recording of user names and passwords during system administration with the telnet clear-text protocol. Storing data on a system without adequate protection may mean that unauthorized users can copy or modify it. One example is when passwords can be read out when they stored in an inadequate secured way (e.g. usage of unsecure hashing algorithms like MD5 or SHA-1) or even without encryption on a system or manipulation of firmware to affected the system integrity.

For this requirement the following threats are relevant:

- Unauthorized access or tapping of data
- Unauthorized modification of data

Req 21 If the system is not located in a room with at least protection class "high" (PC3), used data storages must be fully encrypted.

The Protection Classes (PC) are defined in the Annex 1, "Physical Security of Buildings", of the Group Policy on "Physical Security". Typically, Datacenters are compliant to the requirements of PC3.

Data storages are all disks and flash memory in the systems.

Motivation: Access to devices which are operated outside of data centers with protected access is relatively easy. Physical data storage media can be easily stolen as a result.

Implementation example: On Windows Server 2008, the administrator can activate drive encryption using BitLocker and key storage on a TPM.

For this requirement the following threats are relevant:

- Unauthorized access or tapping of data
- Unauthorized modification of data

Req 22 Information with need of protection must not be contained in files, outputs or messages that are accessible by unauthorized users.

Information with need of protection must not be accessible in files, outputs or messages of the system by unauthorized users. This includes information relating to the operating system, used middleware or applications such as vendor, product name, product identifier, installed software versions, installed service packs, patches, hot fixes and serial numbers. Examples for system messages which must be free of sensitive data are:

- Comments in downloadable files
- Error and system messages
- Stack traces
- Network protocols
- Login windows and dialogs

Furthermore, details of implementation and information relating, e.g., to backend software/systems, function calls, SQL instructions or structure of database, must not be contained in error messages. Excluded from this are displays and outputs that can be viewed and retrieved by authorized users who are logged in. In addition, an internal transfer of system internal information for error analysis is allowed in an adequate dimension. In this case the continuable regulations or guidelines (e.g. of data privacy) must be noticed.

Motivation: The information named above can be used by an attacker to prepare specific attacks on a system. In this way an attacker could, for example, use the precise software version to identify vulnerabilities in the product and, in a second step, exploit them.
For this requirement the following threats are relevant:
• Unauthorized access or tapping of data
• Attacks motivated and facilitated by information disclosure or visible security weaknesses

Req 23  All partitions must be operated with the New Technology File System (NTFS).

For compatibility reasons, Windows Server offers the option to use alternative file systems. Administrators shall only use the NTFS. Even a data partition assigned to a Windows Failover Cluster in the SAN shall only be operated with NT-FS by the administrators. It can ReFS (Resilient File System) can be used if this reflects the semantics of NTFS.

Motivation: NTFS is the only system to offer security settings such as authorizations and monitoring settings.

For this requirement the following threats are relevant:
• Unauthorized access or tapping of data
• Unauthorized modification of data
• Denial of executed activities
• Unnoticeable feasible attacks

Protecting availability and integrity

Req 24  The system must be robust against overload situations.

A system must provide security measures to deal with overload situations. In particular, partial or complete impairment of system availability must be avoided. Potential protective measures include:
• Restricting of available RAM per application
• Restricting of maximum sessions for a Web application
• Defining the maximum size of a dataset
• Restricting CPU resources per process
• Prioritizing processes
• Limiting of amount or size of transactions of an user or from an IP address in a specific time range

Motivation: An attacker can effect systems availability through targeted exploitation of vulnerabilities in resources administration. An attacker can, for example, send bulk queries to a Web server and, by doing so, compromise its availability if the maximum number of permissible Web sessions is not restricted.

For this requirement the following threats are relevant:
• Disruption of availability

Req 25  If an overload situation cannot be prevented, the system must act in a predictable way.

A system must be built in this way that it can react on a overload situation in a controlled way. However it is possible that a situation happens where the security measures are not longer sufficient.

In such case it must be ensured that the system cannot reach an undefined and thus potentially insecure state. In an extreme case this means that a controlled system shutdown is preferable to uncontrolled failure of the security func-tions and thus loss of system protection.
Motivation: With denial-of-service attacks can an attacker try to overload a system to effect its availability or integrity. Unpredictable acting of the system is then a risk for functionality and data and possibly also for other systems.

For this requirement the following threats are relevant:
• Disruption of availability

---

**Req 26**  
The system must be robust against unexpected input.

During transmission of data to a system it is necessary to validate this before processing. This includes all data which are send to the system. Examples for this are user input, values in arrays and content in protocols. The following typical implementation mistakes must not be done:

• No validation on the lengths of transferred data
• Incorrect assumptions about data formats
• No validation that received data complies with the specification
• Insufficient handling of protocol errors in received data
• Insufficient restriction on recursion when parsing complex data formats
• White listing or escaping for inputs outside the values margin

Motivation: An attacker can try to put a system in an unsecure state through targeted manipulation of transmitted data. The object of such an attack is to compromise the usability, availability or integrity of individual services or of the entire system. For instance a unclean memory handling can lead to a buffer overflow that allows an attacker to execute arbitrary code on the effected system.

For this requirement the following threats are relevant:
• Unauthorized access to the system
• Unauthorized use of services or resources
• Disruption of availability

---

**Req 27**  
Growing (dynamic) content must not influence system functions.

Growing log data and uploads must not influence system functions.

Motivation: A filled up filesystem could stop the system from operations.

Implementation example: Usage of dedicated filesystems, separated from main system functions, or quotas, or at least a file system monitoring.

For this requirement the following threats are relevant:
• Disruption of availability

---

**Req 28**  
Systems must not process IP packets which source address is not reachable via the incoming inter-face.

It is necessary to ensure there are no unneeded default routes which is typically the case for internal systems.

Motivation: In such a case the IP packet comes from an untrusted source (spoofed address) or a routing error exists in the network. In both cases the packet has to be dropped.

Implementation example: Use of "Reverse Path Filter" (RPF) which provides this feature.

For this requirement the following threats are relevant:
• Disruption of availability
Req 29 The processing of ICMPv4 and ICMPv6 packets which are not required for operation must be dis-abled.

There are different types of ICMPv4 and ICMPv6 that are not used in most networks, but represent a risk. These types must be disabled or filtered and not be answered, send or processed. The following ICMP types are permitted and may be used:

- Echo Request [Type 8 (v4), Type 128 (v6)]
- Echo Reply [Type 0 (v4), Type 129 (v6)]
- Destination Unreachable [Type 3 (v4), Type 1 (v6)]
- Time Exceeded [Type 11 (v4), Type 3 (v6)]
- Parameter Problem [Type 12 (v4), Type 4 (v6)]
- Packet Too Big [Type 2 (only v6)]
- Neighbor Solicitation [Type 135 (only v6)]
- Neighbor Advertisement [Type 136 (only v6)]

It is possible that other types will be necessary. This should be checked in each individual case. The ICMPv4 types "Timestamp Reply (14)," "Netmask Reply (18)," "Information Reply (16)" and "Redirect (5)" and ICMPv6 types "Router Solicitation" (133), "Router Advertisement" (134) und "Redirect" (137) must not be responded to or processed under any circumstances.

Motivation: ICMPv4 and v6 packets can be used by an attacker to request specific information which can be helpful for planning further attacks. In addition, it may be possible to influence the availability of systems.

For this requirement the following threats are relevant:
- Disruption of availability
- Attacks motivated and facilitated by information disclosure or visible security weaknesses

---

Req 30 IP packets with unnecessary options or extension headers must not be processed.

IP options and extension headers (e.g., source routing) are only required in exceptional cases. So, all packets with en-abled IP options or extension headers must be filtered.

Motivation: Packets with IP options require extended processing. An attacker can exploit this in order to carry out deni-al-of-service attacks against an affected device.

For this requirement the following threats are relevant:
- Disruption of availability
Authentication and authorization

Req 31 System functions with a need of protection must not be used or accessed without successful authentication and authorization.

The usage of a system functions or access of data classified as internal, confidential or strictly confidential must only be possible unambiguously user identification and successful authentication on basis of the user name and at least one authentication attribute. Excepted from this are functions for public use such as those for a Web server on the Internet, via which information is made available to the public. Examples for functions which require a prior authentication are network services (like SSH, SFTP, Web services), local access via a management console, local usage of operating systems and applications. The following examples are possibilities that could be used for authentication.

- Query user name and password
- Use of cryptographic keys and certificates (e.g. as Smartcard)

This requirement must also be applied to accounts that are only used for communication between systems (M2M).

Motivation: The authentication is necessary to doubtless identify a user because the allocated authorization, and therefore the access on data and services of the system depends on that.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized use of services or resources
- Denial of executed activities

Req 32 If the system is not located in a room with at least protection class "high" (PC3), the BIOS must be secured against unauthorized changes.

Servers operated in public or customer areas must be especially protected against unauthorized access and changes: The BIOS settings must be protected against export and tampering. In case passwords are used, these must be exclusive to the individual server and must not allow conclusions to be drawn about a distinguishing feature of the server.

The BIOS must be configured in such a way that only the designated operating system can be started with it from the designated partition.

Motivation: Changing BIOS settings can facilitate attacks. Since, for example, local rooms with technical installations seldom offer access protection to the servers, attackers could change the startup sequence of data storage media when the server is started in the BIOS without the password protection described. This would make it possible to start an alternative operating system which circumvents the security mechanisms of the implemented operating system.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data

Req 33 Accounts must be used that allow unambiguous identification of the user.

Users must be identified unambiguously by the system. This can typically be reached by using a unique account per user. So named group accounts, i.e. the use of one account for several persons, must not be used. On exception of this requirement are so named machine accounts. These will be used for authentication and authorization from system to each other or for
applications on a system and can’t be assigned to a single person. Such accounts must be assigned on a per system or per application basis. In this connection, it has to be guaranteed that this account can’t be misused. Possibilities to protect these accounts are:

- Configuring of a Password that fulfills the security requirements and is known by less than possible circle of administrators.
- Configuring the account that only a local use is possible and a interactive login isn’t possible.
- Use of a technique for authentication of the specific account with public and private key or certificates.
- Limiting the access over the network for legitimised systems.

Additional solution must be checked on their usability per individual case.

Motivation: Unambiguous user identification is a prerequisite for assigning a user the rights that he requires to perform his tasks on the system. This is the only way to adequately control access to system data and services and to prevent misuse. Furthermore, it makes it possible to log activities and actions on a system and to assign them to individual users.

For this requirement the following threats are relevant:

- Unauthorized access to the system
- Unauthorized use of services or resources
- Denial of executed activities

Req 34 Accounts must be protected against unauthorized use by at least one authentication attribute.

The various user and machine accounts on a system must be protected from misuse. To this end, an authentication attribute is typically used, which, when combined with the user name, enables unambiguous authentication and identification of the authorized user.

Authentication attributes include:

- Cryptographic keys
- Token
- Passwords
- PINs

This means that authentication based on a parameter that can be spoofed (e.g. phone numbers, public IP addresses or VPN membership) is not permitted. Exceptions are attributes that cannot be faked or spoofed by an attacker. Several of the above options can be combined (dual-factor authentication) to achieve a higher level of security. Whether or not this is suitable and necessary depends on the protection needs of the individual system and its data and must be evaluated for individual cases.

Motivation: Accounts that are not protected with a secret authentication attribute can be used by an attacker to gain unauthorized access to a system and the data and applications stored on it.

For this requirement the following threats are relevant:

- Unauthorized access to the system
- Unauthorized use of services or resources
- Denial of executed activities

Req 35 Accounts with extensive rights must be protected with two authentication attributes.

Accounts, for example used for administration, maintenance and troubleshooting, have extensive rights. Therefore a single protection (e.g. a password), as for normal user accounts with less rights, is not suitable. To get a higher protection level it is necessary to use more than one authentication attribute. For this a combination of an attribute that the user knows and an attribute that the user owns will be used often. This kind of authentication will be named as 2-factor authentication. Examples for 2-factor authentication are:
3.15 Windows Servers

- Private key with Passphrase
- Secure-ID Token with Password

Motivation: Accounts with extensive rights as used for system administration have a higher risk for system’s security. An attacker can get extensive rights by compromising such an account to get access to wide parts of the system and stored data.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized use of services or resources
- Denial of executed activities

Req 36 There must not be a privilege escalation method which allow gaining administrator/root privileges from an user account without anew authentication with two authentication attributes.

Privilege escalation methods include always the risk that more permissions are gained than needed. The number of exploits in such mechanisms shows the complexity and vulnerability of this solutions which therefore cannot be trusted.

Motivation: If an attacker compromises an account which has permission to a privilege escalation, it may be possible that the attacker get access to wide parts of the system and stored data.

Implementation example: (Re-)Login directly into the account with the needed permissions.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Denial of executed activities

Req 37 Predefined accounts must be deleted or disabled.

Many systems have default accounts (e.g. guest, ctxsys), some of which are preconfigured with or without known pass-words. These standard users must be deleted or disabled. Should this measure not be possible the accounts must be locked for remote login. In any case disabled or locked accounts must configured with a complex password (12 character and more, use of upper/lower case, numbers and special characters). This is necessary to prevent unauthorised use of such a account in case of misconfiguration.

Exceptions to this requirement to delete or disable accounts are accounts that are used only internal on the system in-volved and that are required for one or more applications on the system to function. Also for this accounts remote ac cess or local login must be forbidden to prevent a abusive use by users of the system.

Motivation: Standard users are typically generally known and can be used by an attacker for targeted brute force and dictionary attacks. Standard user accounts represent a special risk if they do not use a password or only use a stand ard password that is generally known. Such standard user accounts can easily be exploited by an attacker in order to gain access to the system involved without being authorized to do so.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized use of services or resources
- Denial of executed activities
Predefined authentication attributes must be deleted or disabled.

Normally, authentication attributes such as password or cryptographic keys will be preconfigured from producer, vendor or developer of a system. Such authentication attributes must be changed an own attribute not known by other parties.

Motivation: Authentication attributes like password or cryptographic keys preconfigured from third parties are not trustable. Such authentication attributes can be used to compromise systems or their data.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized use of services or resources
- Denial of executed activities

The authorizations for accounts and applications must be reduced to the minimum required for the tasks they have to perform.

Authorizations to a system must be restricted to a level in which a user can only access data and use functions that he needs in the course of his work. Suitable authorizations must also be assigned for access to files that are components of the operating system or of applications or that are generated by the same (e.g. configuration and logging files).

Alongside access to data, execution of applications and components must also take place with rights that are as low as possible. Applications should not be executed with administrator or system rights.

Motivation: If the rights granted to a user on a system are too broad, it could be possible for him to access data and applications which he is not permitted to view or use. This would give him the opportunity to disclose or modify confidential data and to manipulate system files. Applications with rights that are too broad can be used by a user to extend his own authorizations and thus to gain access to files and system components to which he would not have had access with his authorizations under normal circumstances.

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources

The system must be connected to a central system for user administration.

For provisioning of these authorization information must the system provide a central interface (e.g. LDAPs for authorization, Kerberos for authentication, locking information for certificates) or decentralized mechanisms (e.g. public-key authentication). A central solution for identity management must be preferred.

In areas where a central identity management system is not available a central system such as LDAP, TACACS+ or RADIUS server for the administration of accounts and their authentication and authorization must be used.

Motivation: Central administration of identity of accounts and their rights means that they only have to be maintained once instead of separately on each system. From the aspect of security, the advantage is that an account and its rights only known on a single central side. This information can be transmitted from a central side to systems (provisioning), central administrated (reconciliation) and central deleted (deprovisioning). This reduces the risk of
accounts being forgotten during changing or deletion since they are configured on multiple systems. This could give a user wrong system rights or continued access to a system.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources
3.15 Windows Servers

Req 41  Active Directory accounts must be used for technical administration of domain members.

Windows Server operating systems are usually members of an Active Directory domain that enables central administration of user accounts. The administrator can grant a user account in the Active Directory, for example, the authority to manage the server completely through membership in the local administrator group on the server that is to be managed.

Motivation: The use of accounts of the central directory services makes it possible to map an account to an individual and to implement security policies (e.g., smartcard login required) for these centrally managed accounts.

Implementation example: An employee who wants to manage file releases in a network of several file and print servers on a regular basis is assigned an account in the Active Directory, which is a member of the local group of printer administrators on all servers in the network.

For this requirement the following threats are relevant:
• Unauthorized access to the system
• Unauthorized use of services or resources
• Denial of executed activities

Req 42  The number of local accounts needed for operation must be minimized.

All operating systems have high security requirements regarding local accounts. The administrator must ensure that all unused local accounts are deactivated.

The passwords needed to login at these accounts must be unique for every machine and account. It is recommended to store them properly, e.g., in a safe or online in a password vault where access is granted only when needed.

Motivation: Local accounts are additional points of attack which can be used by attackers or unauthorized individuals. This requires that only absolutely necessary local accounts required for operation exist in the operating system.

For this requirement the following threats are relevant:
• Unauthorized access to the system

Req 43  In operating system installations that are members of an Active Directory domain, the first local administrator account of the operating system must be blocked.

All Windows Server operating systems come with an initial local administrator account, whose technical security ID (SID) is identical for all Windows Server operating systems. The owner shall deactivate this in order to effectively prevent any use of the account.

Note: If the operation system can no longer be deactivated by means of a domain account, the administrator would have to interrupt server operation and start the operating system in safe mode. This local account is automatically re-activated in this process.

Motivation: The initial local administrator account of a Windows Server represents a vulnerability since it has the same technical security ID (SID) in all systems. It is not automatically blocked if an attacker attempts to guess the password through recurrent login attempts.

Implementation example: The administrator can deactivate the local administrator account by means of a group policy, for example.
For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources
- Denial of executed activities

Req 44 The predefined local guest account must be blocked.

All Windows Server operating systems come with a predefined local and deactivated guest account. The administrator shall deactivate this account in order to effectively prevent any use of the account.

Motivation: The predefined local guest account represents a vulnerability, since this identity can be used to obtain information from the operating system that would make it easier to launch attacks.

Implementation example: The administrator can deactivate the predefined guest account by means of a group policy, for example.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources
- Denial of executed activities

Authentication parameter password

Req 45 If a password is used as an authentication attribute, it must have at least 8 characters and contain three of the following categories: upper cases, lower case, numbers and special characters.

A system may only accept passwords that comply with the following complexity:
- Minimum length of 8 characters.
- Comprising at least three of the following categories: upper/lower case letters, numbers and special characters

When a password is assigned, the system must ensure that the password meets these requirements. If a central system will be used for user authentication this function can be forwarded or delegated to this system.

Motivation: Passwords with the above complexity offer high robustness against attacks coupled with acceptable user friendliness. Passwords with this level of complexity have proved their efficiency in practice. Trivial passwords that are too short are susceptible to brute force and dictionary attacks and are therefore easy for attackers to determine. Once a password has been ascertained it can be used by an attacker for unauthorized access to the system and the data on it.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized use of services or resources
- Denial of executed activities

Req 46 If a password is used as an authentication attribute, users must be enable to independently change the password anytime.
The system must offer a function that enables a user to change his password at any time. When an external centralized system for user authentication will be used it is possible to redirect or implement this function on this system.

Motivation: The fact that a user can change his authentication attribute himself at any time enables him to change it promptly if he suspects that it could have been accessed by a third party.

For this requirement the following threats are relevant:

- Unauthorized access to the system
- Unauthorized use of services or resources
- Denial of executed activities

---

Req 47 If a password is used as an authentication attribute, a protection against brute force and dictionary attacks that hinder password guessing must be implemented.

Brute force and dictionary attacks aim to use automated guessing to ascertain passwords for user and machine accounts. Various measures or a combination of this measures can be taken to prevent this.

The most commonly used protection measures are:

- Increasing the delay (e.g. doubling wait times at each attempt) for each renewed password input following an incorrect entry (“tar pit”).
- Blocking an account following a specified number of incorrect attempts (typically 5). However has to take into account that this solution needs a process for unlocking and an attacker can force this to deactivate accounts and make them unusable.
- Using CAPTCHA to prevent automated attempts (often used for Web applications).

In order to achieve higher security, it is often meaningful to combine two or more of the measures named here. This must be evaluated in individual cases and implemented accordingly.

Motivation: Implementation of one or more of the above measures can prevent successful execution of brute force and dictionary attacks to ascertain passwords.

For this requirement the following threats are relevant:

- Unauthorized access to the system
- Unauthorized use of services or resources
- Denial of executed activities

---

Req 48 If a password is used as an authentication attribute, they must be hidden when displayed on screen.

When a user enters a password or the password is otherwise displayed on screen, it must be made unreadable. Typically, the individual characters of the password are replaced by a character such as “*”. Under certain circumstances it may be permissible for an individual character to be displayed briefly during input. Such a function is used, for example, on Smartphone’s to make input easier. However, the entire password is never output to the display in plaintext.

Motivation: To prevent another person reading a password by chance or intentionally on screen during input, the password must be rendered unreadable during input or display.

For this requirement the following threats are relevant:

- Unauthorized access to the system
- Unauthorized use of services or resources
- Denial of executed activities
3.15 Windows Servers

Logging

Req 49 Security relevant events must be logged with a precise timestamp and a unique system reference.

Systems must log the occurrence of security-relevant incidents. So that these events can be evaluated and classified, they must be logged together with a unique system reference (e.g., host name, IP or MAC address) and the exact time the incident occurred. Furthermore, legal and data privacy regulations (e.g. time of storage of logging data) must be proved and followed. The following table lists events that are relevant to security and the corresponding data that typically has to be logged by a system. Exceptions are systems for which no or only restricted logging applies. Examples of such systems are customer devices such as Smartphone’s or IADs/Homegateways (e.g. Speedport).

Logging must be done considering the currently valid legal, wage and company regulations. This regulations state among others that logging of events can be done only earmarked. Logging of events for doing a work control of employees is not allowed.

Typical event that reasonable should be logged in many cases are:

<table>
<thead>
<tr>
<th>Event</th>
<th>Event data to be logged</th>
</tr>
</thead>
<tbody>
<tr>
<td>Incorrect login attempts</td>
<td>• Account, • No. of failed attempts, • Source (IP address) of remote access</td>
</tr>
<tr>
<td>System access with accounts with administrator rights</td>
<td>• Account, • Access timestamp, • Length of session, • Source (IP address) of remote access</td>
</tr>
<tr>
<td>Account administration</td>
<td>• Administrator account, • Administered account, • Activity performed (configure, delete, enable and disable)</td>
</tr>
<tr>
<td>Change of group membership for accounts</td>
<td>• Administrator account, • Administered account, • Activity performed (group added or removed)</td>
</tr>
<tr>
<td>Critical rise in system values such as disk space, CPU load over a longer period</td>
<td>• Value exceeded, • Value reached (Here suitable threshold values must be defined depending on the individual system.)</td>
</tr>
</tbody>
</table>

Logging of additional security-relevant events may be meaningful. This must be verified in individual cases and implemented accordingly where required.

Motivation: Logging security-relevant events is a basic requirement for detecting ongoing attacks as well as attacks that have already occurred. This is the only way in which suitable
measures can be taken to maintain or restore system security. Furthermore, the logging data is used as evidence so that legal steps can be taken against attackers.

For this requirement the following threats are relevant:
- Denial of executed activities
- Unnoticeable feasible attacks

Req 50 Security relevant logging data must be send to an external system direct after their creation.

Logging data must be forwarded to an external system in appropriate logging files as well as being stored locally. Standard protocols like Syslog, SNMPv3 must be preferred.

Motivation: If logging data is only stored locally it can be manipulated by an attacker who succeeds in compromising the system in order to conceal his attack and any manipulation he has performed on the system. This is the reason why the forwarding must be done immediately after the event occurred.

For this requirement the following threats are relevant:
- Denial of executed activities
- Unnoticeable feasible attacks
3.17 JBOSS Application Servers

Introduction

This security requirement has been prepared based on the provisions of the Group Policy on IT/NT Security.

The security requirement is used as a basis for an approval in the PSA process, among other things. It also serves as an implementation standard for provisions of the Group Policy on IT/NT Security in units which do not participate in the PSA process. These requirements shall be taken into account from the very beginning, including during the planning and decision-making processes. When implementing these security requirements, the precedence of national, international and supranational law shall be observed.

This document assumes, that a web server is running in front of the application server. If this is not the case, then the security requirements on web servers must be fulfilled by the application server or another component, which receives http requests before the application server. A load balancer could be such component.

The examples are valid for JBoss version 7.x.

Security requirements

---

**Req 1** Software and hardware components that are no longer supported by vendor, producer or developer must not be used.

Components that have reached end-of-life or end-of-support must not be used. Excluded are components that have a special support contract. This contract must guarantee the correction of vulnerabilities over components life-time.

*Motivation:* Hardware and software components that have reached end of life or end of support represent a risk for a system. This means that a vendor does not supply remedial updates or patches for a component should errors or vulnerabilities occur. This means that vulnerabilities cannot be fixed when they occur and could be exploited to compromise the system or to impair its availability.

For this requirement the following threats are relevant:

- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Disruption of availability

---

**Req 2** Known vulnerabilities in software and hardware of the system must be fixed or protected.

Prior to installation of a software or hardware component, users must check whether any vulnerability has been discovered and published for the version they are installing. Any component that proves to have a vulnerability must not be installed or used. Excluded from this
rule are components for which the vendor has already provided a measure to remedy the vulnerability, e.g. a patch, update or workaround. In this case, the additional measure must be implemented on the system. Furthermore it is a ongoing process during the complete life cycle of the system to fix upcoming vulnerabilities promptly.

**Motivation:** Publication of vulnerabilities increases the risk of successful exploitation by an attacker, especially since the published information usually includes details on how to exploit the vulnerability and tools that make exploitation possible. One example of this is when Web servers are compromised due to not patched vulnerabilities.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Disruption of availability

---

**Req 3** The application server must run under a dedicated (operating-system) account that only has the permissions required for operation.

The account must not have administrator permissions. If the application server needs to be launched with administrator permissions, then it has to be switched to another account once launched. This is comparable to normal web server situations.

**Motivation:** Every service that runs with administrator permissions constitutes a higher risk in respect of possible vulnerabilities or misuse.

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data
- Unauthorized modification of data

---

**Req 4** Unused services and protocols must be deactivated.

Some application servers provide multiple interfaces for requests to the applications as well as for the administration of the application server. Usually only one of each interface is used. All interfaces and their connected services, which are not required, must be deactivated.

**Motivation:** Services and protocols that are not required for system operation increase the potential attack surface and thus the risk of the system being compromised. This risk is further increased by the fact that a security inspection and an appropriate optimisation of the configuration for unused services and protocols will not be done.

Implementation example: Only services, that are needed, are assigned to sockets. This is configured in the section `<socket-binding-group>` in the configuration file.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Disruption of availability

---

**Req 5** Automatic deployment of applications must be disabled.

**Motivation:** Applications are started by automatic deployment after they are copied in a specific directory. This function could be misused.

Implementation example: Remove the subsystem "deployment-scanner" in the configuration file.
If the automatic deployment on startup of the application server is needed, then set the value for "scan-interval" to a value less than 1.

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Disruption of availability

---

**Req 6** Sample applications must be deleted.

*Motivation:* Sample application could contain vulnerabilities and provide points of attack.

Implementation example: JBoss comes with a welcome page, that shall be deactivated:
Set in the configuration file:
```xml
<param name="enable-welcome-root" value="false"/>
```
Or set in WEB-INF/jboss-web.xml of the own application the context-root:
```xml
<jboss-web><context-root>/</context-root></jboss-web>
```

For this requirement the following threats are relevant:
- Attacks motivated and facilitated by information disclosure or visible security weaknesses

---

**Req 7** The access to the console used to administer the application server (console) must use encryption.

*Motivation:* Communication with the administration application usually contains confidential information, such as login information (name and password). This information must not be transmitted in unencrypted form.

Implementation example: After creating appropriate keys and certificates for http encryption, the management realm and the management interface is configured in the configuration file:
```xml
<management>
  <security-realms>
    <security-realm name="ManagementRealm">
      <server-identities>
        <ssl protocol="TLSv1">
          <keystore path="server.keystore" relative-to="jboss.server.config.dir" key-store-password="<jbosspassword>"/>
          ...</ssl>
        </server-identities>
    </security-realm>
    ...
  </security-realms>
  ...<management-interfaces>
  ...
  <http-interface security-realm="ManagementRealm">
    <socket-binding https="management-console-https"/>
  </http-interface>
</management>
```

*Note:* There is a bug in some JBoss versions: Instead of "keystore-password" only "password" must be used.
The appropriate socket is bind to the earlier defined interface:

```xml
<socket-binding-group name="standard-sockets"
  default-interface="public" port-offset="${jboss.socket.binding.port-offset:0}">
  ...
  <socket-binding name="management-console-https" interface="management" port="9991"/>
  ...
</socket-binding-group>
```

For this requirement the following threats are relevant:

- Unauthorized access to the system

---

**Req 8** The application server must use different IP addresses and/or different TCP ports for administration purposes and the applications itself.

**Motivation:** Even if the TCP port were not available for the application, say as a result of an attack, it would be possible to access the administration application. An attacker cannot access administration via the application TCP port by extending permissions. Firewalls for application accesses and administration can be configured differently in the case of different TCP ports. In this way, the administration applications cannot be accessed from the Internet. By using different IP addresses it is possible to separate traffic for administration and productive data.

Implementation example: Different interfaces are defined in the configuration file:

```xml
<interfaces>
  <interface name="management">
    <inet-address value="1.1.1.1"/>
  </interface>
  <interface name="public">
    <inet-address value="1.2.3.4"/>
  </interface>
</interfaces>
```

Then services are assigned to interfaces and ports.

```xml
<socket-binding-group name="standard-sockets"
  default-interface="public" port-offset="${jboss.socket.binding.port-offset:0}">
  <socket-binding name="management-console-https" interface="management" port="9991"/>
  <socket-binding name="ajp" port="8009"/>
  <socket-binding name="http" port="8080"/>
</socket-binding-group>
```

For this requirement the following threats are relevant:

- Unauthorized access to the system

---

**Req 9** Each application must run in a separate domain or a separate standalone server.

JBoss distinguishes between standalone and domain configuration. In the case of JBoss, domains are a group of servers that share a common configuration.

**Motivation:** It should not be possible to influence an application by modifying the configuration of another application.

For this requirement the following threats are relevant:

- Unauthorized access or tapping of data
- Unauthorized modification of data
Req 10 Access to the application server must be logged.

The application server log must contain the following information:

- Access timestamp
- Source (IP address)
- Account (if known)
- URL
- HTTP status code of application server response

Logging must be done considering the currently valid legal, wage and company regulations. This regulation states among others that logging of events can be done only earmarked. Logging of events for doing a work control of employees is not allowed.

Motivation: For the analysis of security incidents it is very important to have basic information on how the attack has been carried out.

For this requirement the following threats are relevant:

- Unnoticeable feasible attacks
3.20 Third Party Access

Introduction

General Note
This security requirement has been prepared based on the provisions of the Group Policy on IT/NT Security.
The security requirement is used as a basis for an approval in the PSA process, among other things. It also serves as an implementation standard for provisions of the Group Policy on IT/NT Security in units which do not participate in the PSA process. These requirements shall be taken into account from the very beginning, including during the planning and decision-making processes.
When implementing these security requirements, the precedence of national, international and supranational law shall be observed.

Scenarios
This document deals with eight different categories of interface for third parties.

<table>
<thead>
<tr>
<th>Scenario</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application use</td>
<td>User access to applications in networks</td>
</tr>
<tr>
<td>Application development</td>
<td>Software development in environments</td>
</tr>
<tr>
<td>Testing</td>
<td>Testing of IT systems in environments</td>
</tr>
<tr>
<td>Production support/maintenance</td>
<td>IT system/software/hardware support/operation for/of production systems</td>
</tr>
<tr>
<td>Operation</td>
<td>Operation of systems</td>
</tr>
<tr>
<td>Monitoring</td>
<td>Hardware/software monitoring of internal systems</td>
</tr>
<tr>
<td>Business-to-business content</td>
<td>B2B interfaces with content providers</td>
</tr>
<tr>
<td>Business-to-business service</td>
<td>Generic B2B interfaces (file/data transfer, etc.)</td>
</tr>
</tbody>
</table>

Table 1 – Interface categories for third party access

Technical Requirements

System Hardening

Req 1 Unused services and protocols must be deactivated.

After installation of systems and software products typically local or remote reachable services and protocols are active, which are not necessarily needed for operation and functionality of the system. Such services and protocols must be completely disabled on the system. Additionally it is important that a deactivation survives a system reboot.
This kind of system hardening must be done before the system is reachable from the network. Otherwise an attacker has the possibility to attack and maybe compromise the unsecured system.

Motivation: Services and protocols that are not required for system operation increase the potential attack surface and thus the risk of the system being compromised. This risk is further increased by the fact that a security inspection and an appropriate optimisation of the configuration for unused services and protocols will not be done.

For this requirement the following threats are relevant:
• Unauthorized access to the system
• Unauthorized access or tapping of data
• Unauthorized use of services or resources
• Disruption of availability
• Attacks motivated and facilitated by information disclosure or visible security weaknesses

Req 2 The reachability of services must be restricted.

Typically services that are enabled in the basic configuration are accessible over all interfaces of the system and can be reached from systems in connected networks. This availability is often not needed or meaningful for system functioning. For this reason, services should only be enabled on interfaces where their usage is required. On interfaces where services are active, the reachability must be limited to legitimate communication peers. This limitation must be realized on the system itself (without measures (e.g. firewall) at network side).

Motivation: Disabling services on interfaces which do not require system accessibility or by limiting the reachability can greatly reduce the potential vulnerabilities offered to an attacker. For example, access to a system via SSH from the Internet is not necessary. If this service could be accessed from the interface connected to the Internet, this would greatly increase the risk of attacks on the service.

For this requirement the following threats are relevant:
• Unauthorized access to the system
• Unauthorized use of services or resources
• Disruption of availability
• Attacks motivated and facilitated by information disclosure or visible security weaknesses

Req 3 Unused software must not be installed or must be uninstalled.

During installation of a system often software components will be installed or parts of software will be activated which are not needed for the operation or functionality of the system. This includes also parts of a software, which will be in-stalled as examples but typically not be used (e.g. default web pages, example databases, test data). Such compon-ents should not be installed or must be deleted after installation.

Motivation: Vulnerabilities in software of a system offer an attack window for attackers to infiltrate the system. Uninstalling components that are not required can therefore reduce the possibility of a successful compromise of the system.

For this requirement the following threats are relevant:
• Unauthorized access to the system
• Disruption of availability

Req 4 Unused functions of the operated software and hardware must be deactivated.
3.20 Third Party Access

During installation of software and hardware often functions will be activated that are not necessarily needed for operation or function of the system. Functions of software are currently inherent part which could not be deleted or deinstalled individually. Such functions must be deactivated in the configuration of the system permanently.

Beside the functions of the software also hardware functions are active which are not necessary for a system. Functions like unused interfaces must permanently deactivated. Permanent means that they must not be reactivated again after system reboot.

*Motivation:* The hardware or software of a system often contains functions which are not used and so will be a risk for system security. Such functions give an attacker the possibility to manipulate the system. Furthermore it is possible to get unauthorized access other areas or data of the system. An example is a debugging function in software which can be used for troubleshooting but must not be activated during normal operation. Or a hardware interface that will not be used and so is unsecured allows possibly unauthorized access to the system.

For this requirement the following threats are relevant:

- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Disruption of availability

---

Req 5 Software and hardware components that are no longer supported by vendor, producer or developer must not be used.

Components that have reached end-of-life or end-of-support must not be used. Excluded are components that have a special support contract. This contract must guarantee the correction of vulnerabilities over components life-time.

*Motivation:* Hardware and software components that have reached end of life or end of support represent a risk for a system. This means that a vendor does not supply remedial updates or patches for a component should errors or vulnerabilities occur. This means that vulnerabilities cannot be fixed when they occur and could be exploited to compromise the system or to impair its availability.

For this requirement the following threats are relevant:

- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Disruption of availability

---

Req 6 Known vulnerabilities in software and hardware of the system must be fixed or protected.

Prior to installation of a software or hardware component, users must check whether any vulnerability has been discovered and published for the version they are installing. Any component that proves to have a vulnerability must not be installed or used. Excepted from this rule are components for which the vendor has already provided a measure to remedy the vulnerability, e.g. a patch, update or workaround. In this case, the additional measure must be implemented on the system. Furthermore it is a ongoing process during the complete life cycle of the system to fix upcoming vulnerabilities promptly.

*Motivation:* Publication of vulnerabilities increases the risk of successful exploitation by an attacker, especially since the published information usually includes details on how to exploit
the vulnerability and tools that make exploitation possible. One example of this is when Web servers are compromised due to not patched vulnerabilities.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Disruption of availability

Architecture

**Req 7** All connections must be terminated in a DMZ.

There must be no direct connection between a third party (3P) and a target system or vice versa.

An access platform for third parties in a DMZ (3rd party access platform, 3PAP) must have the functions necessary for fulfilling the security requirements.

*Motivation: There must be an effective control point for implementing the security requirements.*

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources
- Disruption of availability

**Req 8** The network segregation of the 3PAP and adjacent services must follow their different functional elements.

The individual functional elements must be contained in different (V)LANs protected by a firewall gateway.

*Motivation: Attacks targeted at a specific functional element remain restricted to its security zone (implemented via a (V)LAN).*

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources
- Disruption of availability

**Req 9** The 3PAP must be protected by stateful firewall gateways.

*Motivation: State-of-the-art network layer protective mechanisms contribute to the overall security of the 3PAP.*

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources
- Disruption of availability
3.20 Third Party Access

Req 10  The system must check every file exchange processed via the 3PAP using a malware scanner.

Motivation: Such a review shall prevent the infiltration of malware in both directions.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources
- Disruption of availability

Req 11 Only visual presentation protocols such as RDP, ICA or RFB (with regard to third parties) must be permitted by the 3PAP.

Application protocols must not be directly available. This means that applications must run within the visual presentation window. This can be achieved using, for example, Windows Terminal Services, Citrix Metaframe, VNC or Oracle Secure Global Desktop.

In the case of hardware/software monitoring of internal systems or B2B interfaces, visual presentation protocols can be dispensed with.

This requirement is only applicable in the following scenarios:
- Application use: User access to applications
- Operation: Operation of systems
- Production support/maintenance: IT system/software/hardware support/maintenance for production systems
- Application development: Software development in environments
- Testing: Testing of IT systems in environments

Motivation: The visual presentation constitutes an abstraction layer which protects the integrity of the client software and its environment.

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data

Req 12  Connections with command line protocols must be protected using an application layer proxy.

In certain scenarios, a command line protocol must be used between the third party and the 3PAP instead of the graphic representation. In such a case, the connection must be protected by an application layer proxy for the relevant protocol. The application layer proxy must provide required security enforcement features. This is the basis for fulfilling all further requirements for the third party access.

Motivation: A holistic protection can only be achieved by controlling all access channels.

Implementation example: Use of the BalaBit Shell Control Box (non-transparent and bastion mode).

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources
- Disruption of availability
Req 14 A set of security monitoring measures tailored to the specific design of the 3PAP must be implemented.

The required security measures can be composed of a combination of different technologies specific to the 3PAP architecture (e.g. IDS, flow data analysis, dedicated checkscripts).

**Motivation:** Intrusions on the 3PAP should be identified promptly.

For this requirement the following threats are relevant:
- Denial of executed activities
- Unnoticeable feasible attacks

Req 15 The system must be robust against overload situations.

A system must provide security measures to deal with overload situations. In particular, partial or complete impairment of system availability must be avoided. Potential protective measures include:
- Restricting of available RAM per application
- Restricting of maximum sessions for a Web application
- Defining the maximum size of a dataset
- Restricting CPU resources per process
- Prioritizing processes
- Limiting of amount or size of transactions of an user or from an IP address in a specific time range

**Motivation:** An attacker can effect systems availability through targeted exploitation of vulnerabilities in resources administration. An attacker can, for example, send bulk queries to a Web server and, by doing so, compromise its availability if the maximum number of permissible Web sessions is not restricted.

For this requirement the following threats are relevant:
- Disruption of availability

Req 16 If an overload situation cannot be prevented, the system must act in a predictable way.

A system must be built in this way that it can react on a overload situation in a controlled way. However it is possible
that a situation happens where the security measures are not longer sufficient.

In such case it must be ensured that the system cannot reach an undefined and thus potentially insecure state. In an extreme case this means that a controlled system shutdown is preferable to uncontrolled failure of the security functions and thus loss of system protection.

Motivation: With denial-of-service attacks can an attacker try to overload a system to effect its availability or integrity. Unpredictable acting of the system is then a risk for functionality and data and possibly also for other systems.

For this requirement the following threats are relevant:
- Disruption of availability

---

**Req 17** The system must be robust against unexpected input.

During transmission of data to a system it is necessary to validate this before processing. This includes all data which are send to the system. Examples for this are user input, values in arrays and content in protocols. The following typical implementation mistakes must not be done:

- No validation on the lengths of transferred data
- Incorrect assumptions about data formats
- No validation that received data complies with the specification
- Insufficient handling of protocol errors in received data
- Insufficient restriction on recursion when parsing complex data formats
- White listing or escaping for inputs outside the values margin

Motivation: An attacker can try to put a system in an unsecure state through targeted manipulation of transmitted data. The object of such an attack is to compromise the usability, availability or integrity of individual services or of the entire system. For instance a unclean memory handling can lead to a buffer overflow that allows an attacker to execute arbitrary code on the effected system.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized use of services or resources
- Disruption of availability

---

**Authentication and Role Based Access**

**Req 18** System functions with a need of protection must not be used or accessed without successful authentication and authorization.

The usage of a system functions or access of data classified as internal, confidential or strictly confidential must only be possible unambiguous user identification and successful authentication on basis of the user name and at least one authentication attribute. Excepted from this are functions for public use such as those for a Web server on the Internet, via which information is made available to the public. Examples for functions which require a prior authentication are network services (like SSH, SFTP, Web services), local access via a management console, local usage of operating systems and applications. The following examples are possibilities that could be used for authentication.

- Query user name and password
- Use of cryptographic keys and certificates (e.g. as Smartcard)
This requirement must also be applied to accounts that are only used for communication between systems (M2M).

**Motivation:** The authentication is necessary to doubtless identify a user because the allocated authorization, and therefore the access on data and services of the system depends on that.

For this requirement the following threats are relevant:

- Unauthorized access to the system
- Unauthorized use of services or resources
- Denial of executed activities

**Req 19**  Accounts must be used that allow unambiguous identification of the user.

Users must be identified unambiguously by the system. This can typically be reached by using a unique account per user. So named group accounts, i.e. the use of one account for several persons, must not be used. On exception of this requirement are so named machine accounts. These will be used for authentication and authorization from system to each other or for applications on a system and can’t be assigned to a single person. Such accounts must be assigned on a per system or per application basis. In this connection, it has to be guaranteed that this account can’t be misused. Possibilities to protect these accounts are:

- Configuring of a Password that fulfills the security requirements and is known by less than possible circle of administrators.
- Configuring the account that only a local use is possible and an interactive login isn’t possible.
- Use of a technique for authentication of the specific account with public and private key or certificates.
- Limiting the access over the network for legitimized systems.

Additional solution must be checked on their usability per individual case.

**Motivation:** Unambiguous user identification is a prerequisite for assigning a user the rights that he requires to perform his tasks on the system. This is the only way to adequately control access to system data and services and to prevent misuse. Furthermore, it makes it possible to log activities and actions on a system and to assign them to individual users.

For this requirement the following threats are relevant:

- Unauthorized access to the system
- Unauthorized use of services or resources
- Denial of executed activities

**Req 20**  Accounts with extensive rights must be protected with two authentication attributes.

Accounts, for example used for administration, maintenance and troubleshooting, have extensive rights. Therefore a single protection (e.g. a password), as for normal user accounts with less rights, is not suitable. To get a higher protection level it is necessary to use more than one authentication attribute. For this a combination of an attribute that the user knows and an attribute that the user owns will be used often. This kind of authentication will be named as 2-factor authentication. Examples for 2-factor authentication are:

- Smartcard with PIN
- Private key with Passphrase
- Secure-ID Token with Password
Motivation: Accounts with extensive rights as used for system administration have a higher risk for system’s security. An attacker can get extensive rights by compromising such an account to get access to wide parts of the system and stored data.

For this requirement the following threats are relevant:
• Unauthorized access to the system
• Unauthorized use of services or resources
• Denial of executed activities

Req 21 Third party users or business interfaces must be authenticated using a sufficiently secure method.

A 2-factor authentication must be implemented for user interfaces. For system-to-system interfaces, a bidirectional, key-based authentication is required.

Motivation: Only authorized users from a third party can gain access to the target systems.

For this requirement the following threats are relevant:
• Unauthorized access to the system

Req 22 If a password is used as an authentication attribute, it must have at least 8 characters and contain three of the following categories: upper cases, lower case, numbers and special characters.

A system may only accept passwords that comply with the following complexity:
• Minimum length of 8 characters.
• Comprising at least three of the following categories: upper/lower case letters, numbers and special characters

When a password is assigned, the system must ensure that the password meets these requirements. If a central system will be used for user authentication this function can be forwarded or delegated to this system.

Motivation: Passwords with the above complexity offer high robustness against attacks coupled with acceptable user friendliness. Passwords with this level of complexity have proved their efficiency in practice. Trivial passwords that are too short are susceptible to brute force and dictionary attacks and are therefore easy for attackers to determine. Once a password has been ascertained it can be used by an attacker for unauthorized access to the system and the data on it.

For this requirement the following threats are relevant:
• Unauthorized access to the system
• Unauthorized use of services or resources
• Denial of executed activities

Req 23 If a password is used as an authentication attribute, users must be enable to independently change the password anytime.

The system must offer a function that enables a user to change his password at any time.

When an external centralized system for user authentication will be used it is possible to redirect or implement this function on this system.

Motivation: The fact that a user can change his authentication attribute himself at any time enables him to change it promptly if he suspects that it could have been accessed by a third party.

For this requirement the following threats are relevant:
• Unauthorized access to the system
• Unauthorized use of services or resources
• Denial of executed activities
Req 24 If a password is used as an authentication attribute, a protection against brute force and dictionary attacks that hinder password guessing must be implemented.

Brute force and dictionary attacks aim to use automated guessing to ascertain passwords for user and machine accounts. Various measures or a combination of this measures can be taken to prevent this.

The most commonly used protection measures are:
- Increasing the delay (e.g. doubling wait times at each attempt) for each renewed password input following an incorrect entry (“tar pit”).
- Blocking an account following a specified number of incorrect attempts (typically 5). However has to take in account that this solution needs a process for unlocking and an attacker can force this to deactivate accounts and make them unusable.
- Using CAPTCHA to prevent automated attempts (often used for Web applications).

In order to achieve higher security, it is often meaningful to combine two or more of the measures named here. This must be evaluated in individual cases and implemented accordingly.

Motivation: *Implementation of one or more of the above measures can prevent successful execution of brute force and dictionary attacks to ascertain passwords.*

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized use of services or resources
- Denial of executed activities

Req 25 If a password is used as an authentication attribute, they must be hidden when displayed on screen.

When a user enters a password or the password is otherwise displayed on screen, it must be made unreadable. Typically, the individual characters of the password are replaced by a character such as “*”. Under certain circumstances it may be permissible for an individual character to be displayed briefly during input. Such a function is used, for example, on Smartphone’s to make input easier. However, the entire password is never output to the display in plaintext.

Motivation: *To prevent another person reading a password by chance or intentionally on screen during input, the password must be rendered unreadable during input or display.*

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized use of services or resources
- Denial of executed activities

Req 26 The system must be connected to a central system for user administration.

Accounts and their rights will be administrated on central identity management systems. For provisioning of these authorization information must the system provide a central inter-face (e.g. LDAPs for authorization, Kerberos for authentication, locking information for certificates) or decentralized mechanisms (e.g. public-key authentication). A central solution for identity management must be preferred.
In areas where a central identity management system is not available a central system such as LDAP, TACACS+ or Radius server for the administration of accounts and their authentication and authorization must be used.

**Motivation:** Central administration of identity of accounts and their rights means that they only have to be maintained once instead of separately on each system. From the aspect of security, the advantage is that an account and its rights only known on a single central side. This information can be transmitted from a central side to systems (provisioning), central administrated (reconciliation) and central deleted (deprovisioning). This reduces the risk of accounts being forgotten during changing or deletion since they are configured on multiple systems. This could give a user wrong system rights or continued access to a system.

For this requirement the following threats are relevant:

- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources
3.20 Third Party Access

Req 27 Predefined accounts must be deleted or disabled.

Many systems have default accounts (e.g. guest, ctxsys), some of which are preconfigured with or without known pass-words. These standard users must be deleted or disabled. Should this measure not be possible the accounts must be locked for remote login. In any case disabled or locked accounts must configured with a complex password (12 character and more, use of upper/lower case, numbers and special characters). This is necessary to prevent unauthorised use of such a account in case of misconfiguration.

Exceptions to this requirement to delete or disable accounts are accounts that are used only internal on the system in-volved and that are required for one or more applications on the system to function. Also for this accounts remote ac-cess or local login must be forbidden to prevent a abusive use by users of the system.

Motivation: Standard users are typically generally known and can be used by an attacker for targeted brute force and dictionary attacks. Standard user accounts represent a special risk if they do not use a password or only use a stand-ard password that is generally known. Such standard user accounts can easily be exploited by an attacker in order to gain access to the system involved without being authorized to do so.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized use of services or resources
- Denial of executed activities

Req 28 Predefined authentication attributes must be deleted or disabled.

Normally, authentication attributes such as password or cryptographic keys will be preconfigured from producer, vendor or developer of a system. Such authentication attributes must be changed an own attribute not known by other parties.

Motivation: Authentication attributes like password or cryptographic keys preconfigured from third parties are not trustable. Such authentication attributes can be used to compromise systems or their data.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized use of services or resources
- Denial of executed activities

Req 29 Access by users from the third party must comply with a role-based access model.

The 3PAP must implement a role-based model with regard to the target system and the services offered on the plat-form.

Example: A 3P role effects a technical restriction of access to specific target systems. Other systems are unavailable and thus are not threatened. For example, if the 3PAP offers access to files for the 3P (proxy functions), the access roles define exactly which upload or download directories are available.

Motivation: Tailored access roles technically prevent unauthorized access.

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources
3.20 Third Party Access

Req 30 The authorizations for accounts and applications must be reduced to the minimum required for the tasks they have to perform.

Authorizations to a system must be restricted to a level in which a user can only access data and use functions that he needs in the course of his work. Suitable authorizations must also be assigned for access to files that are components of the operating system or of applications or that are generated by the same (e.g. configuration and logging files).

Alongside access to data, execution of applications and components must also take place with rights that are as low as possible. Applications should not be executed with administrator or system rights.

Motivation: If the rights granted to a user on a system are too broad, it could be possible for him to access data and applications which he is not permitted to view or use. This would give him the opportunity to disclose or modify confidential data and to manipulate system files. Applications with rights that are too broad can be used by a user to extend his own authorizations and thus to gain access to files and system components to which he would not have had access with his authorizations under normal circumstances.

For this requirement the following threats are relevant:

- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources

Req 31 Access to production systems must only be enabled on a temporary basis and must be based on an individual incident or support case.

The 3PAP is generally the entity that technically implements the temporary character of this type of access. Privileged access rights can cover administrative levels.

This requirement is only applicable in the following scenario:

- Production support/maintenance: IT system/software/hardware support/maintenance for/of production systems

Motivation: In this scenario, production systems are subject to high security risks. Access restrictions must therefore be as strict as possible.

For this requirement the following threats are relevant:

- Unauthorized access to the system
Sessions

3.20 Third Party Access

Req 32  Sessions must be protected against high jacking.

A function must be implemented for all user sessions to prevent other users assuming or pursuing control of the sessions of an authorized user. Such protection can be achieved by implementing the following measures or a combination of these measures:

- Using an appropriate cryptographic algorithm
- On network level: Use of TCP protocol (with sequence number) and filter lists.
- On transport level: SSL/TLS
- On application level: Negotiation of a random, secret value between sender and recipient (e.g. session ID, sequence number, time stamp)

Motivation: An attacker who gains knowledge about a session which is not protected against assumption of control can continue to participate in such a session and thus gain unauthorized access to the system involved. One example is an attacker who records the session cookie for a Web application during non-encrypted communication. The attacker can then view and take control of the session of the original user.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources
- Denial of executed activities

Req 33  It must be possible that users can logoff their session.

The system must have a function that allows a signed in user to logout at any time.

Motivation: A user of a system must have the possibility to protect a session and therefore its data against unauthorized access. Therefore a logout function must be available that can be used to end a session. This prevents that the current session will stay open and can be reactivated and used by an unauthorized person.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources
- Denial of executed activities

Req 34  If no activity is detected on a user/business interface for a specific period, the relevant session must close automatically.

According to expectations, a typical timeout period for a session is less than an hour.

Motivation: Inactive sessions are at risk of being hijacked or misused.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data

Req 35  A 3P session must be limited to a fixed time window.

If necessary, the time frame can be extended in a controlled way. According to expectations, typical time frames are less than 12 hours.

Motivation: Limited time frames minimize the attack surface.
For this requirement the following threats are relevant:
• Unauthorized access to the system
• Unauthorized access or tapping of data
• Unauthorized modification of data

Req 36  An employee must be able to close a 3P session at any time.

For this requirement the following threats are relevant:
• Unauthorized access to the system

Req 37 Specified connection rules must define all permitted connections from the third party to the 3PAP and from the 3PAP to the target system(s) (individually for 3P units and 3P user/business inter-faces).

The connection rules are configured using configuration service programs. The connection rules restrict the possible connections between 3P, 3PAP and the target system(s).

Motivation: It is important to limit 3P interactions in accordance with the need-to-know principle.

Implementation example: Connection rules are usually a combination of definitions on various devices of the 3PAP and a surrounding firewall gateway.

For this requirement the following threats are relevant:
• Unauthorized access to the system

Encryption

Req 38 Data with need of protection must be secured against unauthorized viewing and manipulation during transmission and storage.

Adequate security measures for transmission and storage must be implemented of data with a need for protection that are classified as internal, confidential or strictly confidential. The chosen measure depends on the classification for the data and other factors such as the type of network used during transmission, the storage location for data, etc. Furthermore must be guaranteed that confidential and strictly confidential data will not be unprotected during temporary storage (e.g. in web cache, temporary folders).

All authentication data such as user names, passwords, PINs, etc. must be protected against unauthorized viewing and manipulation. This applies equally to permanent storage and transmission. Typical measures taken to protect authentication data or combinations of these are:

For storage:
• Client systems: encryption or obfuscation of authentication data, no persistent storage, limitation of access rights.
• Server systems: hashing of authentication data with PBKDF like scrypt or bcrypt or when this is not possible hashes with salt.

For transmission:
• Usage of cryptographically protected network protocols.
• Challenge response method (Disadvantage of this solution is that passwords needed in clear-text on server. If this solution is feasible depends on the individual threat scenario)
3.20 Third Party Access

Files of a system that are needed for the functionality must also be protected against manipulation. This is necessary because system’s integrity can be damaged when the system access this kind of files. An example is the use of check-sum or cryptographic methods to validate if e.g. firmware images, patches, drivers or kernel modules are free of manipulations.

For transmission of data with a need of protection network protocols that are insecure due to insufficient security measures shall not be used. Examples are: SSLv3, SSHv1, FTP, Telnet, SNMPv1 and 2c. In case of these protocols a newer version without vulnerabilities or a secure alternative must be used.

Motivation: If data with a need of protection will not be secured an attacker could record or manipulate the data during transmission over a network. An example is the recording of user names and passwords during system administration with the telnet clear-text protocol. Storing data on a system without adequate protection may mean that unauthorized users can copy or modify it. One example is when passwords can be read out when they stored in an inadequate secured way (e.g. usage of unsecure hashing algorithms like MD5 or SHA-1) or even without encryption on a system or manipulation of firmware to affected the system integrity.

For this requirement the following threats are relevant:
• Unauthorized access or tapping of data
• Unauthorized modification of data

---

Req 39 Only secure protocols must be used for communication via business interfaces.

Machine-to-machine connections are understood to include B2B interfaces. Sufficiently secure protocols include, for example, SSL/TLS envelopes, SFTP, SNC.

This requirement is only applicable in the following scenarios:
• Monitoring: Hardware/software monitoring of internal systems
• Business-to-business content: B2B interfaces with content providers
• Business-to-business service: Generic B2B interfaces (file/data transfer, etc.)

Motivation: There are many B2B communication protocols; security functions must be an integral part of these protocols (generally in the form of authentication, encryption and integrity protection).

For this requirement the following threats are relevant:
• Unauthorized access or tapping of data
Logging and Monitoring

Req 40 Information with need of protection must not be contained in files, outputs or messages that are accessible by unauthorized users.

Information with need of protection must not be accessible in files, outputs or messages of the system by unauthorized users. This includes information relating to the operating system, used middleware or applications such as vendor, product name, product identifier, installed software versions, installed service packs, patches, hot fixes and serial numbers. Examples for system messages which must be free of sensitive data are:

- Comments in downloadable files
- Error and system messages
- Stack traces
- Network protocols
- Login windows and dialogs

Furthermore, details of implementation and information relating, e.g., to backend software/systems, function calls, SQL instructions or structure of database, must not be contained in error messages.

Excluded from this are displays and outputs that can be viewed and retrieved by authorized users who are logged in.

In addition, an internal transfer of system internal information for error analysis is allowed in an adequate dimension. In this case the continuative regulations or guidelines (e.g. of data privacy) must be noticed.

Motivation: The information named above can be used by an attacker to prepare specific attacks on a system. In this way an attacker could, for example, use the precise software version to identify vulnerabilities in the product and, in a second step, exploit them.

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data
- Attacks motivated and facilitated by information disclosure or visible security weaknesses

Req 41 Security relevant events must be logged with a precise timestamp and a unique system reference.

Systems must log the occurrence of security-relevant incidents. So that these events can be evaluated and classified, they must be logged together with a unique system reference (e.g., host name, IP or MAC address) and the exact time the incident occurred. Furthermore, legal and data privacy regulations (e.g. time of storage of logging data) must be proved and followed.

The following table lists events that are relevant to security and the corresponding data that typically has to be logged by a system. Exceptions are systems for which no or only restricted logging applies. Examples of such systems are customer devices such as Smartphone’s or IADs/Homegateways (e.g. Speedport).

Logging must be done considering the currently valid legal, wage and company regulations. This regulations state among others that logging of events can be done only earmarked. Logging of events for doing a work control of em-ployees is not allowed.
Typical event that reasonable should be logged in many cases are:

<table>
<thead>
<tr>
<th>Event</th>
<th>Event data to be logged</th>
</tr>
</thead>
<tbody>
<tr>
<td>Incorrect login attempts</td>
<td>• Account,</td>
</tr>
<tr>
<td></td>
<td>• No. of failed attempts,</td>
</tr>
<tr>
<td></td>
<td>• Source (IP address) of remote access</td>
</tr>
<tr>
<td>System access with accounts with administrator</td>
<td>• Account,</td>
</tr>
<tr>
<td>rights</td>
<td>• Access timestamp,</td>
</tr>
<tr>
<td></td>
<td>• Length of session,</td>
</tr>
<tr>
<td></td>
<td>• Source (IP address) of remote access</td>
</tr>
<tr>
<td>Account administration</td>
<td>• Administrator account,</td>
</tr>
<tr>
<td></td>
<td>• Administered account,</td>
</tr>
<tr>
<td></td>
<td>• Activity performed (configure, delete, enable and disable)</td>
</tr>
<tr>
<td>Change of group membership for accounts</td>
<td>• Administrator account,</td>
</tr>
<tr>
<td></td>
<td>• Administered account,</td>
</tr>
<tr>
<td></td>
<td>• Activity performed (group added or removed)</td>
</tr>
<tr>
<td>Critical rise in system values such as disk</td>
<td>• Value exceeded,</td>
</tr>
<tr>
<td>space, CPU load over a longer period</td>
<td>• Value reached</td>
</tr>
<tr>
<td></td>
<td>(Here suitable threshold values must be defined depending on the</td>
</tr>
<tr>
<td></td>
<td>individual system.)</td>
</tr>
</tbody>
</table>

Logging of additional security-relevant events may be meaningful. This must be verified in individual cases and implemented accordingly where required.

Motivation: Logging security-relevant events is a basic requirement for detecting ongoing attacks as well as attacks that have already occurred. This is the only way in which suitable measures can be taken to maintain or restore system security. Furthermore, the logging data is used as evidence so that legal steps can be taken against attackers.

For this requirement the following threats are relevant:
- Denial of executed activities
- Unnoticeable feasible attacks

Req 42 Production support sessions must be recorded consistently.

Recording means capturing the visual content and/or text content of a support session, including all meta data and transmitted files (binary recording). The result is generally a film that shows the session in real time with all 3P interactions, and copies of the transmitted files.

This requirement is only applicable in the following scenarios:
- Production support/maintenance: IT system/software/hardware support/maintenance for/of production sys-tems

Motivation: Due to their critical importance, a detailed audit trail is required for sessions held in the course of production support.

For this requirement the following threats are relevant:
- Denial of executed activities
Req 43  The system must log every successful/failed attempt to access the 3PAP.

*Motivation: In the event of security incidents, audit trails provide important information.*

For this requirement the following threats are relevant:

- Denial of executed activities
- Unnoticeable feasible attacks

---

Req 44  The session log must contain defined parameters for successful access attempts to the 3PAP.

These are the defined parameters:

- Date, timezone and time of session initiation and session termination
- Source object (3P) name
- Source object (3P) IP address
- 3P account name (user)/ID of business interface
- Destination object (target system/application) name and IP address
- Destination service name and TCP port
- Action (session established/session request rejected)

For correct date and time information, all logging systems for a service must be connected to a time synchronization mechanism.

*Motivation: In the event of security incidents, audit trails provide important information.*

For this requirement the following threats are relevant:

- Denial of executed activities
- Unnoticeable feasible attacks

---

Req 45  The session log must provide additional parameters.

The additional parameters are:

- Number of the fault ticket
- Defined time frame
- ID of the initiating representative
- In the case of abnormal termination (e.g., expiry of the time frame): Specification of abnormal termination and the corresponding reason

The majority of this information is registered when a representative creates a session ticket for temporary access for production support.

**This requirement is only applicable in the following scenarios:**

- Production support/maintenance: IT system/software/hardware support/maintenance for/of production systems

*Motivation: In the event of security incidents, audit trails provide important information.*

For this requirement the following threats are relevant:

- Denial of executed activities
- Unnoticeable feasible attacks
3.20 Third Party Access

**Req 46** The 3PAP must attach a digital signature to session recordings when they are created.

**This requirement is only applicable in the following scenarios:**
- Production support/maintenance: IT system/software/hardware support/maintenance for/of production systems

**Motivation: It is important to ensure the authenticity of audit trails.**
For this requirement the following threats are relevant:
- Denial of executed activities

**Req 47** The 3PAP must save session logs in a separate log archive.

Session logs must be transmitted to the log server securely (e.g., via sftp). The interval between protocol transmissions must be as short as possible.

**Motivation: It is important to protect audit trails. Thus they should not remain in at risk components.**
For this requirement the following threats are relevant:
- Denial of executed activities

### Organisational Requirements

**Req 48** Each 3P connection must be registered in a 3P access directory.

The directory can be either central or LBU-specific.

**Motivation: Up-to-date documentation of external connections is a basic requirement for ensuring the security status of the networks.**
For this requirement the following threats are relevant:
- Unnoticeable feasible attacks

**Req 49** Defined details of the 3P connection must be registered.

The directory can be either central or LBU-specific. The following details must be registered:
- Responsible requestor of the 3P connection (name, e-mail address)
- 3P contact information (name of the company, location/address, contact, e-mail address)
- Description of purpose and tasks and reasons for 3P access
- The 3P access platform used for the connection (including the relevant platform applications managers and/or service provider contacts)
- First media access (e.g. IPSec VPN via the Internet, MPLS, dial-in, etc.)
- Target systems and applications
- Protocol(s) used for the system/application access (e.g. SSH, RDP, ICA, HTTPS, etc.)
- Frequency of access

**Motivation: Up-to-date documentation of external connections is a basic requirement for the provision of a continuous security state of the DT networks.**
For this requirement the following threats are relevant:
- Unnoticeable feasible attacks
3.21 Unix Servers

Introduction
This security requirement has been prepared based on the provisions of the Group Policy on IT/NT Security.

The security requirement is used as a basis for an approval in the PSA process, among other things. It also serves as an implementation standard for provisions of the Group Policy on IT/NT Security in units which do not participate in the PSA process. These requirements shall be taken into account from the very beginning, including during the planning and decision-making processes. When implementing these security requirements, the precedence of national, international and supranational law shall be observed.

Unix systems
System hardening

<table>
<thead>
<tr>
<th>Req 1</th>
<th>Unused services and protocols must be deactivated.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>After installation of systems and software products typically local or remote reachable services and protocols are active, which are not necessarily needed for operation and functionality of the system. Such services and protocols must be completely disabled on the system. Additionally it is important that a deactivation survives a system reboot.</td>
</tr>
</tbody>
</table>

This kind of system hardening must be done before the system is reachable from the network. Otherwise an attacker has the possibility to attack and maybe compromise the unsecured system.

Motivation: Services and protocols that are not required for system operation increase the potential attack surface and thus the risk of the system being compromised. This risk is further increased by the fact that a security inspection and an appropriate optimisation of the configuration for unused services and protocols will not be done.

For this requirement the following threats are relevant:
• Unauthorized access to the system
• Unauthorized access or tapping of data
• Unauthorized use of services or resources
• Disruption of availability
• Attacks motivated and facilitated by information disclosure or visible security weaknesses

<table>
<thead>
<tr>
<th>Req 2</th>
<th>The reachability of services must be restricted.</th>
</tr>
</thead>
</table>
|       | Typically services that are enabled in the basic configuration are accessible over all interfaces of the system and can be reached from systems in connected networks. This availability is often not needed or meaningful for system functioning. For this reason, services should only be enabled on interfaces where their usage is required. On interfaces were services are active, the reachability must be limited to legitimate communication peers. This limitation must be realized on the system itself (without measures (e.g. firewall) at network side).

Motivation: Disabling services on interfaces which do not require system accessibility or by limiting the reachability can greatly reduce the potential vulnerabilities offered to an attacker. For example, access to a system via SSH from the Internet is not necessary. If this service could be accessed from the interface connected to the Internet, this would greatly increase the risk of attacks on the service. |
For this requirement the following threats are relevant:

- Unauthorized access to the system
- Unauthorized use of services or resources
- Disruption of availability
- Attacks motivated and facilitated by information disclosure or visible security weaknesses

**Req 3**  Unused software must not be installed or must be uninstalled.

During installation of a system often software components will be installed or parts of software will be activated which are not needed for the operation or functionality of the system. This includes also parts of a software, which will be in-stalled as examples but typically not be used (e.g. default web pages, example databases, test data). Such components should not be installed or must be deleted after installation.

*Motivation: Vulnerabilities in software of a system offer an attack window for attackers to infiltrate the system. Uninstalling components that are not required can therefore reduce the possibility of a successful compromise of the system.*

For this requirement the following threats are relevant:

- Unauthorized access to the system
- Disruption of availability

**Req 4**  Unused functions of the operated software and hardware must be deactivated.

During installation of software and hardware often functions will be activated that are not necessarily needed for operation or function of the system. Functions of software are currently inherent part which could not be deleted or de-installed individually. Such functions must be deactivated in the configuration of the system permanently.

Beside the functions of the software also hardware functions are active which are not necessary for a system. Functions like unused interfaces must permanently deactivated. Permanent means that they must not be reactivated again after system reboot.

*Motivation: The hardware or software of a system often contains functions which are not used and so will be a risk for system security. Such functions give an attacker the possibility to manipulate the system. Furthermore it is possible to get unauthorized access other areas or data of the system. An example is a debugging function in software which can be used for troubleshooting but must not be activated during normal operation. Or a hardware interface that will not be used and so is unsecured allows possibly unauthorized access to the system.*

For this requirement the following threats are relevant:

- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Disruption of availability

**Req 5**  Functions not needed for the operation of a server system must be deactivated.

A server must not perform routing functions; consequently the routing function must be disabled. Additionally the answering routine for broadcast ICMP packages must be disabled. Usually this is already configured correctly out-of-the-box.

*Motivation: The routing functions enable misuse scenarios, meaning that an attack can route malicious packets through the server in connected networks.*

Implementation example: (Linux)

Set `/proc/sys/net/ipv4/ip_forward` respectively
`/proc/sys/net/ipv6/conf/all/forward` to “0”. Set
`/proc/sys/net/ipv4/icmp_echo_ignore_broadcasts` to 1.
For this requirement the following threats are relevant:
• Unauthorized use of services or resources
• Disruption of availability

Req 6 If services cannot be bind to the minimal required interfaces by configuration, a local packet filter must regulate the accessibility of the service.

Sometimes, software cannot be bind to dedicated interfaces. A local packet filter or TCP-wrapper can ensure this.

Motivation: Packet filters offer effective protection in order to prevent misuse services from other networks.

For this requirement the following threats are relevant:
• Unauthorized access to the system
• Unauthorized use of services or resources
• Attacks motivated and facilitated by information disclosure or visible security weaknesses
Req 7 The administration of the operating system must be done via a network interface which is independent from the production network.

Administrative access to a server must not be done via an interface which provides productive services. Access must be limited to legitimate systems. The administration of applications can also be done using this network interface.

The restriction can be done with, e.g., filter mechanisms, local access lists or a packet filter. This limitation has to be done as restrictive as possible, i.e., limit to single IP addresses or at least small IP ranges.

Motivation: In the event of a successful attack, an attacker may gain access to confidential information or even to the entire system. By restricting the accessibility to legitimate systems, the group of potential attackers can be reduced, and thus also the likeliness of a successful attack. Furthermore, systems must be manageable even in the case the customer network is down.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unnoticeable feasible attacks
- Attacks motivated and facilitated by information disclosure or visible security weaknesses

Req 8 Services for administration must be bind to exact one interface.

The administrative services (e.g., SSH, HTTPS, RDP) must be bind to exact one interface. Hence the separation of management traffic from user traffic, this is the IP address in the management network. If the system or parts of it is managed by more than one interface, the management services have to be bind to the lowest possible number.

Motivation: This ensures that it can be clearly foreseen under which address these management services are reachable. In addition, a unique address is important for implementing filters and firewall rules and for checking the authenticity of keys and certificates when using cryptographic procedures to secure management traffic.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unnoticeable feasible attacks
- Attacks motivated and facilitated by information disclosure or visible security weaknesses

Req 9 Network based access used for operating system administration must have integrity, be encrypted and securely authenticated.

Access is only permitted by using secure protocols (e.g., SSHv2, HTTPS, SNMPv3). The administrator must ensure that any network connection between his workstation or a management system and the operating system to be administrated is securely authenticated, encrypted and protected against tampering.

Motivation: If the administrator transmits changes to the configuration settings via unencrypted or unsecure connections, there is a risk that unauthorized parties exploit vulnerabilities. Information could be gained (configuration settings, access IDs, etc.) to exploit additional security vulnerabilities.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Denial of executed activities
Req 10 The IPv4 and IPv6 addresses of all interfaces of a server must be configured statically.

IP addresses providing services must not be changed on external influence, even in the case of an enforced reboot. An automatic assignment of IP addresses, e.g., using DHCPv4/v6 or IPv6 auto-configuration is permitted only in the case when deactivated after initial allocation or secured otherwise. IPv6 router advertisements must be ignored.

It is recommended to form the host share of the IPv6 addresses randomly. Due to the very large address space of IPv6, this way it is very time-consuming for an attacker to use scans to discover systems.

Motivation: Avoidance of bypassing authorized systems by man-in-the-middle attacks: The automatic configuration of IP addresses enables an attacker to divert traffic or impair communication.

For this requirement the following threats are relevant:

- Unauthorized access or tapping of data
- Disruption of availability
Req 11  Kernel based network functions not needed for the operation as a server must be deactivated.

Routing functions are not needed on a server; consequently the routing function must be disabled. Additionally the answering routine for broadcast ICMP packages must be disabled. Usually this and other network features are already configured correctly out-of-the-box.

Motivation: The routing functions enable misuse scenarios, meaning that an attack can route malicious packets through the server to connected networks.

For this requirement the following threats are relevant:
- Disruption of availability
- Unnoticeable feasible attacks

Req 12  Secure Shell (SSH) must be the only service allowing a remote administration.

Motivation: This requirement reduces the attack surface to a minimum and enables a secure control of the administrative accounts.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized use of services or resources

Req 13 If available, the function for “rp_filter” (reverse path filter) or the corresponding function of the utilized distribution must be enabled. Likewise, “strict destination multihoming” must be enabled.

Motivation: “rp_filter” filters out incoming packets containing a source address for which the interface over which the packets arrive has no return route. “Strict destination multihoming” (configurable in Linux in the proc file system under “arp_ignore” and “arp_announce”) prevents packets being routed to any address other than that configured at the incoming interface. These two functions shall be taken into special account on all systems with more than one network interface.

Implementation example: Administrators configure rp_filter in Linux by using the switch /proc/sys/net/ipv4/conf/all/rp_filter. This switch can be set, depending on the distribution, in a system-wide configuration file. The switch for arp_announce and arp_ignore is located under /proc/sys/net/ipv4/conf/all/.

For this requirement the following threats are relevant:
- Disruption of availability
- Unnoticeable feasible attacks
Req 14  The automatic launch of applications on removable media must be deactivated.

Removable media such as CD-, DVD-, USB-Sticks or USB-Storage drives shall not automatically start any applications they contain.

Motivation: Automatic application launch could inadvertently launch malware.

For this requirement the following threats are relevant:
• Unnoticeable feasible attacks

Req 15  Support for Trusted Hosts must be disabled.

Motivation: Trusted host relationships make it possible to use or provide access to various system resources without adequate authentication. Trusted hosts relationships therefore present major attack surface.

Implementation example: The shall be not .rhosts file in the filesystem.

For this requirement the following threats are relevant:
• Unauthorized use of services or resources
• Disruption of availability
• Unnoticeable feasible attacks

Req 16  Protection from buffer overflows must be enabled.

Some Unix systems offer protection from buffer overflows within the operating system, which is usually enabled by de-fault on most Unix systems. The administrator shall enable this operating system function or leave it enabled.

Motivation: Buffer overflows are major attack vectors and shall be intercepted at the earliest possible point, i.e., at operating system level.

Implementation example: Solaris: Set the following lines in the /etc/system file
  set noexec_user_stack=1
Linux: Set the following lines in the file /etc/sysctl.conf
  kernel.exec-shield = 1
  kernel.randomize_va_space = 1

For this requirement the following threats are relevant:
• Unauthorized use of services or resources
• Disruption of availability
• Unnoticeable feasible attacks

Req 17 Application and system logfile must be stored on a dedicated file system, which has been solely created for this purpose.

A separate file system must be created for directories containing log files.

Motivation: There are various possibilities, why a server creates a huge amount of log data. Storing the log files in a separate file system should ensure, that no file systems relevant for the availability or data integrity of the server itself will be flooded and then blocks the operation of the server.
For this requirement the following threats are relevant:

- Disruption of availability
- Denial of executed activities
- Unnoticeable feasible attacks

---

**Req 18** Passwords that are stored on the system must be stored in such a way that unauthorized persons cannot access them.

Passwords must be stored only as hashes (bcrypt, scrypt), never in clear text. Files containing password hashes must be protected against unauthorized access.

*Motivation:* Passwords are in need of protection that only account owners or authorized persons may know and change. This measure is designed to ensure that unauthorized persons cannot gain knowledge of these passwords or have the chance to change them.

Implementation example: For system passwords, the file `/etc/shadow` shall be used in Linux. For other operating systems, the respective equivalent file shall be used, which is only readable for the root and only contains the hashes of the system passwords.

For this requirement the following threats are relevant:

- Unauthorized access to the system
- Denial of executed activities
- Unnoticeable feasible attacks

---

**Req 19** The path variable of all accounts must not contain the current directory `"."`.

System commands have to be started based on a relative or absolute path or can be found in the current PATH variable.

*Motivation:* Malicious / unwanted code may be executed based on the fact, that the user is e.g. in `/temp` directory containing malicious codes.

Implementation example: Check: Enter `echo $PATH` and check the output of the command.

For this requirement the following threats are relevant:

- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources
- Attacks motivated and facilitated by information disclosure or visible security weaknesses

---

**Req 20** Configuration files read by processes with root permissions, executable files executed with root permissions (e.g. via cron, "init"-scripts etc.) and directories containing these files must be protected in such a way, that only root users can change these files.

Executable files in system directories must be generally only writeable for root users. Especially this is valid for system directories such as `/bin`, `/sbin`, `/usr`, `/root`, `/etc`, `/lib`, `/var`. For subdirectories it may be necessary to make them accessible for other users / groups.

*Motivation:* A file with general write permissions can be misused / changed in such a way, that a root user executes code from an attacker, which leads to a privilege escalation.

Implementation example: All relevant files must be changed with `chmod go-w`. 
For this requirement the following threats are relevant:
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources
- Denial of executed activities

---

**Req 21** When creating a file, the file mode must be set in such a way, that a modification is only possible by the owner or the respective group.

*Motivation:* The blocking of manipulations of files from non authorized persons, the access permissions of files have be set very strictly to avoid misuse.

Implementation example: Setting the umask on 027 for all accounts achieves this requirement, i.e. files are writable for the owner and only readable for the group.

The operations and testing department must take care, that permissions of files within deployment/software packages are set correctly by the supplier.

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data

---

**Req 22** All processes must be started with the minimal necessary permissions.

No application, except for cron, syslog and some kernel processes, are allowed to run with UID 0.

*Motivation:* This requirement is necessary to block (un)intended manipulations of the system. Additionally the possibility of a privilege escalation can be reduced.

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data
- Unauthorized use of services or resources

---

**Req 23** There must be no executable files with SUID root.

The SUID bit of files, which belongs to the root user, has to be removed. This is especially valid for SUDO. If there exists situations where the SUID bit cannot be removed, the security department must approve this issue.

*Motivation:* The SUID-Bit enables a user to execute a file / process with different / higher privileges. Therefore the attack surface will be increased, if users do not act careful.

Implementation example: SUID bit files can be found with „find / -type f -perm +6000“ and removed with „chmod –s“.

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data
- Unauthorized use of services or resources

---

**Req 24** The mount points for external data storage media which can be mounted by users without root permissions must be assigned the options “nodev” and “nosuid”.

In Linux systems, administrators shall set the options nodev and nosuid in the `/etc/fstab` for all filesystems, which also have the “user” option.

*Motivation:* The aim is to effectively prevent an unauthorized extension of privileges for users, who can mount external data storage mediums. This option prevents users from misusing an
external data storage medium with alternative “dev” files and adapted permissions or executable files, which enable an extension of privileges, set by the SUID bit.

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources

**System Update**

Req 25 Software and hardware components that are no longer supported by vendor, producer or developer must not be used.

Components that have reached end-of-life or end-of-support must not be used. Excluded are components that have a special support contract. This contract must guarantee the correction of vulnerabilities over components life-time.

Motivation: Hardware and software components that have reached end of life or end of support represent a risk for a system. This means that a vendor does not supply remedial updates or patches for a component should errors or vulnerabilities occur. This means that vulnerabilities cannot be fixed when they occur and could be exploited to compromise the system or to impair its availability.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Disruption of availability

Req 26 Known vulnerabilities in software and hardware of the system must be fixed or protected.

Prior to installation of a software or hardware component, users must check whether any vulnerability has been discovered and published for the version they are installing. Any component that proves to have a vulnerability must not be installed or used. Excluded from this rule are components for which the vendor has already provided a measure to remedy the vulnerability, e.g. a patch, update or workaround. In this case, the additional measure must be implemented on the system. Furthermore it is an ongoing process during the complete life cycle of the system to fix upcoming vulnerabilities promptly.

Motivation: Publication of vulnerabilities increases the risk of successful exploitation by an attacker, especially since the published information usually includes details on how to exploit the vulnerability and tools that make exploitation possible. One example of this is when Web servers are compromised due to not patched vulnerabilities.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Disruption of availability
Protection of Data and Information

Req 27 Information with need of protection must not be contained in files, outputs or messages that are accessible by unauthorized users.

Information with need of protection must not be accessible in files, outputs or messages of the system by unauthorized users. This includes information relating to the operating system, used middleware or applications such as vendor, product name, product identifier, installed software versions, installed service packs, patches, hot fixes and serial numbers. Examples for system messages which must be free of sensitive data are:

- Comments in downloadable files
- Error and system messages
- Stack traces
- Network protocols
- Login windows and dialogs

Furthermore, details of implementation and information relating, e.g., to backend software/systems, function calls, SQL instructions or structure of database, must not be contained in error messages. Excluded from this are displays and outputs that can be viewed and retrieved by authorized users who are logged in. In addition, an internal transfer of system internal information for error analysis is allowed in an adequate dimension. In this case the continuative regulations or guidelines (e.g. of data privacy) must be noticed.

Motivation: The information named above can be used by an attacker to prepare specific attacks on a system. In this way an attacker could, for example, use the precise software version to identify vulnerabilities in the product and, in a second step, exploit them.

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data
- Attacks motivated and facilitated by information disclosure or visible security weaknesses

Req 28 If the system is not located in a room with at least protection class "high" (PC3), used data storages must be fully encrypted.

The Protection Classes (PC) are defined in the Annex 1, "Physical Security of Buildings", of the Group Policy on "Physical Security". Typically, Datacenters are compliant to the requirements of PC3.

Data storages are all disks and flash memory in the systems.

Motivation: Access to devices which are operated outside of data centers with protected access is relatively easy. Physical data storage media can be easily stolen as a result.

Implementation example: On Windows Server 2008, the administrator can activate drive encryption using BitLocker and key storage on a TPM.

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data
- Unauthorized modification of data
Req 29 Accounts (except for root) that are configured for the administration of specific services on a Unix system (e.g., an account for database or web server maintenance) must be configured in such a way that only the work that is necessary in these roles can be executed with them (minimum of permissions - Least Privilege principle).

Motivation: Equipping a specific administration account with only a minimum of permissions is designed to ensure that no unauthorized changes can be made to the system and no data can be viewed without the appropriate authorization. A second benefit of this approach is that, should a party gain unauthorized access to the system via an error in an application (e.g., the web server), their permissions are restricted to such an extent that they cannot manipulate the system in any way and potential damage is thus kept to a minimum.

Implementation example: If several services are installed on a server (for instance a database and a web server), administrators shall set up unprivileged users for each of the services who are allowed to only administrate the one particular service in each case. By assigning the appropriate permissions to the files and directories of the services it is ensured that a database administrator can configure and administrate the database only, and not any other service such as a web server.

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources

Availability and Integrity

Req 30 The system must be robust against overload situations.

A system must provide security measures to deal with overload situations. In particular, partial or complete impairment of system availability must be avoided. Potential protective measures include:

- Restricting of available RAM per application
- Restricting of maximum sessions for a Web application
- Defining the maximum size of a dataset
- Restricting CPU resources per process
- Prioritizing processes
- Limiting of amount or size of transactions of an user or from an IP address in a specific time range

Motivation: An attacker can effect systems availability through targeted exploitation of vulnerabilities in resources administration. An attacker can, for example, send bulk queries to a Web server and, by doing so, compromise its availability if the maximum number of permissible Web sessions is not restricted.

For this requirement the following threats are relevant:
- Disruption of availability
If an overload situation cannot be prevented, the system must act in a predictable way.

A system must be built in this way that it can react on an overload situation in a controlled way. However it is possible that a situation happens where the security measures are not longer sufficient.

In such case it must be ensured that the system cannot reach an undefined and thus potentially insecure state. In an extreme case this means that a controlled system shutdown is preferable to uncontrolled failure of the security functions and thus loss of system protection.

Motivation: With denial-of-service attacks can an attacker try to overload a system to effect its availability or integrity. Unpredictable acting of the system is then a risk for functionality and data and possibly also for other systems.

For this requirement the following threats are relevant:
• Disruption of availability

The system must be robust against unexpected input.

During transmission of data to a system it is necessary to validate this before processing. This includes all data which are send to the system. Examples for this are user input, values in arrays and content in protocols. The following typical implementation mistakes must not be done:
• No validation on the lengths of transferred data
• Incorrect assumptions about data formats
• No validation that received data complies with the specification
• Insufficient handling of protocol errors in received data
• Insufficient restriction on recursion when parsing complex data formats
• White listing or escaping for inputs outside the values margin

Motivation: An attacker can try to put a system in an unsecure state through targeted manipulation of transmitted data. The object of such an attack is to compromise the usability, availability or integrity of individual services or of the entire system. For instance a unclean memory handling can lead to a buffer overflow that allows an attacker to execute arbitrary code on the effected system.

For this requirement the following threats are relevant:
• Unauthorized access to the system
• Unauthorized use of services or resources
• Disruption of availability

Growing (dynamic) content must not influence system functions.

Growing log data and uploads must not influence system functions.

Motivation: A filled up filesystem could stop the system from operations.

Implementation example: Usage of dedicated filesystems, separated from main system functions, or quotas, or at least a file system monitoring.

For this requirement the following threats are relevant:
• Disruption of availability
Req 34 Systems must not process IP packets which source address is not reachable via the incoming interface.

It is necessary to ensure there are no unneeded default routes which is typically the case for internal systems.

**Motivation:** In such a case the IP packet comes from an untrusted source (spoofed address) or a routing error exists in the network. In both cases the packet has to be dropped.

Implementation example: Use of "Reverse Path Filter" (RPF) which provides this feature.

For this requirement the following threats are relevant:

- Disruption of availability

---

Req 35 The processing of ICMPv4 and ICMPv6 packets which are not required for operation must be disabled.

There are different types of ICMP4 and ICMPv6 that are not used in most networks, but represent a risk. These types must be disabled or filtered and not be answered, sent or processed. The following ICMP types are permitted and may be used:

- Echo Request [Type 8 (v4), Type 128 (v6)]
- Echo Reply [Type 0 (v4), Type 129 (v6)]
- Destination Unreachable [Type 3 (v4), Type 1 (v6)]
- Time Exceeded [Type 11 (v4), Type 3 (v6)]
- Parameter Problem [Type 12 (v4), Type 4 (v6)]
- Packet Too Big [Type 2 (only v6)]
- Neighbor Solicitation [Type 135 (only v6)]
- Neighbor Advertisement [Type 136 (only v6)]

It is possible that other types will be necessary. This should be checked in each individual case. The ICMPv4 types "Timestamp Reply (14)," "Netmask Reply (18)," "Information Reply (16)" and "Redirect (5)" and ICMPv6 types "Router Solicitation" (133), "Router Advertisement" (134) und "Redirect" (137) must not be responded to or processed under any circumstances.

**Motivation:** ICMPv4 and v6 packets can be used by an attacker to request specific information which can be helpful for planning further attacks. In addition, it may be possible to influence the availability of systems.

For this requirement the following threats are relevant:

- Disruption of availability
- Attacks motivated and facilitated by information disclosure or visible security weaknesses
Authentication and Authorization

Req 36 If the system is not located in a room with at least protection class "high" (PC3), the BIOS must be secured against unauthorized changes.

Servers operated in public or customer areas must be especially protected against unauthorized access and changes: The BIOS settings must be protected against export and tampering. In case passwords are used, these must be exclusive to the individual server and must not allow conclusions to be drawn about a distinguishing feature of the server.

The BIOS must be configured in such a way that only the designated operating system can be started with it from the designated partition.

Motivation: Changing BIOS settings can facilitate attacks. Since, for example, local rooms with technical installations seldom offer access protection to the servers, attackers could change the startup sequence of data storage media when the server is started in the BIOS without the password protection described. This would make it possible to start an alternative operating system which circumvents the security mechanisms of the implemented operating system.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data

Req 37 There must not be a privilege escalation method which allow gaining administrator/root privileges from an user account without anew authentication with two authentication attributes.

Privilege escalation methods include always the risk that more permissions are gained than needed. The number of exploits in such mechanisms shows the complexity and vulnerability of this solutions which therefore cannot be trusted.

Motivation: If an attacker compromises an account which has permission to a privilege escalation, it may be possible that the attacker get access to wide parts of the system and stored data.

Implementation example: (Re-)Login directly into the account with the needed permissions.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Denial of executed activities
Req 38  IP packets with unnecessary options or extension headers must not be processed.

IP options and extension headers (e.g., source routing) are only required in exceptional cases. So, all packets with enabled IP options or extension headers must be filtered.

Motivation: Packets with IP options require extended processing. An attacker can exploit this in order to carry out denial-of-service attacks against an affected device.

For this requirement the following threats are relevant:
• Disruption of availability

Req 39 System functions with a need of protection must not be used or accessed without successful authentication and authorization.

The usage of a system functions or access of data classified as internal, confidential or strictly confidential must only be possible unambiguously user identification and successful authentication on basis of the user name and at least one authentication attribute. Excepted from this are functions for public use such as those for a Web server on the Internet, via which information is made available to the public. Examples for functions which require a prior authentication are network services (like SSH, SFTP, Web services), local access via a management console, local usage of operating systems and applications. The following examples are possibilities that could be used for authentication.

• Query user name and password
• Use of cryptographic keys and certificates (e.g. as Smartcard)

This requirement must also be applied to accounts that are only used for communication between systems (M2M).

Motivation: The authentication is necessary to doubtless identify a user because the allocated authorization, and therefore the access on data and services of the system depends on that.

For this requirement the following threats are relevant:
• Unauthorized access to the system
• Unauthorized use of services or resources
• Denial of executed activities

Req 40  Accounts must be protected against unauthorized use by at least one authentication attribute.

The various user and machine accounts on a system must be protected from misuse. To this end, an authentication attribute is typically used, which, when combined with the user name, enables unambiguous authentication and identification of the authorized user.

Authentication attributes include:
• Cryptographic keys
• Token
• Passwords
• PINs
This means that authentication based on a parameter that can be spoofed (e.g. phone numbers, public IP addresses or VPN membership) is not permitted. Exceptions are attributes that cannot be faked or spoofed by an attacker. Several of the above options can be combined (dual-factor authentication) to achieve a higher level of security. Whether or not this is suitable and necessary depends on the protection needs of the individual system and its data and must be evaluated for individual cases.

**Motivation:** Accounts that are not protected with a secret authentication attribute can be used by an attacker to gain unauthorized access to a system and the data and applications stored on it.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized use of services or resources
- Denial of executed activities

---

**Req 41** Accounts with extensive rights must be protected with two authentication attributes.

Accounts, for example used for administration, maintenance and troubleshooting, have extensive rights. Therefore a single protection (e.g. a password), as for normal user accounts with less rights, is not suitable. To get a higher protection level it is necessary to use more than one authentication attribute. For this a combination of an attribute that the user knows and an attribute that the user owns will be used often. This kind of authentication will be named as 2-factor authentication. Examples for 2-factor authentication are:
- Smartcard with PIN
- Private key with Passphrase
- Secure-ID Token with Password

**Motivation:** Accounts with extensive rights as used for system administration have a higher risk for system's security. An attacker can get extensive rights by compromising such an account to get access to wide parts of the system and stored data.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized use of services or resources
- Denial of executed activities

---

**Req 42** Predefined accounts must be deleted or disabled.

Many systems have default accounts (e.g. guest, ctxsys), some of which are preconfigured with or without known pass-words. These standard users must be deleted or disabled. Should this measure not be possible the accounts must be locked for remote login. In any case disabled or locked accounts must be configured with a complex password (12 character and more, use of upper/lower case, numbers and special characters). This is necessary to prevent unauthorised use of such a account in case of misconfiguration.

Exceptions to this requirement to delete or disable accounts are accounts that are used only internal on the system in-volved and that are required for one or more applications on the system to function. Also for this accounts remote access or local login must be forbidden to prevent an abusive use by users of the system.
Motivation: Standard users are typically generally known and can be used by an attacker for targeted brute force and dictionary attacks. Standard user accounts represent a special risk if they do not use a password or only use a standard password that is generally known. Such standard user accounts can easily be exploited by an attacker in order to gain access to the system involved without being authorized to do so.

For this requirement the following threats are relevant:
• Unauthorized access to the system
• Unauthorized use of services or resources
• Denial of executed activities

Req 43 Predefined authentication attributes must be deleted or disabled.

Normally, authentication attributes such as password or cryptographic keys will be preconfigured from producer, vendor or developer of a system. Such authentication attributes must be changed an own attribute not known by other parties.

Motivation: Authentication attributes like password or cryptographic keys preconfigured from third parties are not trustable. Such authentication attributes can be used to compromise systems or their data.

For this requirement the following threats are relevant:
• Unauthorized access to the system
• Unauthorized use of services or resources
• Denial of executed activities

Req 44 The authorizations for accounts and applications must be reduced to the minimum required for the tasks they have to perform.

Authorizations to a system must be restricted to a level in which a user can only access data and use functions that he needs in the course of his work. Suitable authorizations must also be assigned for access to files that are components of the operating system or of applications or that are generated by the same (e.g. configuration and logging files).

Alongside access to data, execution of applications and components must also take place with rights that are as low as possible. Applications should not be executed with administrator or system rights.

Motivation: If the rights granted to a user on a system are too broad, it could be possible for him to access data and applications which he is not permitted to view or use. This would give him the opportunity to disclose or modify confidential data and to manipulate system files. Applications with rights that are too broad can be used by a user to extend his own authorizations and thus to gain access to files and system components to which he would not have had access with his authorizations under normal circumstances.

For this requirement the following threats are relevant:
• Unauthorized access or tapping of data
• Unauthorized modification of data
• Unauthorized use of services or resources

Req 45 The system must be connected to a central system for user administration.

For provisioning of these authorization information must the system provide a central inter-face (e.g. LDAPs for authorization, Kerberos for authentication, locking information for certificates) or decentralized mechanisms (e.g. public-key authentication). A central solution for identity management must be preferred.
In areas where a central identity management system is not available a central system such as LDAP, TACACS+ or Radius server for the administration of accounts and their authentication and authorization must be used.

Motivation: Central administration of identity of accounts and their rights means that they only have to be maintained once instead of separately on each system. From the aspect of security, the advantage is that an account and its rights only known on a single central side. This information can be transmitted from a central side to systems (provisioning), central administrated (reconciliation) and central deleted (deprovisioning). This reduces the risk of accounts being forgotten during changing or deletion since they are configured on multiple systems. This could give a user wrong system rights or continued access to a system.

For this requirement the following threats are relevant:
• Unauthorized access to the system
• Unauthorized access or tapping of data
• Unauthorized modification of data
• Unauthorized use of services or resources

---

Req 46 The number of local accounts needed for operation must be minimized.

All operating systems have high security requirements regarding local accounts. The administrator must ensure that all unused local accounts are deactivated.

The passwords needed to login at these accounts must be unique for every machine and account. It is recommended to store them properly, e.g., in a safe or online in a password vault where access is granted only when needed.

Motivation: Local accounts are additional points of attack which can be used by attackers or unauthorized individuals. This requires that only absolutely necessary local accounts required for operation exist in the operating system.

For this requirement the following threats are relevant:
• Unauthorized access to the system

---

Req 47 The identification of users of system accounts (root, oracle, wlsadmin, ...) must be unambiguous and must be carried out by an individual, strong authentication scheme. Shared passwords shall not used for remote login.

The unambiguous identification is ensured by authentication with user-specific credentials (e.g. private key on the smartcard). By using SSH keys users do not have to share credentials (passwords). Passwords are only necessary in combination with physical console-access protected by admission control.

Personalized accounts on a Unix server are problematic when the privileges have to be extended by SUID programs for performing administrative tasks. In practice, these more complex configurations are error-prone and lead to vulnerable configurations. Complex SUID programs (e.g. sudo) themselves have a large attack surface, which are exploited for privilege escalation.

Functional accounts (root, oracle, ...) shall be favored over user-specific accounts for system administration, since due to systemic nature of system account they are needed anyway and the lack of user-specific accounts reduces complexity (no privilege escalation, manage fewer accounts, no complex "RBAC "configurations), and thus increases security.
However, the person responsible shall ensure that the functional accounts are kept strictly separate from each other. For this purpose, they shall implement suitable measures, depending on the operating system and functions it offers. Restrictively setting file and directory permissions must be the minimum requirement.

It can also be worked with personal accounts, which must be member of a dedicated group based on the needed function. Also in this case the use of SUDO is not necessary, if the relevant resources (log files etc.) are available for the members of the dedicated group.

Example:
A Unix system running a database as its only application requires no more than two accounts in the simplest case: A root account for system administration and a database account for application administration at system level.

By not using personalized accounts, those responsible will not need SUID root binaries such as SUDO, which means that user administration is significantly simplified. Using SSH keys, for instance, ensures that access data (passwords) is not known to several persons.

Motivation: Restrictive user administration for Unix systems is important to ensure that only authorized persons gain access to a system and can only perform activities there that are necessary for their work. Those responsible should also make sure here that they are not “introducing” additional security problems (SUDO) and that administration remains as simple as possible. This will increase clarity and help to avoid user errors. These measures will also automatically increase user-friendliness (no need to enter a user name and password on every individual system at login) for administrators and simplify user administration.

Implementation example: To make sure that unprivileged users can still perform certain tasks, the administrator may, for example, bind an Apache web server to an unprivileged port (if located behind a load-balancer), so that it can also be launched with the permissions of a normal user.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources
- Disruption of availability
Each system account must have a unique UID.

Motivation: If accounts other than the root account bear UID 0, this often indicates that a successful attack has been carried out on a UNIX system. In order to prevent misunderstandings, maintain system clarity and take account the requirement, which specifies that functional accounts instead of user-specific accounts shall be used for certain administrative tasks, it is necessary that one Unix account only may have UID 0.

For this requirement the following threats are relevant:
- Denial of executed activities

**Authentication Parameter Password**

If a password is used as an authentication attribute, it must have at least 8 characters and contain three of the following categories: upper cases, lower case, numbers and special characters.

A system may only accept passwords that comply with the following complexity:
- Minimum length of 8 characters.
- Comprising at least three of the following categories: upper/lower case letters, numbers and special characters.

When a password is assigned, the system must ensure that the password meets these requirements. If a central system will be used for user authentication this function can be forwarded or delegated to this system.

Motivation: Passwords with the above complexity offer high robustness against attacks coupled with acceptable user friendliness. Passwords with this level of complexity have proved their efficiency in practice. Trivial passwords that are too short are susceptible to brute force and dictionary attacks and are therefore easy for attackers to determine. Once a password has been ascertained it can be used by an attacker for unauthorized access to the system and the data on it.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized use of services or resources
- Denial of executed activities

If a password is used as an authentication attribute, users must be able to independently change the password anytime.

The system must offer a function that enables a user to change his password at any time.

When an external centralized system for user authentication will be used it is possible to redirect or implement this function on this system.

Motivation: The fact that a user can change his authentication attribute himself at any time enables him to change it promptly if he suspects that it could have been accessed by a third party.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized use of services or resources
- Denial of executed activities
Req 51 If a password is used as an authentication attribute, a protection against brute force and dictionary attacks that hinder password guessing must be implemented.

Brute force and dictionary attacks aim to use automated guessing to ascertain passwords for user and machine accounts. Various measures or a combination of these measures can be taken to prevent this.

The most commonly used protection measures are:

- Increasing the delay (e.g. doubling wait times at each attempt) for each renewed password input following an incorrect entry ("tar pit").
- Blocking an account following a specified number of incorrect attempts (typically 5). However, one must take into account that this solution requires a process for unlocking and that an attacker can force the system to deactivate accounts and make them unusable.
- Using CAPTCHA to prevent automated attempts (often used for Web applications).

In order to achieve higher security, it is often meaningful to combine two or more of these measures named here. This must be evaluated in individual cases and implemented accordingly.

Motivation: Implementation of one or more of the above measures can prevent successful execution of brute force and dictionary attacks to ascertain passwords.

For this requirement, the following threats are relevant:

- Unauthorized access to the system
- Unauthorized use of services or resources
- Denial of executed activities

Req 52 If a password is used as an authentication attribute, they must be hidden when displayed on screen.

When a user enters a password or the password is otherwise displayed on screen, it must be made unreadable. Typically, the individual characters of the password are replaced by a character such as “*”. Under certain circumstances, it may be permissible for an individual character to be displayed briefly during input. Such a function is used, for example, on Smartphone’s to make input easier. However, the entire password is never output to the display in plaintext.

Motivation: To prevent another person reading a password by chance or intentionally on screen during input, the password must be rendered unreadable during input or display.

For this requirement, the following threats are relevant:

- Unauthorized access to the system
- Unauthorized use of services or resources
- Denial of executed activities
Logging

Req 53 Security relevant events must be logged with a precise timestamp and a unique system reference.

Systems must log the occurrence of security-relevant incidents. So that these events can be evaluated and classified, they must be logged together with a unique system reference (e.g., host name, IP or MAC address) and the exact time the incident occurred. Furthermore, legal and data privacy regulations (e.g. time of storage of logging data) must be proved and followed. The following table lists events that are relevant to security and the corresponding data that typically has to be logged by a system. Exceptions are systems for which no or only restricted logging applies. Examples of such systems are customer devices such as Smartphone’s or IADs/Homegateways (e.g. Speedport).

Logging must be done considering the currently valid legal, wage and company regulations. This regulations state among others that logging of events can be done only earmarked. Logging of events for doing a work control of employees is not allowed.

Typical event that reasonable should be logged in many cases are:

<table>
<thead>
<tr>
<th>Event</th>
<th>Event data to be logged</th>
</tr>
</thead>
<tbody>
<tr>
<td>Incorrect login attempts</td>
<td>• Account,</td>
</tr>
<tr>
<td></td>
<td>• No. of failed attempts,</td>
</tr>
<tr>
<td></td>
<td>• Source (IP address) of remote access</td>
</tr>
<tr>
<td>System access with accounts with</td>
<td>• Account,</td>
</tr>
<tr>
<td>administrator rights</td>
<td>• Access timestamp,</td>
</tr>
<tr>
<td></td>
<td>• Length of session,</td>
</tr>
<tr>
<td></td>
<td>• Source (IP address) of remote access</td>
</tr>
<tr>
<td>Account administration</td>
<td>• Administrator account,</td>
</tr>
<tr>
<td></td>
<td>• Administered account,</td>
</tr>
<tr>
<td></td>
<td>• Activity performed (configure, delete, enable and disable)</td>
</tr>
<tr>
<td>Change of group membership for accounts</td>
<td>• Administrator account,</td>
</tr>
<tr>
<td></td>
<td>• Administered account,</td>
</tr>
<tr>
<td></td>
<td>• Activity performed (group added or removed)</td>
</tr>
<tr>
<td>Critical rise in system values such</td>
<td>• Value exceeded,</td>
</tr>
<tr>
<td>as disk space, CPU load over a longer</td>
<td>• Value reached</td>
</tr>
<tr>
<td>period</td>
<td>(Here suitable threshold values must be defined depending on the individual system.)</td>
</tr>
</tbody>
</table>

Logging of additional security-relevant events may be meaningful. This must be verified in individual cases and implemented accordingly where required.

Motivation: Logging security-relevant events is a basic requirement for detecting ongoing attacks as well as attacks that have already occurred. This is the only way in which suitable measures can be taken to maintain or restore system security. Furthermore, the logging data is used as evidence so that legal steps can be taken against attackers.

For this requirement the following threats are relevant:
• Denial of executed activities
• Unnoticeable feasible attacks
Req 54 Security relevant logging data must be send to an external system direct after their creation.

Logging data must be forwarded to an external system in appropriate logging files as well as being stored locally. Standard protocols like Syslog, SNMPv3 must be preferred.

Motivation: If logging data is only stored locally it can be manipulated by an attacker who succeeds in compromising the system in order to conceal his attack and any manipulation he has performed on the system. This is the reason why the forwarding must be done immediately after the event occurred.

For this requirement the following threats are relevant:
• Denial of executed activities
• Unnoticeable feasible attacks
3.23 Routers and Switches

Introduction

This security requirement has been prepared based on the provisions of the Group Policy on IT/NT Security. The security requirement is used as a basis for an approval in the PSA process, among other things. It also serves as an implementation standard for provisions of the Group Policy on IT/NT Security in units which do not participate in the PSA process. These requirements shall be taken into account from the very beginning, including during the planning and decision-making processes. When implementing these security requirements, the precedence of national, international and supranational law shall be observed.

System hardening

<table>
<thead>
<tr>
<th>Req 1</th>
<th>Any services and protocols that are not secure and not used must be disabled.</th>
</tr>
</thead>
</table>

Many network devices offer services which may not be used in networks on account of known security vulnerabilities such as non-encrypted transmission or inadequate authentication, etc. These services must be completely disabled. The services not to be used are:

- FTP
- TFTP
- Telnet
- rlogin, RCP, RSH
- HTTP
- SNMPv1 and v2
- SSHv1
- TCP/UDP Small Servers (Echo, Chargen, Discard und Daytime)
- Finger
- BOOTP server
- Discovery protocols (CDP, LLDP)
- IP Identification Service (Identd)
- PAD
- MOP

As an alternative to disabling the HTTP service, it is also possible for this service to remain active for reasons of user friendliness. In this case, however, queries to the web service may not be answered directly on this port but must be diverted to a port on which the encrypted HTTPS protocol is used.

Discovery protocols such as the Cisco Discovery Protocol (CDP) or the Link Layer Discovery Protocol (LLDP) must be completely disabled. These protocols may be used in well-founded, exceptional cases. However, it must be ensured that the protocols are only active on internal links. Discovery protocols must be disabled on interfaces to customers or devices.

Should additional services be available on a network devices, a check should be carried out in each case to establish whether the services are necessary for the operation of the network devices. Otherwise these services shall be dis-abled.

Motivation: The protocols named display various security vulnerabilities. A large proportion of the protocol messages is transmitted in plain text, for example. An attacker who is able to record such communication is then able to obtain confidential data such as user names and passwords. Another vulnerability inherent in the aforementioned services is the susceptibility to
denial-of-service attacks (DoS). These can be used by attackers to compromise network device availability.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized use of services or resources
- Disruption of availability
- Attacks motivated and facilitated by information disclosure or visible security weaknesses

Req 2  The proxy ARP function must be disabled.

ARP (Address Resolution Protocol) requests are used by systems to request the MAC address of other systems in the same network based on the known IP address. This is a layer-2 protocol which can only be used within a network and not across routers. The proxy ARP function cancels this limitation whereby the router or switch functions as the broker for such requests. The proxy ARP function is typically not used in networks and should be disabled due to the resulting risk. Should this function actually be required in a network, a check should be carried out in each case to establish whether the associated risk is acceptable.

Motivation: An attacker can use the proxy ARP function under certain circumstances in order to carry out typical layer-2 attacks such as ARP spoofing or denial-of-service attacks.

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data
- Disruption of availability

Req 3  Gratuitous ARP requests may not be accepted.

Gratuitous ARPs are ARP messages which a system can send in a network in order to inform other systems about its IP address. ARP requests are normally requests of a system which are sent as and when required. Gratuitous ARP messages are usually not necessary and should be disabled on routers and switches. Under certain circumstances (e.g., when HSRP is used), gratuitous ARP shall be permitted. This depends on the individual case and shall be checked accordingly.

Motivation: An attacker can use gratuitous ARP messages to tamper with the address tables of other systems and thus divert traffic or carry out a denial-of-service attack.

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data
- Disruption of availability

Req 4  The IPv4 and IPv6 addresses of all the interfaces must be statically configured.

The IPv4 and IPv6 addresses of all the interfaces must be permanently-configured. This means that the automatic assignment of IP addresses, e.g., using DHCPv4/v6 or IPv6 auto-configuration for network devices, is not permitted. IPv6 router advertisements must be ignored by network devices.

It is recommended to form the host share of the IPv6 addresses randomly. Due to the very large address area of IPv6, it is very time-consuming for an attacker to use scans to discover systems.

Motivation: The automatic configuration of IP addresses enables a connected attacker to tamper with the network device, so that he may be able to divert traffic or impair communication.
For this requirement the following threats are relevant:
  • Unauthorized access to the system
  • Disruption of availability

---

**Req 5**  Management traffic must be separated from signalling and commercial traffic.

Data traffic for the management of a network device must be physically or logically separated from any other traffic. Physical separation is, for example, a separate dedicated interface via which the network device is connected to a separate management network. Logical separation can be achieved via VPNs or VLANs. In this case, traffic is transmitted within the same physical network as other traffic but is logically separate. Thus direct access to the management of the network device from the production network i.e., possible access by customers, is therefore prevented.

*Motivation:* The management services and traffic are an attractive target for attackers. By recording management traffic, an attacker may obtain important information which can be used to prepare and carry out attacks. Direct accessibility of the management services by customers or from within the Internet increases the risk of a successful attack against a network device. Since system administration with high-level access rights normally takes place via such services, an attacker might compromise the entire network device via this and, in doing so, gain unauthorized access to the network device and any networks connected to it.

For this requirement the following threats are relevant:
  • Unauthorized access to the system
  • Disruption of availability

---

**Req 6**  The accessibility of management services must be restricted to legitimate systems.

Access can be restricted, for example, through filters, access lists or a local firewall. The restriction must be as strict as possible. This means to host or network addresses to achieve that the management services can only be reached from legitimate systems.

*Motivation:* Management services enable access to network devices in order to perform operational tasks. In the event of a successful attack, an attacker may gain access to confidential information or even to the entire system. By restricting the accessibility to legitimate systems, the group of potential attackers can be reduced, and thus also the likeliness of a successful attack.

For this requirement the following threats are relevant:
  • Unauthorized access to the system
  • Disruption of availability

---

**Req 7**  Management services must be permanently connected to an address.

The management services (e.g., SSH, HTTPS or SNMP) that are active on a network device must be permanently connected to an address of the network device. Hence the required separation of management traffic from control and user traffic is the appropriate address from the management address range. This ensures that the relevant traffic always comes from a fixed sender address and on the other side the management service can be reached under the same address at any time.

*Motivation:* Without the implementation of this measure, it cannot be clearly foreseen with which sender address packets of the management services of the network device are sent out or on which address management services are reachable. This causes a number of disadvantages. Thus recognition of attacks in logging and monitoring and the analysis of the
data arising therefrom is made much more difficult. In addition, a permanent sender address is important for implementing filters and firewall rules and for checking the authenticity of keys and certificates when using cryptographic procedures to secure management services and traffic.

For this requirement the following threats are relevant:

• Denial of executed activities
• Unnoticeable feasible attacks

---

Req 8 Unused software must not be installed or must be uninstalled.

During installation of a system often software components will be installed or parts of software will be activated which are not needed for the operation or functionality of the system. This includes also parts of a software, which will be installed as examples but typically not be used (e.g. default web pages, example databases, test data). Such components should not be installed or must be deleted after installation.

Motivation: Vulnerabilities in software of a system offer an attack window for attackers to infiltrate the system. Uninstalling components that are not required can therefore reduce the possibility of a successful compromise of the system.

For this requirement the following threats are relevant:

• Unauthorized access to the system
• Disruption of availability
Mark A. Reiter, Ulrich Schröter, Christian Schwegler

3.23 Routers and Switches

Req 9  Unused interfaces must be disabled.

Unused interfaces of a network device shall be disabled. It must be assured that interfaces remain inactive after a re-boot.

Motivation: Unused interfaces are usually not taken into account in the configuration process of a network device. As a result, these interfaces are operated with the manufacturer’s default configuration. This may enable an attacker who has direct physical access to such a network device to gain unauthorized access to the system or to networks connected to it.

For this requirement the following threats are relevant:
• Unauthorized access to the system
• Unauthorized access or tapping of data
• Disruption of availability

System update

Req 10  Software and hardware components that are no longer supported by vendor, producer or developer must not be used.

Components that have reached end-of-life or end-of-support must not be used. Excluded are components that have a special support contract. This contract must guarantee the correction of vulnerabilities over components life-time.

Motivation: Hardware and software components that have reached end of life or end of support represent a risk for a system. This means that a vendor does not supply remedial updates or patches for a component should errors or vulnerabilities occur. This means that vulnerabilities cannot be fixed when they occur and could be exploited to compromise the system or to impair its availability.

For this requirement the following threats are relevant:
• Unauthorized access to the system
• Unauthorized access or tapping of data
• Unauthorized modification of data
• Disruption of availability

Req 11  Known vulnerabilities in software and hardware of the system must be fixed or protected.

Prior to installation of a software or hardware component, users must check whether any vulnerability has been discovered and published for the version they are installing. Any component that proves to have a vulnerability must not be installed or used. Excepted from this rule are components for which the vendor has already provided a measure to remedy the vulnerability, e.g. a patch, update or workaround. In this case, the additional measure must be implemented on the system. Furthermore it is an ongoing process during the complete life cycle of the system to fix upcoming vulnerabilities promptly.

Motivation: Publication of vulnerabilities increases the risk of successful exploitation by an attacker, especially since the published information usually includes details on how to exploit the vulnerability and tools that make exploitation possible. One example of this is when Web servers are compromised due to not patched vulnerabilities.

For this requirement the following threats are relevant:
• Unauthorized access to the system
• Unauthorized access or tapping of data
• Unauthorized modification of data
• Disruption of availability
Protecting data and information

Req 12 Data with need of protection must be secured against unauthorized viewing and manipulation during transmission and storage.

Adequate security measures for transmission and storage must be implemented for data with a need for protection that are classified as internal, confidential or strictly confidential. The chosen measure depends on the classification for the data and other factors such as the type of network used during transmission, the storage location for data, etc. Furthermore must be guaranteed that confidential and strictly confidential data will not be unprotected during temporary storage (e.g. in web cache, temporary folders).

All authentication data such as user names, passwords, PINs, etc. must be protected against unauthorized viewing and manipulation. This applies equally to permanent storage and transmission. Typical measures taken to protect authentication data or combinations of these are:

For storage:
- Client systems: encryption or obfuscation of authentication data, no persistent storage, limitation of access rights.
- Server systems: hashing of authentication data with PBKDF like scrypt or bcrypt or when this is not possible hashes with salt.

For transmission:
- Usage of cryptographically protected network protocols.
- Challenge response method (Disadvantage of this solution is that passwords need to be in clear-text on server. If this solution is feasible depends on the individual threat scenario)

Files of a system that are needed for functionality must also be protected against manipulation. This is necessary because system’s integrity can be damaged when the system access this kind of files. An example is the use of checksum or cryptographic methods to validate if e.g. firmware images, patches, drivers or kernel modules are free of manipulations.

For transmission of data with a need of protection network protocols that are insecure due to insufficient security measures shall not be used. Examples are: SSLv3, SSHv1, FTP, Telnet, SNMPv1 and 2c. In case of these protocols a newer version without vulnerabilities or a secure alternative must be used.

Motivation: If data with a need of protection will not be secured an attacker could record or manipulate the data during transmission over a network. An example is the recording of user names and passwords during system administration with the telnet clear-text protocol. Storing data on a system without adequate protection may mean that unauthorized users can copy or modify it. One example is when passwords can be read out when they stored in an inadequate secured way (e.g. usage of unsecure hashing algorithms like MD5 or SHA-1) or even without encryption on a system or manipulation of firmware to affected the system integrity.

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data
- Unauthorized modification of data
3.23 Routers and Switches

Req 13  Encrypted protocols must be used for management.

Access to management services may only take place by means of secure protocols (e.g., SSHv2, HTTPS or SNMPv3). This is necessary because when accessing management services of a network device, data requiring protection such as user names, passwords or configuration data is transmitted. In addition, the use of encrypted protocols is also necessary for the transmission of new operating system versions and for updates and patches, etc.

Motivation: When plain text protocols such as Telnet, HTTP, FTP, TFTP or SNMP (version 1 and 2) are used for the management of network devices, an attacker record and manipulate data or possibly hijack the current session. In his next step, he can then use the information gained for attacks on the network device.

For this requirement the following threats are relevant:
• Unauthorized access or tapping of data
• Unauthorized modification of data
• Attacks motivated and facilitated by information disclosure or visible security weaknesses

Req 14 Information with need of protection must not be contained in files, outputs or messages that are accessible by unauthorized users.

Information with need of protection must not be accessible in files, outputs or messages of the system by unauthorized users. This includes information relating to the operating system, used middleware or applications such as vendor, product name, product identifier, installed software versions, installed service packs, patches, hot fixes and serial numbers. Examples for system messages which must be free of sensitive data are:

• Comments in downloadable files
• Error and system messages
• Stack traces
• Network protocols
• Login windows and dialogs

Furthermore, details of implementation and information relating, e.g., to backend software/systems, function calls, SQL instructions or structure of database, must not be contained in error messages.

Excluded from this are displays and outputs that can be viewed and retrieved by authorized users who are logged in. In addition, an internal transfer of system internal information for error analysis is allowed in an adequate dimension. In this case the continuative regulations or guidelines (e.g. of data privacy) must be noticed.

Motivation: The information named above can be used by an attacker to prepare specific attacks on a system. In this way an attacker could, for example, use the precise software version to identify vulnerabilities in the product and, in a second step, exploit them.

For this requirement the following threats are relevant:
• Unauthorized access or tapping of data
• Attacks motivated and facilitated by information disclosure or visible security weaknesses
Protecting availability and integrity

Req 15  The system must be robust against overload situations.

A system must provide security measures to deal with overload situations. In particular, partial or complete impairment of system availability must be avoided. Potential protective measures include:

- Restricting of available RAM per application
- Restricting of maximum sessions for a Web application
- Defining the maximum size of a dataset
- Restricting CPU resources per process
- Prioritizing processes
- Limiting of amount or size of transactions of an user or from an IP address in a specific time range

Motivation: An attacker can effect systems availability through targeted exploitation of vulnerabilities in resources ad-ministration. An attacker can, for example, send bulk queries to a Web server and, by doing so, compromise its availability if the maximum number of permissible Web sessions is not restricted.

For this requirement the following threats are relevant:
- Disruption of availability

Req 16 Packets which must be processed via the operating system of the network device must not lead to impairment of availability, even when they occur in great numbers.

Particular packets mean that processing implemented within the hardware is no longer possible and processing of the relevant task must be undertaken by the operating system. This has a direct impact on system resources like CPU and memory of the network device. Examples of such behavior are:

- Generating response packets to ICMPv4 and ICMPv6 queries.
- Generating ICMP response packets such as Destination Unreachable, Packet too Big etc. for particular pack-ets.
- Generating response packets for packets with expired TTL/Hop Limit value.
- Processing IPv6 packets with Hop-by-Hop or Destinations Options headers.

The behaviour of the network device can often be configured that or only a certain number of such packets are pro-cessed (rate limit) or such packets are rejected.

Motivation: Packets that a network device will operate by the operating system lead to load of the processor and working memory. Without an applicable protection a attacker can use massively send packets to compromise the availabil-ity of the network device.

For this requirement the following threats are relevant:
- Disruption of availability

Req 17  Directed broadcasts must be disabled

IP-directed broadcasts are packets which are sent to a broadcast address of a subnet with which the sending system has no direct connection. IP-directed broadcasts are used in a network in exceptional cases only. IP-directed broad-casts can be misused for denial-of-service attacks. The option to pass on such packets on a router must therefore be disabled.

Motivation: IP-directed broadcasts can be exploited by an attacker to perform SMURF denial-of-service attacks.
3.23 Routers and Switches

For this requirement the following threats are relevant:
• Disruption of availability

---

**Req 18**  
Packets with IPv4 options must be ignored.

IP options (e.g., source routing) are only required in modern networks in exceptional cases. Because appropriate packets are a threat for a network device the handling of packets with IP options enabled must be disabled or filtered.

*Motivation:* Packages with IP options require extended processing by the network device. This means that processing such packets at the hardware level is not possible. An attacker can exploit this in order to carry out denial-of-service attacks against an affected network device.

For this requirement the following threats are relevant:
• Disruption of availability

---

**Req 19**  
Packets with IPv6 routing header must be ignored.

Two routing header types are defined in the IPv6 standard. These are types 0 (for source routing) and 2 (for mobile IPv6). Both options should not occur in a network. Such packets must therefore be ignored by a network element.

*Motivation:* Type 0 IPv6 routing headers can be exploited by an attacker in order to interfere with decisions regarding pathways. This can be exploited, on the one hand to evade security mechanisms, and on the other to impair availability.

For this requirement the following threats are relevant:
• Disruption of availability

---

**Req 20**  
The IPv6 hop-by-hop header of packets that do not come from trustworthy senders must be ignored.

The IPv6 standard defines that packets with Hop-by-Hop extension headers must be processed by every system on the path between the source and the destination of a communication, because they may contain further instructions. This processing is time-consuming and normally cannot be performed at the hardware level of a network device. It must therefore be ensured that such packets forwarded by the network device without processing, or only be processed if they come from the address of a trustworthy sender.

*Motivation:* An attacker may exploit packets with IPv6 Hop-by-Hop extension headers in order to produce a high load on a network device. This may lead to the impairment of the availability of the network device.

For this requirement the following threats are relevant:
• Disruption of availability

---

**Req 21**  
Packets with IPv6 Destination Options headers that are sent to an address of the network device and do not come from a trustworthy sender must be rejected.

The IPv6 standard defines that packets with Destination Options headers must be processed by the recipient's system, since they may contain further instructions. This processing is time-consuming and normally cannot be performed at the hardware level of a network device. It must therefore be ensured that such packets are not processed at all, or only if they come from the address of a trustworthy sender.
Motivation: An attacker may exploit packets with IPv6 Destination Options headers in order to produce a high load on a network device. This may lead to the impairment of the availability of the network device.

For this requirement the following threats are relevant:
• Disruption of availability

---

**Req 22** Fragmented IPv6 packets that are sent to an address of the network device must be rejected.

The occurrence of fragmented IPv6 packets that are sent to an address of a network device is very unlikely and should not happen in a properly designed network. Therefore all fragmented IPv6 packets that are sent to one or more of a network device's addresses must be rejected by that network device. When necessary fragmented packets can be allowed inside the infrastructure address range of the network.

Motivation: An attacker may exploit fragmented IPv6 packets in order to produce a high load on a network device. This may lead to the impairment of the availability of the network device.

For this requirement the following threats are relevant:
• Disruption of availability

---

**Req 23** ICMPv4 and v6 types which are not required for operation must be disabled.

There are different types of ICMPv4 and v6 that are not used in most networks, but represent a risk. These types must be disabled or filtered and not be answered, send or processed by the network device. The following ICMP types are permitted and may be used in networks:

- Echo Request [Type 8 (v4), Type 128 (v6)]
- Echo Reply [Type 0 (v4), Type 129 (v6)]
- Destination Unreachable [Type 3 (v4), Type 1 (v6)]
- Time Exceeded [Type 11 (v4), Type 3 (v6)]
- Parameter Problem [Type 12 (v4), Type 4 (v6)]
- Packet Too Big [Type 2 (only v6)]
- Neighbor Solicitation [Type 135 (only v6)]
- Neighbor Advertisement [Type 136 (only v6)]

It is possible that other types will be necessary. This should be checked in each individual case. The ICMPv4 types "Timestamp Reply (14)," "Netmask Reply (18)," "Information Reply (16)" and "Redirect (5)" must not be responded to or processed under any circumstances.

Motivation: ICMPv4 and v6 packets can be used by an attacker to request specific information from a network device which can be helpful for planning further attacks. In addition, it may be possible to influence the availability of systems.

For this requirement the following threats are relevant:
• Disruption of availability

---

**Req 24** The maximum number of ICMPv4 and v6 response packages sent per second must be restricted.

Different packets or events lead to a situation where these have to be answered by a network device with an ICMP packet. Such packets or events include, for example, ICMP queries such as Echo Reply, packets with an expired TTL/Hop Limit field or packets that exceed the permitted MTU, but must not be fragmented. Since the production of ICMP response packets cannot be processed at the hardware level with many network devices, this leads to a strain on the processor and the RAM. For this reason, the maximum number of ICMP response packets that may be produced per second must be limited (rate limit).
Motivation: An attacker can send large numbers of tampered packets to a network device in order to compel the production of ICMP response packets. This may lead to a high load and thus to an impairment of the availability of the network device.

For this requirement the following threats are relevant:
• Disruption of availability

Req 25 If an overload situation cannot be prevented, the system must act in a predictable way.

A system must be built in this way that it can react on an overload situation in a controlled way. However it is possible that a situation happens where the security measures are not longer sufficient.

In such case it must be ensured that the system cannot reach an undefined and thus potentially insecure state. In an extreme case this means that a controlled system shutdown is preferable to uncontrolled failure of the security functions and thus loss of system protection.

Motivation: With denial-of-service attacks can an attacker try to overload a system to effect its availability or integrity. Unpredictable acting of the system is then a risk for functionality and data and possibly also for other systems.

For this requirement the following threats are relevant:
• Disruption of availability

Req 26 The system must be robust against unexpected input.

During transmission of data to a system it is necessary to validate this before processing. This includes all data which are send to the system. Examples for this are user input, values in arrays and content in protocols. The following typical implementation mistakes must not be done:
• No validation on the lengths of transferred data
• Incorrect assumptions about data formats
• No validation that received data complies with the specification
• Insufficient handling of protocol errors in received data
• Insufficient restriction on recursion when parsing complex data formats
• White listing or escaping for inputs outside the values margin

Motivation: An attacker can try to put a system in an unsecure state through targeted manipulation of transmitted data. The object of such an attack is to compromise the usability, availability or integrity of individual services or of the entire system. For instance a unclean memory handling can lead to a buffer overflow that allows an attacker to execute arbitrary code on the effected system.

For this requirement the following threats are relevant:
• Unauthorized access to the system
• Unauthorized use of services or resources
• Disruption of availability

Req 27 Manipulated packets, that are sent to an address of the network device, must not lead to an impairment of availability.

A network device shall be not effected in its availability or robustness by packets that are manipulated or differing the norm. This means that appropriate packets must be detected as invalid and be discarded. The process shall not be effect the performance of the network device. This robustness must be just as effective for a great mass of invalid pack-ets as for individual or a small number of packets.
Examples of such packets are:

- Mass-produced TCP packets with a set SYN flag to produce half-open TCP connections (SYN flooding attack)
- Packets with the same IP sender address and IP recipient address (Land attack)
- Mass-produced ICMP packets with the broadcast address of a network as target address (Smurf attack)
- Fragmented IP packets with overlapping offset fields (Teardrop attack)
- ICMP packets that are larger than the maximum permitted size (65,535 Bytes) of IPv4 packets (Ping-of-death attack)

Sometimes the relevant behaviour of the network device must be configured. In other cases, the behaviour of the network device may only be verified by the relevant tests.

**Motivation:** An attacker can use tampered packets to perform so-called denial-of-service attacks, in order to impair the availability of the network device as a whole or in part. Sometimes it only requires individual packets, or a few of them, to make a vulnerable network device crash.

For this requirement the following threats are relevant:

- Disruption of availability

---

**Authentifizierung und Autorisierung**

**Authentication and authorization**

**Req 28** System functions with a need of protection must not be used or accessed without successful authentication and authorization.

The usage of a system functions or access of data classified as internal, confidential or strictly confidential must only be possible unambiguously user identification and successful authentication on basis of the user name and at least one authentication attribute. Excepted from this are functions for public use such as those for a Web server on the Internet, via which information is made available to the public. Examples for functions which require a prior authentication are network services (like SSH, SFTP, Web services), local access via a management console, local usage of operating systems and applications. The following examples are possibilities that could be used for authentication.

- Query user name and password
- Use of cryptographic keys and certificates (e.g. as Smartcard)

This requirement must also be applied to accounts that are only used for communication between systems (M2M).

**Motivation:** The authentication is necessary to doubtless identify a user because the allocated authorization, and therefore the access on data and services of the system depends on that.

For this requirement the following threats are relevant:

- Unauthorized access to the system
- Unauthorized use of services or resources
- Denial of executed activities

**Req 29** If the network device is operated in an insecure environment, the restoration or bypassing of start and system passwords must be prevented.

Many network devices have a function that resets the current system password (password reset). For network devices operated in public areas or areas in which access cannot be
controlled, this function shall be disabled. If this should not be possible, it is to be ensured that after resetting the password, an attacker cannot access the configuration of the network device, in which the configuration is irrecoverably deleted.

**Motivation:** An attacker with access to a network device could reset or bypass the system password if this measure was not complied with, and thus gain unauthorized access to the relevant network device, its configuration or the networks connected to it. Furthermore, analysis of the configuration could provide information that could be used for attacks on other systems.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data

---

**Req 30** Accounts must be used that allow unambiguous identification of the user.

Users must be identified unambiguously by the system. This can typically be reached by using a unique account per user. So named group accounts, i.e. the use of one account for several persons, must not be used. On exception of this requirement are so named machine accounts. These will be used for authentication and authorization from system to each other or for applications on a system and can’t be assigned to a single person. Such accounts must be assigned on a per system or per application basis. In this connection, it has to be guaranteed that this account can’t be misused. Possibilities to protect these accounts are:
- Configuring of a Password that fulfils the security requirements and is known by less than possible circle of administrators.
- Configuring the account that only a local use is possible and a interactive login isn’t possible.
- Use of a technique for authentication of the specific account with public and private key or certificates.
- Limiting the access over the network for legitimised systems.

Additional solution must be checked on their usability per individual case.

**Motivation:** Unambiguous user identification is a prerequisite for assigning a user the rights that he requires to perform his tasks on the system. This is the only way to adequately control access to system data and services and to prevent misuse. Furthermore, it makes it possible to log activities and actions on a system and to assign them to individual users.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized use of services or resources
- Denial of executed activities

---

**Req 31** Accounts must be protected against unauthorized use by at least one authentication attribute.

The various user and machine accounts on a system must be protected from misuse. To this end, an authentication attribute is typically used, which, when combined with the user name, enables unambiguous authentication and identification of the authorized user.

Authentication attributes include:
- Cryptographic keys
- Token
- Passwords
- PINs

This means that authentication based on a parameter that can be spoofed (e.g. phone numbers, public IP addresses or VPN membership) is not permitted. Exceptions are attributes that cannot be faked or spoofed by an attacker. Several of the above options can be combined
Motivation: Accounts that are not protected with a secret authentication attribute can be used by an attacker to gain unauthorized access to a system and the data and applications stored on it.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized use of services or resources
- Denial of executed activities

Req 32 Accounts with extensive rights must be protected with two authentication attributes.

Accounts, for example used for administration, maintenance and troubleshooting, have extensive rights. Therefore a single protection (e.g. a password), as for normal user accounts with less rights, is not suitable. To get a higher protection level it is necessary to use more than one authentication attribute. For this a combination of an attribute that the user knows and an attribute that the user owns will be used often. This kind of authentication will be named as 2-factor authentication. Examples for 2-factor authentication are:
- Smartcard with PIN
- Private key with Passphrase
- Secure-ID Token with Password

Motivation: Accounts with extensive rights as used for system administration have a higher risk for system’s security. An attacker can get extensive rights by compromising such an account to get access to wide parts of the system and stored data.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized use of services or resources
- Denial of executed activities

Req 33 Predefined accounts must be deleted or disabled.

Many systems have default accounts (e.g. guest, ctxsys), some of which are preconfigured with or without known pass-words. These standard users must be deleted or disabled. Should this measure not be possible the accounts must be locked for remote login. In any case disabled or locked accounts must configured with a complex password (12 character and more, use of upper/lower case, numbers and special characters). This is necessary to prevent unauthorised use of such a account in case of misconfiguration.

Exceptions to this requirement to delete or disable accounts are accounts that are used only internal on the system in-volved and that are required for one or more applications on the system to function. Also for this accounts remote ac cess or local login must be forbidden to prevent a abusive use by users of the system.

Motivation: Standard users are typically generally known and can be used by an attacker for targeted brute force and dictionary attacks. Standard user accounts represent a special risk if they do not use a password or only use a stand-ard password that is generally known. Such standard user accounts can easily be exploited by an attacker in order to gain access to the system involved without being authorized to do so.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized use of services or resources
- Denial of executed activities
Req 34  Predefined authentication attributes must be deleted or disabled.

Normally, authentication attributes such as password or cryptographic keys will be preconfigured from producer, vendor or developer of a system. Such authentication attributes must be changed an own attribute not known by other parties.

Motivation: Authentication attributes like password or cryptographic keys preconfigured from third parties are not trustable. Such authentication attributes can be used to compromise systems or their data.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized use of services or resources
- Denial of executed activities

Req 35 The authorizations for accounts and applications must be reduced to the minimum required for the tasks they have to perform.

Authorizations to a system must be restricted to a level in which a user can only access data and use functions that he needs in the course of his work. Suitable authorizations must also be assigned for access to files that are components of the operating system or of applications or that are generated by the same (e.g. configuration and logging files).

Alongside access to data, execution of applications and components must also take place with rights that are as low as possible. Applications should not be executed with administrator or system rights.

Motivation: If the rights granted to a user on a system are too broad, it could be possible for him to access data and applications which he is not permitted to view or use. This would give him the opportunity to disclose or modify confidential data and to manipulate system files. Applications with rights that are too broad can be used by a user to extend his own authorizations and thus to gain access to files and system components to which he would not have had access with his authorizations under normal circumstances.

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources

Req 36 The system must be connected to a central system for user administration.

For provisioning of these authorization information must the system provide a central inter-face (e.g. LDAPs for authorization, Kerberos for authentication, locking information for certificates) or decentralized mechanisms (e.g. public-key authentication). A central solution for identity management must be preferred.

In areas where a central identity management system is not available a central system such as LDAP, TACACS+ or Radius server for the administration of accounts and their authentication and authorization must be used.

Motivation: Central administration of identity of accounts and their rights means that they only have to be maintained once instead of separately on each system. From the aspect of security, the advantage is that an account and its rights only known on a single central side. This information can be transmitted from a central side to systems (provisioning), central administrated (reconciliation) and central deleted (deprovisioning). This reduces the risk of accounts being for-gotten during changing or deletion since they are configured on multiple systems. This could give a user wrong system rights or continued access to a system.
3.23 Routers and Switches

For this requirement the following threats are relevant:
• Unauthorized access to the system
• Unauthorized access or tapping of data
• Unauthorized modification of data
• Unauthorized use of services or resources

Protecting sessions

Req 37 Sessions must be protected against high jacking.

A function must be implemented for all user sessions to prevent other users assuming or pursuing control of the sessions of an authorized user. Such protection can be achieved by implementing the following measures or a combination of these measures:
• Using an appropriate cryptographic algorithm
• On network level: Use of TCP protocol (with sequence number) and filter lists.
• On transport level: SSL/TLS
• On application level: Negotiation of a random, secret value between sender and recipient (e.g. session ID, sequence number, time stamp)

Motivation: An attacker who gains knowledge about a session which is not protected against assumption of control can continue to participate in such a session and thus gain unauthorized access to the system involved. One example is an attacker who records the session cookie for a Web application during non-encrypted communication. The attacker can then view and take control of the session of the original user.

For this requirement the following threats are relevant:
• Unauthorized access to the system
• Unauthorized access or tapping of data
• Unauthorized modification of data
• Unauthorized use of services or resources
• Denial of executed activities

Req 38 It must be possible that users can logoff their session.

The system must have a function that allows a signed in user to logout at any time.

Motivation: A user of a system must have the possibility to protect a session and therefore its data against unauthorized access. Therefore a logout function must be available that can be used to end a session. This prevents that the current session will stay open and can be reactivated and used by an unauthorized person.

For this requirement the following threats are relevant:
• Unauthorized access to the system
• Unauthorized access or tapping of data
• Unauthorized modification of data
• Unauthorized use of services or resources
• Denial of executed activities

Req 39 Session must be terminated after an adjusted period of inactivity for the purpose of use.

It is necessary that a session will be terminated automatically after a specified time of inactivity. For this reason a time-out must be set for sessions. The selected period depends on use and if applicable the physical environment. This means for example that a time-out for an application in an unsecure environment must be shorter (less minutes) than the time out for an application used of operational staff for monitoring tasks used in an protected area (60 minutes and longer).
Motivation: In case of an open or unused session exists the possibility that an unauthorized user can high-jack and use this session to get access to the effected system and its data. Furthermore open session will allocate system re-sources.

For this requirement the following threats are relevant:
• Unauthorized access to the system
• Unauthorized access or tapping of data
• Unauthorized modification of data
• Unauthorized use of services or resources
• Denial of executed activities
Authentication parameter password

Req 40 If a password is used as an authentication attribute, it must have at least 8 characters and contain three of the following categories: upper cases, lower case, numbers and special characters.

A system may only accept passwords that comply with the following complexity:

- Minimum length of 8 characters.
- Comprising at least three of the following categories: upper/lower case letters, numbers and special characters

When a password is assigned, the system must ensure that the password meets these requirements. If a central system will be used for user authentication this function can be forwarded or delegated to this system.

Motivation: Passwords with the above complexity offer high robustness against attacks coupled with acceptable user friendliness. Passwords with this level of complexity have proved their efficiency in practice. Trivial passwords that are too short are susceptible to brute force and dictionary attacks and are therefore easy for attackers to determine. Once a password has been ascertained it can be used by an attacker for unauthorized access to the system and the data on it.

For this requirement the following threats are relevant:

- Unauthorized access to the system
- Unauthorized use of services or resources
- Denial of executed activities

---

Req 41 If a password is used as an authentication attribute, users must be enable to independently change the password anytime.

The system must offer a function that enables a user to change his password at any time.

When an external centralized system for user authentication will be used it is possible to redirect or implement this function on this system.

Motivation: The fact that a user can change his authentication attribute himself at any time enables him to change it promptly if he suspects that it could have been accessed by a third party.

For this requirement the following threats are relevant:

- Unauthorized access to the system
- Unauthorized use of services or resources
- Denial of executed activities

---

Req 42 If a password is used as an authentication attribute, a protection against brute force and dictionary attacks that hinder password guessing must be implemented.

Brute force and dictionary attacks aim to use automated guessing to ascertain passwords for user and machine accounts. Various measures or a combination of this measures can be taken to prevent this.

The most commonly used protection measures are:

- Increasing the delay (e.g. doubling wait times at each attempt) for each renewed password input following an incorrect entry (“tar pit”).
3.23 Routers and Switches

- Blocking an account following a specified number of incorrect attempts (typically 5). However, it has to take into account that this solution needs a process for unlocking and an attacker can force this to deactivate accounts and make them unusable.
- Using CAPTCHA to prevent automated attempts (often used for Web applications).

In order to achieve higher security, it is often meaningful to combine two or more of the measures named here. This must be evaluated in individual cases and implemented accordingly.

Motivation: Implementation of one or more of the above measures can prevent successful execution of brute force and dictionary attacks to ascertain passwords.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized use of services or resources
- Denial of executed activities

---

**Req 43** If a password is used as an authentication attribute, it must be hidden when displayed on screen.

When a user enters a password or the password is otherwise displayed on screen, it must be made unreadable. Typcially, the individual characters of the password are replaced by a character such as “*”. Under certain circumstances it may be permissible for an individual character to be displayed briefly during input. Such a function is used, for example, on Smartphone’s to make input easier. However, the entire password is never output to the display in plaintext.

Motivation: To prevent another person reading a password by chance or intentionally on screen during input, the pass-word must be rendered unreadable during input or display.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized use of services or resources
- Denial of executed activities
Logging

Req 44  Security-relevant events must be logged with a precise time stamp and a unique system reference.

Network elements must log the occurrence of security-relevant events. So that these events can be evaluated and classified, they must be logged together with a unique system reference (e.g., host name, IP or MAC address) and the exact time the event occurred.

When logging, the applicable statutory, collective agreement and operating provisions must be taken into account; these include the statement that the logging of events may only take place for the intended purpose. Logging events in order to exploit these for the control of employees’ work is not permitted.

The following security-relevant events must be logged by a network element:

<table>
<thead>
<tr>
<th>Event</th>
<th>Event data to be logged</th>
</tr>
</thead>
<tbody>
<tr>
<td>Failed login attempts</td>
<td>• Account,</td>
</tr>
<tr>
<td></td>
<td>• No. of failed attempts,</td>
</tr>
<tr>
<td></td>
<td>• Source (IP address) of remote access</td>
</tr>
<tr>
<td>Changes to configuration</td>
<td>• Change made,</td>
</tr>
<tr>
<td></td>
<td>• User</td>
</tr>
<tr>
<td>Reboot/shutdown/crash etc.</td>
<td>• Action performed (reboot, shutdown,</td>
</tr>
<tr>
<td></td>
<td>• User (for intentional actions)</td>
</tr>
<tr>
<td>Change to the status of interfaces</td>
<td>• Interface name and type,</td>
</tr>
<tr>
<td>(e.g., shutdown)</td>
<td>• Status (shutdown, missing link, etc.)</td>
</tr>
<tr>
<td>Critical rise in system values</td>
<td>• Value exceeded,</td>
</tr>
<tr>
<td>such as high memory or CPU load over</td>
<td>• Value reached</td>
</tr>
<tr>
<td>a longer period</td>
<td>(Here suitable threshold values must be defined depending on the individual system.)</td>
</tr>
<tr>
<td>Change to neighbourhood relationships (routing)</td>
<td>• Protocol</td>
</tr>
<tr>
<td>Port security violation (switch)</td>
<td>• IP address of the neighbour</td>
</tr>
<tr>
<td>Identification of fake BPDU packets</td>
<td>• Port name</td>
</tr>
<tr>
<td>in the network (switch)</td>
<td>• MAC address of the triggering system</td>
</tr>
<tr>
<td></td>
<td>• MAC address of the sending system</td>
</tr>
</tbody>
</table>

Suitable thresholds are to be defined depending on the system type and hardware used. Logging of additional security-relevant events may be appropriate. This shall be verified in individual cases and implemented accordingly where required.

**Motivation:** Logging security-relevant events is a basic requirement for detecting ongoing attacks as well as attacks that have already occurred. This is the only way in which suitable measures can be taken to maintain or restore system security. Furthermore, the logging data is used as evidence so that legal action can be taken against attackers.

For this requirement the following threats are relevant:
• Denial of executed activities
• Unnoticeable feasible attacks
Req 45  Security relevant logging data must be send to an external system direct after their creation.

Logging data must be forwarded to an external system in appropriate logging files as well as being stored locally. Standard protocols like Syslog, SNMPv3 must be preferred.

Motivation: If logging data is only stored locally it can be manipulated by an attacker who succeeds in compromising the system in order to conceal his attack and any manipulation he has performed on the system. This is the reason why the forwarding must be done immediately after the event occurred.

For this requirement the following threats are relevant:
• Denial of executed activities
• Unnoticeable feasible attacks

Layer 2 protocols

Req 46  VLAN 1 must not be used.

VLAN 1 is the default VLAN at some manufacturers. This VLAN cannot be deleted or disabled and is automatically assigned to all interfaces which are not mapped to another VLAN. VLAN 1 shall not be used. In addition, VLAN 1 shall not be enabled on trunks.

Motivation: If VLAN 1 is used, an improper configuration may result in a system not intended for this inadvertently gaining access to systems and data in this VLAN as it is automatically mapped to this VLAN. This can become particularly problematic if VLAN 1 is used, for example, to manage the switch.

For this requirement the following threats are relevant:
• Unauthorized access or tapping of data

Trunk

Req 47  A native VLAN must be configured for trunk ports.

In the IEEE 802.1q standard, the packets of the native VLAN are transmitted without tagging. Packets which do not belong to a VLAN due to lack of tagging are mapped to the native VLAN. For this reason, the native VLAN may only be configured on trunks for the transmission of management information between network devices which are connected via this trunk. VLAN 1 shall not be used as a native VLAN. Its use for the transmission of user data is not permitted. Please note that the same VLAN shall be used as “native” on both systems of a trunk.

Motivation: When using the native VLAN for data communication, it may be possible to overcome the logical separation of VLANs using VLAN hopping attacks and thus feed data into a VLAN without authorization.

For this requirement the following threats are relevant:
• Unauthorized access or tapping of data

Req 48  When using VTP, the partners involved must authenticate each other.

The Virtual Trunking Protocol (VTP) is used for automatic dissemination of VLAN information between network elements. In order to prevent any tampering with the VLAN configuration, VTP should be disabled. If VTP is to be used, the parties involved must authenticate each other.
Motivation: An attacker can use the VTP protocol to tamper with the trunk and VLAN configuration of a network element's port in order to gain unauthorized access to VLANs.

For this requirement the following threats are relevant:
• Unauthorized access or tapping of data

Access ports

Req 49 The private VLAN function must be enabled on access ports.

The private VLAN function prevents direct communication between devices which are connected to a switch. As a result, this function should be enabled (in isolated mode) on access ports. It is possible to ignore this requirement if a communication between systems is needed.

The private VLAN function may not be enabled on trunks and uplinks. Private VLANs represent protection on OSI layer 2 only. As a result, it may be possible to bypass this protection on OSI layer 3. A check should therefore be performed in each case to establish whether additional measures to safeguard communication between devices needs to be implemented on layer 3.

Motivation: Communication between the systems that are connected to a switch can be prevented on layer 2 by means of the private VLAN function. As a result, attacks on other systems and the recording of data traffic can be prevented to a large extent.

For this requirement the following threats are relevant:
• Unauthorized access or tapping of data

Req 50 Access ports must not be configured as trunks.

Trunks are used to exchange data of multiple VLANs between network elements. The individual VLANs are marked and differentiated through tagging. Access ports to devices may not be configured as trunks. Systems such as servers with which data from multiple VLANs is to be exchanged explicitly are excluded from this.

Motivation: An attacker who can connect to a trunk port can access all VLANs that are accessible via this trunk and reach systems in these VLANs in order to attack them.

For this requirement the following threats are relevant:
• Unauthorized access or tapping of data

Req 51 DTP must be disabled on access ports.

DTP (Dynamic Trunking Protocol) offers the opportunity to negotiate the trunk status between two network elements that are connected to each other. This protocol must be disabled on access ports. Devices such as servers with which data from multiple VLANs is to be exchanged explicitly are excluded from this.

Motivation: An attacker who is connected to a port and is active on the DTP can use the protocol to configure the port as a trunk and thus possibly gain unauthorized access to VLANs.

For this requirement the following threats are relevant:
• Unauthorized access or tapping of data
• Disruption of availability

Req 52 Protection against fake BPDUs must be used on access ports.

The Spanning Tree Protocol (STP) is a layer-2 protocol that is to prevent endless loops in networks. Endless loops can arise in redundant network paths. A "root bridge" is defined using BPU (Bridge Protocol Data Units) packets and on the basis of a bridge ID. Since there is no authentication in the STP, an attacker can influence the selection of the root bridge. In order to
3.23 Routers and Switches

prevent such attacks, protection against fake BPDUs, such as BPDU Guard, Root Guard or similar access lists must be enabled.

Motivation: Using fake BPDU packets, an attacker can take up the position of the "root bridge" in order to divert data traffic via himself.

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data

---

Req 53  Port security must be enabled on access ports.

Port security should be enabled on access ports to devices. This requirement is especially for access ports used to connect devices like workstations and printers etc. The requirement can be ignored for ports used to connect servers.

This function ensures that only legitimate systems whose MAC address is released for access can connect to the net-work. If a system with an unknown MAC address is connected to the port, the port should be permanently blocked un-till it is manually unblocked by an administrator.

In the configuration less as possible MAC addresses should be al-lowed. Furthermore security is higher with static MAC addresses than with learned MAC addresses. In single cases it can be necessary to learn the MAC address when a device is connected for the first time. In this cases it must be en-sured that this function can not be used abusive.

Motivation: Enabling port security helps to prevent illegitimate systems from being connected to the network. Port se-curity also constitutes effective protection against attacks such as:
  - CAM table flooding – With this attack, an overflow of the MAC table (CAM table) and possibly a crash of the switch concerned is provoked through manipulated ARP packets with fake MAC addresses.
  - DHCP starvation attack – With this attack, all DHCP leases are used up with the help of fake MAC addresses. The aim of this attack is that no further IP addresses are available to connect additional systems.

For this requirement the following threats are relevant:
  - Unauthorized access to the system
3.24 MySQL /MariaDB database systems

Introduction

This security requirement has been prepared based on the provisions of the Group Policy on IT/NT Security.

The security requirement is used as a basis for an approval in the PSA process, among other things. It also serves as an implementation standard for provisions of the Group Policy on IT/NT Security in units which do not participate in the PSA process. These requirements shall be taken into account from the very beginning, including during the planning and decision-making processes. When implementing these security requirements, the precedence of national, international and supranational law shall be observed.

General requirements for MySQL / Maria database systems

Req 1 The MySQL-DB / MariaDB version must be in the active or extended lifecycle period as per the MySQL Lifecycle Policy or in case of Maria DB official supported by commercial supporting companies.

An overview of the current MySQL Lifecycle Policy can be found at http://www.mysql.com/about/legal/lifecycle/. In addition, the platforms supported in the MySQL Enterprise Edition must be within the MySQL lifecycle (active support or extended support). A directory of the platforms supported can be found at: http://www.mysql.com/support/supportedplatforms/enterprise.html. As it stands (May 2011), version 5.0 or higher applies.

Motivation: Oracle offers special support agreements as part of lifecycle support. They are combined with services for fault clearance and the clearance of security vulnerabilities.

Implementation example: The MySQL-DB version installed and its type (Community or Enterprise) can be queried using the following commands:

```sql
show variables like "%version%";
status;
select version();
```

For this requirement the following threats are relevant:

- Unauthorized access or tapping of data
- Unauthorized modification of data
- Disruption of availability
- Unnoticeable feasible attacks

Database system hardening

This section defines the requirements for hardening the database system. The measures are comparable with those for operating system hardening. They reduce the likelihood of successful attacks on database systems and/or the consequences thereof. On UNIX/Linux operating systems, the “mysql_secure_installation” script is provided for hardening MySQL-DB. This bundles hardening measures such as:

- Setting a password for the super user account of the MySQL database system.
- Setting the login rights to local host only for the MySQL ROOT user.
- Deleting anonymous user accounts.
- Deleting the “test” database and its privileges.
It is recommended that the MySQL-DB is run in a chroot-environment. This can be accomplished by either starting with the option \--chroot or adding the making the relevant entry in the my.conf file.

**Req 2** The "test" database must be deleted.

*Motivation:* Any unnecessary information on the system presents a security risk. Attackers could gain additional information about the system from this.

Implementation example: Checking for the existence of and deleting the "test" database:

```
show databases like "test"; drop database "test";
```

On Unix/Linux operating systems, running the "mysql_secure_installation" script when the default database is installed deletes the "test" database. Running the http://dev.mysql.com/doc/refman/5.5/en/mysql-secure-installation.html script on Linux generally hardens the database and implements the following measures:

- Setting a password for the super user account of the MySQL database system.
- Setting the login rights to local host only for the MySQL ROOT user.
- Deleting anonymous user accounts.
- Deleting the "test" database and its privileges.

For this requirement the following threats are relevant:

- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources

**Req 3** It must be ensured that no user accounts without a user name exist (anonymous accounts).

"Anonymous accounts" are user accounts with a blank ("") user name.

*Motivation:* By default, logins to these user accounts are possible and the rights to these can be used by other users under certain circumstances.

Implementation example: Running the SQL command DROP USER ""; deletes relevant user accounts. On Unix/Linux operating systems, running the "mysql_secure_installation" script during the installation deletes the anonymous accounts.

Check: The use of anonymous accounts can be checked using the following SQL command:

```
select user from mysql.user where user = "";
```

For this requirement the following threats are relevant:

- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources

**Req 4** Only one instance of the MySQL / Maria database system must be installed on one operating-system instance (hardware platform or virtualization guest).

*Motivation:* If multiple database instances for different tasks (e.g., Internet and intranet) are running on one operating-system instance, the instances are not separated from one another. There is a risk that attackers could also corrupt the second database system. To separate the individual database instances deploy virtualization solutions.
For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources

---

**Req 5**  
(Default) passwords on (SUPER) user accounts must be changed.

With a standard installation on a Unix/Linux or Windows system, the system creates a super user account without a password or “root”.

*Motivation: Default passwords for databases present a high security risk. The administrator therefore needs to change these.*

Implementation example: On Unix/Linux operating systems, running the "mysql_secure_installation" script sets the SUPER user account (root) password. The following SQL command can be used to check whether user accounts without passwords exist:

```sql
select user, password from mysql.user where length (password) = 0 or password = ""
```

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources

---

**Req 6**  
MySQL / MariaDB must be started with the Option --safe-user-create.

It is important to prevent the GRANT process for a new user with a blank password. The administrator must therefore start the MySQL daemon using the following option:

```
--safe-user-create
```

*Motivation: Blank passwords for databases present a high security risk. The administrator therefore needs to change these.*

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources
Req 7  The start option “old-passwords” must not be used.

*Motivation: The administrator must use the highest available hash strength in order to make brute force attacks difficult. Short password hashes (such as before version 4.1), on the other hand, have minimal security.*

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources

Req 8  The start option secure-auth must be used.

*Motivation: The administrator must use the strongest hashing algorithm available to block brute force attacks. Short hashes (as existing before version 4.1) do not offer sufficient security.*

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources
- Unnoticeable feasible attacks

Req 9  The variable MYSQL_PWD must not be used when saving passwords and/or in scripts.

*Motivation: The use of MYSQL_PWD is highly insecure since the password can be read out with the command “ps”.*

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources
- Unnoticeable feasible attacks

Req 10 If used with Unix/Linux the database daemon must run with a dedicated, non-administrative Unix/Linux account which is used on the system only for the database.

*Motivation: The DB daemon must not run with root rights since otherwise every DB user can create files with FILE privileges.*

Implementation example: The read and write access to the database directory must be restricted to database users. If the daemon starts with the dedicated mysql user account, the following entry should be made in the my.cnf file:

```
[mysqld]
user=mysql
```

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources
Req 11 If used with Windows, the MySQL-DB / MariaDB must run with restricted privileges under a net-work service account.

*Motivation:* The MySQL user, as the network service account, only has restricted rights to the operating system. This reduces the impact of security gaps in the database.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources
- Unnoticeable feasible attacks

Req 12 Unless it is needed, the symlink function must be deactivated.

The symlink function enables a user to access the server’s data directory.

*Motivation:* Every user with write access to the server’s data directory can delete any files on the system.

Implementation example: To deactivate the symlink function, the administrator must start the MySQL-DB using the following option,

```
--skip-symbolic-links
```

The status of the symlink function can be queried using the SQL command show variables like "have_symlink";
This is deactivated if the system returns “Disabled” as the result.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data

Req 13 The FILE privilege must only be restricted to the super user account.

*Motivation:* Users with FILE privileges can read all files on the host which can also be read by the MySQL server. Thus a user can create new files in every directory to which the MySQL server has write access.

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data
- Unauthorized modification of data

Req 14 The read and write access for the FILE privilege must be limited to a defined directory.

*Motivation:* Users with FILE privileges can read all files on the host which can also be read by the MySQL server. Thus a user can create new files in every directory to which the MySQL server has write access.

Implementation example: The start option,

```
--secure-file-priv
```

limits the read and write access of LOAD FILE and LOAD DATA, as well as select…-into outfile commands to a pre-defined directory.

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data
- Unauthorized modification of data
3.24 MySQL/MariaDB database systems

Req 15  The database must be started with the option \texttt{--local-infile=0}.

\textit{Motivation: With this option, no user can transfer files from the client system to the server via LOAD DATA}

Implementation example: The administrator must start the MySQL-DB with the following option (or transfer to the configuration file my.cnf):

--local-infile=0

For this requirement the following threats are relevant:

- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources
- Unnoticeable feasible attacks

Req 16 It must be ensured that only the MySQL/MariaDB user account is granted read and write access to all MySQL/MariaDB data and its subdirectories, as well as to log files and database files.

Depending on the installation type, these directories are available for standard Windows or Unix binary installations: $MYSQL\_HOME/data; Linux RPM distribution: /var/lib/mysql; source distribution: /usr/local/var.

\textit{Motivation: Restricting the file permissions means that the information gained after successfully taking over the system is significantly reduced.}

For this requirement the following threats are relevant:

- Unauthorized access or tapping of data
- Unauthorized modification of data

Req 17 It must be ensured that only the MySQL/MariaDB users and authorized administrators have read and write access to query and binary log files.

The system saves executed commands of the MySQL-DB (system, DDL, DML, DCL and SELECT) in the log files without encryption.

\textit{Motivation: Restricting the file permissions means that the information gained after successfully taking over the system is significantly reduced.}

For this requirement the following threats are relevant:

- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized use of services or resources

Req 18 If used with Linux, it must be ensured that only the MySQL/MariaDB users have read and write access to the file \texttt{~mysql/.mysql\_history}.

\textit{Motivation: The file \texttt{~mysql/.mysql\_history} contains unencrypted passwords and DDL commands. If the user accounts are managed via SQL commands such as CREATE USER, GRANT and SET PASSWORD, the passwords appear in plain text}

For this requirement the following threats are relevant:

- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources
MySQL / Maria -specific requirements

Req 19 It must be ensured that only administrative accounts have access to the MySQL-MariaDB "user table".

**Motivation:** The user table contains all user information. It is therefore extremely important that it is specially protected. Only then can further attacks on the system be reduced.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources

Req 20 It must be ensured that only authorized accounts have access to the MySQL / Maria database tables.

**Motivation:** The database table contains all database information. It is therefore extremely important that it is specially protected. Only then can further attacks on the system be reduced.

Implementation example: The administrator must start MySQL-DB with the following option:
```
--skip-show-database
```
This option can also be added to the file my.cnf. With this option, only those users with the "show databases" authorization may use the instruction "show databases". This privilege should therefore only be granted to specific users.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized access or tapping of data

Req 21 The MySQL / Maria database authorization system must not be deactivated.

**Motivation:** Only if a rights system is implemented is it possible to protect the data and the system as a whole in the long term.

Implementation example: It must be ensured that the database system is NOT started using the following option:
```
--skip-grant-tables
```
This start option deactivates the entire authorization system so that every database user is granted unrestricted access to all databases. If the procedure for restoring the root password is required, the start options must only be changed by the system administrator. It is therefore important to monitor changes in the configuration files as well as daemon re-starts using the modified settings.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources

Req 22 The GRANT option "WITH GRANT" must not be used.

Using the option "WITH GRANT" means that the user with the relevant privilege can transfer his privileges to other users or roles.

**Motivation:** Only if a rights system is implemented is it possible to protect the data and the system as a whole in the long term. Forwarding your own rights increases the risk of the unauthorized use of data.
Implementation example: The users with this privilege can be listed using the following command:

```
select user, host from mysql.user where grant_priv ="y";
```

For this requirement the following threats are relevant:

- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources

---

Req 23 Predominantly administrative privileges must only be assigned for DB administrator accounts (principle of least privilege).

The granting of the following privileges must remain restricted to the DB administrator account:

- SUPER
- PROCESS
- SHUTDOWN
- CREATE USER
- RELOAD
- SHOW DATABASES
- GLOBAL GRANT OPTION
- CREATE TEMPORARY TABLES
- LOCK TABLES

Motivation: Non-administrative users who have such privileges may jeopardize the confidentiality, integrity and availability of the MySQL DB.

Implementation example: With the command show grants an administrator can check the privileges granted. A user with these privileges could procure a list of executable queries which contain sensitive data such as passwords. Furthermore, he can get read and write access to all files in the mysqld process or configure the DB differently.

For this requirement the following threats are relevant:

- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources

---

Req 24 The “privilege tables” must be checked after every upgrade.

Motivation: Sometimes privileges are added to new MySQL versions, which did not exist in the previous versions. For this reason, the administrator must check the privilege tables for correct assignment of privileges after every upgrade. This also means that incompatibilities with the current MySQL version can be identified.

Implementation example: MySQL has a script for checking the tables and privileges after an upgrade. An executable version of “mysql_upgrade” will be installed on all platforms in all MySQL versions higher than 5.1.10. The script checks/repairs the tables (mysqcheck) and checks the privilege tables (mysq_fix_privilege_tables). With older versions the administrator must issue these commands manually. A description of all commands can be found at http://dev.mysql.com/doc/refman/5.5/en/mysql-upgrade.html.

For this requirement the following threats are relevant:

- Unauthorized access to the system
3.24 MySQL /MariaDB database systems

- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources

---

**Req 25** The start option `--allow-suspicious-udfs` must be deactivated.

This option determines whether user-defined functions (UDFs) can be loaded at the start. This function is deactivated by default.

*Motivation:* **User defined functions can run malicious code on the system or be carrier functions for malicious code. They must therefore be deactivated.**

Implementation example: Check: It is important to check whether the following option is used in the `my.cnf` configuration file:
```
--allow-suspicious-udfs
```
If this option is selected, the administrator must deselect it.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources

---

**Req 26** It must be ensured that there are no wildcards ("%") in the host name during the user authentication.

This ensures that only connections from trustworthy sources are possible.

*Motivation:* **To reduce the risk of the accounts of compromised systems jeopardizing the database system, permissions must be stringently assigned.**

Implementation example: Check: The use of wildcards can be checked using the SQL command:
```
select user from mysql.user where host = "%";
```

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources

---

**Req 27** It must be ensured that the super user can only log in to the local host.

*Motivation:* **To reduce the risk of the accounts of compromised systems jeopardizing the database system, permissions must be stringently assigned. This includes the administration of the database itself.**

Implementation example: On Unix/Linux operating systems, running the "mysql_secure_installation" script upon installation restricts the access permissions of the super user account to the local host.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources
- Disruption of availability
- Unnoticeable feasible attacks
Req 28  The resources available per user account must be limited.

Motivation: To reduce the risk of compromised accounts jeopardizing the database system, permissions should be stringently assigned.

Implementation example: The available server resources can be limited using the variables below. This is possible both globally and for each individual user.

- MAX_QUESTIONS
- MAX_USER_CONNECTIONS
- MAX_QUERIES_PER_HOUR
- MAX_UPDATES_PER_HOUR
- MAX_CONNECTIONS_PER_HOUR

Further information can be found at http://dev.mysql.com/doc/refman/5.5/en/user-resources.html.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources
- Disruption of availability

Auditing and monitoring

Req 29  Important database services and instances must be monitored continually for misuse scenarios.

The monitoring of user actions for misuse shall comply with national legislation currently in force (for details see the “Security Requirement on Misuse Detection”).

Motivation: There are many conceivable ways to misuse database systems. Users generate an unusually high data use-age rate or operate at unusual times of day. Attackers utilize unusual and critical commands for database queries, as well as tools and malware to extend their rights. To detect misuse, database systems should be continually monitored for misuse scenarios, e.g. by means of database triggers and log monitoring.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources
3.29 Oracle Database Systems

Introduction
This security requirement has been prepared based on the provisions of the Group Policy on IT/NT Security.
The security requirement is used as a basis for an approval in the PSA process, among other things. It also serves as an implementation standard for provisions of the Group Policy on IT/NT Security in units which do not participate in the PSA process. These requirements shall be taken into account from the very beginning, including during the planning and decision-making processes. When implementing these security requirements, the precedence of national, international and supranational law shall be observed.

1.1. Responsibilities
The system owner must ensure the implementation of the security requirements from this requirements document.

These security requirements are intended for all individuals who are responsible for the development and operation of database systems, and/or who develop database systems (or have them developed) or procure them.

General Information
These security requirements contain numerous minor and major enhancements, updates and error corrections compared with the previous version. It is therefore strongly recommended that the updated security requirements are examined in detail.

Below, the terms “DBA User” and/or “DBA Account” refer to a database account which has DBA (database administrator) authorizations or takes on DBA-related activities according to the principle of segregation of functions (e.g., user administration, backup, etc.).

Basic security requirements
Version of the database used

---

Req 1 The Oracle database software must be approved by the manufacturer for productive operation and must be in in the “Premier Support” or “Extended Support” phase, in accordance with the Oracle Lifetime Support Policy.

11.1: Premier support up to August 2012, extended support up to August 2015.

11.2: Premier support up to January 2015, extended support up to January 2018.

All older versions are in the “Sustaining Support” phase, for which the manufacturer no longer provides regular security updates. This significantly increases the likelihood of an attack.

Reference:
Motivation: Only a database software version for which the manufacturer provides complete support services guarantees secure, stable operation of the product.

For this requirement the following threats are relevant:
• Disruption of availability
• Unnoticeable feasible attacks

Database hardening
This section lists the requirements for hardening the database system. The measures are comparable with those for operating system hardening. The requirements below are also used to minimize the impact of unauthorized access to database systems.

Service elimination

**Req 2** Components of the Oracle-DBMS which are not required must not be installed.

During the installation it should be ensured that only the necessary components (for example, not Apache HTTP server, APEX, Spatial, etc.) are installed.

References:
http://docs.oracle.com/cd/B28359_01/install.111/b32002/install_overview.htm
http://docs.oracle.com/cd/E11882_01/install.112/e24321.pdf

Motivation: Additional components increase the likelihood of weaknesses arising and unnecessarily increase operating costs.

For this requirement the following threats are relevant:
• Unauthorized use of services or resources

**Req 3** The Oracle HTTP server must not be installed on the DBMS server or must be deactivated.

The Oracle HTTP server must not be installed on the same operating system instance as the DBMS or must be deactivated in order to comply with the N-tier architecture.

If use of the Oracle HTTP server is required, those responsible must ensure that it is installed on a dedicated server separate from the database system.

Check:
With the command `$ORACLE_HOME/apache/apache/bin/apachectl` the administrator can check whether the server is active.

Motivation: Additional functionalities result in additional security vulnerabilities

For this requirement the following threats are relevant:
• Unauthorized access to the system
• Disruption of availability
• Unnoticeable feasible attacks

**Req 4** Oracle Application Express (APEX) must not be installed on the DBMS server or must be deactivated.

Oracle Application Express (APEX) must not be installed on the same operating system instance as the DBMS or must be deactivated in order to comply with the N-tier architecture.

Oracle Application Express is pre-installed as standard, together with the database, from Oracle 11g.

If use of Oracle Application Express is required, those responsible must ensure that it is installed on a dedicated server separate from the database system.
Check:
After installing APEX, the files are located in $ORACLE_HOME/apex and user accounts of the type FLOWS_* exist in the database.

References:
http://docs.oracle.com/cd/E14373_01/install.32/e13366/over
view.htm#i46634
http://docs.oracle.com/cd/E14373_01/appdev.32/e11838/sec
.htm http://www.oracle.com/technetwork/developer-
tools/apex/overview/index.html
http://en.wikipedia.org/wiki/Oracle_Application_Express
Motivation: Additional functionalities result in additional security vulnerabilities.

Implementation example: To uninstall APEX, administrators need to log in and run the query @apxremov.sql (see http://download.oracle.com/docs/cd/E17556_01/doc/install.40/e15513.pdf, p. A-6)

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unnoticeable feasible attacks
- Attacks motivated and facilitated by information disclosure or visible security weaknesses

---

Req 5 Only one instance of the Oracle database system must be installed on one operating-system instance (hardware platform or virtualization guest).

Motivation: Motivation: If server hardware is used multiple times by multiple database systems, the risk increases of a larger group of people obtaining unauthorized access to systems for which they are not responsible technically or in terms of administration.

Implementation example: If multiple database instances for different tasks (e.g., Internet and intranet) are running on one operating-system instance, the instances are not separated from one another. There is a risk that attackers could also corrupt the second database system. To separate the individual database instances deploy virtualization solutions.

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data
- Unauthorized use of services or resources
- Disruption of availability

Default schemata

Users and roles

---

Req 6 (Default) user accounts and (default) roles must be deleted if these are not required.

Where this is not possible or sensible, the user status must be set to "EXPIRED & LOCKED".

NB: Database user accounts with the "EXPIRED & LOCKED" status can continue to perform activities in the database instance even though they are blocked. An active audit, where applicable, is also of no benefit here since such activities are not recorded. The blocking procedure is therefore associated with the risk of hidden misuse (automated jobs, for example). This means that blocking users is always the less satisfactory alternative.

The following default users are not required in a production environment:

ADAMS
ADLDEMOANDY
BLAKE
CDEMOCOR
CDEMORID
CDEMOUCB
CDEMO82
CLARK
DEMO
DIANE
FROSTY
HLW
HR
Unless they are required, the following users should have the status “EXPIRED & LOCKED”:

<table>
<thead>
<tr>
<th>Account</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>ANONYMOUS</td>
<td>Oracle XML DB</td>
</tr>
<tr>
<td>CTXSYS</td>
<td>CTXSYS</td>
</tr>
<tr>
<td>DBSNMP</td>
<td>Oracle Enterprise Manager</td>
</tr>
<tr>
<td>DIP</td>
<td>Oracle Label Security</td>
</tr>
<tr>
<td>DSSYS</td>
<td>Dynamic Services Engine</td>
</tr>
<tr>
<td>EXFSYS</td>
<td>Rules Manager &amp; Expression Filter</td>
</tr>
<tr>
<td>FLOWS_30000</td>
<td>Oracle Database Application Express</td>
</tr>
<tr>
<td>FLOWS_FILES</td>
<td>Oracle Database Application Express</td>
</tr>
<tr>
<td>LBACSYS</td>
<td>Oracle Label Security</td>
</tr>
<tr>
<td>MDDATA</td>
<td>Oracle Spatial</td>
</tr>
<tr>
<td>MDSYS</td>
<td>Oracle Spatial/Oracle Multimedia Locator</td>
</tr>
<tr>
<td>MGMT_VIEW</td>
<td>Oracle Enterprise Manager Database Control</td>
</tr>
<tr>
<td>ODM</td>
<td>Oracle Data Mining</td>
</tr>
<tr>
<td>ODM_MTR</td>
<td>Oracle Data Mining</td>
</tr>
<tr>
<td>OLAPSYS</td>
<td>OLAP</td>
</tr>
<tr>
<td>ORDDATA</td>
<td>Oracle Multimedia DICOM</td>
</tr>
<tr>
<td>ORDPLUGINS</td>
<td>Oracle Multimedia</td>
</tr>
<tr>
<td>ORDSYS</td>
<td>Oracle Multimedia</td>
</tr>
<tr>
<td>OUTLN</td>
<td>plan stability</td>
</tr>
<tr>
<td>ORACLE_OCM</td>
<td>Oracle Configuration Manager</td>
</tr>
<tr>
<td>OWBSYS</td>
<td>Oracle Warehouse Builder</td>
</tr>
<tr>
<td>REPADMIN</td>
<td>Replication user</td>
</tr>
</tbody>
</table>
3.29 Oracle Database Systems

<table>
<thead>
<tr>
<th>SI_INFORMTN_SCHEMA</th>
<th>SQL/MM Still Image Standard</th>
</tr>
</thead>
<tbody>
<tr>
<td>SPATIAL_CSW_ADMIN_USR</td>
<td>Oracle Spatial CSW Cache Manager</td>
</tr>
<tr>
<td>SPATIAL_WFS_ADMIN_USR</td>
<td>Oracle Spatial WFS Cache Manager</td>
</tr>
<tr>
<td>TRACESVR</td>
<td>Oracle Trace Server</td>
</tr>
<tr>
<td>WK_TEST</td>
<td>Oracle Ultra Search</td>
</tr>
<tr>
<td>WKPROXY</td>
<td>Oracle Ultra Search</td>
</tr>
<tr>
<td>WKSYS</td>
<td>Oracle Ultra Search</td>
</tr>
<tr>
<td>WMSYS</td>
<td>Oracle Workspace Manager</td>
</tr>
<tr>
<td>XDB</td>
<td>Oracle XML DB</td>
</tr>
<tr>
<td>XS$NULL</td>
<td>internal</td>
</tr>
</tbody>
</table>

References:
Sample schemas 11.2 http://download.oracle.com/docs/cd/E11882_01/server.112/e10831.pdf
Default passwords and their hash codes http://www.petefinnigan.com/default/default_password_list.htm
List of default users http://www.orafaq.com/wiki/List_of_default_database_users
Securing Oracle Database User Accounts http://docs.oracle.com/cd/E11882_01/server.112/e10575/tdpsg_user_accounts.htm

Motivation: Installation programs of database systems automatically set up a large number of users and roles which are not needed by the people who use the database. These roles are envisaged, e.g., for practice and test databases. Knowledge of default users and roles allows an attacker to gain (privileged) access to the database system. Users and roles which are not required should therefore be deleted or blocked (users only).

Implementation example: No generally applicable procedure can be specified for deleting these schemas, please refer to each one individually in “My Oracle Support” (MOS).

To set the status of a user to “EXPIRED & LOCKED”:
ALTER USER <username> ACCOUNT LOCK;
ALTER USER <username> PASSWORD EXPIRE;

Check: Show all users with status:
SELECT Username, Status FROM DBA_USERS;

For this requirement the following threats are relevant:
• Unauthorized access or tapping of data
• Unauthorized use of services or resources
• Disruption of availability

Passwords

Req 7      Default passwords must be changed.

Default passwords must be changed immediately after the system installation and before a network listener is started.

Check:
Display user accounts with default passwords:
SELECT * FROM dba_users_with_defpwd;

Command to change a password:
ALTER USER <username> IDENTIFIED BY <new password>;

References:
http://www.oracle.com/technetwork/articles/sql/11g-security-100258.html

A list of default passwords and their hash codes is available at:
http://www.petefinnigan.com/default/default_password_list.htm

Motivation: Numerous default passwords are known and enable direct access to the system or database.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized use of services or resources
Req 8 The number of failed logins for non-SYSDBA accounts must be limited for all existing profiles by the FAILED_LOGIN_ATTEMPTS parameter.

Check:
SELECT Username, Profile FROM DBA_USERS;
SELECT Profile, Resource_Name, Limit FROM DBA_PROFILES
WHERE Resource_Name = 'FAILED_LOGIN_ATTEMPTS';

Motivation: Limiting the login attempts reduces the risk of a brute force attack on passwords.

Implementation example: The FAILED_LOGIN_ATTEMPTS parameter should be set to an appropriate number for all available profiles.
ALTER PROFILE <profile name> LIMIT FAILED_LOGIN_ATTEMPTS <number>;

For this requirement the following threats are relevant:
• Unauthorized access or tapping of data
• Unauthorized use of services or resources
• Disruption of availability

Req 9 The password lock time for non-SYSDBA accounts must be defined for all existing profiles.

Check:
SELECT Username, Profile FROM DBA_USERS;
SELECT Profile, Resource_Name, Limit FROM DBA_PROFILES
WHERE Resource_Name = 'PASSWORD_LOCK_TIME';

Motivation: The setting reduces the likelihood of a brute force attack on passwords without enabling a denial-of-service attack on the accounts.

Implementation example: The administrator sets the PASSWORD_LOCK_TIME parameter for all available profiles, e.g., to 1/2,880 days (= 30 seconds).
ALTER PROFILE <profile name> LIMIT PASSWORD_LOCK_TIME <number>;

For this requirement the following threats are relevant:
• Unauthorized access to the system
• Unauthorized access or tapping of data
• Unauthorized use of services or resources
• Disruption of availability
Req 10 To ensure that strong passwords are assigned, a password verification function must be defined in all profiles.

Oracle delivers a simple example script for a password verification function (UTLPWDMG.SQL).

Check:
SELECT Profile, Limit FROM DBA_PROFILES WHERE Resource_Name = 'PASSWORD_VERIFY_FUNCTION';

These SQL statement only checks whether a password verification function exists in a profile but not the content and function.

Note:
Licence free database software, like
  • Grid / Cloud Control
  • RMAN Catalog
  • OracleVM Manager Repository

must be licensed, if own DDLs (Data Definition Language) as the password verification function will be used.

Motivation: Weak passwords present a high risk and can result in security gaps in the system. Using the function for password verification, those responsible ensure that the selected passwords meet the relevant complexity requirements.

For this requirement the following threats are relevant:
  • Unauthorized access or tapping of data
  • Unauthorized use of services or resources
  • Disruption of availability

Req 11 The function for case-sensitive passwords must be activated.

Check:
The following command shows the parameter setting.
SHOW PARAMETER SEC_CASE_SENSITIVE_LOGON;

Reference:
Creating and Maintaining a Password File
http://docs.oracle.com/cd/B28359_01/server.111/b28310/dba007.htm

Motivation: Distinguishing between upper and lower case makes it much more difficult to guess passwords by expanding the search area.

Implementation example: To do this, the password file (for SYSDBA accounts) should be created with the IGNORE-CASE parameter set to “N” ("No") and the sec_case_sensitive_logon initialization parameter (for normal users) in init.ora should be set to TRUE.

Commands for activating case sensitive passwords:
ALTER SYSTEM SET
SEC_CASE_SENSITIVE_LOGON = TRUE; orapwd
file=orapw entries=10 ignorecase=n;

Note: There is no direct way for the author to see whether or not a password file already created is case sensitive other than trying it out at login.

For this requirement the following threats are relevant:
  • Unauthorized use of services or resources
3.29 Oracle Database Systems

Req 12  Only authorized users and SYS may be entered in the password authentication file.

The password file is used to save usernames and passwords which should have the SYSDBA privilege and/or the SYSOPER privilege. The password file enables the authentication of these users even if a database instance is not running. Due to these high levels of authorization, they should be limited to a minimum number. A regular check of the user accounts included in the password file should be established.

Check:
The database view SYS.V$PWFILE_USERS shows the users in the password file:SELECT Username, Sysdba, Sysoper FROM V$PWFILE_USERS; or
SELECT Username, Sysdba, Sysoper FROM sys.V_$PWFILE_USERS;

Motivation: Minimizing the area open to attack by minimizing authorizations.

For this requirement the following threats are relevant:

• Unauthorized access to the system
• Unauthorized access or tapping of data
• Unauthorized modification of data
• Unauthorized use of services or resources

Principle of least privilege

Req 13  All rights to the DBMS_OBFUSCATION_TOOLKIT must be revoked from the PUBLIC pseudo role.

Although Oracle has replaced the DBMS_OBFUSCATION_TOOLKIT with the DBMS_CRYPTO package, the former is still required for certain tasks.

Check:
SELECT Table_Name, Owner, Grantee, Privilege FROM DBA_TAB_PRIVS WHERE Grantee = 'PUBLIC' AND Table_Name = 'DBMS_OBFUSCATION_TOOLKIT';

Motivation: By revoking the rights of the PUBLIC role, the administrator prevents unauthorized users from being able to decrypt the data.

Implementation example: REVOKE EXECUTE ON DBMS_OBFUSCATION_TOOLKIT FROM PUBLIC;

For this requirement the following threats are relevant:

• Unauthorized access to the system
• Unauthorized access or tapping of data
• Unauthorized modification of data
• Unauthorized use of services or resources

Req 14  A database service must not run with root rights or other operating system-related administrative rights.

Exception are: Oracle Grid Infrastructure (ASM/RAC), for instance ora_asm, ora_dism

Check: Look at the process lists.

Motivation: If this requirement is not met, a security vulnerability in the database service could cause the operating system to be compromised (e.g., through a buffer overflow).
For this requirement the following threats are relevant:

- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources
- Disruption of availability
- Attacks motivated and facilitated by information disclosure or visible security weaknesses

External procedures

**Req 15** If it is not necessary to call external procedures, the configuration entry “PROGRAM=extproc” must be removed from the listener.ora file.

*Motivation: These configurations make it possible to run external programs under the operating system account under which the database instance is running. These settings should be deactivated in order to safeguard the database system.*

For this requirement the following threats are relevant:

- Disruption of availability

**Req 16** If the “extproc” functionality is required, in addition to the normal database listener, a separate EXT-PROC listener must be used which runs under its own unprivileged account.

*Motivation: There is a risk of attackers infiltrating an external procedure as a Trojan and thus taking control of the super user privileges. A dedicated listener with restricted privileges reduces the risk of this type of attack.*

Implementation example: The administrator creates an additional listener.ora under the unprivileged user account of the operating system and configures the entry for the extproc listener in this.


For this requirement the following threats are relevant:

- Disruption of availability

**Req 17** Calling external procedures must be restricted to only the required functions in the configuration file listener.ora.

*Motivation: In order to make the non-audit-proof exchange of the external procedures which can be called more difficult, only the EXTPROC listener and the system administrator may have access rights to the directories in which the external procedures are saved. Application responsible should have no access rights after that.*

Implementation example: Restriction with the option EXTPROC DLLS=ONLY:DLL1:DLL2 in the extproc.ora file and specifying the absolute path for each required DLL (see [http://download.oracle.com/docs/cd/E11882_01/appdev.112/e17125.pdf](http://download.oracle.com/docs/cd/E11882_01/appdev.112/e17125.pdf) S.14-6).

During implementation, it should be noted that the configuration of the listener.ora (or extproc.ora) imposes a restriction through explicitly numbering and naming the permissible external procedures. The impact of this can be seen with the ONLY option, e.g., EXTPROC DLLS=ONLY:DLL1:DLL2:..., whereby this option requires the absolute path to be specified for the required DLLs.
For this requirement the following threats are relevant:

- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources
- Disruption of availability

SQL extensions with operating-system or network access

Req 18 The use of external jobs (local external job or remote external job) must be restricted to DBA user accounts.

Reference:
http://docs.oracle.com/cd/B28359_01/server.111/b28310/schedover004.htm

Motivation: The functions enable operating system calls to be carried out locally or remotely. The risk is that with this functionality, in reality a login on the database server is replaced (carried out by the function itself). Therefore, this function should only granted to DBA user accounts.

For this requirement the following threats are relevant:

- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources

Req 19 The UTL_FILE_DIR parameter must not be used.

Motivation: Special access rights to directories on the database server or the files it contains cannot be adequately configured using UTL_FILE_DIR. When using the UTL_FILE_DIR initialization parameter, in principle all database users have read and write rights to the files located in the specified directories.

Implementation example: The person responsible must remove the UTL_FILE_DIR parameter from the init.ora file. However, if the user still wants to have read or write access to the files in the directories located on the database server, database objects of the DIRECTORY type must be created for this and approved for read or write access. It is then possible to assign access rights to these directories per user or role. It is also necessary to create directories which are to be approved for read and write access which remain under the control of system administrators. As a result, the CREATE DIRECTORY system privilege may only be assigned to system administrators and, under no circumstances, to application accounts.

For this requirement the following threats are relevant:

- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources
- Disruption of availability
Access rights

Req 20 The “Oracle Software Owner” must be a system account that is only used for the administration of the database system.

Motivation: There is a risk that an unauthorized and uncontrolled change could be made to the installation.

Implementation example: The “root” system account or another system account must not be made available as the Database Software Owner. Generally, the administration account for the database is “Oracle”.

Exception: Oracle Grid Infrastructure (ASM/RAC).

In addition, conversely, this system account must not be used for the installation of other products (e.g., Oracle HTTP server) at the same time.

For this requirement the following threats are relevant:
• Disruption of availability

Req 21 The accounts of the Oracle Software Owner and those of the (personalized) administrators must be the individual members of the “dba” OS group in the “/etc/group” file.

Motivation: To ensure a controlled administration, there should be only one database administrator account or administrator accounts personalized in another way.

For this requirement the following threats are relevant:
• Unauthorized access to the system
• Unauthorized access or tapping of data
• Unauthorized modification of data
• Unauthorized use of services or resources
• Disruption of availability

Req 22 The Oracle Software Owner must be the owner of $ORACLE_HOME/bin and all the files located in it.

Motivation: There is a risk of uncontrolled access or an uncontrolled execution.

Implementation example: Please note: The following executable files in the $ORACLE_HOME/bin directory emtgctl2, extjob, jsu, nmb, nmhs, nmo, oracle, oradism belong to the Oracle Software Owner and/or have the SUID bit set. Administrators should not change this.

For this requirement the following threats are relevant:
• Unauthorized access to the system
• Unauthorized access or tapping of data
• Unauthorized modification of data
• Unauthorized use of services or resources
• Disruption of availability

Req 23 The access rights for the bin directory ($ORACLE_HOME/bin) must be set to 0755 (rwxr-xr-x) or lower.

Motivation: There is a risk of uncontrolled access and uncontrolled execution due to missing or incorrect access rights for the Oracle bin directory.

For this requirement the following threats are relevant:
• Unauthorized access to the system
• Unauthorized access or tapping of data
• Unauthorized modification of data
• Unauthorized use of services or resources
• Disruption of availability
• Unnoticeable feasible attacks

Req 24  Access to files which may contain Oracle data with a need for protection must be restricted to the DBA group.

Files belonging to this category include:
• Files with information on configuration and authentication (e.g., init.ora, spfile.ora, snmp_ro.ora, snmp_rw.ora, catsnmp.sql, orapw<SID>, listener.ora, xsqlconfig.xml, soapConfi.xml, …)
• Net8 trace and log files
• data files
• control files
• online redo files
• archive files
• audit files

Motivation: There is a risk of unauthorized access to confidential information.

For this requirement the following threats are relevant:
• Unauthorized access to the system
• Unauthorized access or tapping of data
• Unauthorized modification of data
• Unauthorized use of services or resources

Data communication

This section summarizes the requirements concerning queries from other database systems or the exchange of data between database systems. This can be implemented either by means of so-called ad-hoc queries, or by setting a data connection to share data between the systems. Oracle realizes data connections between databases via database links.

Req 25  Public database links must not be used.

Motivation: Public database links can be used by anyone, which means that there is potential for misuse. In the case of fixed database links (= a user with a fixed username/password), the opportunity for misuse is obvious. However, even the case of a connected user database link is not desirable because a compromised password on the source database can immediately also be used on the remote database, since these are identical.

Implementation example: Delete public database links and revoke the privilege CREATE PUBLIC DATABASE LINK from owners.

Check:
SELECT Owner, Db_Link FROM DBA_DB_LINKS WHERE Owner = 'PUBLIC';
SELECT COUNT(0), owner, db_link, username FROM sys.dba_db_links
GROUP BY owner, db_link, username ORDER BY 1,3;

This statement should not contain any lines with Owner=’PUBLIC’ or any lines missing a username.

For this requirement the following threats are relevant:
• Unauthorized access or tapping of data
• Unauthorized modification of data
• Unauthorized use of services or resources
• Disruption of availability
Req 26 Private database links must only be used with a “CONNECT TO” clause (fixed private database links).

Reference:
http://docs.oracle.com/cd/E11882_01/server.112/e25494/ds_concepts002.htm

Check:
In the following output, all links must have a username entered and the owner must not be PUBLIC.

```
SELECT Owner, Db_Link, Username, Host FROM DBA_DB_LINKS;
SELECT COUNT(0), owner, db_link, username FROM sys.dba_db_links
GROUP BY owner, db_link, username ORDER BY 1,3;
```

This statement should not contain any lines with Owner=’PUBLIC’ or any lines missing a username.

Motivation: Private database links without a “CONNECT TO” clause require identical user accounts on the local and on the remote database. This results in the local database account being compromised along with the remote user account. A private database link with the “CONNECT TO” clause and a password enables the authorized local user, along with the user specified in the link, to access the remote database. The area open to attack can be reduced by restricting the link as a private link (only the owner can use this) and by restricting it to the users defined in the link.

Implementation example: The example below creates a private fixed database link with the name sales.us.americas.example_auto.com to the remote database ‘sales_us’. To do this, the current user (database link owner) uses the remote user scott and his password “password”.

```
CREATE DATABASE LINK sales.us.americas.example_auto.com CONNECT TO scott IDENTIFIED BY password USING 'sales_us';
```

If an additional or other user now accesses the objects in the remote database, a view of the owner’s objects is created and the local user is given the necessary privileges for this. Thus only one user uses the database link directly.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources

Transport encryption
System monitoring

Logging

Req 27 Accesses to critical database procedures and database content must be logged.

Logging of security-relevant user actions must comply with national legislation currently in force. When implementing measures resulting from this requirement, the applicable participation rights of the responsible employee representatives/trade unions as well as the works and collective agreements shall be observed.

Motivation: Secure, traceable database operation requires important operating information to be logged. This includes, for example, the logging of failed login attempts to uncover possible intrusion attempts.

Implementation example: Using Oracle Auditing and the implementation of triggers, almost any access can be logged.

For this requirement the following threats are relevant:

- Denial of executed activities

Req 28 Database Event Triggers must not contain application objects of any kind. Only those procedures saved in the "SYS" dictionary schema may be called from the Database Event Trigger.

Motivation: If an application procedure is included in a Database Event Trigger, there is a risk that an experienced and malicious user could exploit this trigger by manipulating the process code and extending his user rights.

For this requirement the following threats are relevant:

- Disruption of availability
- Denial of executed activities

Oracle-specific requirements

In addition to the security requirements derived from the general database security requirements, the manufacturer-specific requirements listed below should be taken into account. Below, the terms "DBA User" and/or "DBA Account" refer to a database account which has DBA (database administrator) authorizations or takes on DBA-related activities according to the principle of segregation of functions (e.g., user administration, backup, etc.).

Authentication

Req 29 It must be ensured that the “Proxy Authentication” function is only assigned to technical accounts and not to the accounts of natural persons.

References:
http://docs.oracle.com/cd/E11882_01/java.112/e16548/proxy.htm
Motivation: The Proxy Authentication function is typically used in conjunction with web and application servers and enables these to handle a variety of user logins and sessions using one single database connection. The web or application server, as a "proxy", logs the user into the database using his proxy account. Thus each user of the web application works with his own individual Oracle account and the rights to the database linked with this and not with the global technical account of the web or application server.

For this requirement the following threats are relevant:
• Unauthorized access to the system
• Unauthorized access or tapping of data
• Unauthorized modification of data
• Unauthorized use of services or resources

---

Req 30 The operating system authentication must be deactivated for all non-DBA and non-operational users who do not require automatic login from the operating system.

Check:
Check OS group dba for unwanted members.
Check users of the following query:
SELECT Username FROM DBA_USERS
WHERE PASSWORD = 'EXTERNAL';

Motivation: The database security should depend as little as possible on the security of the underlying operating system.

Implementation example: The number of members in the DBA operating system group should be restricted. The IDENTIFIED EXTERNALLY mode should not be used in user profiles. Accounts which are not used for operational purposes, and for which automatic login via the operating system is not required (e.g., for backup), must not have account names with OS_AUTHENT_PREFIX. The default value for OS_AUTHENT_PREFIX is OPS$.

For this requirement the following threats are relevant:
• Unauthorized access to the system
• Unauthorized access or tapping of data
• Unauthorized modification of data
• Unauthorized use of services or resources

---

Req 31 User login via an external (remote) operating system authentication must be deactivated.

Motivation: Oracle provides the option of relying on an external (remote) operating system authentication for the user login. Since the remote system cannot be relied upon, these options must be set to FALSE.

Implementation example: Administrators must set the parameters in the init.ora configuration file as shown below:
REMOTE_OS_AUTHENT=FALSE
REMOTE_OS_ROLES=FALSE
The default value for Oracle 11g is FALSE.

For this requirement the following threats are relevant:
• Unauthorized access to the system
• Unauthorized access or tapping of data
• Unauthorized modification of data
• Unauthorized use of services or resources
• Unnoticeable feasible attacks

Requirement 32: Database roles must not be assigned via the operating system.

Motivation: Oracle provides the option of having database user roles managed by the operating system. This option carries hidden security risks; the person responsible must therefore deactivate this function. The tasks and responsibilities of DBA and operating system administration must be separated.

Implementation example: The FALSE default value of the parameter “OS_ROLES” in init.ora must not be changed.

For this requirement the following threats are relevant:
• Unauthorized access to the system
• Unauthorized access or tapping of data
• Unauthorized modification of data
• Unauthorized use of services or resources

Listener and network connection

Requirement 33: Access to the TNS listener must be restricted to authorized users through technical measures.

Various technical options are available for restricting the access to authorized users.

Dynamic IP address
For computers with a dynamic IP address, access via a secure jump service / terminal server or certificate-based authentication is required since access restriction at IP level is no longer provided.

Fixed IP address
For computers with a fixed IP address, the access restriction can be implemented using Oracle valid node checking, Access Control Lists (ACLs) and network or host-based systems.

Motivation: Restricting access to authorized users drastically reduces the likelihood of attacks on the database system.

Implementation example: Example of implementation of Oracle Valid Node Checking:
For Oracle valid node checking, the following lines are to be added to the listener sqlnet.ora configuration file (for older systems protocol.ora):
TCP.VALIDNODE_CHECKING = YES
TCP.INVITED_NODES = (<host_1>,
<host_2>, …)
(The value for <host_x> is either an IP address or a DNS name)

Please note: TCP.INVITED_NODES have priority over TCP.EXCLUDED_NODES. Furthermore, the Oracle listener should be protected via the IP filter of the operating system.

For this requirement the following threats are relevant:
• Unauthorized access to the system
• Unauthorized access or tapping of data
• Unauthorized modification of data
• Unauthorized use of services or resources
Req 34  The TNS listener must be protected against unauthorized configuration.

The TNS listener is protected against unauthorized configuration by adding the A D - MIN_RESTRICTIONS_<listenername>=ON parameter to the listener.ora configuration file.

Reference:
Listener Control Utility
http://docs.oracle.com/cd/E14072_01/network.112/e10574. pdf

Motivation: Unauthorized persons must be prevented from reconfiguring the listener since this has an impact on the security of the database system.

For this requirement the following threats are relevant:
• Unauthorized access to the system
• Unauthorized access or tapping of data
• Unauthorized modification of data
• Unauthorized use of services or resources
• Disruption of availability

User accounts
Privileges for users and roles

Req 35 It must be ensured that no database privileges for application objects are granted to the PUBLIC user group.

Check:
SELECT table_name, owner, privilege from dba_tab_privs
where grantee = 'PUBLIC'
and owner not in ('SYS','SYSTEM','OUTLN','DBSNMP');

The exclusion of further user accounts (e.g., XDB,CTXSYS, WMSYS, etc.) to be assigned to the system area may be ne-cessary if these components are to be installed.

Motivation: The assignment of privileges for application objects to “PUBLIC” violates the need-to-know principle and increases the risk of misuse by approving resources and functionality which is not globally required. The authorization concept of an application can also only then be classified as reliable if the accesses to application resources or functionality are granted individually and not just assigned across the board via “PUBLIC”.

Checking the rights assigned to PUBLIC can only ever be worthwhile for application objects and not for objects found in the data dictionary.

For this requirement the following threats are relevant:
• Unauthorized access to the system
• Unauthorized access or tapping of data
• Unauthorized modification of data
• Unauthorized use of services or resources
Req 36  The O7_DICTIONARY_ACCESSIBILITY parameter must be set to FALSE.

Motivation: If the O7_DICTIONARY_ACCESSIBILITY initialization parameter is set to FALSE (default value since Oracle 9i), access to the dictionary objects is no longer possible using ANY privileges.

Implementation example: If access to dictionary objects (e.g., for applications and tools) is required, the administrator must individually grant object privileges for access to these dictionary objects.


For this requirement the following threats are relevant:
• Unauthorized access to the system
• Unauthorized access or tapping of data
• Unauthorized modification of data
• Unauthorized use of services or resources

Req 37  Non-DBA users must not have object privileges (directly or via roles) for the following objects:

- AUD$
- ALL_SOURCE
- DBA_AUDIT_TRAIL
- DBA_ROLE_PRIVS
- DBA_SYS_PRIVS
- DBA_TAB_PRIVS
- DBA_USERS
- DBMS_FILE_TRANSFER
- DBMS_IJOB
- DBMS_ISCHED
- DBMS_SYS_SQL
- DBA_USERS_WITH_DEFPWD
- FGA_LOG$
- LINK$
- USER$
- USER_HISTORY$
- UTL_INADDR (controlled via ACL)
- UTL_TCP (controlled via ACL)
- UTL_HTTP (controlled via ACL)
- UTL_SMTP (controlled via ACL)
- UTL_MAIL (controlled via ACL)

Motivation: The views and packets named either allow access to confidential information (e.g., password hash in LINK$) or the performance of potentially dangerous actions (e.g., sending e-mails from the database with UTL_MAIL).

For this requirement the following threats are relevant:
• Unauthorized access to the system
• Unauthorized access or tapping of data
• Unauthorized modification of data
• Unauthorized use of services or resources

Non-DBA user accounts must not have one of the following roles (directly or indirectly):

• AQ_ADMINISTRATOR_ROLE
• DATAPUMP_IMP_FULL_DATABASE
• DBFS_ROLE
• DBA
• DELETE_CATALOG_ROLE
• EXECUTE_CATALOG_ROLE
• EXP_FULL_DATABASE
• GATHER_SYSTEM_STATISTICS
• IMP_FULL_DATABASE
• LOGSTDBY_ADMINISTRATOR
• OEM_MONITOR
• OEM_ADVISOR
• RECOVERY_CATALOG_OWNER
• SCHEDULER_ADMIN
• SNMPAGENT
• SELECT_CATALOG_ROLE

Motivation: These roles have critical object and/or system privileges. Thus, for example, the DATAPUMP_IMP_FULL_DATABASE role has a system privilege which enables any privilege to be assigned to anyone.

For this requirement the following threats are relevant:
• Unauthorized access to the system
• Unauthorized access or tapping of data
• Unauthorized modification of data
• Unauthorized use of services or resources
3.30 Microsoft SQL Server

Introduction
This security requirement has been prepared based on the provisions of the Group Policy on IT/NT Security.

The security requirement is used as a basis for an approval in the PSA process, among other things. It also serves as an implementation standard for provisions of the Group Policy on IT/NT Security in units which do not participate in the PSA process. These requirements shall be taken into account from the very beginning, including during the planning and decision-making processes. When implementing these security requirements, the precedence of national, international and supranational law shall be observed.

Basic security requirements for Microsoft SQL Server
The sections below describe the security requirements which apply especially to MS SQL Server.

Req 1 The version of the MS SQL Server must be approved for production licensing.

Motivation: These MS SQL Server editions offer the highest security and stability compared with other editions.

Implementation example: Using the following command, the administrator can determine which edition is installed in the MS SQL Server Management Studio:

```
SELECT SERVERPROPERTY ('edition')
```

Req 2 At least MS SQL Server 2005 must be used.

Motivation: The MS SQL Server versions as of 2005 have an adequate security standard and an optimized security design with improved default settings compared with earlier versions (7.0, 2000).

Database system hardening
This section contains the requirements for hardening the database system. The measures are comparable with those relating to operating system hardening and reduce the threat of an attack on the database systems.

Req 3 The standard database administrator account (sa) must be disabled.

Motivation: A large number of automated malware (such as worms) attempts to compromise the default administrator account (sa) using brute force attacks in order to gain unauthorized access to the database. Disabling the account is a simple but effective preventative measure against automated attacks.

Req 4 The Windows Built-in accounts or groups must not be SQL Logins.
Motivation: By default, local administrators have full administrator rights for the database system and all connected databases (this no longer applies for versions of MS SQL Server 2008).

Implementation example: The group can be deleted by running the following stored procedure from the “sysadmin” role:
EXEC sp_dropsrvrolemember 'BUILTIN\Administrators', 'sysadmin';
GO

Req 5 Windows local groups must not be SQL Logins.

Motivation: It is recommended to not allow the usage of any application defined local windows groups (except those created by the Microsoft SQL Server installation for the purpose of providing SQL Services with appropriate permissions such as SQLServer2005MSSQL$User$ComputerName$InstanceName). Allowing local windows groups as SQL Logins provides a loophole whereby anyone with OS level administrator rights (and no SQL Server rights) could add users to the local Windows groups and thereby give themselves or others access to the SQL Server instance.

Req 6 Database access for the guest user must not be enabled in user application databases.

Motivation: Only known users may access the databases. Conversely, the GUEST user is intended explicitly for un-known users.

Implementation example: REVOKE CONNECT FROM GUEST

Req 7 Installed sample databases must be removed from the SQL Server instance.

Known “sample” databases (pubs, Northwind, or any AdventureWorks database) must not be installed on production systems and must be removed if found.

Req 8 If the Trustworthy property is used, Database owner of a Trustworthy non-system database must not be in the sysadmin role.

By default, the TRUSTWORTHY database property is OFF for user databases.

Req 9 The MS SQL Server must not be set up on a domain controller.

Motivation: This is a manufacturer recommendation by Microsoft.

Req 10 Only one instance of the database system must be installed on one operating-system instance (hardware platform or virtualization guest).

Motivation: If multiple database instances for different tasks (e.g., Internet and intranet) are running on one operating-system instance, the instances are no longer separated from one another. This increases the risk of an attack.
Req 11 The authorizations for accounts and applications must be reduced to the minimum required for the tasks they have to perform.

Authorizations to a system must be restricted to a level in which a user can only access data and use functions that he needs in the course of his work. Suitable authorizations must also be assigned for access to files that are components of the operating system or of applications or that are generated by the same (e.g. configuration and logging files).

Alongside access to data, execution of applications and components must also take place with rights that are as low as possible. Applications should not be executed with administrator or system rights.

Motivation: If the rights granted to a user on a system are too broad, it could be possible for him to access data and applications which he is not permitted to view or use. This would give him the opportunity to disclose or modify confidential data and to manipulate system files. Applications with rights that are too broad can be used by a user to extend his own authorizations and thus to gain access to files and system components to which he would not have had access with his authorizations under normal circumstances.

For this requirement the following threats are relevant:

• Unauthorized access or tapping of data
• Unauthorized modification of data
• Unauthorized use of services or resources

Req 12 The operating-system rights for the database files and directories (program, control, trace and log files) must be assigned exclusively to the database system’s operating-system account.

Some database systems store sensitive account data in plain text in trace or log files. Therefore, access rights to sensitive files and directories on the database system MUST be set so that non-administrative users do not have any read, write or execution rights.

Access to essential system files and directories on the database system must be reserved for the database system’s user account.

Motivation: By implementing restrictive access rights, the risk of manipulation can be substantially reduced.

For this requirement the following threats are relevant:

• Unauthorized access to the system
• Unauthorized access or tapping of data
• Unauthorized modification of data
• Unauthorized use of services or resources
Req 13  The MSSQL service account must be a Windows domain account, MSA, or Virtual account.

Motivation: Every additional service with unnecessary rights increases the risk of attack and the level of damage.

Req 14  The SQLAgent service account must be a Windows domain account, MSA, or Virtual account.

Motivation: Every additional service with unnecessary rights increases the risk of attack and the level of damage.

Req 15  If used, the SQLFullText service account must be the same as the MSSQL service account for SQL Server 2005.

Req 16  For SQL Server 2008 and higher, the SQLFullText service account must be different from the MSSQL service account.


Req 17  The service accounts must not have the right “Log on locally”.

Req 18  If a domain account is required for the MS SQL Server, its AD rights must be restricted to "Log on To".

Motivation: Restricting the account rights reduces the risk of a system being compromised.

Req 19  The MSSQL service account, must not be a Windows Administrator.

Motivation: Restricting the account rights reduces the risk of a system being compromised.

Req 20  The SQLAgent service account must not be a Windows Administrator.

Motivation: Restricting the account rights reduces the risk of a system being compromised.
Req 21  The SQL Full-Text Service Account must not be a Windows Administrator.

Motivation: Restricting the account rights reduces the risk of a system being compromised.

Req 22  The PUBLIC role in the msdb database must not be granted access to SQL Agent proxies.

Motivation: This would allow all users to utilize the proxy which may have high privileges.

Req 23  For SQL Server 2005, access to all webtask extended stored procedures must be deactivated.

The following extended stored procedures for MS SQL Server 2005 and earlier versions will be replaced by SQL Server Reporting Services (SSRS) as of MS SQL Server Version 2005. SSRS are to be used instead of the older webtask extended SP.

The webtask extended stored procedures include:
- Xp_cleanupwebtask,
- Xp_convertwebtask,
- Xp_dropwebtask,
- Xp_enumcodepages,
- Xp_makewebtask,
- Xp_readwebtask and
- Xp_runwebtask.

Motivation: Every non-essential additional package potentially increases the area open to attack for the system.

Implementation example: USE [master]
GO
EXECUTE sp_configure 'Web Assistant Procedures', 0;
RECONFIGURE;
GO

Req 24  Access to all OLE automation stored procedures must be deactivated.

The administrator must deactivate the following OLE automation stored procedures:
- Sp_OACreate,
- Sp_OADestroy,
- SP_OAGetMethodInfo,
- Sp_OAGetProperty,
- SP_OAMethod,
- SP_OASetProperty and
- SP_OAStop.

Motivation: Every unnecessary functionality increases the risk of a successful attack on the system.

Implementation example: sp_configure 'show advanced options', 1;
GO
RECONFIGURE;
GO
sp_configure 'Ole Automation Procedures', 0;
GO
RECONFIGURE;
GO
3.30 Microsoft SQL Server

Run the following command to check whether the OLE automation stored procedures are deactivated:

```
EXEC sp_configure 'Ole Automation Procedures';
GO
```

---

**Req 25** SQL Mail must be deactivated.

*Motivation: Every unnecessary functionality increases the risk of a successful attack on the system.*

Implementation example: Access to the following extended stored procedures must be deactivated: Xp_deletemail, Xp_fndnextmsg, Xp_get_mapi_default_profile, Xp_get_mapi_profiles, Xp_readmail, Xp_sendmail, Xp_startmail, and Xp_stopmail.

---

**Req 26** If not used, Database Mail must be deactivated.

---

**Req 27** If granted, the access right of the role/group “Public” for extended stored procedures with direct access to the operating-system level or network must be revoked.

For the following extendedStored Procedures (XPs), the execute permission for public must be revoked:

- xp_availablemedia
- xp_enumgroups
- xp_subdirs
- xp_regaddmultistring
- xp_regdeletekey
- xp_regdeletevalue
- xp_regenumvalues
- xp_regremovemultistring
- xp_regwrite
- xp_servicecontrol

*Motivation: A restrictive set of rights reduces the risk of attackers changing data and thus compromising the system.*

Implementation example: Revoking the “Public” access rights from xp_regread: REVOKE EXECUTE ON xp_regwrite TO Public

How to evaluate: USE [master] GO

```
SELECT OBJECT_NAME(major_id) AS 'extended_procedure', [permission_name], 'PUBLIC' AS 'to_principal'
```
FROM sys.database_permissions
WHERE major_id =
OBJECT_ID('xp_availablemedia') AND
[type] = 'EX'
AND grantee_principal_id = 0;
GO

Req 28 If SSL is used for access to a database instance, the option “ForceEncryption” must be set to “YES” on the server side.

Motivation: The use of SSL can effectively protect the transfer of confidential data.

Users and roles

Principle of least privilege

SQL functions and packages

Data communication

This chapter summarizes the requirements concerning the querying of data from other database systems and the exchange of data between database systems. This can be implemented either by means of so-called ad-hoc queries, or by setting a data connection to share data between the systems. The terminology and the technical implementation of such data connections varies greatly among manufacturers (MS-SQL: linked servers/replication; Oracle: database links; DB2: connections). To simplify matters, the term “database links” is used below in a general sense.

Req 29 The “Remote Server” function must be deactivated.

In the more recent SQL Server versions, “Remote Servers” are only supported for the purpose of backward compatibility. More recent applications must use “Linked Servers” instead. “Linked Servers” have better security functions for querying and executing stored procedures on remote instances of MS SQL Servers and other OLE database data sources (e.g., Oracle, Access, Excel, DB2). “Remote Servers” support the concept of remote stored procedures (RPC). The “Remote Server” option must be activated on both the local and the remote server so that a connection can be successfully established. With MS SQL Server version 2005 or later, the “Remote Server” option is deactivated by default for security reasons. The administrator must also activate the MS SQL Server Browser service.

Motivation: The function represents a possible gateway for attackers, as with every functionality which can be accessed remotely.

Implementation example: The stored procedure SP_CONFIGURE can be used to deactivate the “Remote Server”: EXECUTE sp_configure 'remote access', 0
RECONFIGURE
GO

Req 30 If the “Linked Server” function is used, the default login mapping for the “Public” role must be deleted.

Motivation: The “Public” role represents a default role for all users. It provides potential attacks with a number of approaches.
3.30 Microsoft SQL Server

Implementation example: By default, all “Linked Servers” and “Remote Servers” can see all logins. To deactivate the default login mapping, the stored procedure “sp_droplinkedsrvlogin” must be executed with NULL (zero) as the “local login” parameter.

EXEC sp_droplinkedsrvlogin 'linked-server', NULL;

---

Req 31  The named pipes network protocol must be disabled if not used.

Microsoft Best Practices recommends disabling any protocols not required. TCP/IP is preferred over Named Pipes for WANs and slow networks.

---

Misuse detection and prevention

---

Req 32  Accesses to database systems, as well as critical database procedures and database content must be logged.

Secure, traceable database operation requires important operating information to be logged. This includes, for instance, the logging of failed login attempts to uncover possible intrusion attempts.

Logging of security-relevant user actions shall comply with national legislation currently in force. When implementing measures resulting from this Requirement, the applicable participation rights of the responsible employee representatives/trade unions as well as the works and collective agreements shall be observed.

For this requirement the following threats are relevant:
• Unnoticeable feasible attacks

---

Req 33  The auditing of security events must be activated.

The standard audit setting should suffice for the most applications. With systems that must meet the security standards such as SOX, PCI etc., the person responsible should take into account C2 audit mode. They must, however, be aware that C2 auditing can lead to loss of performance. Administrators should therefore define specific trace events and execute them through a stored procedure. An example can be found on the Microsoft Common Criteria website.

Motivation: Secure, traceable database operation requires important operating information to be logged. This includes, for instance, the logging of failed login attempts to uncover possible intrusion attempts.

Implementation example: C2 audit mode can be activated via the MS SQL Server Management Studio or by executing the following stored procedure as a member of the sysadmin role.

```
sp_configure 'show advanced options', 1 ;
RECONFIGURE ;
sp_configure 'c2 audit mode', 1 ;
RECONFIGURE ;
```

The MS SQL Server needs to be restarted for this setting to take effect.

---

Req 34  The SQL Server default trace must be enabled.

Motivation: Default tracing provides information about configuration and DDL changes should be logged for trace-able database operation.

---

Req 35  The number of Error log must be increased to at least 12.

By default, the SQL Server only creates up to 7 error logs, a new one is created every restart of the server.
Motivation: Secure, traceable database operation requires important operating information to be logged. Error logs therefore should not be overwritten after 7 restarts of the server.

---

**Req 36** The 'xp_cmdshell' Server Configuration Option must be set to 0.

*Motivation: Enabling this option can provide the ability for non-sysadmins to execute OS commands.*

Implementation example:
```
USE [master]
GO
EXECUTE sp_configure 'xp_cmdshell', 0;
RECONFIGURE;
GO
```

---

**Req 37** The 'Remote Admin Connections' Server Configuration Option must be set to 0 on non-clustered instances.

*Motivation: This option enables access to the dedicated admin connection remotely. However, this option must be enabled on clustered instances.*

Implementation example: Do not execute this on a clustered instance.
```
USE [master]
GO
EXECUTE sp_configure 'Remote Admin Connections', 0;
RECONFIGURE;
GO
```

---

**Req 38** The 'CLR Assembly Permission Set' must be set to SAFE_ACCESS for all user-defined CLR Assemblies.

Implementation example: To find user created assemblies, execute:
```
SELECT name AS Assembly_Name, permission_set_desc FROM sys.assemblies
WHERE is_user_defined = 1
and permission_set_desc <> 'SAFE_ACCESS';
```

For compliance, no rows should be returned.

How to remediate:
For each Assembly_Name returned in the query above, execute:
```
USE [master]
GO
ALTER ASSEMBLY Assembly_Name WITH PERMISSION_SET = SAFE;
GO
```

This should first be tested within a test environment prior to production to ensure the assembly still functions as designed with SAFE permission setting.
### MS SQL-specific requirements

In addition to the security requirements derived from the main document, the manufacturer-specific requirements listed below should be taken into account.

<table>
<thead>
<tr>
<th>Req</th>
<th>Description</th>
</tr>
</thead>
</table>
| 39  | Windows authentication mode must be used.  
**Motivation:** Compared with Windows authentication, mixed-mode authentication provides an inadequate standard of security. Windows authentication uses the Kerberos protocol. In addition, the validation of password complexity, account blocking and the password workflow can be enforced using Group policies. |
| 40  | The 'Cross DB Ownership Chaining' Server Configuration Option must be set to 0.  
**Motivation:** The activated configuration option applies to all databases running on the DBMS, i.e., not only for an instance. The area open to attack will be increased unnecessarily for all databases as a result. |
| 41  | The 'DB_CHAINING' Database Property Setting must be set to OFF.  
**Motivation:** Restricting the rights reduces the area open to attack and thus increases security.  
Implementation example: `ALTER DATABASE dbname SET DB_CHAINING OFF; GO` |
| 42  | Do not create the symmetric or asymmetric keys in system databases.  
**Reference:** [http://support.microsoft.com/kb/2162020](http://support.microsoft.com/kb/2162020)  
**Motivation:** Per Microsoft best practices, user-defined objects should not be created in system databases (master, model, msdb, and tempdb). |
3.32 Microsoft IIS

Introduction
This security requirement has been prepared based on the provisions of the Group Policy on IT/NT Security. The security requirement is used as a basis for an approval in the PSA process, among other things. It also serves as an implementation standard for provisions of the Group Policy on IT/NT Security in units which do not participate in the PSA process. These requirements shall be taken into account from the very beginning, including during the planning and decision-making processes. When implementing these security requirements, the precedence of national, international and supranational law shall be observed.

Platform requirements

Req 1 The web server must be the only externally accessible service of a system, unless the web server is exclusively used for an administration interface.

The web server must not be operated on the same system as a database, firewall or a system for the detection of network attacks (with the exception of host-based IDS). There may be multiple web servers instances (web sites) in a system. The “system” may be a virtual system.
If the web server only provides an administration interface to a network component (router, firewall, etc.), for example, the service to be administrated is permitted to be running in parallel to the web server.

Motivation: As soon as security is compromised on one of the components installed on the server system, the entire server system must be considered compromised.

Implementation example: A Windows server, on which an IIS web server is running, must not also be used as a domain controller, file server or Exchange server.

For this requirement the following threats are relevant:
• Unauthorized access to the system

Requirements on web server software

Req 2 Software and hardware components that are no longer supported by vendor, producer or developer must not be used.

Components that have reached end-of-life or end-of-support must not be used. Excluded are components that have a special support contract. This contract must guarantee the correction of vulnerabilities over components life-time.

Motivation: Hardware and software components that have reached end of life or end of support represent a risk for a system. This means that a vendor does not supply remedial updates or patches for a component should errors or vulnerabilities occur. This means that vulnerabilities cannot be fixed when they occur and could be exploited to compromise the system or to impair its availability.

For this requirement the following threats are relevant:
• Unauthorized access to the system
• Unauthorized access or tapping of data
• Unauthorized modification of data
• Disruption of availability
Req 3 Known vulnerabilities in software and hardware of the system must be fixed or protected.

Prior to installation of a software or hardware component, users must check whether any vulnerability has been discovered and published for the version they are installing. Any component that proves to have a vulnerability must not be installed or used. Excepted from this rule are components for which the vendor has already provided a measure to remedy the vulnerability, e.g. a patch, update or workaround. In this case, the additional measure must be implemented on the system. Furthermore it is a ongoing process during the complete life cycle of the system to fix upcoming vulnerabilities promptly.

Motivation: Publication of vulnerabilities increases the risk of successful exploitation by an attacker, especially since the published information usually includes details on how to exploit the vulnerability and tools that make exploitation possible. One example of this is when Web servers are compromised due to not patched vulnerabilities.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Disruption of availability

Installation requirements

Req 4 Each web site must be assigned a separate application pool.

Motivation: If an attacker obtains unauthorized access to a web site, he must be prevented from also accessing other web sites.

Implementation example: For IIS 7, IIS 7.5 and IIS 8, the IIS Manager can be used to display and create application pools. Also application pools can be assigned to web sites here. To do so, select the web site to be configured and then, in the “Actions” pane, click on “Basic settings…”.

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources

Req 5 The identity of an application pool must not be a user account with system privileges.

Motivation: If the web server process runs with administrative access rights, an attacker who obtains control over this process would be able to control the entire system.

Implementation example: For IIS 7, IIS 7.5 and IIS 8, an application pool can be configured in the IIS Manager. After the application pool to be configured has been selected, click on “Advanced settings...” in the “Actions” pane. The “Identity” can be found in the “Process Model” section and modified by first clicking in the value field and then clicking on “…”. Neither the built-in accounts “LocalService” or “LocalSystem” shall be selected here nor a custom account with corresponding privileges.

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data
- Unauthorized modification of data
Configuration requirements

Req 6 The reachability of services must be restricted.

Typically services that are enabled in the basic configuration are accessible over all interfaces of the system and can be reached from systems in connected networks. This availability is often not needed or meaningful for system functioning. For this reason, services should only be enabled on interfaces where their usage is required. On interfaces were services are active, the reachability must be limited to legitimate communication peers. This limitation must be realized on the system itself (without measures (e.g. firewall) at network side).

Motivation: Disabling services on interfaces which do not require system accessibility or by limiting the reachability can greatly reduce the potential vulnerabilities offered to an attacker. For example, access to a system via SSH from the Internet is not necessary. If this service could be accessed from the interface connected to the Internet, this would greatly increase the risk of attacks on the service.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized use of services or resources
- Disruption of availability
- Attacks motivated and facilitated by information disclosure or visible security weaknesses

Req 7 Unused functions of the operated software and hardware must be deactivated.

During installation of software and hardware often functions will be activated that are not necessarily needed for operation or function of the system. Functions of software are currently inherent part which could not be deleted or deinstalled individually. Such functions must be deactivated in the configuration of the system permanently.

Beside the functions of the software also hardware functions are active which are not necessary for a system. Functions like unused interfaces must permanently deactivated. Permanent means that they must not be reactivated again after system reboot.

Motivation: The hardware or software of a system often contains functions which are not used and so will be a risk for system security. Such functions give an attacker the possibility to manipulate the system. Furthermore it is possible to get unauthorized access other areas or data of the system. An example is a debugging function in software which can be used for troubleshooting but must not be activated during normal operation. Or a hardware interface that will not be used and so is unsecured an allows possibly unauthorized access to the system.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Disruption of availability

Req 8 HTTP methods that are not required must be deactivated.

Standard requests to web servers only use GET and POST. If other methods are required, they must be processed securely.

Motivation: HTTP TRACE could be misused by an attacker. This method allows for debugging and trace analysis of connections between the client and the web server. The Microsoft IIS web server uses the TRACK alias for this method. Other HTTP methods could also be used to obtain information about the server, or they could be directly misused by an attacker.

Implementation example: For IIS 7, IIS 7.5 and IIS 8, HTTP methods that are not required can be deactivated for a web site as follows:
In the IIS manager select the web site to be configured and then open “Request Filtering” in the “IIS” section of the “Features View”. Then, in the “Actions” pane, select “Edit Feature Settings...” and, if necessary, deactivate the item “Allow unlisted verbs”. Now select the tab “HTTP verbs” and enter all permitted HTTP methods via “Allow Verb ...” in the “Actions” pane.

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Attacks motivated and facilitated by information disclosure or visible security weaknesses

---

Req 9 Unless they are required Windows features regarding Internet Information services or “Web Server” role services must not be activated.

If they are not required, the following IIS features/"Web Server” role services must not be activated:
- FTP server
- IIS Management Service for Remote Administration
- All Application Development features (ASP, CGI, ISAPI in particular)
- WebDAV Publishing
- Server Side Includes (SSI)

Motivation: Each Windows feature can have security vulnerabilities and should therefore be deactivated if it is not required.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources
- Unnoticeable feasible attacks

---

Req 10 If CGI is used, CGI wrappers must be used for the initial parameter review/validation and setup of the correct runtime environment for CGI programs.

Motivation: Using CGI, the web server communicates with runtime environments which could be misused to execute malicious software.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unnoticeable feasible attacks

---

Req 11 If CGI is used, the CGI directory must not include compilers or interpreters (e.g., PERL interpreter, PHP interpreter/compiler, Tcl interpreter/compiler or operating system shells).

Motivation: Using CGI, the web server communicates with runtime environments which could be misused to execute malicious software.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources
- Disruption of availability
• Unnoticeable feasible attacks

 Req 12 If CGI is used, the CGI directory must not be used for uploads.

 Motivation: Using CGI, the web server communicates with runtime environments which could be misused to execute malicious software.

 For this requirement the following threats are relevant:
 • Unauthorized access or tapping of data
 • Unauthorized modification of data
 • Unnoticeable feasible attacks

 Req 13 If Server Side Includes (SSI) are active, the execution of system commands must be deactivated.

 Motivation: The Server Side Includes (SSI) technology, which is implemented in most web server products as an additionally loadable module, can potentially be used by attackers. The "exec" function of SSI, in particular, could be used to execute system commands, which represents a risk.

 Implementation example: For IIS 7, IIS 7.5 and IIS 8, the execution of system commands can be deactivated for a web site as follows:
 In the IIS Manager select the web site to be configured and then open the “Configuration Editor” in the “Management” section of the “Features View”. Now, in the “system.webServer/serverSideInclude” section, change the value of “ssiExecDisable” to “True”.

 For this requirement the following threats are relevant:
 • Unauthorized access to the system
 • Unauthorized access or tapping of data
 • Unauthorized modification of data
 • Unauthorized use of services or resources
 • Disruption of availability
 • Unnoticeable feasible attacks

 Req 14 If WebDAV is used for writing files, access must not be granted without successful authentication.

 Motivation: WebDav makes it possible to update content online which has been made available by the web server. This function could therefore be misused to change website content.

 For this requirement the following threats are relevant:
 • Unauthorized modification of data

 Req 15 If WebDAV is used, access rights must be configured restrictively.

 Access rights to all files accessible by WebDAV must be configured as restrictively as possible. Additionally, if Web-DAV is used, WebDAV access must be restricted to the directories required.

 Motivation: WebDav makes it possible to update content online which has been made available by the web server. This function could therefore be misused to change website content.

 For this requirement the following threats are relevant:
 • Unauthorized access or tapping of data
 • Unauthorized modification of data

 Req 16 Access rights for web server configuration files must only be granted to the owner of the web server process or a user with system privileges.

 Motivation: Configuration files may only be written by the owner of the web server process or a user with system privileges. Otherwise it would be possible for unauthorized users to change
the configuration of the web server or to obtain configuration information which could be used for an attack.

Implementation example: Delete “read” and “write” access rights for “others.” Only grant “write” access to the user who configures the web server.

For this requirement the following threats are relevant:
• Unauthorized access or tapping of data
• Unauthorized modification of data
• Unauthorized use of services or resources
• Disruption of availability

Req 17 If the “Default Web Site” is not used, it must be deleted.

Motivation: The „Default Web Site“ is delivered with example files in a standard configuration. If an attacker obtains access to the „Default Web Site“, he can therefore draw conclusions about the system used.

Implementation example: For IIS 7, IIS 7.5 and IIS 8, right click on the “Default Web Site” and select "Remove".

For this requirement the following threats are relevant:
• Unauthorized access or tapping of data
• Attacks motivated and facilitated by information disclosure or visible security weaknesses

Req 18 Default files in a website’s directory must be deleted.

A newly created web site's directory may contain default files. Usually this is an index HTML file and an image file, but there may be additional example files or tutorials. These files must be deleted. This concerns, in particular, all files in the directory of the “Default Web Site”, if this web site is used.

Motivation: By using examples, information could be obtained about the installed software (version). Examples can include security vulnerabilities.

For this requirement the following threats are relevant:
• Unauthorized access or tapping of data
• Attacks motivated and facilitated by information disclosure or visible security weaknesses

Req 19 Write access to executable files that are executed with system privileges must be restricted to users with system privileges.

Other user groups must not be able to modify these files.

This is highlighted for all script/batch files which (by default) belong to the web server user but which are executed during server startup or by a scheduler with administrator rights.

Motivation: A non-administrative owner could modify the file to execute other functions or to obtain administrator rights.

For this requirement the following threats are relevant:
• Unauthorized access or tapping of data
• Unauthorized modification of data
• Unauthorized use of services or resources
• Disruption of availability
Req 20 The Windows feature “Directory Browsing” of the Internet Information Services or the “Web Server” role service “Directory Browsing”, respectively, must be deactivated.

**Motivation:** Directory listings contain information about files and directory structures which could be misused.

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data
- Attacks motivated and facilitated by information disclosure or visible security weaknesses

Req 21 The HTTP “Server” header must not include information on the software and version of the web server.

**Motivation:** Any information about the web server could allow conclusions to be drawn about security vulnerabilities.

Implementation example: When using the UrlScan tool, the entry

```
RemoveServerHeader=1
```

or

```
AlternateServerName=<Webserver>
```

is made in the “[Options]” section of the UrlScan configuration file, whereby `<Webserver>` can be replaced by the string “Webserver”, for example.

For this requirement the following threats are relevant:
- Attacks motivated and facilitated by information disclosure or visible security weaknesses

Req 22 Additional HTTP response headers with information about the software and version of the web server or components used, must not be set.

**Motivation:** Any information about the web server or the components used could allow conclusions to be drawn about security vulnerabilities.

Implementation example: For IIS 7, IIS 7.5 and IIS 8, HTTP response headers can be configured as follows:

In the IIS manager select the server or the web site to be configured and then open “HTTP Response Headers” in the “IIS” section of the “Features View”. Headers which contain non-permissible information such as the “Xpowered- by” header can now be deleted.

For this requirement the following threats are relevant:
- Attacks motivated and facilitated by information disclosure or visible security weaknesses

Req 23 Information about the webserver in error pages, that are being delivered by the web server, must be
deleted.

Default error pages must be replaced with user-defined error pages. User-defined error pages must not include version information about the web server and the modules/addons used. Error messages must not include internal information such as internal server names, error codes, etc.

Motivation: Any information about the web server could allow conclusions to be drawn about security vulnerabilities.

Implementation example: For IIS 7, IIS 7.5 and IIS 8, error pages can be configured as follows: In the IIS manager select the server or the website to be configured and then open the “Error pages” in the “IIS” section of the “Features View”. Now either edit the error files under the paths specified here or enter different paths to specific new error pages without information about the web server product and version. By default, the error pages are defined at server level and then passed on to the individual web sites. If this is not changed, it is therefore sufficient to modify the error files defined at server level.

For this requirement the following threats are relevant:
- Attacks motivated and facilitated by information disclosure or visible security weaknesses

Req 24 File type mappings that are not required must be deleted.

File type mappings that are not required must be deleted, e.g., php, phtml, js, sh, csh, bin, exe, pl, vbe, vbs.

Motivation: File type mappings define how scripts and programs are executed on the server. These mappings could provide a runtime environment for files which were placed by an attacker. This file execution must be blocked.

Implementation example: For IIS 7, IIS 7.5 and IIS 8 file type mappings can be configured via the IIS manager. After the server or an individual web site has been selected, open “Handler mappings” in the “IIS” section of the “Features View”. Entries with paths that have the file suffixes specified above shall be deactivated, in particular.

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources
- Unnoticeable feasible attacks

Req 25 Unauthorized changes to Web.config files must be prevented.

Microsoft IIS allows for site, application or directory specific configuration in Web.config files. Access rights to these files must be set as restrictive as possible. In particular, these files must not be writable for users without administrative privileges. Restrictive access rights are already assigned to all Web.config files that have been created by IIS Manager. Especially for (virtual) directories that are integrated into the document tree it is important to make sure that no unauthorized user has the possibility to create or modify those files.

Motivation: A web site’s configuration could be modified by other users with the help of Web.config files. This way, for example, a user could get unauthorized access to a web site’s files.

For this requirement the following threats are relevant:
3.32 Microsoft IIS

- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources
- Disruption of availability

Req 26  Configurable settings in Web.config files must be as restrictive as possible.

All sections in the central configuration file ApplicationHost.config that refer to security relevant or critical aspects must be locked, unless individual settings on web site level are compulsory. By default this is the case, changes must not be made without legitimate reason.

Motivation: Settings in sections that are not locked may be changed by distributed Web.config files that are located in document directories. This may make it easier for a user to accomplish unauthorized configuration changes.

Implementation example: ApplicationHost.config is located in %windir%\system32\inetsrv\config. The different sections’ properties are defined within a special <configSections> section. A section is locked if its „overrideModeDefault“ attribute is set to “Deny”, for example:

```xml
<section name="defaultDocument" overrideModeDefault="Deny" />
```

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources
- Disruption of availability

Req 27  The web server may only deliver files which are meant to be delivered.

Restrictive access rights must be assigned to all files which are directly or indirectly (e.g., via links or in virtual directories) in the web server’s document directory. In particular, the web server must not be able to access files which are not meant to be delivered. For IIS, in particular virtual directories used to integrate the content of other applications, for example, must be thoroughly configured.

Motivation: If additional files or directories are integrated via links or virtual directories into the document directory of the web server, in particular, it is possible that a user can access files via the web server which he should not be allowed to view. This must be prevented through careful configuration.

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data

Req 28  The web server must be robust against overload situations.

A web server must provide security measures to deal with overload situations. In particular, partial or complete impairment of web server availability must be avoided. Potential protective measures include:

- Restricting the maximum number of HTTP sessions per IP address
- Defining the maximum size of a HTTP request
- Defining a timeout for HTTP request

Restrictions must be implemented in consideration of the application to be protected and its characteristics. The following values may be used as a guideline:

If the web server will not be used for uploads:
- Maximum number of HTTP sessions per IP address: 50
- Maximum size of a HTTP request: 20000 bytes
• Timeout for HTTP requests: 30 seconds

If the web server may also be used for uploads:
  • Maximum number of HTTP sessions per IP address: 50
  • Maximum size of a HTTP request: 10000000 bytes or, if known, maximum size of expected upload
  • Timeout for HTTP requests: 60 seconds or, if known, time to complete maximum upload

Motivation: Attackers often try to bring a web server into an overload situation by using denial-of-service (DoS) attacks. If such an attack is successful the web server's availability or integrity may be impaired.

Implementation example: In order to restrict the number of HTTP sessions per IP address the required Windows feature or IIS role service "IP and domain restrictions", respectively, must be installed.

All necessary configuration may be done from within the IIS manager. The number of HTTP sessions per IP address and the maximum request size may be configured either for the web server or single web sites. The timeout may only be configured for web sites.

To configure the number of HTTP sessions per IP address either select the server or a web site and then open the feature "IP Address and Domain Restrictions". Now within "Actions" click on "Edit Dynamic Restriction Settings...". In the window opening check "Deny IP Address based on the number of concurrent requests".

To configure the maximum HTTP request size either select the server or a web site and then open the feature "Request Filtering". Now within "Actions" click on "Edit Feature Settings...". In the window opening the request size may be configured in the field "Maximum allowed content length (Bytes)".

To configure the HTTP request timeout select a web site and then within "Actions" under "Configure" click on "Limits...". In the window opening the timeout may be configured in the field "Connection time-out (in seconds)".
HTTPS requirements

Req 29 Data with need of protection must be secured against unauthorized viewing and manipulation during transmission and storage.

Adequate security measures for transmission and storage must be implemented of data with a need for protection that are classified as internal, confidential or strictly confidential. The chosen measure depends on the classification for the data and other factors such as the type of network used during transmission, the storage location for data, etc. Furthermore must be guaranteed that confidential and strictly confidential data will not be unprotected during temporary storage (e.g. in web cache, temporary folders).

All authentication data such as user names, passwords, PINs, etc. must be protected against unauthorized viewing and manipulation. This applies equally to permanent storage and transmission. Typical measures taken to protect authentication data or combinations of these are:

For storage:
- Client systems: encryption or obfuscation of authentication data, no persistent storage, limitation of access rights.
- Server systems: hashing of authentication data with PBKDF like scrypt or bcrypt or when this is not possible hashes with salt.

For transmission:
- Usage of cryptographically protected network protocols.
- Challenge response method (Disadvantage of this solution is that passwords needed in clear-text on server. If this solution is feasible depends on the individual threat scenario)

Files of a system that are needed for the functionality must also be protected against manipulation. This is necessary because system’s integrity can be damaged when the system access this kind of files. An example is the use of check-sum or cryptographic methods to validate if e.g. firmware images, patches, drivers or kernel modules are free of manipulations.

For transmission of data with a need of protection network protocols that are insecure due to insufficient security measures shall not be used. Examples are: SSLv3, SSHv1, FTP, Telnet, SNMPv1 and 2c. In case of these protocols a newer version without vulnerabilities or a secure alternative must be used.

Motivation: If data with a need of protection will not be secured an attacker could record or manipulate the data during transmission over a network. An example is the recording of user names and passwords during system administration with the telnet clear-text protocol. Storing data on a system without adequate protection may mean that unauthorized users can copy or modify it. One example is when passwords can be read out when they stored in an inadequatesecured way (e.g. usage of unsecure hashing algorithms like MD5 or SHA-1) or even without encryption on a system or manipulation of firmware to affected the system integrity.

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data
- Unauthorized modification of data

Req 30 For encryption with HTTPS the TLS protocol must be used.

SSL must be considered outdated and thus may not be activated or must be deactivated, respectively.
Motivation: Particularly SSLv2 has a number of weaknesses that make it impossible to use from a security point of view. TLS is the further development of SSL. It is already established for years so there is no need for further use of SSL.

Implementation example: SSL/TLS is configured through corresponding registry entries. The following entries must be made for exclusive use of TLS:

```
[HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\SSL 2.0\Server] "Enabled"=dword:00000000
[HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\SSL 3.0\Server] "Enabled"=dword:00000000
[HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\TLS 1.0\Server] "Enabled"=dword:fffffff
[HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\TLS 1.1\Server] "Enabled"=dword:fffffff
[HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\TLS 1.2\Server] "Enabled"=dword:fffffff
```

The value of “Enabled” specifies in each case whether the protocol may be used (value not equal 0) or not (value 0). Non-existent keys must be created if necessary.

TLS 1.1 and TLS 1.2 are supported starting with Windows Server 2008 R2. Thus for older versions of the operating system the last two entries may be omitted.

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data
- Unauthorized modification of data

---

Req 31 The web server must be configured in such a way that the use of the latest version of the TLS protocol is enabled.

In particular, the web server must be configured for the use of TLS 1.2.

Motivation: The latest version of the protocol offers the best possible protection and contains fixes to known vulnerabilities in previous versions of the protocol.

Implementation example: Starting with Windows Server 2008 R2 the registry must contain the following entries:

```
[HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\TLS 1.0\Server] "Enabled"=dword:fffffff
[HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\TLS 1.1\Server] "Enabled"=dword:fffffff
[HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\Protocols\TLS 1.2\Server] "Enabled"=dword:fffffff
```

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data
- Unauthorized modification of data

---

Req 32 The TLS configuration must not use any unsecure cipher suites.

Cipher suites with one of the following characteristics are not secure
- no server authentication,
- no encryption,
- encryption with DES, RC4 or any other algorithm with a key length of less than 128 bits
- use of MD5 as hash function
Motivation: Cipher suites known to be unsecure do not offer sufficient protection.

For this requirement the following threats are relevant:
• Unauthorized access or tapping of data
• Unauthorized modification of data

Req 33 The TLS configuration must provide that the cipher suite considered most secure is being chosen with highest priority.

Motivation: When a TLS connection is being established a cipher suite is selected based on the cipher suites available both on client and on server side. In order to ensure a high compatibility to all kinds of client systems the web server must not only allow for the cipher suites considered most secure. To make sure that nevertheless for each client the best possible cipher suite is selected and thus the connection is best protected the configuration must contain an according prioritization.

• At a command prompt, enter gpedit.msc. The Group Policy Object Editor appears.
• Now expand "Computer Configuration", then "Administrative Templates" and finally "Network". Now click "SSL Configuration Settings".
• Under "SSL Configuration Settings", double-click "SSL Cipher Suite Order setting".
• In the "Options" section of the "SSL Cipher Suite Order pane" the cipher suite order may be specified now. Please follow the instructions in the "Help" section. In particular, the string entered may not contain more than 1023 characters.
• The new settings will take effect after a reboot.

On a Windows Server 2012 the following cipher suites must be used. The cipher suites are listed from highest priority at the top to the lowest priority at the bottom. The cipher suites at the bottom of the list offer the least security but they ensure that HTTPS connections to the web server may use TLS 1.0. For operating systems before Windows Server 2012 the list may have to be shortened.

```
TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384_P521
TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384_P384
TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384_P256
TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256_P521
TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256_P384
TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256_P256
TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384_P384
TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384_P256
TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256_P384
TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256_P256
TLS_DHE_DSS_WITH_AES_256_CBC_SHA256
TLS_DHE_DSS_WITH_AES_128_CBC_SHA256
TLS_RSA_WITH_AES_256_CBC_SHA256
TLS_RSA_WITH_AES_128_CBC_SHA256
TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA_P521
TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA_P384
TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA_P256
TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA_P384
TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA_P256
TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA_P384
TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA_P256
TLS_DHE_DSS_WITH_AES_256_CBC_SHA
TLS_DHE_DSS_WITH_AES_128_CBC_SHA
TLS_DHE_DSS_WITH_3DES_EDE_CBC_SHA
TLS_RSA_WITH_AES_256_CBC_SHA
TLS_RSA_WITH_AES_128_CBC_SHA
```
For this requirement the following threats are relevant:
• Unauthorized access or tapping of data
• Unauthorized modification of data

Req 34 Certificates must be issued by a certification authority whose certificates are recognized by the commonly used web browsers.

For critical applications that can be used via the Internet, use of an extended validation certificate (EV certificate) is recommended.

Motivation: Only if the certificate authority (CA) is contained in the CA list of the browser being used the browser can verify the authenticity of the server or web application. Strictly issuing criteria apply to EV certificates. If an EV certificate is used, this is visualized in the browser. Even if EV certificates do not improve security, their use increases the trustworthiness of the server for the user.

For this requirement the following threats are relevant:
• Attacks motivated and facilitated by information disclosure or visible security weaknesses

Req 35 Certificates must lose their validity after a maximum of 36 months.

Motivation: The methods used for analysing and breaking cryptographic processes are improved continuously. Therefore, the security of the certificates can be ensured for a limited period only. But, according to a general estimation, the security of the certificates is ensured for the required validity period of three years, if an appropriate key length is used.

For this requirement the following threats are relevant:
• Unauthorized access or tapping of data
• Unauthorized modification of data

Req 36 Certificates must have a key length of at least 2048 bits when using RSA or 256 bits when using ECC.

Motivation: In order to guarantee the security of certificates over the validity period, the cryptographic keys must have an appropriate length. According to a general estimation, a key length of 2048 bits provides sufficient protection for the next years. For ECC algorithms, shorter key lengths already provide the same level of security.

For this requirement the following threats are relevant:
• Unauthorized access or tapping of data
• Unauthorized modification of data

Logging

Req 37 Access to the web server must be logged.

The web server log must contain the following information:
• Access timestamp
• Source (IP address)
• Account (if known)
• URL
• Status code of web server response
3.36 Apache Web Servers

Logging must be done considering the currently valid legal, wage and company regulations. This regulations state among others that logging of events can be done only earmarked. Logging of events for doing a work control of employees is not allowed.

Motivation: For the analysis of security incidents it is very important to have basic information on how the attack has been carried out. Since a webserver represents an external interface certain information about an attack is only available on the webserver, even if the attack is aimed at a downstream system. Thus logging on a web server is mandatory.

For this requirement the following threats are relevant:

• Denial of executed activities
• Unnoticeable feasible attacks
3.36 Apache Web Servers

Introduction
The security requirement is used as a basis for an approval in the PSA process, among other things. It also serves as an implementation standard for provisions of the Group Policy on IT/NT Security in units which do not participate in the PSA process. These requirements shall be taken into account from the very beginning, including during the planning and decision-making processes.
When implementing these security requirements, the precedence of national, international and supranational law shall be observed.

Platform

Req 1 The web server must be the only externally accessible service of a system, unless the web server is exclusively used for an administration interface.

The web server must not be operated on the same system as a database, firewall or a system for the detection of network attacks (with the exception of host-based IDS).
There may be multiple web servers in a system. The “system” may be a virtual system.
If the web server only provides an administration interface to a network component (router, firewall, etc.), for example, the service to be administrated is permitted to be running in parallel to the web server.

Motivation: As soon as security is compromised on one of the components installed on the server system, the entire server system must be considered compromised.
For this requirement the following threats are relevant:
• Unauthorized access to the system

Software

Req 2 Software and hardware components that are no longer supported by vendor, producer or developer must not be used.

Components that have reached end-of-life or end-of-support must not be used. Excluded are components that have a special support contract. This contract must guarantee the correction of vulnerabilities over components life-time.

Motivation: Hardware and software components that have reached end of life or end of support represent a risk for a system. This means that a vendor does not supply remedial updates or patches for a component should errors or vulnerabilities occur. This means that vulnerabilities cannot be fixed when they occur and could be exploited to compromise the system or to impair its availability.
For this requirement the following threats are relevant:
• Unauthorized access to the system
• Unauthorized access or tapping of data
• Unauthorized modification of data
• Disruption of availability
Req 3  Known vulnerabilities in software and hardware of the system must be fixed or protected.

Prior to installation of a software or hardware component, users must check whether any vulnerability has been discovered and published for the version they are installing. Any component that proves to have a vulnerability must not be installed or used. Excepted from this rule are components for which the vendor has already provided a measure to remedy the vulnerability, e.g. a patch, update or workaround. In this case, the additional measure must be implemented on the system. Furthermore it is an ongoing process during the complete life cycle of the system to fix upcoming vulnerabilities promptly.

Motivation: Publication of vulnerabilities increases the risk of successful exploitation by an attacker, especially since the published information usually includes details on how to exploit the vulnerability and tools that make exploitation possible. One example of this is when Web servers are compromised due to not patched vulnerabilities.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Disruption of availability

Installation

Req 4  Different instances of the Apache web server on a system must be executed as dedicated user, respectively.

Motivation: If an attacker obtains unauthorized access to an instance of a web server, he must be prevented from accessing other instances as well.

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data
- Unauthorized modification of data

Req 5  Web server processes must not run with system privileges.

If a process is started by a user with system privileges, execution must be transferred to a different user without system privileges after the start. One httpd process runs as the user, who started the web server, e.g. root. This is common for Apache web servers and is accepted.

Motivation: If the web server process runs with administrative privileges, an attacker who obtains control over this process may control the entire system.

Implementation example: User and group of the webserver processes are defined in the configuration file:

```
User apache
Group apache
```

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized use of services or resources
- Disruption of availability
Configuration

Req 6 The reachability of services must be restricted.

Typically services that are enabled in the basic configuration are accessible over all interfaces of the system and can be reached from systems in connected networks. This availability is often not needed or meaningful for system functioning. For this reason, services should only be enabled on interfaces where their usage is required. On interfaces were services are active, the reachability must be limited to legitimate communication peers. This limitation must be realized on the system itself (without measures (e.g. firewall) at network side).

Motivation: Disabling services on interfaces which do not require system accessibility or by limiting the reachability can greatly reduce the potential vulnerabilities offered to an attacker. For example, access to a system via SSH from the Internet is not necessary. If this service could be accessed from the interface connected to the Internet, this would greatly increase the risk of attacks on the service.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized use of services or resources
- Disruption of availability
- Attacks motivated and facilitated by information disclosure or visible security weaknesses

Req 7 Access rights for web server configuration files must only be granted to the owner of the web server process or a user with system privileges.

To enhance the security it is recommended to create an additional non privileged user, who has write access to web server configuration files and to grant only read access to the owner of the web server process. Then an attacker, who got control over the web server process, could not modify configuration files directly.

Motivation: Configuration files may only be written by the owner of the web server process or a user with system privileges. Otherwise it would be possible for unauthorized users to change the configuration of the web server or to obtain configuration information which could be used for an attack.

Implementation example: Delete “read” and “write” access rights for “others.” Only grant “write” access to the user who configures the web server.

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data
- Unauthorized modification of data

Req 8 Apache modules that are not required must be deactivated.

In particular, the modules for
- CGI: mod_cgi and mod_cgid
- Server Side Includes (SSI): mod_include
- WebDAV: mod_dav

must be deactivated, if they are not required.

Motivation: Each add-on, component or function can have security vulnerabilities.
3.36 Apache Web Servers

Implementation example: Modules are loaded with the directive `LoadModule` in a configuration file. Depending on the organisation of the configuration files there are different ways for the configuration.

If the `LoadModule` directives are in a configuration file, they can be deactivated with a comment sign:

```
#LoadModule ...
```

If the modules are configured by links in the `mods-enabled` directory, the links can be deleted to deactivate the modules.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources
- Disruption of availability
- Unnoticeable feasible attacks

---

**Req 9** If Server Side Includes (SSI) are active, execution of system commands must be deactivated.

The execution of system commands for SSI can be deactivated by setting the "IncludesNoExec" option in the "Directory" directive.

**Motivation:** The Server Side Includes (SSI) technology, which is implemented as an additionally loadable module, can potentially be used by attackers. The “exec” function of SSI, in particular, could be used to execute system commands, which represents a risk.

Implementation example: Enter options in the configuration file in the "Directory" directive:

```
<Directory document_directory>
  Options +IncludesNoExec ...
</Directory>
```

Replace text in italics appropriately.

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data
- Attacks motivated and facilitated by information disclosure or visible security weaknesses

---

**Req 10** HTTP methods that are not required must be deactivated.

Standard requests to web servers only use GET and POST. If other methods are required, they must be processed securely.

**Motivation:** HTTP TRACE could be misused by an attacker. This method allows for debugging and the trace analysis for connections between the client and the web server. Other HTTP methods could also be used to obtain information about the server, or they could be directly misused by an attacker.

Implementation example: The configuration of Apache versions 2.2 and 2.4 are different.

A new directive was introduced in Apache 2.4: `AllowMethods`

Add the following lines in the configuration file:

```
<Location />
  AllowMethods GET POST
</Location>
```

The `AllowMethods` directive is marked as "experimental" currently (Feb. 2014). If the expected functionality can not be achieved by this, then the following setup for version 2.2 can be used.
For version 2.2, add the following lines in the configuration file in the “Directory” directive:

```xml
<LimitExcept GET POST>
  deny from all
</LimitExcept>
```

In all versions the TRACE method needs to be deactivated explicitly. Add TraceEnable off to the configuration file outside the “Directory” directive.

For this requirement the following threats are relevant:

- Unauthorized access or tapping of data
- Attacks motivated and facilitated by information disclosure or visible security weaknesses

---

**Req 11** If symbolic links are not required, the function to follow symbolic links must be deactivated.

*Motivation: By using symbolic links, it is possible to access files outside the specified document master directory. This could lead to unwanted file access. The relevant risk can be reduced by restricting access rights.*

Implementation example: Enter options in the configuration file in the “Directory” directive:

```xml
<Directory document_directory>
  Options -FollowSymLinks
  ...
</Directory>
```

Replace text in italics appropriately.

For this requirement the following threats are relevant:

- Unauthorized access or tapping of data
- Attacks motivated and facilitated by information disclosure or visible security weaknesses

---

**Req 12** If symbolic links are required, file access must be restricted to those files, which belong to the owner of the link.

*Motivation: This limitation reduces the risk, to deliver files by the web server, which are not intended to be delivered.*

Implementation example: Enter options in the configuration file in the “Directory” directive:

```xml
<Directory document_directory>
  Options +SymLinksIfOwnerMatch
  ...
</Directory>
```

Replace text in italics appropriately.

For this requirement the following threats are relevant:

- Unauthorized access or tapping of data

---

**Req 13** If CGI is used, CGI wrappers must be used for the initial parameter review/validation and setup of the correct runtime environment for CGI programs.

*Motivation: Using CGI, the web server communicates with runtime environments which could be misused to execute malicious software.*

For this requirement the following threats are relevant:

- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unnoticeable feasible attacks
Req 14 If CGI is used, the CGI directory must not include compilers or interpreters (e.g., PERL interpreter, PHP interpreter/compiler, Tcl interpreter/compiler or operating system shells).

*Motivation: Using CGI, the web server communicates with runtime environments which could be misused to execute malicious software.*

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources
- Disruption of availability
- Unnoticeable feasible attacks

Req 15 If CGI is used, the CGI directory must not be used for uploads.

*Motivation: Using CGI, the web server communicates with runtime environments which could be misused to execute malicious software.*

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unnoticeable feasible attacks

Req 16 If WebDAV is used for writing files, access must not be granted without successful authentication.

*Motivation: WebDAV makes it possible to update content online which has been made available by the web server. This function could therefore be misused to change website content.*

For this requirement the following threats are relevant:
- Unauthorized modification of data

Req 17 If WebDAV is used, access rights must be configured restrictively.

Access rights to all files accessible by WebDAV must be configured as restrictively as possible. Additionally, if WebDAV is used, WebDAV access must be restricted to the directories required.

*Motivation: WebDAV makes it possible to update content online which has been made available by the web server. This function could therefore be misused to change website content.*

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data
- Unauthorized modification of data

Req 18 Directory listings (indexing) must be deactivated.

*Motivation: Directory listings provide information on files and directory structures which could be misused.*
3.36 Apache Web Servers

Implementation example: Enter options in the configuration file in the “Directory” directive:

```xml
<Directory document_directory>
  Options -Indexes
...  
</Directory>
```

Replace text in italics appropriately.

For this requirement the following threats are relevant:

- Unauthorized access or tapping of data

---

**Req 19** Script assignments that are not required must be deleted.

Script assignments that are not required must be deleted, e.g., php, phtml, js, sh, csh, bin, exe, pl, vbe, vbs.

*Motivation: Define script assignments as to how scripts and programs are executed on the server. These assignments could provide a runtime environment for files which were placed by an attacker. This file execution must be blocked.*

Implementation example: Delete script file extensions in mime.types and “AddType” lines in the configuration file which are not required.

For this requirement the following threats are relevant:

- Unauthorized access or tapping of data
- Unauthorized modification of data
- Disruption of availability
- Unnoticeable feasible attacks

---

**Req 20** The web server must be robust against overload situations.

A web server must provide security measures to deal with overload situations. In particular, partial or complete impairment of web server availability must be avoided. Potential protective measures include:

- Restricting the maximum number of HTTP sessions per IP address
- Defining the maximum size of a HTTP request
- Defining a timeout for HTTP requests

*Motivation: Attackers often try to bring a web server into an overload situation by using denial-of-service (DoS) attacks. If such an attack is successful the web server's availability or integrity may be impaired.*

Implementation example: Configure protecting measures with the following directives. The stated values are examples, which may need to be adapted to a concrete situation. If, for example, files can be uploaded via the web server, the value for LimitRequestBody must be increased accordingly.

Detailed information on the directives can be found in the Apache documentation.

Timeout (in seconds) for receiving HTTP requests:
```
RequestReadTimeout header=10-20,MinRate=500 body=10-20,MinRate=500
```

Timeout (in seconds) for waiting on certain events of HTTP communication:
```
Timeout 30
```

Timeout (in seconds) for subsequent requests on a persistent connection:
```
KeepAliveTimeout 5
```
Limits on HTTP requests:
- LimitRequestBody 10000 (Maximum number of bytes of the body, to be increased for file upload functionality, e.g. 10000000)
- LimitRequestFields 50 (Maximum number of request header fields)
- LimitRequestFieldSize 8190 (Maximum number of bytes of an HTTP header)
- LimitRequestLine 8190 (Maximum number of bytes of the request line)
- LimitXMLRequestBody 10000 (Maximum number of bytes of an XML-based request body)

Limit the number of connections that will be processed simultaneously:
- MaxRequestWorkers 256 (this is differently defined for threaded and non-threaded server, see the documentation)

For this requirement the following threats are relevant:
- Disruption of availability

---

Req 21 Default content must be removed.

Default content (examples, help files, documentation, aliases) that is provided with the standard installation must be removed.

Motivation: By using examples, information could be obtained about the installed software (version). Examples can include security vulnerabilities.

Implementation example: Directories which could contain default content can be found in the configuration file in the directory directives. There must not be any directory directives for directories which are not to be used by the web server, and the content must be deleted. Configured directories should only contain files which are required for the application.

Directories of standard installations are for example:
- /opt/apache2/
- /var/www/

Alias definitions are applied in the configuration file in alias or script alias directives. These entries should only exist if the alias is actually being used.

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data
- Attacks motivated and facilitated by information disclosure or visible security weaknesses

---

Req 22 Information about the web server in HTTP headers must be minimized.

The HTTP header must not include information on the version of the web server and the modules/add-ons used.

Motivation: Any information about the web server could allow conclusions to be drawn about security vulnerabilities.

Implementation example: There are multiple alternatives to fulfill this requirement:

1. **mod_security**
   Activate mod_security and enter the following in the configuration file:
   ```
   ServerTokens Full
   SecServerSignature "Webserver"
   ```
   To ensure that this directive is able to work, the “ServerTokens” Apache directive must be set to “Full”. ModSecurity will overwrite the server signature data stored in this storage area with the data entered in the directive. If “ServerTokens” is not set to “Full,” it is possible that the storage area is not large enough to include the new data.

2. **Apache configuration “ServerTokens Prod”**
   Set the directive ServerTokens in the configuration file:
3.36 Apache Web Servers

3. Security functionality outside of the web server

A security proxy or a load balancer can modify the HTTP response header “Server” appropriately.

For this requirement the following threats are relevant:
- Attacks motivated and facilitated by information disclosure or visible security weaknesses

---

Req 23 Web server information in error pages, which are delivered by the web server, must be removed.

Default error pages must be replaced with user-defined error pages. User-defined error pages must include version information about the web server and the modules/addons used. Error messages must not include internal information such as internal server names, error codes, etc.

*Motivation: Any information about the web server could allow conclusions to be drawn about security vulnerabilities.*

Implementation example: Configure user-defined error pages in the configuration file with the “ErrorDocument” directive, for example:

```
ErrorDocument 500 /cgi-bin/tester
ErrorDocument 404 /cgi-bin/bad_urls.pl
ErrorDocument 401 /subscription_info.html
ErrorDocument 403 "Sorry can't allow you access today"
```

For this requirement the following threats are relevant:
- Attacks motivated and facilitated by information disclosure or visible security weaknesses

---

Req 24 Unauthorized overwriting of the web server configuration must be prevented.

*Motivation: The web server configuration could be modified by other users, e.g., with the help of .htaccess files. This would modify access rights which must be prevented.*

Implementation example: In the configuration file in the “Directory” directive:

```
<Directory document_directory>
  AllowOverride None
...
</Directory>
```

Replace text in italics appropriately.

For this requirement the following threats are relevant:
- Unauthorized modification of data

---

Req 25 Write access to executable files that are executed with system privileges must be restricted to users with system privileges.

Other user groups must not be able to modify these files. This is highlighted for start/stop/log rotation scripts which (by default) belong to web server user but which are ex-executed during server startup or by cron jobs with administrator/root rights.

*Motivation: A non-administrative owner could modify the file to execute other functions or to obtain administrator rights (e.g., root).*

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources
- Disruption of availability
HTTPS

Req 26 Data with need of protection must be secured against unauthorized viewing and manipulation during transmission and storage.

Adequate security measures for transmission and storage must be implemented on data with a need for protection that are classified as internal, confidential or strictly confidential. The chosen measure depends on the classification for the data and other factors such as the type of network used during transmission, the storage location for data, etc. Furthermore must be guaranteed that confidential and strictly confidential data will not be unprotected during temporary storage (e.g. in web cache, temporary folders).

All authentication data such as user names, passwords, PINs, etc. must be protected against unauthorized viewing and manipulation. This applies equally to permanent storage and transmission. Typical measures taken to protect authentication data or combinations of these are:

For storage:
- Client systems: encryption or obfuscation of authentication data, no persistent storage, limitation of access rights.
- Server systems: hashing of authentication data with PBKDF like scrypt or bcrypt or when this is not possible hashes with salt.

For transmission:
- Usage of cryptographically protected network protocols.
- Challenge response method (Disadvantage of this solution is that passwords needed in clear-text on server. If this solution is feasible depends on the individual threat scenario)

Files of a system that are needed for the functionality must also be protected against manipulation. This is necessary because system’s integrity can be damaged when the system access this kind of files. An example is the use of check-sum or cryptographic methods to validate if e.g. firmware images, patches, drivers or kernel modules are free of manipulations.

For transmission of data with a need of protection network protocols that are insecure due to insufficient security measures shall not be used. Examples are: SSLv3, SSHv1, FTP, Telnet, SNMPv1 and 2c. In case of these protocols a newer version without vulnerabilities or a secure alternative must be used.

**Motivation:** If data with a need of protection will not be secured an attacker could record or manipulate the data during transmission over a network. An example is the recording of user names and passwords during system administration with the telnet clear-text protocol. Storing data on a system without adequate protection may mean that unauthorized users can copy or modify it. One example is when passwords can be read out when they stored in an inadequate secured way (e.g. usage of unsecure hashing algorithms like MD5 or SHA-1) or even without encryption on a system or manipulation of firmware to affected the system integrity.

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data
- Unauthorized modification of data
3.36 Apache Web Servers

Req 27  For encryption with HTTPS the TLS protocol must be used.

SSL must be considered outdated and thus may not be activated or must be deactivated, respectively.

Motivation: Particularly SSLv2 has a number of weaknesses that make it impossible to use from a security point of view. TLS is the further development of SSL. It is already established for years so there is no need for further use of SSL.

Implementation example: Define allowed protocols in the configuration file:

```plaintext
SSLProtocol All -SSLv2 -SSLv3
```

Since Apache 2.4 does not support SSL version 2, the "-SSLv2" can be omitted there.

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data
- Unauthorized modification of data

---

Req 28  The web server must be configured in such a way that the use of the latest version of the TLS protocol is enabled.

In particular, the web server must be configured for the use of TLS 1.2.

Motivation: The latest version of the protocol offers the best possible protection and contains fixes to known vulnerabilities in previous versions of the protocol.

Implementation example: If TLS 1.2 is supported, depends not only on the apache configuration, but also, if the SSL module this supports. mod_ssl uses OpenSSL, which supports TLS 1.2 since version 1.0.1. If this version can not be used, mod_nss can be used instead.

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data
- Unauthorized modification of data

---

Req 29  The TLS configuration must not use any unsecure cipher suites.

Cipher suites with one of the following characteristics are not secure
- no server authentication,
- no encryption,
- encryption with DES, RC4 or any other algorithm with a key length of less than 128 bits
- use of MD5 as hash function

Motivation: Cipher suites known to be unsecure do not offer sufficient protection.

Implementation example: Define allowed ciphers in the configuration file:

```
```

(Source: applied-crypto-hardening.pdf, 03.07.2015 at https://bettercrypto.org)
For this requirement the following threats are relevant:
- Unauthorized access or tapping of data
- Unauthorized modification of data

---

**Req 30** SSL compression must be deactivated.

*Motivation: The SSL compression contains a vulnerability, which allows the "crime" attack.*

Implementation example: Deactivate the SSL compression in the configuration file:

```
SSLCompression off
```

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data
- Unauthorized modification of data

---

**Req 31** The TLS configuration must provide that the cipher suite considered most secure is being chosen with highest priority.

A cipher suite contains the definition of four algorithms. These are used for key exchange, authentication, encryption and as a hash function. General guidelines for the prioritization are

- For the key exchange the Diffie-Hellman method must be preferred because it offers perfect forward secrecy. Cipher suites using the Diffie-Hellman method usually may be identified by the strings DHE or ECDHE. ECDHE has higher priority than DHE.
- For encryption the Advanced Encryption Standard (AES) or Camellia with a key length as big as possible has to be used.
- As a hash function SHA-2 has to be used. This function usually may be identified by the string SHA followed by a number (256, 384 or 512). Warning: if the string SHA is not followed by a number this identifies the SHA-1 function which is significantly less secure.

*Motivation: When a TLS connection is being established a cipher suite is selected based on the cipher suites available both on client and on server side. In order to ensure a high compatibility to all kinds of client systems the web server must not only allow for the cipher suites considered most secure. To make sure that nevertheless for each client the best possible cipher suite is selected and thus the connection is best protected the configuration must contain an according prioritization.*

Implementation example: Activate the option SSLHonorCipherOrder in the configuration file:

```
SSLHonorCipherOrder On
```

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data
- Unauthorized modification of data

---

**Req 32** Certificates must be issued by a certification authority whose certificates are recognized by the commonly used web browsers.

For critical applications that can be used via the Internet, use of an extended validation certificate (EV certificate) is recommended.

*Motivation: Only if the certificate authority (CA) is contained in the CA list of the browser being used the browser can verify the authenticity of the server or web application*
3.36 Apache Web Servers

Stricter issuing criteria apply to EV certificates. If an EV certificate is used, this is visualized in the browser. Even if EV certificates do not improve security, their use increases the trustworthiness of the server for the user.

For this requirement the following threats are relevant:
• Attacks motivated and facilitated by information disclosure or visible security weaknesses

---

<table>
<thead>
<tr>
<th>Req 33</th>
<th>Certificates must lose their validity after a maximum of 36 months.</th>
</tr>
</thead>
</table>

**Motivation:** The methods used for analysing and breaking cryptographic processes are improved continuously. Therefore the security of the certificates can be ensured for a limited period only. But, according to a general estimation, the security of the certificates is ensured for the required validity period of three years, if an appropriate key length is used.

For this requirement the following threats are relevant:
• Unauthorized access or tapping of data
• Unauthorized modification of data

---

<table>
<thead>
<tr>
<th>Req 34</th>
<th>Certificates must have a key length of at least 2048 bits when using RSA or 256 bits when using ECC.</th>
</tr>
</thead>
</table>

**Motivation:** In order to guarantee the security of certificates over the validity period, the cryptographic keys must have an appropriate length. According to a general estimation, a key length of 2048 bits provides sufficient protection for the next years. For ECC algorithms, shorter key lengths already provide the same level of security.

For this requirement the following threats are relevant:
• Unauthorized access or tapping of data
• Unauthorized modification of data

---

**Logging**

<table>
<thead>
<tr>
<th>Req 35</th>
<th>Access to the webserver must be logged.</th>
</tr>
</thead>
</table>

The web server log must contain the following information:
• Access timestamp
• Source (IP address)
• Account (if known)
• URL
• Status code of web server response

Logging must be done considering the currently valid legal, wage and company regulations. These regulations state among others that logging of events can be done only earmarked. Logging of events for doing a work control of employees is not allowed.

**Motivation:** For the analysis of security incidents it is very important to have basic information on how the attack has been carried out. Since a webserver represents an external interface certain information about an attack is only available on the webserver, even if the attack is aimed at a downstream system. Thus logging on a web server is mandatory.

Implementation example: Logging of accesses is activated in the default configuration. A typical configuration for the access log might look as follows:
```
LogFormat "%h %l %u %t "%r" %>s %b" common
CustomLog logs/access_log common
```

For this requirement the following threats are relevant:
• Unnoticeable feasible attacks
3.39 Tomcat Application Servers

Introduction

This security requirement has been prepared based on the provisions of the Group Policy on IT/NT Security. The security requirement is used as a basis for an approval in the PSA process, among other things. It also serves as an implementation standard for provisions of the Group Policy on IT/NT Security in units which do not participate in the PSA process. These requirements shall be taken into account from the very beginning, including during the planning and decision-making processes. When implementing these security requirements, the precedence of national, international and supranational law shall be observed.

This document assumes, that a web server is running in front of the application server. If this is not the case, then the security requirements on web servers must be fulfilled by the application server or another component, which receives http requests before the application server. A load balancer could be such component.

The examples are valid for Tomcat version 8.0.

Security requirements

Req 1 Software and hardware components that are no longer supported by vendor, producer or developer must not be used.

Components that have reached end-of-life or end-of-support must not be used. Excluded are components that have a special support contract. This contract must guarantee the correction of vulnerabilities over components life-time.

Motivation: Hardware and software components that have reached end of life or end of support represent a risk for a system. This means that a vendor does not supply remedial updates or patches for a component should errors or vulnerabilities occur. This means that vulnerabilities cannot be fixed when they occur and could be exploited to compromise the system or to impair its availability.

For this requirement the following threats are relevant:
• Unauthorized access to the system
• Unauthorized access or tapping of data
• Unauthorized modification of data
• Disruption of availability

Req 2 Known vulnerabilities in software and hardware of the system must be fixed or protected.

Prior to installation of a software or hardware component, users must check whether any vulnerability has been discovered and published for the version they are installing. Any component that proves to have a vulnerability must not be installed or used. Excepted from this rule are components for which the vendor has already provided a measure to remedy the vulnerability, e.g. a patch, update or workaround. In this case, the additional measure must be implemented on the system. Furthermore it is a ongoing process during the complete life cycle of the system to fix upcoming vulnerabilities promptly.

Motivation: Publication of vulnerabilities increases the risk of successful exploitation by an attacker, especially since the published information usually includes details on how to exploit...
the vulnerability and tools that make exploitation possible. One example of this is when Web servers are compromised due to not patched vulnerabilities.

For this requirement the following threats are relevant:
• Unauthorized access to the system
• Unauthorized access or tapping of data
• Unauthorized modification of data
• Disruption of availability

Req 3 The application server must run under a dedicated (operating-system) account that only has the permissions required for operation.

The account must not have administrator permissions. If the application server needs to be launched with administrat-or permissions, then it has to be switched to another account once launched. This is comparable to normal web server situations.

Motivation: Every service that runs with administrator permissions constitutes a higher risk in respect of possible vul-nerabilities or misuse.

For this requirement the following threats are relevant:
• Unauthorized access or tapping of data
• Unauthorized modification of data

Req 4 Unused services and protocols must be deactivated.

Some application servers provide multiple interfaces for requests to the applications. Usually only one of each inter-face is used. All interfaces and their connected services, which are not required, must be deactivated.

Motivation: Services and protocols that are not required for system operation increase the potential attack surface and thus the risk of the system being compromised. This risk is further increased by the fact that a security inspection and an appropriate optimisation of the configuration for unused services and protocols will not be done.

Implementation example: Connectors, that are not used, are deleted in the file $CATALINA_HOME/conf/server.xml.

For this requirement the following threats are relevant:
• Unauthorized access to the system
• Unauthorized access or tapping of data
• Unauthorized modification of data
• Disruption of availability

Req 5 If the shutdown port is not needed, it must be deactivated.

Motivation: Via the shutdown port (default 8005) it is possible to terminate all web applications that are running under Tomcat at that moment, and thus provoke a DoS scenario. This option is generally not required and should therefore not be available.

Implementation example: Deactivation of the shutdown port:
In the file CATALINA_HOME/conf/server.xml, the shutdown port is set to -1 and thus deactivated:

```xml
<Server port="-1" shutdown="SHUTDOWN">
```

For this requirement the following threats are relevant:
• Disruption of availability
Req 6 If the shutdown port is needed, it must be secured by a strong password.

Motivation: Via the shutdown port (default 8005) it is possible to terminate all web applications that are running under Tomcat at that moment, and thus provoke a DoS scenario. This option must be secured against unauthorized usage.

Implementation example: Protection of the shutdown port by configuring a complex shutdown command:
In the file CATALINA_HOME/conf/server.xml, the shutdown command is filled with a complex value, thus making unauthorized usage more difficult:

```
<Server port="8005" shutdown="Rea!!yC0mplexW0rd">
```

For this requirement the following threats are relevant:
- Disruption of availability

Req 7 Automatic deployment of applications must be disabled.

Motivation: Applications are started by automatic deployment after they are copied in a specific directory. This function could be misused.

Implementation example: The following configuration of the Host element in the $CATALINA_HOME/conf/server.xml file disables automatic deployment.

```
<Host ... autoDeploy="false" .... >
```

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Disruption of availability

Req 8 Information with need of protection must not be contained in files, outputs or messages that are accessible by unauthorized users.

Information with need of protection must not be accessible in files, outputs or messages of the system by unauthorized users. This includes information relating to the operating system, used middleware or applications such as vendor, product name, product identifier, installed software versions, installed service packs, patches, hot fixes and serial numbers. Examples for system messages which must be free of sensitive data are:

- Comments in downloadable files
- Error and system messages
- Stack traces
- Network protocols
- Login windows and dialogs

Furthermore, details of implementation and information relating, e.g., to backend software/systems, function calls, SQL instructions or structure of database, must not be contained in error messages. Excluded from this are displays and outputs that can be viewed and retrieved by authorized users who are logged in. In addition, an internal transfer of system internal information for error analysis is allowed in an adequate dimension. In this case the continuative regulations or guidelines (e.g. of data privacy) must be noticed.
Motivation: The information named above can be used by an attacker to prepare specific attacks on a system. In this way an attacker could, for example, use the precise software version to identify vulnerabilities in the product and, in a second step, exploit them.

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data
- Attacks motivated and facilitated by information disclosure or visible security weaknesses

---

**Req 9** Sample applications and unnecessary standard tools must be deleted.

The Tomcat application server includes samples and administration tools in the standard configuration.

**Motivation:** Sample applications and administration tools may provide points of attack.

**Implementation example:** Delete all unnecessary directories below

```text
CATALINA_HOME/webapps
```

For this requirement the following threats are relevant:
- Attacks motivated and facilitated by information disclosure or visible security weaknesses

---

**Req 10** If the “manager” application is used, this must be protected against unauthorized use.

To use the “manager” application securely, it is mandatory to set up an administration account and to configure the access via https. Especially in automated roll-outs ensure that a password is only used in one configuration.

**Motivation:** Without sufficient access protection, the “manager” application can be misused.

**Implementation example:** Define administrator accounts in the file

```text
CATALINA_HOME/conf/tomcat-users.xml.
```

```xml
<user username="Admin1" password="fbb574b6ce19ae918a63e255f13904db4976ce30"
 roles="manager-gui"/>
<user username="Admin2" password="7f45df11051861a541d280d0037a6b79de95b29d"
 roles="manager-gui"/>
```

The name for the administrator and the password must be suitably replaced. Passwords are not stored in clear-text, but as SHA-1 hashes. To create hashes Tomcat contains the tools "digest.sh" resp. "digest.bat" in the bin directory. (More secure hash algorithms are not available.)

To configure the SHA hashing, the parameter “digest” with the value “sha” needs to be added in the UserData-Base-Realm in the file CATALINA_HOME/conf/server.xml:

```xml
<Realm className="org.apache.catalina.realm.UserDatabaseRealm"
digest="sha" resourceName="UserDatabase"/>
```

**Note:**
The used role name was changed from “manager” with effect from Version 6.0.30 since this version introduced various roles with different permissions. Details can be found in the Tomcat documentation.

Redirecting the http port to https:
The setting up of an https connector is described in the Tomcat documentation (“SSL Configuration HOW-TO”).

https is redirected by adjusting the file CATALINA_HOME/webapps/manager/WEB-INF/web.xml in the <security-constraint> tags of all <web-resource-name> entries. Here three lines are added under the </authconstraint> in each case:
For this requirement the following threats are relevant:

- Unauthorized access to the system

---

**Req 11** Access to the application server must only be possible from approved IP addresses (IP whitelisting).

*Motivation: In most cases the application server is only accessed from certain, i.e., known web servers. The restriction to selected source addresses prevents possible misuse from other computers and substantially reduces the risk of an attack.*

Implementation example: The allowed IP addresses can be defined for each application separately. Is is done in the file

```xml
$CATALINA_HOME/webapps/<application>/META-INF/context.xml.

  <Context ...>
    <Valve className="org.apache.catalina.valves.RemoteAddrValve"
        allow="127\.
```
For each application, a separate service must be configured within the application server.

**Motivation:** It should not be possible to influence an application by modifying the configuration of another application.

Implementation example: For each application a separate service is defined in the file $CATALINA_HOME/conf/server.xml. Each Service uses an own directory in $CATALINA_HOME/webapps. This example shows two services:

```xml
<Service name="service1">
  <Connector port="8080" protocol="HTTP/1.1"
    connectionTimeout="20000"
    redirectPort="8443" />
  <Engine name="engine1" defaultHost="localhost">
    ...

    <Host name="localhost" appBase="webapps/engine1"
      unpackWARs="true" autoDeploy="false"
      deployOnStartup="true">
      <Valve class="org.apache.catalina.valves.AccessLogValve" directory="logs"
        prefix="localhost_access_log" suffix=".txt"
        pattern="service1:%h %l %u %t "%r" %s %b" />
    </Host>
  </Engine>
</Service>

<Service name="service2">
  <Connector port="8081" protocol="HTTP/1.1"
    connectionTimeout="20000"
    redirectPort="8444" />
  <Engine name="engine2" defaultHost="localhost">
    ...

    <Host name="localhost" appBase="webapps/engine2"
      unpackWARs="true" autoDeploy="false"
      deployOnStartup="true">
      <Valve class="org.apache.catalina.valves.AccessLogValve" directory="logs"
        prefix="localhost_access_log" suffix=".txt"
        pattern="service2:%h %l %u %t "%r" %s %b" />
    </Host>
  </Engine>
</Service>
```

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data
- Unauthorized modification of data
Req 13  Access to the application server must be logged.

The application server log must contain the following information:

- Access timestamp
- Source (IP address)
- Account (if known)
- URL
- HTTP status code of application server response

Logging must be done considering the currently valid legal, wage and company regulations. This regulations state among others that logging of events can be done only earmarked. Logging of events for doing a work control of employees is not allowed.

**Motivation:** For the analysis of security incidents it is very important to have basic information on how the attack has been carried out.

For this requirement the following threats are relevant:
- Unnoticeable feasible attacks
3.45 SNMP

Introduction
This security requirement has been prepared based on the provisions of the Group Policy on IT/NT Security.

The security requirement is used as a basis for an approval in the PSA process, among other things. It also serves as an implementation standard for provisions of the Group Policy on IT/NT Security in units which do not participate in the PSA process. These requirements shall be taken into account from the very beginning, including during the planning and decision-making processes. When implementing these security requirements, the precedence of national, international and supranational law shall be observed.

General requirements for the use of SNMPv3

<table>
<thead>
<tr>
<th>Req 1</th>
<th>It must be used version 3 of SNMP protocol.</th>
</tr>
</thead>
</table>

Motivation: In today's networks, the older versions of SNMP no longer offer the features required for secure management. For example, secure authentication and data transmission (encryption) is not possible with older variants. Only SNMPv3 supports encryption algorithms which correspond to the state of the art and for which no attacks are known.

For this requirement the following threats are relevant:

- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unnoticeable feasible attacks
- Attacks motivated and facilitated by information disclosure or visible security weaknesses

Access control

<table>
<thead>
<tr>
<th>Req 2</th>
<th>The SNMP server must avoid, that the value of the HMAC length becomes too small.</th>
</tr>
</thead>
</table>

Many current applications allow SNMP clients to define the length of the HMAC themselves. This is a potential security vulnerability.

Motivation: For example, if the client sets the HMAC value to a number of bits that is too low, it would become very easy to guess the key and an attacker could easily gain access to the system.

For this requirement the following threats are relevant:

- Unauthorized access or tapping of data

<table>
<thead>
<tr>
<th>Req 3</th>
<th>Predefined authentication attributes must be deleted or disabled.</th>
</tr>
</thead>
</table>

Normally, authentication attributes such as password or cryptographic keys will be preconfigured from producer, vendor or developer of a system. Such authentication attributes must be changed an own attribute not known by other parties.

Motivation: Authentication attributes like password or cryptographic keys preconfigured from third parties are not trustable. Such authentication attributes can be used to compromise systems or their data.
For this requirement the following threats are relevant:

- Unauthorized access to the system
- Unauthorized use of services or resources
- Denial of executed activities

**Req 4** Accounts must be protected against unauthorized use by at least one authentication attribute.

The various user and machine accounts on a system must be protected from misuse. To this end, an authentication attribute is typically used, which, when combined with the user name, enables unambiguous authentication and identification of the authorized user.

Authentication attributes include:

- Cryptographic keys
- Token
- Passwords
- PINs

This means that authentication based on a parameter that can be spoofed (e.g. phone numbers, public IP addresses or VPN membership) is not permitted. Exceptions are attributes that cannot be faked or spoofed by an attacker. Several of the above options can be combined (dual-factor authentication) to achieve a higher level of security. Whether or not this is suitable and necessary depends on the protection needs of the individual system and its data and must be evaluated for individual cases.

*Motivation:* Accounts that are not protected with a secret authentication attribute can be used by an attacker to gain unauthorized access to a system and the data and applications stored on it.

For this requirement the following threats are relevant:

- Unauthorized access to the system
- Unauthorized use of services or resources
- Denial of executed activities

**Req 5** If a password is used as an authentication attribute, it must have at least 8 characters and contain three of the following categories: upper cases, lower case, numbers and special characters.

A system may only accept passwords that comply with the following complexity:

- Minimum length of 8 characters.
- Comprising at least three of the following categories: upper/lower case letters, numbers and special characters

When a password is assigned, the system must ensure that the password meets these requirements. If a central system will be used for user authentication this function can be forwarded or delegated to this system.

*Motivation:* Passwords with the above complexity offer high robustness against attacks coupled with acceptable user friendliness. Passwords with this level of complexity have proved their efficiency in practice. Trivial passwords that are too short are susceptible to brute force and dictionary attacks and are therefore easy for attackers to determine. Once a password has been ascertained it can be used by an attacker for unauthorized access to the system and the data on it.

For this requirement the following threats are relevant:

- Unauthorized access to the system
- Unauthorized use of services or resources
- Denial of executed activities
Req 6 Authentication and encryption must be activated depending on the "need for protection" of the handled data.

Base for this is the relating User Security Model (USM) "Auth" requires an authentication of the user before he’s able to set a command. With activation of "Privacy" messages are transmitted in encrypted format between the SNMP manager and SNMP agent with the help of a "shared key."

Minimum specifications for a User Model:

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>Auth</th>
<th>Auth / Priv</th>
</tr>
</thead>
<tbody>
<tr>
<td>write</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>read</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>alarming</td>
<td>X</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Data with need for protection must be encrypted

*Motivation: With the additional activation of “Privacy,” the security level is raised to an acceptable level. An attacker then needs to guess the HMAC and also the key used.*

**Third-party SNMP access to customer CPEs**

Req 7 Information with need of protection must not be contained in files, outputs or messages that are accessible by unauthorized users.

Information with need of protection must not be accessible in files, outputs or messages of the system by unauthorized users. This includes information relating to the operating system, used middleware or applications such as vendor, product name, product identifier, installed software versions, installed service packs, patches, hot fixes and serial numbers. Examples for system messages which must be free of sensitive data are:

- Comments in downloadable files
- Error and system messages
- Stack traces
- Network protocols
- Login windows and dialogs

Furthermore, details of implementation and information relating, e.g., to backend software/systems, function calls, SQL instructions or structure of database, must not be contained in error messages. Excluded from this are displays and outputs that can be viewed and retrieved by authorized users who are logged in. In addition, an internal transfer of system internal information for error analysis is allowed in an adequate dimension. In this case the continuative regulations or guidelines (e.g. of data privacy) must be noticed.

*Motivation: The information named above can be used by an attacker to prepare specific attacks on a system. In this way an attacker could, for example, use the precise software version to identify vulnerabilities in the product and, in a second step, exploit them.*

For this requirement the following threats are relevant:

- Unauthorized access or tapping of data
- Attacks motivated and facilitated by information disclosure or visible security weaknesses
Req 8 If there's a customer contract based option for gaining access to components, it has to be assured, that this access is a restricted “read only” one and that no data with need for protection can be retrieved.

Under certain circumstances, business customers may be granted read SNMP access to the CPEs and components.

For this requirement the following threats are relevant:
• Unauthorized access or tapping of data
• Attacks motivated and facilitated by information disclosure or visible security weaknesses
3.50 Cryptographic Algorithms and IP Security (IPSec)

Introduction

1.1. General Introduction

This security requirement has been prepared based on the provisions of the Group Policy on IT/NT Security.

The security requirement is used as a basis for an approval in the PSA process, among other things. It also serves as an implementation standard for provisions of the Group Policy on IT/NT Security in units which do not participate in the PSA process.

These requirements shall be taken into account from the very beginning, including during the planning and decision-making processes. When implementing these security requirements, the precedence of national, international and supranational law shall be observed.

1.2. Remarks

This document is a general base for the work with cryptographic algorithms. This means that not every requirement is relevant if system or application specific requirements regarding the cryptographic algorithms are already in place.

Cryptographic algorithms for mobile network are out of scope of this document. This requirements are already addressed through the requirement document "3.38 Security Parameters for Mobile Networks" and defined in specifications published by the 3rd Generation Partnership Project (3GPP) and the International Telecommunication Union (ITU).

General Principles for Cryptographic Algorithms

Req 1 Cryptographic algorithms recommended by at least two acknowledged institutions must be used.

All algorithms which are listed in this document fulfill this requirement. Therefore this requirement is only applicable if deviating algorithms shall be used.

Acknowledged institutions are:

- German Information Security Agency - Bundesamt für Sicherheit in der Informationstechnik (BSI)
- European Union Agency for Network and Information Security Agency (ENISA)
- European Network of Excellence in Cryptography II (ECRYPT II)
- National Institute of Standards and Technology (NIST)

Motivation: Only well investigated cryptographic algorithms have a sufficient systematic protection against cryptographic attacks. The security of a cryptographic system must be approved by at least two of four entities listed above. It shall be ensured that the recommendations are based on a common functional consent.

Implementation example: The AES operation mode CFB is recommended by NIST. However this operation mode is not recommended by BSI because of potential weaknesses. Potential security doubts might not be identified if one relied on only one single recommendation.
For this requirement the following threats are relevant:

- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Denial of executed activities
- Attacks motivated and facilitated by information disclosure or visible security weaknesses

---

**Symmetric Encryption Algorithms**

**Req 3** The following symmetric encryption algorithms must be used: AES-256, AES-192 or AES-128.

The following symmetric encryption algorithms are allowed as alternative to AES as well:

- 3DES-168 (3 keys)
- 3DES (2 keys)
- Camellia-256
- Camellia-192
- Camellia-128

AES is the short form for "Advanced Encryption Standard". It is also called "Rijndael" with AES-128, AES-192 and AES-256 standing for the different key lengths.

**Restrictions on 3DES (3 keys):**

3DES-168 stands for a triple execution of the Data Encryption Standard (DES). For every execution a dedicated 56 bit part of the 168 bit key is used. This algorithm cannot anymore be seen as secure for new applications. Nevertheless it is approved for legacy use.

**Restrictions on 3DES (2 keys):**
The algorithm 3DES (with 2 keys) has the same design as 3DES-168. The only difference between the two variants of 3DES is the key usage. For the 3DES the first and the third part of the 168 bit key are equal. This is the reason why the security level of the mentioned algorithm is fewer than 3DES-168. As a result the algorithm is only allowed for exceptional cases, if no other (stronger) encryption is available.

Restrictions on Camellia:
The Camellia algorithm is one of the available encryption algorithms of the Transport Layer Security (TLS) protocol. The algorithm can be operated in three different key lengths (128 bit, 192 bit and 256 bit). Camellia can be applied as alternative to AES for TLS.

_Motivation:_ Strong encryption ensures the confidentiality of sensitive data for a long time.

Implementation example: A webserver use a cipher suite with AES to ensure a client TLS-session:

_TLS_DHE_DSS_WITH_AES_256_CBC_SHA256_

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data
- Attacks motivated and facilitated by information disclosure or visible security weaknesses

---

__Req 4__ The following operation modes must be used: Galois-Counter-Mode (GCM), Cipher-Block-Chain-ing (CBC), Counter Mode (CTR) or Output Feedback (OFB).

If the input data is longer than the AES input block length of 128 bit, a Operation Mode of AES must be used to encrypt all data.

The following table represents the valid applications and operation conditions for the operation modes:

<table>
<thead>
<tr>
<th>Operation Mode</th>
<th>Random Access Encryption</th>
<th>Stream Encryption</th>
<th>Block Encryption</th>
</tr>
</thead>
<tbody>
<tr>
<td>GCM</td>
<td>NO</td>
<td>NO</td>
<td>YES</td>
</tr>
<tr>
<td>CBC</td>
<td>NO</td>
<td>NO</td>
<td>YES</td>
</tr>
<tr>
<td>CTR</td>
<td>YES</td>
<td>YES</td>
<td>NO</td>
</tr>
<tr>
<td>OFB</td>
<td>NO</td>
<td>YES</td>
<td>NO</td>
</tr>
</tbody>
</table>

**Random Access Encryption**
Applications with random access encryption require a access of any encrypted data fields. An example is the random file access of an encrypted file in a case of a hard disk encryption.

**Stream Encryption**
Continuous encryption / decryption of a data stream with an undefined length.

**Block Encryption**
Encryption / decryption of data blocks with a fixed length.

**Special Operation Conditions are only relevant for developers:**

**Special Operation Conditions on GCM:**
- The counter values in the counter part of the Initialization Vector (IV) must not be repeated in case of an un-changed secret key.
• A dedicated Initialization Vector for each input message is required.
• The implementation of the Initialization Vector (IV) must ensures that no external entity is able to change the value of the IV.
• Padding is required.

**Special Operation Conditions on CBC:**
• Only unpredictable Initialization Vectors (IV) are allowed.
• Dedicated Initialization Vectors for each input message are required.
• The implementation of the initialization vector (IV) must ensure that no external entity is able to change the value of the IV.
• Padding is required.

**Special Operation Conditions on CTR:**
• The counter values in the counter part of the Initialization Vector (IV) must not be repeated in case of an un-changed secret key.
• Padding is *not* required.

**Special Operation Conditions on OFB:**
• Only unpredictable Initialization Vectors (IV) are allowed.
• Dedicated Initialization Vectors for each input message are required.
• The implementation of the Initialization Vector (IV) must ensure that no external entity is able to change the value of the IV.
• Padding is *not* required.

*Motivation: Wrong implemented operation modes can cause a weakening of cryptographic systems.*

For this requirement the following threats are relevant:
• Unauthorized access or tapping of data
• Unauthorized modification of data
• Denial of executed activities
• Attacks motivated and facilitated by information disclosure or visible security weaknesses
Req 5 The following Padding procedures must be applied: ISO-Padding, CMS-Padding or ESP-Padding.

Readership of this requirement should be developers and suppliers.

Padding procedures must be applied (to fill up with missing bits) if the length of the input data block is smaller than the multiple of the fixed 128 bit input block length of AES.

Ciphertext Stealing:

Padding is unnecessary by the help of the special operation mode CBC with "Ciphertext stealing" (CBC-CS). Missing bits will be filled up with ciphertext.

The security "Ciphertext Stealing" is not recommended by at least two institutions. However the usage of Ciphertext Stealing is valid for reasonable cases (e.g. no other padding function is available / technical limitations).

Specification Sources:
- ISO-Padding [ISO/IEC 7816-4]
- CMS-Padding [IETF-RFC 5652]
- ESP-Padding [IETF-RFC 4303]
- Ciphertext Stealing CBC-CS1, CBC-CS2, CBC-CS3 [NIST-SP 800-38A - Addendum]

Motivation: Symmetric block ciphers (e.g. AES, 3DES) have fixed input block lengths. In case of CBC mode the input length is the multiple of 128 bit (e.g. 2x128 bit = 256 bit). Padding functions ensure a "secure" fill up of the input block to the next multiple of 128 bit (AES).

Implementation example: A 200 bit long data block shall be encrypted by AES-CBC. The length of the input block doesn't match with the multiple of 128 bit (128 bit, 256 bit, 384 bit,...). 56 bits are filled up to the next multiple (in this case: 256 bit) with ISO-Padding.

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Denial of executed activities
- Attacks motivated and facilitated by information disclosure or visible security weaknesses
Asymmetric Encryption Algorithms

Req 6 The following asymmetric encryption algorithms must be used: RSA-OAEP, RSA-KEM, PSEC-KEM or ECIES-KEM.

The following table represents the valid applications and minimum key lengths:

<table>
<thead>
<tr>
<th>Asymmetric Algorithm</th>
<th>Minimum Length of Private Key Parameter</th>
<th>Minimum Length of Public Key Parameter</th>
</tr>
</thead>
<tbody>
<tr>
<td>RSA-OAEP</td>
<td>( k \geq 2048 ) bit</td>
<td>no restriction</td>
</tr>
<tr>
<td>RSA-KEM</td>
<td>( k \geq 2048 ) bit</td>
<td>no restriction</td>
</tr>
<tr>
<td>PSEC-KEM (ElGamal)</td>
<td>( N \geq 224 ) bit</td>
<td>( L \geq 2048 ) bit</td>
</tr>
<tr>
<td>PSEC-KEM (ElGamal with elliptic curves)</td>
<td>( f \geq 224 ) bit</td>
<td>no restriction</td>
</tr>
<tr>
<td>ECIES-KEM (ElGamal with elliptic curves)</td>
<td>( f \geq 224 ) bit</td>
<td>no restriction</td>
</tr>
</tbody>
</table>

Remark on RSA-PKCS#1 v1.5:
RSA PKCS#1 v1.5 cannot more seen as secure. That is the reason why this scheme may not be used for encryption.

Restrictions on elliptic curves:
Following curve parameter may only be used
- \( \text{brainpoolP}224r1 \)
- \( \text{brainpoolP}256r1 \)
- \( \text{brainpoolP}320r1 \)
- \( \text{brainpoolP}384r1 \)
- \( \text{brainpoolP}512r1 \)
- \( \text{Curve}25519 \)

NIST-curves may only be used as backup (no other brainpool curve is available).

Abbreviations:
ECIES = Elliptic Curve Integrated Encryption Scheme
KEM = Key Encapsulation Mechanism
OAEP=Optimal Asymmetric Encryption Padding
PSEC = Provable Secure Elliptic Curve (encryption)
RSA = Rivest-Shamir-Adleman

Remark:
Listed parameters provide a comparable security level. Algorithms based on elliptic curve cryptography require a significant shorter key for a comparable security.

Motivation: The algorithms provide a sufficient protection and were recommended by acknowledged institutions.

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data
- Attacks motivated and facilitated by information disclosure or visible security weaknesses
Hash Functions

Req 7 The following hash functions from the SHA-2 family must be used: SHA-512, SHA-384, SHA-256 or SHA-224.

Following table represents the valid application of different hash functions for new systems:

<table>
<thead>
<tr>
<th>Hash</th>
<th>Confidentiality Protection (Passwords, Secret Keys)</th>
<th>Integrity Protection (Firmware, Software)</th>
<th>Digital Fingerprint (Signatures)</th>
<th>Message Authentication (MAC)</th>
<th>Key Derivation (KDF)</th>
<th>Random Number Generation (RNG)</th>
</tr>
</thead>
<tbody>
<tr>
<td>SHA-512</td>
<td>YES</td>
<td>YES</td>
<td>YES</td>
<td>YES</td>
<td>YES</td>
<td>YES</td>
</tr>
<tr>
<td>SHA-384</td>
<td>YES</td>
<td>YES</td>
<td>YES</td>
<td>YES</td>
<td>YES</td>
<td>YES</td>
</tr>
<tr>
<td>SHA-256</td>
<td>YES</td>
<td>YES</td>
<td>YES</td>
<td>YES</td>
<td>YES</td>
<td>YES</td>
</tr>
<tr>
<td>SHA-224</td>
<td>YES</td>
<td>YES</td>
<td>YES</td>
<td>YES</td>
<td>YES</td>
<td>YES</td>
</tr>
<tr>
<td>SHA-1</td>
<td>NO</td>
<td>NO</td>
<td>NO</td>
<td>NO</td>
<td>NO</td>
<td>NO</td>
</tr>
<tr>
<td>RIPEMD-160</td>
<td>NO</td>
<td>NO</td>
<td>NO</td>
<td>YES</td>
<td>YES</td>
<td>YES</td>
</tr>
<tr>
<td>MD5</td>
<td>NO</td>
<td>NO</td>
<td>NO</td>
<td>NO</td>
<td>NO</td>
<td>NO</td>
</tr>
<tr>
<td>RIPEMD-128</td>
<td>NO</td>
<td>NO</td>
<td>NO</td>
<td>NO</td>
<td>NO</td>
<td>NO</td>
</tr>
</tbody>
</table>

Restrictions on SHA-1/RIPEMD-160:
SHA-1/RIPEMD-160 usage is only allowed for Message Authentication Codes (MAC) and Key Derivation Functions (KDF) and should be exchanged against stronger hash functions (SHA-2 family).

Restrictions on MD5/RIPEMD-128:
MD5/RIPEMD-128 usage is only allowed on legacy systems in case that no stronger hash function is available. MD5/ RIPEMD-128 must be exchanged against stronger hash functions (SHA-2 family) soon.

Prospects to SHA-3:
The standardization procedure for SHA-3 hash functions is still on going. That is the reason why no (public) entity has recommended this hash function yet.

Following variants will likely be provided:
- SHA3-224
- SHA3-256
- SHA3-384
- SHA3-512

From security perspective SHA-3 will basically provide the same cryptography strength than SHA-2 but the algorithm design will become more efficient and therefore SHA-3 should be faster than SHA-2.

Motivation: Secure hash functions are applied in many applications to protect the confidentiality and integrity of sensitive data.

Implementation example: Store of a hashed password in a database record.
For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Denial of executed activities
- Attacks motivated and facilitated by information disclosure or visible security weaknesses

### Message Authentication Codes (MAC)

**Req 8** The following Message Authentication Codes (MAC) must be used: CMAC (based on AES algorithm operated in CBC mode) or HMAC (based on SHA hash functions).

The following table represents the valid algorithms and parameter:

<table>
<thead>
<tr>
<th>MAC</th>
<th>Approved Algorithms</th>
<th>Special Operation Conditions / Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>CMAC</td>
<td>• AES-CBC-256</td>
<td>• Key lengths are used according to used AES variant (128 bit, 192 bit, 256 bit).</td>
</tr>
<tr>
<td></td>
<td>• AES-CBC-192</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• AES-CBC-128</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• 3DES-CBC (3 keys)</td>
<td></td>
</tr>
<tr>
<td>HMAC</td>
<td>• SHA-512</td>
<td>• Key lengths must be at least as long as the output value of the hash function.</td>
</tr>
<tr>
<td></td>
<td>• SHA-384</td>
<td>• Key length must be at least 128 bit long.</td>
</tr>
<tr>
<td></td>
<td>• SHA-256</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• SHA-224</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• SHA-1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• RIPEMD-160</td>
<td></td>
</tr>
</tbody>
</table>

**Restrictions on MD5/RIPEMD-128:**
The use of MD5/RIPEMD-128 for Message Authentication Codes (MAC) is only valid for legacy systems if no stronger hash function (minimum SHA-1) is available.

**Motivation:** Message Authentication Codes (MAC) shall protect the integrity of messages in security protocols as IPSec or TLS/SSL. These messages won't be stored persistently. Therefore the cryptographic protection isn't required for a long time. That's why the weak hash function SHA-1 is allowed as well.

Implementation example: Integrity protection of billing data records by HMAC-SHA-1.

For this requirement the following threats are relevant:
- Unauthorized modification of data
- Denial of executed activities

### Signature Algorithms

**Req 9** The following signature algorithms must be used: RSA-PSS, DSA / DSS or EC-DSA.

Signature algorithms are used to verify the authenticity of the sender and the integrity of a signed message. The verification is performed by decryption with the public key. Signature algorithms always require a hash function to calculate a fingerprint.

The following table represents the valid parameters for digital signatures:
<table>
<thead>
<tr>
<th>Signature Algorithm</th>
<th>Minimum Length of Private Key Parameter</th>
<th>Minimum Length of Public Key Parameter</th>
<th>Valid Hash Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>RSA (-PSS)</td>
<td>k&gt;=3072 bit</td>
<td>no restriction</td>
<td>• SHA-224</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• SHA-256</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• SHA-384</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• SHA-512</td>
</tr>
<tr>
<td>DSA / DSS</td>
<td>N&gt;=256 bit</td>
<td>L&gt;=3072 bit</td>
<td>• SHA-224</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• SHA-256</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• SHA-384</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• SHA-512</td>
</tr>
<tr>
<td>EC-DSA</td>
<td>f&gt;=256 bit</td>
<td>no restriction</td>
<td>• SHA-224</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• SHA-256</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• SHA-384</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• SHA-512</td>
</tr>
</tbody>
</table>

Restrictions on RSA PKCS#1 v1.5:

The use of RSA PKCS#1 v1.5 is only allowed on legacy systems and should be substituted by stronger signature algorithms.

Restrictions on MD5/SHA-1/RIPEMD-128/RIPEMD-160:

MD5/SHA-1/RIPEMD-128/RIPEMD-160 hash functions must not be applied for new signatures. However MD5/SHA-1/RIPEMD-128/RIPEMD-160 may be used for verification of legacy signatures.

Restrictions on elliptic curves:
Following curve parameter may only be used

• brainpoolP224r1
• brainpoolP256r1
• brainpoolP320r1
• brainpoolP384r1
• brainpoolP512r1
• Curve25519

NIST-curves may only be used as backup (no other brainpool curve is available).

Abbreviations:

DSA = Digital Signature Algorithm
DSS = Digital Signature Standard
EC = Elliptic Curve
PKCS = Public Key Cryptography Standards
PSS = Probabilistic Signature Scheme

Motivation: Secure digital signatures are the base of a unique (digital) identity.

Implementation example: x509 certificate mit following signature algorithm: sha-224WithRSAEncryption
For this requirement the following threats are relevant:
• Unauthorized modification of data
• Denial of executed activities

**IP Security (IPSec)**

If requirements for IP Security (IPSec) are needed, then only chapter 8 is applicable.

**General IPSec Parameter**

<table>
<thead>
<tr>
<th>Req 10</th>
<th>IPSec must be used with Internet Key Exchange (IKE) Version 2.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>If IKEv2 is supported, then IKEv2 must be used. If IKEv2 is not available, IKEv1 must be used.</td>
</tr>
</tbody>
</table>

*Motivation: IKEv2 has stronger Key Derivation Functions (KDF) in place. The tunnel establishment procedure for IKEv2 is faster as IKEv1 as well.*

For this requirement the following threats are relevant:
• Unauthorized access to the system
• Unauthorized access or tapping of data
• Unauthorized modification of data
• Denial of executed activities
• Attacks motivated and facilitated by information disclosure or visible security weaknesses

<table>
<thead>
<tr>
<th>Req 11</th>
<th>The following encryption algorithms for IPSec must be used: AES-256, AES-192, AES-128 or 3DES.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>The &quot;Encryption&quot; parameter has to be defined for the phase 1 (ISAKMP) and for the phase 2 (Quick Mode / ESP).</td>
</tr>
</tbody>
</table>

*Remark on AES:*

The IPSec specification only defines CBC mode and CTR mode for operation. That's why almost all IPSec products (e.g. IPSec Gateways) do not provide configuration functions for the mode of operation.

*Remark on 3DES (TDES / Triple-DES):*

3DES may be used in combination of Perfect Forward Secrecy (PFS). However this cipher should be substituted with AES in the near future.

*Motivation: Strong encryption protects the confidentiality of transferred data.*

*Implementation example: Cisco configuration example for the encryption of ESP:*

```plaintext
CE1_R1_4(config)#crypto ipsec transform-set test esp-aes
```

For this requirement the following threats are relevant:
• Unauthorized access or tapping of data
• Attacks motivated and facilitated by information disclosure or visible security weaknesses

<table>
<thead>
<tr>
<th>Req 12</th>
<th>The following Diffie-Hellman Groups for IPSec must be used: (DH) Group 14, (DH) Group 5 or (DH) Group 2.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>The parameter (DH) Group defines the method of key exchange and key establishment in the phase 1 and phase 2 (in combination of Perfect Forward Secrecy).</td>
</tr>
<tr>
<td></td>
<td>The difference between the (DH) groups is the different length of the modulus. The higher the modulus the higher the security level.</td>
</tr>
</tbody>
</table>
Restrictions on (DH) group 2:
(DH) group 2 may only be used in combination with 3DES as encryption algorithm.

Motivation: Key exchange and key establishment based on Diffie-Hellman shall ensure that only the tunnel endpoints have knowledge of the secret key even if it is used an insecure channel.

Implementation example: Cisco configuration example of Diffie-Hellman group in phase 1:
CE1_R1_4(config-isakmp)#group 14

Cisco configuration example of Diffie-Hellman group in phase 2:
CE1_R1_4(ipsec-profile)#set pfs group14

For this requirement the following threats are relevant:
• Unauthorized access or tapping of data
• Attacks motivated and facilitated by information disclosure or visible security weaknesses
Req 13 The following hash functions must be used for IPSec: SHA-512, SHA-384, SHA-256, SHA-224 or SHA-1.

Hash functions are the base for Message Authentication Codes (MAC) to protect the integrity of IPSec messages.

Restrictions on MD5:
MD5 may only be used on legacy systems and IPSec connections. MD5 must be substituted towards stronger hash functions (minimum SHA-1) in the near future.

Motivation: Message Authentication ensures that no tampered packets shall be processed or forwarded. Message Authentication Codes also prevents successful spoofs of source addresses in IPSec headers.

Implementation example: Cisco configuration example for hash in phase 1:
CE1_R1_4(config-isakmp)#hash sha

For this requirement the following threats are relevant:
• Unauthorized modification of data
• Denial of executed activities

ID: 3.50-13/2.0

8.2. Phase 1 (ISAKMP Security Association)
The following table represents the approved parameters for new systems and IPSec security associations:

<table>
<thead>
<tr>
<th>Authentication</th>
<th>Encryption</th>
<th>Hash</th>
<th>Group (Diffie-Hellman)</th>
<th>Mode (IKEv1)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Certificate-based</td>
<td>AES-256</td>
<td>SHA-512</td>
<td>Group 14</td>
<td>Main Mode</td>
</tr>
<tr>
<td>Pre-Shared Key (PSK)-based</td>
<td>AES-192</td>
<td>SHA-384</td>
<td>Group 5</td>
<td></td>
</tr>
<tr>
<td></td>
<td>AES-128</td>
<td>SHA-256</td>
<td>Group 2</td>
<td></td>
</tr>
<tr>
<td></td>
<td>3DES-168</td>
<td>SHA-224</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>SHA-1</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Req 14 IKEv1 must be operated in Main Mode.

Motivation: Main Mode can only be configured with IKEv1.

For this requirement the following threats are relevant:
• Unauthorized access or tapping of data
• Attacks motivated and facilitated by information disclosure or visible security weaknesses
Req 15  
A certificate-based authentication must be preferred against Pre-Shared Key (PSK).

If a Pre-Shared Key (PSK) based authentication is applied, the following conditions must be fulfilled:

- A PSK must be at least 128 bit long.
- A PSK must be equally defined by both tunnel endpoints.
- A PSK must be transferred in an encrypted manner.
- It must be defined dedicated PSKs for each communication association.
- A PSK must be renewed regularly (at least every 24 months)

**Motivation:** A secure PSK ensures a sufficient security of the cryptographic system.

Implementation example: Each of the tunnel endpoints A and B defining an 64 bit long partial key (8 characters long ASCII-based secret). The partial secrets will be exchanged between A and B via encrypted emails.

For this requirement the following threats are relevant:

- Unauthorized access to the system
- Unauthorized access or tapping of data

**Phase 2 (Quick Mode)**

The following table represents the approved parameters for phase 2:

<table>
<thead>
<tr>
<th>Encapsulated Security Payload (ESP)</th>
<th>Authentication Header (AH)</th>
<th>Encryption</th>
<th>Hash</th>
<th>Perfect Forward Secrecy (PFS)</th>
<th>Anti-Replay (only Gateway)</th>
</tr>
</thead>
<tbody>
<tr>
<td>YES</td>
<td>NO</td>
<td>AES-256</td>
<td>SHA-512</td>
<td>Group 14</td>
<td>YES</td>
</tr>
<tr>
<td></td>
<td></td>
<td>AES-192</td>
<td>SHA-384</td>
<td>Group 5</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>AES-128</td>
<td>SHA-256</td>
<td>Group 2</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>3DES-168</td>
<td>SHA-224</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>SHA-1</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Encapsulated Security Payload (ESP):**
The ESP provides functions for confidentiality and integrity protection. Therefore the use of ESP is mandatory.

**Authentication Header (AH):**
The Authentication Header can encapsulates the ESP. Considering that the ESP already provides all required features for integrity protection and also for Anti-Replay, there is no need to use AH.

**Encryption:**
The IPSec specification defines AES encryption only in CBC or CTR mode. That is the reason why there is only the capability to choose an algorithm with a key length. The 3DES/TDES algorithm may be used as fallback.

**Hash:**
The chosen hash function is the base of Message Authentication (MAC). For this usage a persistent protection of the data is not needed.
That is the reason why SHA-1 may be used as well. MD5 may only be used on legacy systems and must be substituted towards SHA-1 or higher in the near future.
Perfect Forward Secrecy (PFS):
Perfect Forward Secrecy (PFS) ensures that encrypted messages which are logged by an attacker can not afterwards be decrypted.

Even if the attacker gains knowledge of the master key, Diffie-Hellman Group 1 cannot be seen as secure anymore. Therefore groups greater than 1 have to be applied.

Anti-Replay:
The Anti Replay function protects against manipulated packets sent by an attacker which have the spoofed identity of the original source.
3.50 Cryptographic Algorithms and IP Security (IPSec)

**Req 16** Encapsulated Security Payload (ESP) must be operated in Tunnel Mode.

**Remark on Authentication Header (AH):**
Authentication Header (AH) can be used in combination with ESP. Nevertheless does ESP already provide all required features for integrity protection and Anti-Replay. That is the reason why the use of AH is not necessary.

*Motivation: ESP already provides all required features for integrity protection and Anti-Replay.*

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Denial of executed activities
- Attacks motivated and facilitated by information disclosure or visible security weaknesses

**Key Management**

**General Key Management**

**Req 17** Physical random number generators or strong pseudo-random number generators (deterministic sources) must be used to generate cryptographic keys for algorithms.

If a pseudo-random number generator is used to generate a cryptographic key, a non-deterministic value must be used to calculate the seed value. That non-deterministic source must have a sufficient high entropy (random).

For both kinds of random number generation must be ensured that the output values (symbols) occur independent with the same probability.

*Motivation: If output values of a random number generator occur with different probabilities, then it can be exploited to deduce or derivate a key through statistical methods (analyses).*

Implementation example: Usage of the random function in Linux `/dev/random` to generate a seed value for a pseudo-random number generator.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Denial of executed activities
- Attacks motivated and facilitated by information disclosure or visible security weaknesses

**Req 18** Only secure Key Establishment methods may be used. An attacker may not be able to derivate or calculate a secret key from transferred messages.

Secure Key Establishment procedures are usually based on Diffie-Hellman Key Exchange. The advantage of this Key Exchange is, that the complete key will never be transported over the communication channel.

*Motivation: The security of an cryptographic system is in general based on following factors:*
- Quality of the key secrecy
- (cryptographic) length of the secret key
- Security of the cryptographic algorithm
3.50 Cryptographic Algorithms and IP Security (IPSec)

Implementation example: An example of a secure key establishment method is the (pseudo-) random number based key generation in combination with a Diffie-Hellman key exchange. This is for instance used by the Perfect Forward Secrecy (PFS) function in the Transport Layer Security (TLS) protocol.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Denial of executed activities
- Attacks motivated and facilitated by information disclosure or visible security weaknesses

Req 19 The following Key Derivation Functions (KDF) must be used to derivate cryptographic keys: NIST-SP800-108-KDF, NIST-SP800-56-KDF-A, NIST-SP800-56-KDF-B or NIST-SP800-56-KDF-C.

In dedicated cases of a shared-secrets (pre-shared key, key) usage it can be required to have several sub-keys (e.g. one as session key and one key for message authentication).

Key Derivation Functions (KDF) can generate several derivated sub-keys from one single secret key (master key) that an applicable for dedicated cases.

Key Derivation Functions (KDF) are based on a smart interconnection of pseudo-random number generators. These pseudo-random number generators often based on Keyed-Hash Message Authentication Codes (HMAC) and CBC-based Message Authentication Codes (CMAC).

Application of Key Derivation Functions:
- Binding protocol data to a key (e.g. source name, destination name, ...)
- Derivation of session keys or keys for dedicated tasks
- Postprocessing of random numbers with imbalanced probabilities

Motivation: Well investigated and recommended (by acknowledged institutions) Key Derivation Functions provide a sufficient protection and ensure comparable cryptographical strengths of the derivated sub-keys.

Implementation example: Generation of a HMAC key from a master key through the use of a KDF.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Denial of executed activities
- Attacks motivated and facilitated by information disclosure or visible security weaknesses

Req 20 The encryption method to transport or store a secret key (session keys, pre-shared keys, etc.) must have at least the same security level as the algorithm of the secret key.

Motivation: The security of an cryptographic system is in general based on following factors:
- Quality of the key secrecy
- (cryptographic) length of the secret key
3.50 Cryptographic Algorithms and IP Security (IPSec)

- Security of the cryptographic algorithm

Implementation example: Encryption of a AES-128 key (key length = 128 bit) with RSA-3072 (key length k=3072 bit).

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Denial of executed activities
- Attacks motivated and facilitated by information disclosure or visible security weaknesses

Req 21 Private keys of asymmetric cryptographic algorithms may only be known by the key owner.

Private keys are used for authentication, decryption of encrypted messages and for a digital signature of messages. A digital signature (= comparable with a personal signature) can only be proceeded with a private key and ensures the authenticity. That is the reason why private keys may not be located in central storage systems nor be distributed. Private Keys must be protected against any misuse via a personal secret (PIN or Password) as well.

Motivation: If an attacker gets knowledge of a private key, he is able to masquerate himself as the original user / system.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Denial of executed activities
- Attacks motivated and facilitated by information disclosure or visible security weaknesses

Req 22 Cryptographic keys (pre-shared keys, private keys, public keys, etc.) must be regularly renewed / changed.

The duration of validity is driven by the strength of the cryptographic algorithm. This means weak ciphers need a shorter renew-cycle than stronger algorithms. Renewed/changed keys may not be derived from existing keys.

Motivation: Compromised keys which have a too long validity might be used to decrypt further messages until the validity is expired.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Denial of executed activities
- Attacks motivated and facilitated by information disclosure or visible security weaknesses

Req 23 Key which are required no more or compromised must be destroyed irrecoverable by secure methods immediately.

If an attacker gets knowledge of an "old" pre-shared key, it is possible to decrypt logged messages (messages which were encrypted with the "old" PSK).

Motivation: If an attacker gets knowledge of an "old" pre-shared key, it is possible to decrypt logged messages (messages which were encrypted with the "old" PSK).

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized access or tapping of data
High-Security Key Management

Req 24 To generate keys for applications with high security requirements, a hardware security module (HSM) with physical noise generator must be used.

The HSM must be designed in such a way that the generated keys are unable to leave the HSM.

Motivation: Tamper-proof HSMs are provide a secure storage for secret keys. HSMs are ensure a defined access to the stored keys. They also provide a protection against a plaintext disclosure of cryptographic keys.

Implementation example:

- Smartcards with cryptographic processors for e-mail encryption
- HSM box to protect operator specific parameters for the user authentication in mobile networks

For this requirement the following threats are relevant:

- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Denial of executed activities
- Attacks motivated and facilitated by information disclosure or visible security weaknesses

Req 25 Encryption algorithms used for applications with high security requirements must be implemented in a Hardware Security Module (HSM).

Secret keys and other sensitive data may not leave the device.

The use of tamper-proof devices (HSM) ensures that the session keys cannot be compromised under any circumstances. Furthermore a HSM protects the cryptographic algorithm against side channel attacks.

Motivation: The use of tamper-proof devices (HSM) ensures that the session keys cannot be compromised under any circumstances. Furthermore a HSM protects the cryptographic algorithm against side channel attacks.

For this requirement the following threats are relevant:

- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Denial of executed activities
- Attacks motivated and facilitated by information disclosure or visible security weaknesses
### Table of Comparable Security Levels for Cryptographic Algorithms

<table>
<thead>
<tr>
<th>Security Level (the higher the better)</th>
<th>Symmetric Algorithms</th>
<th>Asymmetric Algorithms DSA / Diffie-Hellman</th>
<th>Asymmetric Algorithms RSA</th>
<th>Asymmetric Algorithms ECDSA (elliptic curves)</th>
<th>Hash for Signatures, Passwords + Data-integrity</th>
<th>Hash for HMAC, Key Derivation + Random Number Generation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3DES-168</td>
<td>L=2048/ N=224</td>
<td>k=2048</td>
<td>f=224-255</td>
<td>SHA-224 SHA-256 SHA-384 SHA-512</td>
<td>SHA-1 SHA-224 SHA-256 SHA-384 SHA-512</td>
</tr>
<tr>
<td>2</td>
<td>AES-128 Camellia-128</td>
<td>L=3072/ N=256</td>
<td>k=3072</td>
<td>f=256-383</td>
<td>SHA-256 SHA-384 SHA-512</td>
<td>SHA-1 SHA-224 SHA-256 SHA-384 SHA-512</td>
</tr>
<tr>
<td>3</td>
<td>AES-192 Camellia-192</td>
<td>L=7680/ N=384</td>
<td>k=7680</td>
<td>f=384-511</td>
<td>SHA-384 SHA-512</td>
<td>SHA-224 SHA-256 SHA-384 SHA-512</td>
</tr>
<tr>
<td>4</td>
<td>AES-256 Camellia-256</td>
<td>L=15360/ N=512</td>
<td>k=15360</td>
<td>f=512+</td>
<td>SHA-512</td>
<td>SHA-256 SHA-384 SHA-512</td>
</tr>
</tbody>
</table>

**Restrictions on elliptic curves:**
Following curve parameter may be used
- `brainpoolP224r1`
- `brainpoolP256r1`
- `brainpoolP320r1`
- `brainpoolP384r1`
- `brainpoolP512r1`
3.55 Network Based Storage Systems

Introduction
This security requirement has been prepared based on the provisions of the Group Policy on IT/NT Security.
The security requirement is used as a basis for an approval in the PSA process, among other things. It also serves as an implementation standard for provisions of the Group Policy on IT/NT Security in units which do not participate in the PSA process. These requirements shall be taken into account from the very beginning, including during the planning and decision-making processes. When implementing these security requirements, the precedence of national, international and supranational law shall be observed.

Storage Area Network (SAN)

Req 1 System functions with a need of protection must not be used or accessed without successful authentication and authorization.

The usage of a system functions or access of data classified as internal, confidential or strictly confidential must only be possible unambiguous user identification and successful authentication on basis of the user name and at least one authentication attribute. Excepted from this are functions for public use such as those for a Web server on the Internet, via which information is made available to the public. Examples for functions which require a prior authentication are network services (like SSH, SFTP, Web services), local access via a management console, local usage of operating systems and applications. The following examples are possibilities that could be used for authentication.

- Query user name and password
- Use of cryptographic keys and certificates (e.g. as Smartcard)

This requirement must also be applied to accounts that are only used for communication between systems (M2M).

Motivation: The authentication is necessary to doubtless identify a user because the allocated authorization, and therefore the access on data and services of the system depends on that.

For this requirement the following threats are relevant:

- Unauthorized access to the system
- Unauthorized use of services or resources
- Denial of executed activities

Req 2 Systems with a high protection requirement, mutual authentication of the FC components (switch, HBA, storage) must take place.

Access Control Lists (ACLs) offer a means of identification based on worldwide names (WWNs) when an FC component wants to join a fabric. For critical systems this type of basic identification is no longer adequate. For critical systems, such as save credit card information, is such a simple identification not adequate any more. Therefore, the majority of FC component manufacturers provides FC component authentication on the basis of various protocols such as the Switch Link Authentication Protocol (SLAP), Fibre Channel Authentication Protocol (FCAP) and Diffie Hellman - Challenge Handshake Authentication Protocol (DH-CHAP). SLAP and FCAP are based on authentication with certificates while DH-CHAP relies on the exchange of a mutual secret. Certificate-based authentication is recommended.
Motivation: These measures ensure that components cannot join the FC fabric without having the appropriate certificates or passwords on the component. As a result, the configuration of the fabric cannot be read or manipulated. Furthermore, an attacker who connects his component to the fabric does not obtain access to the fabric in order to eaves-drop on data traffic.

For this requirement the following threats are relevant:
• Unauthorized access to the system

Req 3 The access mode for the default zone must be denied to additional FC components (default zone – no access).

Only components that are members of the same zone can communicate with each other. Components that do not belong to any zone are automatically assigned to the default zone. To ensure that an unknown component cannot communicate via the default zone within a fabric, it must be denied access to the default zone.

Motivation: An attacker who connects to the FC switch with his host cannot communicate with any component or read communications between partners.

For this requirement the following threats are relevant:
• Unauthorized access to the system
3.55 Network Based Storage Systems

<table>
<thead>
<tr>
<th>Req 4</th>
<th>The reachability of services must be restricted.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Typically services that are enabled in the basic configuration are accessible over all interfaces of the system and can be reached from systems in connected networks. This availability is often not needed or meaningful for system functioning. For this reason, services should only be enabled on interfaces where their usage is required. On interfaces where services are active, the reachability must be limited to legitimate communication peers. This limitation must be realized on the system itself (without measures (e.g., firewall) at the network side).</td>
</tr>
<tr>
<td></td>
<td><strong>Motivation:</strong> Disabling services on interfaces which do not require system accessibility or by limiting the reachability can greatly reduce the potential vulnerabilities offered to an attacker. For example, access to a system via SSH from the Internet is not necessary. If this service could be accessed from the interface connected to the Internet, this would greatly increase the risk of attacks on the service.</td>
</tr>
<tr>
<td></td>
<td>For this requirement the following threats are relevant:</td>
</tr>
<tr>
<td></td>
<td>• Unauthorized access to the system</td>
</tr>
<tr>
<td></td>
<td>• Unauthorized use of services or resources</td>
</tr>
<tr>
<td></td>
<td>• Disruption of availability</td>
</tr>
<tr>
<td></td>
<td>• Attacks motivated and facilitated by information disclosure or visible security weaknesses</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Req 5</th>
<th>All ports must have a fixed port type (e.g., E, F port) defined according to their specific use.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Port types shall be defined with a fixed type for all ports. Hosts with an N port are connected to an F port. An FC switch is connected via the E port type and can exchange data with the other switch about configuration parameters. When only hosts are connected to an FC switch, the F port type shall be configure for all ports on the switch.</td>
</tr>
<tr>
<td></td>
<td><strong>Motivation:</strong> A port on an FC switch has the ability to change the port type dynamically to suit the connected component. The measure described here prevents an attacker from forcing a port type change and reading or manipulating the FC switch configuration.</td>
</tr>
<tr>
<td></td>
<td>For this requirement the following threats are relevant:</td>
</tr>
<tr>
<td></td>
<td>• Unauthorized access to the system</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Req 6</th>
<th>Hardware-based enforcement of zoning must be used.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Zoning can be executed in two ways (soft, hard zoning). Each component finds other components in the fabric by querying the name server which is located on the FC switch and resolves WWNs to FC IDs. When a component logs in to the name server, the latter returns a list of components that can communicate with each other. In soft zoning (information-based zoning), the restrictions are only applied to interaction between the name server and the component. If a component knows the fabric address of a component outside its own zone, it can communicate with the other component. Hard zoning (hardware-based enforcement of zoning) is enforced by the hardware in each frame. When a frame arrives at a switch, the combination of source and target address is verified. If the combination is valid, the frame is forwarded.</td>
</tr>
<tr>
<td></td>
<td><strong>Motivation:</strong> The use of hardware-based enforcement of zoning prevents components communicating across zones.</td>
</tr>
</tbody>
</table>
For this requirement the following threats are relevant:
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Disruption of availability

Req 7  The configuration of hardware-based enforcement zoning must be based on port WWNs.

Each host bus adapter has a node WWN, which identifies the host bus adapter (HBA), and one or more port WWNs, which identify the HBA port. Furthermore, each host bus adapter has a fabric address assigned to it, consisting of do-main/area and port ID sections. In the case of N-Port Identifier Virtualization (NPIV), virtual WWNs (vWWNs) are added for each virtual machine. Zone configuration can take place on the basis of domains, port identification (D, P)/ node or port WWNs.

Motivation: D,P identification is not globally unique and refers to individual switch ports. Each component that is connected to a port can communicate within the zone configured for that switch port. Identification based on port WWNs is globally unique. Furthermore, zone configuration based on port WWNs guarantees administration flexibility and the use of extended functions such as Fibre Channel Routing and Fibre Channel Write Acceleration.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Disruption of availability

Req 8  Port security must be enabled for all frontend FC switch ports.

In most cases, port security is achieved with access control lists (ACL). These define which WWN may occur at which port. If a WWN occurs at a port where it is not permitted, the FC switch deactivates the port and logs a warning or error message to this effect. Port security shall be configured for all connected SAN components (host, switch, network storage). The connection from the network based storage, which provides the logical units (LU), to the FC switch is called as backend and the connection from the Hosts to the FC switch is called as frontend. In case of storage virtualization, the connection from the network based storage to the storage virtualization is called as backend.

Motivation: Activating port security makes WWN spoofing more difficult.

For this requirement the following threats are relevant:
- Unauthorized access to the system

Req 9  LUNs on network storage must be protected with LUN masking based on port WWNs.

Logical Unit Numbers (LUN) associate logical units (logical hard disks) with a host or a port WWN. Only this host can communicate with the LUN and read/write data on the LUN. Other hosts do not have access to the LUN and cannot see it. Some manufacturers offer key-based authentication between network storage and the host. Authentication based on certificates or keys shall be preferred.

Motivation: LUNs are protected by LUN masking and only authorized HBAs have access to a LUN. Therefore, an attacker would have to implement WWN spoofing. However, this does not produce the desired result, namely to read or write the LUN, if the measures described in this document are implemented (e.g., enabling port security).

For this requirement the following threats are relevant:
- Unauthorized access to the system
3.55 Network Based Storage Systems

Req 10  Unique static domain IDs must be used.

It is possible for an FC switch to assume a new ID after booting, in particular in cases when an FC switch is added to a fabric or after a power failure. To prevent this, a static domain ID must be assigned. Manufacturer Brocade refers to static domain IDs as 'insistent domain IDs' (IDID) while Cisco calls them 'static domain ID' or 'persistent FC ID.' An IDID or 'persistent FC ID' continues to exist after a reboot.

Motivation: The use of IDIDs or 'persistent FC IDs' simplifies subsequent troubleshooting. Nor can an attacker's FC switch join the fabric since the IDID is not registered in the fabric.

For this requirement the following threats are relevant:
• Disruption of availability

IP based network storage NAS/ iSCSI

Req 11  Mutual target and initiator authentication must be enabled.

As a rule, mutual authentication of initiators and targets is implemented via CHAP. Other features such as the iSCSI Qualified Name (IQN) and IP address shall also be used in authentication. The IQN is the name of the initiator. If the product supports other cryptographic procedures such as Kerberos, these shall be preferred.

Motivation: An attacker has to assume control of the initiator and compromise the user name and password through a brute force attack; cryptographic procedures shall therefore be preferred.

For this requirement the following threats are relevant:
• Unauthorized access to the system

Req 12  NFS version 4 or later must be used.

NFS version 4 defines Kerberos support as a compulsory element of the protocol. It also has improved user name mapping, only uses a TCP port and is able to establish authentication and encrypted communication (transport en-cryption) between client and server with Kerberos version 5. The use of NFS version 4 implies that NFS versions 1, 2 and 3 are deactivated.

Motivation: Firewall activation is confined to a single port, thus reducing attack surface. Deactivation of NFS versions 1,2 and 3 eliminates some points of attack, since version 4 no longer requires a port mapper. The use of Kerberos is another factor that makes it possible to achieve a higher level of security.

For this requirement the following threats are relevant:
• Unauthorized access to the system

Req 13  Client root access (UID = 0) must be mapped to a non-privileged user.

The NFS-Server trust the UID’s that are sent from the client to him. The authenticated user can manipulate his identity in such a way that the server thinks the authenticated user is root. Thus, the NFS-server gives the authenticated user root permissions. The unprivileged users must not have access to any share of the NFS-server, which means that an-onymous access without authentication MUST be disabled.

Motivation: Prevent an attacker to gain elevated permissions of the NFS server.

For this requirement the following threats are relevant:
• Unauthorized access to the system
Req 14 Anonymous access without authentication must be disabled.

Non-authenticated users can use the hidden share ($IPC) to collect information relating to user names, group names, group members, administrator groups and open releases.

*Motivation: This makes it difficult for an attacker to collect information for further attacks.*

For this requirement the following threats are relevant:
- Unauthorized access to the system

System hardening

Req 15 Unused services and protocols must be deactivated.

After installation of systems and software products typically local or remote reachable services and protocols are active, which are not necessarily needed for operation and functionality of the system. Such services and protocols must be completely disabled on the system. Additionally it is important that a deactivation survives a system reboot.

This kind of system hardening must be done before the system is reachable from the network. Otherwise an attacker has the possibility to attack and maybe compromise the unsecured system.

*Motivation: Services and protocols that are not required for system operation increase the potential attack surface and thus the risk of the system being compromised. This risk is further increased by the fact that a security inspection and an appropriate optimisation of the configuration for unused services and protocols will not be done.*

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized use of services or resources
- Disruption of availability
- Attacks motivated and facilitated by information disclosure or visible security weaknesses

Req 16 Unused software must not be installed or must be uninstalled.

During installation of a system often software components will be installed or parts of software will be activated which are not needed for the operation or functionality of the system. This includes also parts of a software, which will be in-stalled as examples but typically not be used (e.g. default web pages, example databases, test data). Such components should not be installed or must be deleted after installation.

*Motivation: Vulnerabilities in software of a system offer an attack window for attackers to infiltrate the system. Uninstalling components that are not required can therefore reduce the possibility of a successful compromise of the system.*

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Disruption of availability
Req 17 Unused functions of the operated software and hardware must be deactivated.

During installation of software and hardware often functions will be activated that are not necessarily needed for operation or function of the system. Functions of software are currently inherent part which could not be deleted or deinstalled individually. Such functions must be deactivated in the configuration of the system permanently.

Beside the functions of the software also hardware functions are active which are not necessary for a system. Functions like unused interfaces must permanently deactivated. Permanent means that they must not be reactivated again after system reboot.

Motivation: The hardware or software of a system often contains functions which are not used and so will be a risk for system security. Such functions give an attacker the possibility to manipulate the system. Furthermore it is possible to get unauthorized access other areas or data of the system. An example is a debugging function in software which can be used for troubleshooting but must not be activated during normal operation. Or a hardware interface that will not be used and so is unsecured allows possibly unauthorized access to the system.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Disruption of availability
System update

Req 18 Software and hardware components that are no longer supported by vendor, producer or developer must not be used.

Components that have reached end-of-life or end-of-support must not be used. Excluded are components that have a special support contract. This contract must guarantee the correction of vulnerabilities over components life-time.

Motivation: Hardware and software components that have reached end of life or end of support represent a risk for a system. This means that a vendor does not supply remedial updates or patches for a component should errors or vulnerabilities occur. This means that vulnerabilities cannot be fixed when they occur and could be exploited to compromise the system or to impair its availability.

For this requirement the following threats are relevant:
• Unauthorized access to the system
• Unauthorized access or tapping of data
• Unauthorized modification of data
• Disruption of availability

Req 19 Known vulnerabilities in software and hardware of the system must be fixed or protected.

Prior to installation of a software or hardware component, users must check whether any vulnerability has been discovered and published for the version they are installing. Any component that proves to have a vulnerability must not be installed or used. Excepted from this rule are components for which the vendor has already provided a measure to remedy the vulnerability, e.g. a patch, update or workaround. In this case, the additional measure must be implemented on the system. Furthermore it is an ongoing process during the complete life cycle of the system to fix upcoming vulnerabilities promptly.

Motivation: Publication of vulnerabilities increases the risk of successful exploitation by an attacker, especially since the published information usually includes details on how to exploit the vulnerability and tools that make exploitation possible. One example of this is when Web servers are compromised due to not patched vulnerabilities.

For this requirement the following threats are relevant:
• Unauthorized access to the system
• Unauthorized access or tapping of data
• Unauthorized modification of data
• Disruption of availability

Protecting data and information

Req 20 Data with need of protection must be secured against unauthorized viewing and manipulation during transmission and storage.

Adequate security measures for transmission and storage must be implemented of data with a need for protection that are classified as internal, confidential or strictly confidential. The chosen measure depends on the classification for the data and other factors such as the type of network used during transmission, the storage location for data, etc. Furthermore must be guaranteed that confidential and strictly confidential data will not be unprotected during temporary storage (e.g. in web cache, temporary folders).

All authentication data such as user names, passwords, PINs, etc. must be protected against unauthorized viewing and manipulation. This applies equally to permanent storage and
transmission. Typical measures taken to protect authentication data or combinations of these are:

For storage:
- Client systems: encryption or obfuscation of authentication data, no persistent storage, limitation of access rights.
- Server systems: hashing of authentication data with PBKDF like scrypt or bcrypt or when this is not possible hashes with salt.

For transmission:
- Usage of cryptographically protected network protocols.
- Challenge response method (Disadvantage of this solution is that passwords needed in clear-text on server. If this solution is feasible depends on the individual threat scenario)

Files of a system that are needed for the functionality must also be protected against manipulation. This is necessary because system’s integrity can be damaged when the system access this kind of files. An example is the use of check-sum or cryptographic methods to validate if e.g. firmware images, patches, drivers or kernel modules are free of manipulations.

For transmission of data with a need of protection network protocols that are insecure due to insufficient security measures shall not be used. Examples are: SSLv3, SSHv1, FTP, Telnet, SNMPv1 and 2c. In case of these protocols a newer version without vulnerabilities or a secure alternative must be used.

Motivation: If data with a need of protection will not be secured an attacker could record or manipulate the data during transmission over a network. An example is the recording of user names and passwords during system administration with the telnet clear-text protocol. Storing data on a system without adequate protection may mean that unauthorized users can copy or modify it. One example is when passwords can be read out when they stored in an inadequate se-cured way (e.g. usage of unsecure hashing algorithms like MD5 or SHA-1) or even without encryption on a system or manipulation of firmware to affected the system integrity.

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data
- Unauthorized modification of data

Req 21 Information with need of protection must not be contained in files, outputs or messages that are accessible by unauthorized users.

Information with need of protection must not be accessible in files, outputs or messages of the system by unauthorized users. This includes information relating to the operating system, used middleware or applications such as vendor, product name, product identifier, installed software versions, installed service packs, patches, hot fixes and serial numbers. Examples for system messages which must be free of sensitive data are:
- Comments in downloadable files
- Error and system messages
- Stack traces
- Network protocols
- Login windows and dialogs

Furthermore, details of implementation and information relating, e.g., to backend software/systems, function calls, SQL instructions or structure of database, must not be contained in error messages. Excluded from this are displays and outputs that can be viewed and retrieved by authorized users who are logged in. In addition, an internal transfer of system internal information for error
analysis is allowed in an adequate dimension. In this case the continuative regulations or guidelines (e.g. of data privacy) must be noticed.

Motivation: The information named above can be used by an attacker to prepare specific attacks on a system. In this way an attacker could, for example, use the precise software version to identify vulnerabilities in the product and, in a second step, exploit them.

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data
- Attacks motivated and facilitated by information disclosure or visible security weaknesses

Protecting availability and integrity

<table>
<thead>
<tr>
<th>Req 22</th>
<th>The system must be robust against overload situations.</th>
</tr>
</thead>
</table>

A system must provide security measures to deal with overload situations. In particular, partial or complete impairment of system availability must be avoided. Potential protective measures include:
- Restricting of available RAM per application
- Restricting of maximum sessions for a Web application
- Defining the maximum size of a dataset
- Restricting CPU resources per process
- Prioritizing processes
- Limiting of amount or size of transactions of an user or from an IP address in a specific time range

Motivation: An attacker can effect systems availability through targeted exploitation of vulnerabilities in resources ad-ministration. An attacker can, for example, send bulk queries to a Web server and, by doing so, compromise its availability if the maximum number of permissible Web sessions is not restricted.

For this requirement the following threats are relevant:
- Disruption of availability

<table>
<thead>
<tr>
<th>Req 23</th>
<th>If an overload situation cannot be prevented, the system must act in a predictable way.</th>
</tr>
</thead>
</table>

A system must be built in this way that it can react on a overload situation in a controlled way. However it is possible that a situation happens where the security measures are not longer sufficient.

In such case it must be ensured that the system cannot reach an undefined and thus potentially insecure state. In an extreme case this means that a controlled system shutdown is preferable to uncontrolled failure of the security func-tions and thus loss of system protection.

Motivation: With denial-of-service attacks can an attacker try to overload a system to effect its availability or integrity. Unpredictable acting of the system is then a risk for functionality and data and possibly also for other systems.

For this requirement the following threats are relevant:
- Disruption of availability
Req 24  The system must be robust against unexpected input.

During transmission of data to a system it is necessary to validate this before processing. This includes all data which are send to the system. Examples for this are user input, values in arrays and content in protocols. The following typical implementation mistakes must not be done:

- No validation on the lengths of transferred data
- Incorrect assumptions about data formats
- No validation that received data complies with the specification
- Insufficient handling of protocol errors in received data
- Insufficient restriction on recursion when parsing complex data formats
- White listing or escaping for inputs outside the values margin

Motivation: An attacker can try to put a system in an unsecure state through targeted manipulation of transmitted data. The object of such an attack is to compromise the usability, availability or integrity of individual services or of the entire system. For instance a unclean memory handling can lead to a buffer overflow that allows an attacker to execute arbitrary code on the effected system.

For this requirement the following threats are relevant:

- Unauthorized access to the system
- Unauthorized use of services or resources
- Disruption of availability

Authentication and authorization

Req 25  Accounts must be used that allow unambiguous identification of the user.

Users must be identified unambiguously by the system. This can typically be reached by using a unique account per user. So named group accounts, i.e. the use of one account for several persons, must not be used. On exception of this requirement are so named machine accounts. These will be used for authentication and authorization from sys-tem to each other or for applications on a system and can’t be assigned to a single person. Such accounts must be assigned on a per system or per application basis. In this connection, it has to be guaranteed that this account can’t be misused. Possibilities to protect these accounts are:

- Configuring of a Password that fulfils the security requirements and is known by less than possible circle of ad-ministrators.
- Configuring the account that only a local use is possible and a interactive login isn’t possible.
- Use of a technique for authentication of the specific account with public and private key or certificates.
- Limiting the access over the network for legitimised systems.

Additional solution must be checked on their usability per individual case.

Motivation: Unambiguous user identification is a prerequisite for assigning a user the rights that he requires to perform his tasks on the system. This is the only way to adequately control access to system data and services and to prevent misuse. Furthermore, it makes it possible to log activities and actions on a system and to assign them to individual users.
3.55 Network Based Storage Systems

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized use of services or resources
- Denial of executed activities

---

**Req 26** Accounts must be protected against unauthorized use by at least one authentication attribute.

The various user and machine accounts on a system must be protected from misuse. To this end, an authentication attribute is typically used, which, when combined with the user name, enables unambiguous authentication and identification of the authorized user. Authentication attributes include:

- Cryptographic keys
- Token
- Passwords
- PINs

This means that authentication based on a parameter that can be spoofed (e.g. phone numbers, public IP addresses or VPN membership) is not permitted. Exceptions are attributes that cannot be faked or spoofed by an attacker. Several of the above options can be combined (dual-factor authentication) to achieve a higher level of security. Whether or not this is suitable and necessary depends on the protection needs of the individual system and its data and must be evaluated for individual cases.

Motivation: Accounts that are not protected with a secret authentication attribute can be used by an attacker to gain unauthorized access to a system and the data and applications stored on it.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized use of services or resources
- Denial of executed activities

---

**Req 27** Accounts with extensive rights must be protected with two authentication attributes.

Accounts, for example used for administration, maintenance and troubleshooting, have extensive rights. Therefore a single protection (e.g. a password), as for normal user accounts with less rights, is not suitable. To get a higher protection level it is necessary to use more than one authentication attribute. For this a combination of an attribute that the user knows and an attribute that the user owns will be used often. This kind of authentication will be named as 2-factor authentication. Examples for 2-factor authentication are:

- Smartcard with PIN
- Private key with Passphrase
- Secure-ID Token with Password

Motivation: Accounts with extensive rights as used for system administration have a higher risk for system’s security. An attacker can get extensive rights by compromising such an account to get access to wide parts of the system and stored data.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized use of services or resources
- Denial of executed activities
3.55 Network Based Storage Systems

Req 28  
Predefined accounts must be deleted or disabled.

Many systems have default accounts (e.g. guest, ctxsys), some of which are preconfigured with or without known pass-words. These standard users must be deleted or disabled. Should this measure not be possible the accounts must be locked for remote login. In any case disabled or locked accounts must configured with a complex password (12 character and more, use of upper/lower case, numbers and special characters). This is necessary to prevent unauthorized use of such a account in case of misconfiguration.

Exceptions to this requirement to delete or disable accounts are accounts that are used only internal on the system involved and that are required for one or more applications on the system to function. Also for this accounts remote access or local login must be forbidden to prevent a abusive use by users of the system.

**Motivation:** Standard users are typically generally known and can be used by an attacker for targeted brute force and dictionary attacks. Standard user accounts represent a special risk if they do not use a password or only use a standard password that is generally known. Such standard user accounts can easily be exploited by an attacker in order to gain access to the system involved without being authorized to do so.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized use of services or resources
- Denial of executed activities

Req 29  
Predefined authentication attributes must be deleted or disabled.

Normally, authentication attributes such as password or cryptographic keys will be preconfigured from producer, vendor or developer of a system. Such authentication attributes must be changed an own attribute not known by other parties.

**Motivation:** Authentication attributes like password or cryptographic keys preconfigured from third parties are not trustable. Such authentication attributes can be used to compromise systems or their data.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized use of services or resources
- Denial of executed activities

Req 30  
The authorizations for accounts and applications must be reduced to the minimum required for the tasks they have to perform.

Authorizations to a system must be restricted to a level in which a user can only access data and use functions that he needs in the course of his work. Suitable authorizations must also be assigned for access to files that are components of the operating system or of applications or that are generated by the same (e.g. configuration and logging files).

Alongside access to data, execution of applications and components must also take place with rights that are as low as possible. Applications should not be executed with administrator or system rights.

**Motivation:** If the rights granted to a user on a system are too broad, it could be possible for him to access data and applications which he is not permitted to view or use. This would give him the opportunity to disclose or modify confidential data and to manipulate system files.
Applications with rights that are too broad can be used by a user to extend his own authorizations and thus to gain access to files and system components to which he would not have had access with his authorizations under normal circumstances.

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources

Req 31 The system must be connected to a central system for user administration.

For provisioning of these authorization information must the system provide a central inter-face (e.g. LDAPs for authorization, Kerberos for authentication, locking information for certificates) or decentralized mechanisms (e.g. public-key authentication). A central solution for identity management must be preferred.

In areas where a central identity management system is not available a central system such as LDAP, TACACS+ or Ra-dius server for the administration of accounts and their authentication and authorization must be used.

Motivation: Central administration of identity of accounts and their rights means that they only have to be maintained once instead of separately on each system. From the aspect of security, the advantage is that an account and its rights only known on a single central side. This information can be transmitted from a central side to systems (provisioning), central administrated (reconciliation) and central deleted (deprovisioning). This reduces the risk of accounts being for-gotten during changing or deletion since they are configured on multiple systems. This could give a user wrong sys-tem rights or continued access to a system.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources

Protecting sessions

Req 32 Sessions must be protected against high jacking.

A function must be implemented for all user sessions to prevent other users assuming or pursuing control of the ses-sions of an authorized user. Such protection can be achieved by implementing the following measures or a combina-tion of these measures:
- Using an appropriate cryptographic algorithm
- On network level: Use of TCP protocol (with sequence number) and filter lists.
- On transport level: SSL/TLS
- On application level: Negotiation of a random, secret value between sender and recipient (e.g. session ID, se-quence number, time stamp)

Motivation: An attacker who gains knowledge about a session which is not protected against assumption of control can continue to participate in such a session and thus gain unauthorized access to the system involved. One example is an attacker who records the session cookie for a Web application during non-encrypted communication. The attack-er can then view and take control of the session of the original user.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized access or tapping of data
• Unauthorized modification of data
• Unauthorized use of services or resources
• Denial of executed activities

---

**Req 33** It must be possible that users can logoff their session.

The system must have a function that allows a signed in user to logout at any time.

*Motivation: A user of a system must have the possibility to protect a session and therefore its data against unauthorized access. Therefore a logout function must be available that can be used to end a session. This prevents that the current session will stay open and can be reactivated and used by an unauthorized person.

For this requirement the following threats are relevant:

- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources
- Denial of executed activities

---

**Req 34** Session must be terminated after an adjusted period of inactivity for the purpose of use.

It is necessary that a session will be terminated automatically after a specified time of inactivity. For this reason a time-out must be set for sessions. The selected period depends on use and if applicable the physical environment. This means for example that a time-out for an application in an unsecure environment must be shorter (less minutes) than the time out for an application used of operational staff for monitoring tasks used in a protected area (60 minutes and longer).

*Motivation: In case of an open or unused session exists the possibility that an unauthorized user can high-jack and use this session to get access to the effected system and its data. Furthermore open session will allocate system resources.

For this requirement the following threats are relevant:

- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources
- Denial of executed activities
authentication parameter password

Req 35 If a password is used as an authentication attribute, it must have at least 8 characters and contain
three of the following categories: upper cases, lower case, numbers and special characters.

A system may only accept passwords that comply with the following complexity:

- Minimum length of 8 characters.
- Comprising at least three of the following categories: upper/lower case letters, numbers and special characters

When a password is assigned, the system must ensure that the password meets these requirements. If a central system will be used for user authentication this function can be forwarded or delegated to this system.

Motivation: Passwords with the above complexity offer high robustness against attacks coupled with acceptable user friendliness. Passwords with this level of complexity have proved their efficiency in practice. Trivial passwords that are too short are susceptible to brute force and dictionary attacks and are therefore easy for attackers to determine. Once a password has been ascertained it can be used by an attacker for unauthorized access to the system and the data on it.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized use of services or resources
- Denial of executed activities

Req 36 If a password is used as an authentication attribute, users must be enable to independently change the password anytime.

The system must offer a function that enables a user to change his password at any time.

When an external centralized system for user authentication will be used it is possible to redirect or implement this function on this system.

Motivation: The fact that a user can change his authentication attribute himself at any time enables him to change it promptly if he suspects that it could have been accessed by a third party.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized use of services or resources
- Denial of executed activities

Req 37 If a password is used as an authentication attribute, a protection against brute force and dictionary attacks that hinder password guessing must be implemented.

Brute force and dictionary attacks aim to use automated guessing to ascertain passwords for user and machine accounts. Various measures or a combination of this measures can be taken to prevent this.

The most commonly used protection measures are:

- Increasing the delay (e.g. doubling wait times at each attempt) for each renewed password input following an incorrect entry (“tar pit”).
• Blocking an account following a specified number of incorrect attempts (typically 5). However has to take in account that this solution needs a process for unlocking and an attacker can force this to deactivate accounts and make them unusable.
• Using CAPTCHA to prevent automated attempts (often used for Web applications).

In order to achieve higher security, it is often meaningful to combine two or more of the measures named here. This must be evaluated in individual cases and implemented accordingly.

Motivation: Implementation of one or more of the above measures can prevent successful execution of brute force and dictionary attacks to ascertain passwords.

For this requirement the following threats are relevant:
• Unauthorized access to the system
• Unauthorized use of services or resources
• Denial of executed activities

Req 38 If a password is used as an authentication attribute, they must be hidden when displayed on screen.

When a user enters a password or the password is otherwise displayed on screen, it must be made unreadable. Typically, the individual characters of the password are replaced by a character such as “*”. Under certain circumstances it may be permissible for an individual character to be displayed briefly during input. Such a function is used, for example, on Smartphone’s to make input easier. However, the entire password is never output to the display in plaintext.

Motivation: To prevent another person reading a password by chance or intentionally on screen during input, the pass-word must be rendered unreadable during input or display.

For this requirement the following threats are relevant:
• Unauthorized access to the system
• Unauthorized use of services or resources
• Denial of executed activities

Logging

Req 39 Security relevant events must be logged with a precise timestamp and a unique system reference.

Systems must log the occurrence of security-relevant incidents. So that these events can be evaluated and classified, they must be logged together with a unique system reference (e.g., host name, IP or MAC address) and the exact time the incident occurred. Furthermore, legal and data privacy regulations (e.g. time of storage of logging data) must be proved and followed. The following table lists events that are relevant to security and the corresponding data that typically has to be logged by a system. Exceptions are systems for which no or only restricted logging applies. Examples of such systems are customer devices such as Smartphone’s or IADs/Homegateways (e.g. Speedport).

Logging must be done considering the currently valid legal, wage and company regulations. This regulations state among others that logging of events can be done only earmarked. Logging of events for doing a work control of em-ployees is not allowed.
Typical event that reasonable should be logged in many cases are:

<table>
<thead>
<tr>
<th>Event</th>
<th>Event data to be logged</th>
</tr>
</thead>
<tbody>
<tr>
<td>Incorrect login attempts</td>
<td>• Account,</td>
</tr>
<tr>
<td></td>
<td>• No. of failed attempts,</td>
</tr>
<tr>
<td></td>
<td>• Source (IP address) of remote access</td>
</tr>
<tr>
<td>System access with accounts with administrator rights</td>
<td>• Account,</td>
</tr>
<tr>
<td></td>
<td>• Access timestamp,</td>
</tr>
<tr>
<td></td>
<td>• Length of session,</td>
</tr>
<tr>
<td></td>
<td>• Source (IP address) of remote access</td>
</tr>
<tr>
<td>Account administration</td>
<td>• Administrator account,</td>
</tr>
<tr>
<td></td>
<td>• Administered account,</td>
</tr>
<tr>
<td></td>
<td>• Activity performed (configure, delete, enable and disable)</td>
</tr>
<tr>
<td>Change of group membership for accounts</td>
<td>• Administrator account,</td>
</tr>
<tr>
<td></td>
<td>• Administered account,</td>
</tr>
<tr>
<td></td>
<td>• Activity performed (group added or removed)</td>
</tr>
<tr>
<td>Critical rise in system values such as disk space, CPU load over a longer period</td>
<td>• Value exceeded,</td>
</tr>
<tr>
<td></td>
<td>• Value reached</td>
</tr>
<tr>
<td></td>
<td>(Here suitable threshold values must be defined depending on the individual system.)</td>
</tr>
</tbody>
</table>

Logging of additional security-relevant events may be meaningful. This must be verified in individual cases and implemented accordingly where required.

Motivation: Logging security-relevant events is a basic requirement for detecting ongoing attacks as well as attacks that have already occurred. This is the only way in which suitable measures can be taken to maintain or restore system security. Furthermore, the logging data is used as evidence so that legal steps can be taken against attackers.

For this requirement the following threats are relevant:
• Denial of executed activities
• Unnoticeable feasible attacks

Req 40 Security relevant logging data must be send to an external system direct after their creation.

Logging data must be forwarded to an external system in appropriate logging files as well as being stored locally. Standard protocols like Syslog, SNMPv3 must be preferred.

Motivation: If logging data is only stored locally it can be manipulated by an attacker who succeeds in compromising the system in order to conceal his attack and any manipulation he has performed on the system. This is the reason why the forwarding must be done immediately after the event occurred.

For this requirement the following threats are relevant:
• Denial of executed activities
• Unnoticeable feasible attacks
3.57 Architecture of the access and transport network

Introduction

This security requirement has been prepared based on the provisions of the Group Policy on IT/NT Security.

The security requirement is used as a basis for an approval in the PSA process, among other things. It also serves as an implementation standard for provisions of the Group Policy on IT/NT Security in units which do not participate in the PSA process. These requirements shall be taken into account from the very beginning, including during the planning and decision-making processes. When implementing these security requirements, the precedence of national, international and supranational law shall be observed.

General

Req 1 The operational network and its systems must be entirely separate from the test and development systems.

The operational network and its systems must be entirely separate from the test and development systems. Physical separation would be preferable. Logical separation is only permitted if the logical separation of both system types can not be avoided. Furthermore, it must be ensured that production security will not be affected by activities on the test and development systems.

Any communication required between the operational systems and the test/development systems must be via a secure connection. This means that communication between the systems must take place via a separate system such as a firewall or a router with access control list, which is necessary to ensure implementation of the most restrictive rules possible for communication.

Motivation: A sufficiently secure system status cannot be assumed in the case of test and development systems because these systems are typically subject to permanent changes. If operational systems or networks are used for test and development activities, and if communication between the various system types is not secure, the operational environment may be accessed by an unauthorized party from the test/development environment or the stability and availability of this environment may be impaired. Another benefit of a complete separation of the operational environment from the test and development systems is that the latter can then be permanently used for acceptance tests. This ensures, for example, that security updates can be tested and made available more quickly even during critical phases (or "frozen zones").

For this requirement the following threats are relevant:

• Unauthorized access to the system
• Unauthorized access or tapping of data
• Unauthorized modification of data
• Disruption of availability

Req 2 The IPv4 and IPv6 infrastructure address space of the transport network must not be accessible from customer lines and connected networks.

The infrastructure address space of the transport network is used for accessibility within the network. These addresses are, for example, used to transmit the control plane traffic. These addresses must not be accessible, or only to a very limited extent, from the outside, i.e., from
3.57 Architecture of the access and transport network

customer lines or connected networks such as the Internet. Accessibility must therefore be restricted. There are various possible approaches to achieve this. One of them is, for example, to use access control lists on routers to protect the infrastructure address space (iACL).

**Motivation:** If the infrastructure address space of the transport network is accessible without restrictions, this can be used by attackers to carry out denial-of-service attacks against individual systems or the entire network. Furthermore, services (routing and other control plane protocols) in the network can also be accessed as a result. This makes it possible for third parties to directly manipulate the infrastructure.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Disruption of availability

## System management

### Req 3
A reliable transfer of control and management traffic must be ensured.

Data packets used to exchange control plane information and to manage different systems in a network must be transferred with the highest priority. To do this, relevant packets must, for example, be transported in a sufficiently prioritized quality-of-service class.

**Motivation:** If control, management and user traffic in a network was handled in the same way, this could mean that in extreme situations the necessary control information could no longer be exchanged and access to management services would no longer be possible. It would also mean that the affected systems could no longer be accessed in order to perform maintenance and troubleshooting tasks. Such a situation could lead to the network becoming unstable or even availability being impaired. An attacker could exploit this in order to cause network outage by means of denial-of-service attacks.

For this requirement the following threats are relevant:
- Disruption of availability

### Req 4
If data needs to be exchanged between management plane and planes for user and control traffic, this communication must be protected.

Management traffic must generally be separated from control and user plane traffic. It may, however, be necessary to exchange data between the different planes. If this is required, the exchange may only take place via secure connections. The bidirectional restriction of the permissible communication relationships to necessary sender and recipient is required here for protection purposes. Any communication relationships that are not necessary must be prevented.

**Motivation:** If unprotected communication between different network planes is possible, an attacker can exploit this situation to attack system management services. If such an attack is successful, the result is generally that the relevant system is fully compromised.

For this requirement the following threats are relevant:
- Unauthorized access to the system

### Req 5
After a successful authentication on a network device it must not possible to access other network devices or systems without anew authentication.

Following successful authentication to a network device, it shall not be possible to gain access to the management of another network device or system without renewed authentication. Renewed authentication shall therefore be enforced for login to another network device.

**Motivation:** This measure prevents an attacker using a compromised network device to gain access to other network devices or systems without renewed authentication.
For this requirement the following threats are relevant:
• Unauthorized access to the system

Control plane protocols

Req 6 Mutual authentication must be used for internal signalling protocols through which path information is exchanged.

For signalling protocols, such as routing protocols (IBGP, OSPF, IS-IS, etc.) and other protocols, via which path information is distributed (LDP, RSVP, etc.), mutual authentication of the communication partners must be used. The data used for authentication must be protected against viewing and tampering by means of a cryptographic procedure. As secure a procedure as possible must be selected here.

Motivation: An attacker can interfere with signalling protocols in order to tamper with path decisions in the network, to divert traffic or to disrupt communication. Signalling protocols and the services open for them on a network device also bring the risk of denial-of-service attacks. These attacks can also be prevented by the use of authentication of the involved communication parties for the relevant signalling protocols.

For this requirement the following threats are relevant:
• Unauthorized access to the system
• Unauthorized access or tapping of data
• Unauthorized modification of data
• Disruption of availability

Req 7 Routing protocols must be disabled on interfaces where routing updates need not be sent or received.

Routing protocols, and thus the sending of routing updates, may only be enabled on interfaces to other network elements where there is a neighborhood relationship in place within the framework of the relevant routing protocol. On all other interfaces, the routing protocol must be disabled.

Motivation: An attacker could glean information from recorded routing updates that contains details about the network architecture. This information could be used to plan and implement further attacks.

For this requirement the following threats are relevant:
• Unauthorized access or tapping of data
• Disruption of availability
• Attacks motivated and facilitated by information disclosure or visible security weaknesses

Req 8 TTL security must be used for EBGP.

EBGP packets (External Border Gateway Protocol) are sent with a TTL value of 1 in order to ensure that they are not transferred beyond a router. The TTL security function means that EBGP packets are sent with the highest possible TTL value. The TTL value is calculated using the following formula: “255 – [maximum number of necessary hops].” On the BGP neighbor, the function must be enabled with the same TTL value. This is the only way to ensure that BGP packets with a value that is smaller than the predefined one are discarded. The TTL security function is not supported by all manufacturers.

Motivation: An unprotected BGP service can be taken over by an attacker. It is also possible for denial-of-service attacks to be carried out on accessible BGP ports and thus on the relevant router. By implementing this measure, attacks using fake BGP packets from external networks can largely be prevented.

For this requirement the following threats are relevant:
3.57 Architecture of the access and transport network

- Unauthorized modification of data
- Disruption of availability

Req 9 If prefixes are exchanged with a customer via a routing protocol, the maximum number of prefixes must be limited.

When a router is informed about new routes via routing updates, it enters them in its routing table. The size of the routing table has a direct influence on the memory and CPU usage of the router. This is why the maximum number of learnable routes should be limited depending on the hardware used.

*Motivation:* A routing table that is too large can lead to critical utilization levels for system resources such as RAM and CPU. An attacker can exploit this specifically to compromise the availability of a network element or even entire sections of the network by means of faked routes.

For this requirement the following threats are relevant:
- Disruption of availability

**Access network**

Req 10 A standardized authentication procedure must be used in order to ensure the authenticity of a customer device.

It may be necessary to verify the authenticity of devices connected to the access network via authentication. By this means it can be ensured that only known and thus trustworthy systems can be connected to the network. This is mainly useful for systems and network elements that are operated in environments that cannot be monitored, or for customer systems. Possible solutions include IKE/IPSec as well as IEEE 802.1X (with EAP) for port-based network access control (PNAC).

*Motivation:* Network elements that are operated in non-secure environments (e.g., multifunctional street cabinets, CPE) are subject to an increased risk of manipulation or unauthorized exchange of devices. Appropriate device authentication means that only trustworthy systems can be connected to the network.

For this requirement the following threats are relevant:
- Unauthorized access to the system

Req 11 If addresses are assigned dynamically, they must be protected against manipulation.

If a dynamic assignment of IPv4 or IPv6 addresses such as DHCP is used, relevant systems must be protected against the following scenarios:
- Manipulation of address assignment using fake response packets.
- Using up the available address pool.

In the case of DHCP, protection against fake response packets and/or against a fake DHCP server must be set up via the DHCP snooping function. Protection against the manipulative allocation of the available address space may involve, for example, limiting the maximum number of addresses that can be accessed per line, and/or the MAC address. However, this requires the implementation of a further measure that stops the fake MAC addresses on a line.

*Motivation:* The automated assignment of addresses can be manipulated by an attacker. One possible way of doing this is to redirect traffic using fake response packets in order to record data in need of protection. Another attack scenario would be using up the available address pool.
3.57 Architecture of the access and transport network pool, whereby no more addresses are available for other systems and the functionality of the systems is impaired.

For this requirement the following threats are relevant:
• Unauthorized access or tapping of data
• Disruption of availability

---

Req 12 If control plane protocols are exchanged with customers a rate limit must be set.

Control plane protocols are an important control mechanism in the network and are also exchanged with customers, e.g., in order to exchange routing and authentication data. This results in a risk as insufficient protection can compromise the transport platform. For this reason, a rate limit must be configured for control plane protocols that are exchanged with customers. Dimensioning depends on the available performance of the network elements used.

Motivation: The resources (e.g., CPU, memory) of a system are limited. This can be deliberately exploited or, via targeted attacks, also impair system availability. In this case, mass-generated control plane protocol packets are used to generate a high load on the system affected.

For this requirement the following threats are relevant:
• Disruption of availability

---

Req 13 With Ethernet-based customer lines, the maximum number of MAC addresses that can be learned on the network side must be limited.

If it is necessary for MAC addresses of customer devices to be learned on the network side, the maximum number of addresses per customer must be limited. Lines on which MAC addresses must be learned include, for example, Ethernet-based and VPLS/Layer 2 VPN customer lines. The maximum number of MAC addresses to be learned depends on each individual case and on the hardware used and cannot therefore be specified authoritatively.

Motivation: An attacker can use fake packets with a high number of different MAC addresses to overbook existing resources so that the functionality, and thus, availability of systems is impaired.

For this requirement the following threats are relevant:
• Disruption of availability

---

Req 14 VLAN tags that originate from customer lines or external networks must not be used to control traffic.

VLAN tags can be freely manipulated and are thus classified as not trustworthy if they are assigned on the customer side. VLAN tags must therefore be handled appropriately at the network boundary. Depending on the usage model, the following methods can be used:

• With a Layer 2 VPN service, the customer sends packets with a VLAN tag (C-tag). This C-tag must not be analyzed within the network; instead, the traffic must be transparently routed to another customer location. This means that within the network the traffic must be encapsulated. This can, for example, be achieved using an additional provider tag (P-tag) assigned at the network boundary.
• If no traffic with VLAN tags is expected from a customer, packets with VLAN tags must be rejected.

If, in deviation from this requirement, VLAN tags from the customer are used for network-internal traffic control, a check must be carried out at the network entry point to determine whether the VLAN tag is correct for this customer and that there is no multiple tagging (Q-in-Q).
Motivation: Using manipulated VLAN tags, an attacker may break out from his permitted communication channel in order to access other systems in the network or systems of other customers.

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data
- Disruption of availability

Req 15 QoS values in packets that originate from customer lines or external networks must not be trusted.

Quality-of-service (QoS) is used in the network to prioritize traffic flows. This is used within the network for the prioritized transfer of control and management traffic. Furthermore, different traffic classes for customers are implemented using this technology. It is therefore necessary for the parameters used for this to be assigned in packets (TOS-bits) or frames (p-bits) in trustworthy systems. This means that relevant values, in packets or frames, which originate from customer lines or from connected third-party networks, must not be trusted. On network elements at network edges, these values must be overwritten and changed to in-house values.

Motivation: If an attacker can manipulate the QoS class of packets, it may be possible to upgrade traffic and thus to use services that do not comply with his contract or to influence the transport of other customers' data and the transport platform.

For this requirement the following threats are relevant:
- Unauthorized use of services or resources
- Disruption of availability

Protection against spoofing

Req 16 Spoofed ARP packets must be identified and rejected.

Protection must be provided on a network element or on a separate system in the network through which the spoofed ARP packets are identified. The preferred measure for this is to use ARP inspection to monitor the ARP cache and traffic and thus to identify ARP spoofing attacks. Other, and in some cases, additional measures include deactivating proxy ARP and gratuitous ARP as well as using static ARP entries.

Motivation: An attacker can use ARP spoofing attacks to prepare, e.g., man-in-the-middle attacks on systems located in the same network as the attacker. To do this, he sends spoofed ARP packets to systems in the network, whereby any traffic that is to be sent to the originating system is redirected via the attacker's system.

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data
- Disruption of availability

Req 17 Frames of customer lines with the same MAC sender and destination address must be rejected.

Identifying and filtering frames with fake MAC addresses is time-consuming and only possible to a limited extent. For this reason, at least those frames that contain an invalid MAC sender and destination address combination must be identified and rejected. These are frames whose MAC sender and destination address are identical. A network element must identify these frames as invalid and reject them. Further processing is not permitted.

Motivation: An attacker can use manipulated frames with the same MAC sender and destination address to impair the availability of vulnerable systems.

For this requirement the following threats are relevant:
- Disruption of availability
3.57 Architecture of the access and transport network

Req 18 Frames with the same MAC sender address for different customer lines on the same network element must be rejected.

Network elements save the MAC source addresses of frames forwarded for the first time in a table so that subsequent frames can be processed more quickly. If a MAC address on a customer line has been identified and saved in the MAC address table of the network element, frames with the same MAC source address from another customer line must be rejected.

Motivation: An attacker can send a frame with a MAC source address of the victim that has been faked. A network element that does not notice this overwrites the existing MAC address in the MAC address table with the port number of the attacker’s line. For as long as the victim does not send a new frame, the attacker will receive the victim’s data.

For this requirement the following threats are relevant:
• Unauthorized access or tapping of data
• Disruption of availability

Req 19 Packets with spoofed IPv4 or v6 source addresses must be identified and rejected.

Protection against packets with spoofed IPv4 or v6 source addresses must be provided on routers at network boundaries, i.e., at gateways to external networks and network elements to customer lines, in order to identify and reject the relevant packets. Possible solutions include:
• Activating the Unicast RPF (Unicast Reverse Path) function.
• Access control list via which packets that originate from an external network and have an IP source address from the internal network are identified and rejected.

On routers used for peering only spoofed packets addressed to the infrastructure addresses of the backbone must be filtered. This means transit traffic shall not be filtered.

There is no need to implement this measure on the router where a firewall providing IP spoofing protection is used for the network gateways.

Motivation: An attacker can use packets with a spoofed IP source address to hide his actual IP source address during attacks and to access systems for which his IP source address has been blocked.

For this requirement the following threats are relevant:
• Unauthorized access or tapping of data
• Disruption of availability

Multi Protocol Label Switching (MPLS)

Req 20 The internal structure of the MPLS transport network must not be visible from the outside.

The internal structure of the MPLS transport network must not be visible from the outside. This means that the label switch router (LSR) of the transport network must not appear as a hop on a trace route from a customer line or from external networks. The transport network thus only appears in the path as two hops (ingress and egress LER). This can be done by configuring the relevant routers accordingly. This suppresses the propagation of the TTL field of the IP packet into the preceding MPLS label.

Motivation: Implementing this measure makes it more difficult for an attacker to obtain information.

For this requirement the following threats are relevant:
• Unauthorized access to the system
MPLS layer 2 and 3 VPNs

Req 21 A separate connection must be used to connect a customer router to the MPLS edge router.

To ensure the separation of VPNs, a separate connection must be used every time a customer router (CE) is connected to a label edge router (LER). The connection can either be physically or logically separated. If, in the case of logical separation, a switch is used, make sure that a separate VLAN is used for each CE/LER connection.

Motivation: If multiple customer routers share the same Layer 2 infrastructure for the LER connection, fake packets may be imported by a VPN customer in order to gain access to a VPN of another customer on the same LER.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Disruption of availability

Req 22 Customer VPNs must be entirely separate from one another and from the underlying transport network.

Virtual networks can be implemented for customers on an MPLS-based transport network using Layer 2 and Layer 3 VPNs. In this way, every customer can use its own IP address structure without influencing other customers. Where such VPN solutions are implemented, it must be ensured that the traffic and the IP address structure of the various customers are separated from one another and from the transport platform used.

Motivation: It must be ensured that customers cannot break out from their VPN. If it is possible to overcome the VPN boundaries, an attacker can potentially feed packets into external customer VPNs or access systems in the transport network.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Disruption of availability

Req 23 If the customer uses its own routing in a customer VPN, this must be entirely separate from the routing for other VPNs and the transport platform.

Since every VPN customer can use its own IP address space within its VPN, it is possible for several customers to be using identical IP addresses. For this reason, in addition to a strict separation of the VPN traffic, an independent routing entity per customer VPN is also required.
The separation of the routing for a VPN is typically achieved via the configuration of "route targets" for VRF (Virtual Routing and Forwarding). Route targets are used to define which routes on an LER are imported or exported to a VRF for a customer VPN.

**Motivation:** The routing entities in the various customer VPNs must be entirely separate from each other as otherwise unauthorized accessibility of systems between customer VPNs would be possible. The greatest threat to the separation of VPNs is therefore posed by misconfiguring network elements.

For this requirement the following threats are relevant:

- Unauthorized access to the system
- Disruption of availability

---

**Req 24** The components of the MPLS transport network must not be accessible from customer VPNs.

The MPLS transport network provides data transport services for the connected customers. For a customer, the MPLS transport network to which the CEs of its locations are connected is not visible. To utilize the transport service, no more than the IP addresses between the CE and LER need to be visible to connect the VPN to the MPLS transport network. There is no need for the IP addresses of the core network to be accessible.

**Motivation:** Direct accessibility of the MPLS transport network from a customer VPN increases the risk of attack.

For this requirement the following threats are relevant:

- Unauthorized access to the system
- Disruption of availability

---

**Req 25** Layer 2 and 3 control plane traffic from customers must not affect the transport network.

One service feature of Layer 2 and Layer 3 VPNs is the exchange of control plane protocols with the customer. This is used to distribute routing information between customer locations or to exchange route decisions for transport in the VPN. Make sure that the control plane protocols used do not have any influence on the transport network.

**Motivation:** An attacker can specifically exploit an interpretation of the control plane protocols by the transport platform to impair the platform's availability or to influence traffic.

For this requirement the following threats are relevant:

- Disruption of availability

---

**Req 26** It must be ensured that targeted LDP packets are only accepted by trustworthy sender systems.

Targeted LDP packets (LDP = Label Distribution Protocol) are exchanged between routers that are not directly connected to each other in order to establish layer 2 VPNs, for example. Since such packets can be sent across several hops, it must be ensured that they are only accepted by trustworthy senders.

One option here is to limit the acceptance of targeted LDP packets to configured neighbors. However, this approach is only supported by some router manufacturers. Another option is to use filter lists so that only targeted LDP packets from authorized IP sender addresses are permitted.

**Motivation:** Accepting all targeted LDP requests on a destination system allows an attacker to send false information to a destination system, thus potentially compromising the stability of individual VPNs and/or the platform.
3.57 Architecture of the access and transport network

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data
- Disruption of availability
3.58 Architecture of datacenter and cloud infrastructures

Introduction
This document has been prepared based on the provisions of the Group Policy on IT/NT Security.

The security requirement is used as a basis for an approval in the PSA process, among other things. It also serves as an implementation standard for provisions of the Group Policy on IT/NT Security in units which do not participate in the PSA process. These requirements shall be taken into account from the very beginning, including during the planning and decision-making processes.
When implementing these security requirements, the precedence of national, international and supranational law shall be observed.

Infrastructure

Req 1 A datacenter infrastructure or (virtual) cloud platform must be able to separate systems, which are operated on top of this infrastructure, appropriately in line with their protection requirements.

Critical systems, in particular, always have to be separated from other systems including other critical systems. Similarly, systems on which personal data is processed must be protected against unwanted access or data flows from the same network or other networks by using state-of-the-art measures. Systems for an externally accessible presentation layer are to be implemented on physically separated infrastructure components.

Motivation: It is more likely for a less protected system to be compromised. This must not result in other systems with higher protection requirements being attacked by this compromised system.

For this requirement the following threats are relevant:
• Unauthorized access to the system
• Unauthorized access or tapping of data
• Unauthorized use of services or resources
• Disruption of availability
• Unnoticeable feasible attacks

Req 2 If systems are accessible from external networks, they must be implemented on an infrastructure which is physically separated from internal systems.

To support a physical separation, as required in the layer model, all externally reachable machines and the corresponding infrastructure elements like hypervisors, and network elements such as switches, routers, firewalls and load-balancers must be physically separated from internal systems.

Motivation: This requirement ensures that there is only one path from the systems accessible from the Internet to the underlying application and database systems. The firewall or filter elements cannot be circumvented via components that are accessible from the Internet as well as from internal networks. This has also the benefit that infrastructure incidents only affect a sub-part while other parts remain available.
Implementation example: Independent switches and routers are used to connect a presentation layer to the internet.

For this requirement the following threats are relevant:
• Unauthorized access to the system
• Disruption of availability
• Unnoticeable feasible attacks

Req 3 Infrastructure systems which are used for internal and external purposes must have a partitioning of their data and functions with respect to internal and external aspects.

In case of central storage systems these must ensure that internal data is not readable from the outside. An implementation can use physically divided storage pools, different SAN fabrics or a storage virtualization.

The system management must in this case occur over the management network for internal systems.

Motivation: *Internal data must not be compromised by external reachable components.*

For this requirement the following threats are relevant:
• Unauthorized access or tapping of data
• Unauthorized modification of data
• Disruption of availability
Req 4 Services in the infrastructure must be configured in such a way that accessing systems are only able to read and write designated data.

In case of storage systems a logical partitioning (vFiler) is recommended and usage of an authentication mechanism like Kerberos.

**Motivation:** Client separation and implementation of the principles of need-to-know and need-to-see.

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data
- Unauthorized modification of data

Req 5 Common used infrastructure services must especially be secured and available.

Failure of main services and security equipment can have an adverse effect on the protection of systems. They must be designed with the relevant security and availability in mind and not have any single point of failures. This includes:
- DNS servers for name and address resolution
- NTP servers for time synchronization
- Outgoing (web) proxies for controlled web access
- Jumphosts for system access internally/externally (3rd party access)
- System management and monitoring

Corresponding redundancies also make it possible to maintain central systems in ongoing operations.

**Motivation:** Main infrastructure services have a high need of protection and must especially be secured and available.

Implementation example: Jumphosts should be redundant, so that all systems can be administrated all the time, even by third parties. Outgoing (web) proxies should always be able to deliver software updates.

For this requirement the following threats are relevant:
- Unauthorized use of services or resources
- Disruption of availability

Req 6 IP addresses of infrastructure elements must be configured statically.

**Motivation:** Avoidance of unintended change of IP addresses of these elements.

For this requirement the following threats are relevant:
- Disruption of availability

Req 7 Attacks on the infrastructure platform must be recognized to ensure early countermeasures.

In order to detect attacks on the platform as early as possible and to minimize potential impacts, it is necessary to implement appropriate measures for monitoring and correlation of security events.

**Motivation:** Especially virtual environments provide additional layer of attack surface by using virtualisation software. This must be monitored and protected through the appropriate safety features, such as those already used in other technology areas. Only through active monitoring, detection and resulting subsequent actions, a continuous protection is possible on operated machinery.
For this requirement the following threats are relevant:

- Disruption of availability
- Unnoticeable feasible attacks
Virtualization

Req 8 Data and functions of virtual machines must be reliably separated by the virtualization environment.

Virtual machines, including their data, services and functions offered in the network, must be protected from each other. To achieve this, security functions of the virtualization solution must be used across all levels (physical, logical, net-work based, in the management, etc.).

Especially it has be ensured, that a compromised virtual instance does not allow access to another virtual instance or to the hypervisor. That means virtual machines must be protected against each other and no direct communication must be possible.

Virtual machines and their data must be grouped in such a form, that in the case of a break of the security features of the virtualization environment, the potential damage is limited to an acceptable level (residual risk). A consideration of the extent of possible damage is just as necessary as an assessment of protection needs, criticality, of the threat po-tential of individual systems, as well as the data to be processed. These indicators are to assess and to consider in the planning and implementation accordingly.

Motivation: The consistent partition of virtual instances limits the risk to compromise all virtual instances. To keep the potential effects of an attack as low as possible, for example in the form of an acquired virtual machine to another virtu-al machine, including the data, is to implement additional segmentation.

Implementation example: A corresponding grouping could, for example, follow the n-tier architecture model. By imple-menting, compromised virtual machines that are placed in the presentation layer, a vulnerability in the virtualization layer, granting no access via the hypervisor to the application and database layers on other machines.

For this requirement the following threats are relevant:
• Unauthorized access to the system
• Disruption of availability
• Unnoticeable feasible attacks

Req 9 Deployment of virtualization must not make it possible to circumvent security interfaces, functions and intrusion detection systems.

Using virtualization may sometimes make it possible to circumvent certain specified security mechanisms, e.g., the au-thorizations, rules or routes configured on a host or the underlying network. Circumvention of these security functions may jeopardize the security of other systems.

Motivation: If network gateways are interconnected directly via the virtualization software, attacks cannot be detected, i.e., network services are not protected from unauthorized access.

For this requirement the following threats are relevant:
• Unauthorized access to the system
• Unauthorized use of services or resources
Req 10 The networks needed for the operation of a virtualization environment (hypervisor management, live migration, heart beat, ...) must be completely separated from the networks (including management network) of the virtual machines running on top of the virtualization environment.

Traffic data on the virtual machines must be separated from the administration of the virtualization environment. A physical separation is recommended.

Motivation: The (hardware based) platform and the corresponding management must always be under control of the operator. It is designed to prevent a virtual machine influencing the virtualization environment or gaining access to it.

Implementation example: A physical separation complies to this requirement.

For this requirement the following threats are relevant:

- Unauthorized access to the system
- Disruption of availability
- Attacks motivated and facilitated by information disclosure or visible security weaknesses
Networks

Req 11 Production networks (external and internal), management networks, office networks, test & development networks, transport networks and other network areas must be separated from each other.

A physical separation of these network areas is to be preferred. Gateways between these network areas must be set up via appropriate security systems (e.g., firewalls and jumphosts). It is recommended to use clearly distinguishable IP address ranges for these different kinds of network areas in the “1st level segregation”.

Motivation: This way, systems are separated according to their purpose.

For this requirement the following threats are relevant:
• Unauthorized access to the system
• Unauthorized access or tapping of data
• Unauthorized use of services or resources
• Disruption of availability
• Unnoticeable feasible attacks

Req 12 A network separation on the client side of systems must be implemented with the same separating effect on the management network as well as all other interfaces.

Systems and system components must be separated with an identical effect at all points via which communication is possible. A network-based separation of systems (e.g., through VLANs, private VLANs or other layer-2 techniques) must take place in all connected network areas (e.g., production network, management network, storage/backup network) according to the same logic. The same applies to other network technologies (e.g., Fiber Channel (FC)).

Special attention must be paid to system management and backup systems: In many cases, multiple systems are managed via a single instance. Please note that it must not be possible to communicate between systems or have one system compromise another one using the management network. So internally and externally reachable systems must not be managed within the same VLAN.

Motivation: A separation with the intention to prevent other systems from being attacked following a compromise only generates genuine added value if it is executed on all interfaces in the same way.

Implementation example: From a “customer perspective”, presentation and database layer are separated on the network side. However, the administrative management interfaces would be connected in the same network and communication among systems would be possible. If the presentation layer was compromised, an attacker could access the systems of the database layer on layer-2 by circumventing a possible firewall.

For this requirement the following threats are relevant:
• Unauthorized access to the system
• Disruption of availability
• Unnoticeable feasible attacks
Req 13 “Site-to-Site” connections, e.g., generic IPSec tunnels connecting internal networks between data centers, must be terminated in such a way that the tunneled communication can be checked at the relevant firewall / the packet filter of the target network.

This also concerns generically configured tunnels/VPNs between data centers which are used by multiple systems. For example, untrustworthy network sections are bridged this way.

Motivation: *Encrypted connections on the IP level must be terminated in such a way that it is still possible to check the communications matrix.*

For this requirement the following threats are relevant:
• Unauthorized access to the system
• Unauthorized use of services or resources

Req 14 At the network border of the datacenter IP packets with invalid addresses or which are logically in-correct must be dropped incoming and outgoing.

Packets with spoofed source addresses and invalid packets must be dropped early in the network. This is relevant for the whole datacenter network and is independent from the protection of individual systems by specialized filter elements.

Motivation: *Protection of systems in the datacenter against wrong requests and malicious IP packets. Protection of foreign systems against invalid packets sent out of the datacenter.*

Implementation example: Denying packets using private IP addresses (as source or destination) at the border gateway to the public internet; dropping packets with addresses which are locally to the datacenter network.

For this requirement the following threats are relevant:
• Disruption of availability
Firewalls

Req 15 Filter elements for protecting systems (e.g., a firewall or loadbalancer) must be independent of the systems to be protected.

When connecting networks the protection function (e.g., a firewall or a session border controller) must not also “run” on a system that is to be protected. In the case of virtual firewalls another hypervisor is required as for the systems to be protected.

Sometimes, loadbalancers take on a double role, when used for application balancing and also acting as a filter on IP level.

Motivation: The separation of the security elements

• ensures a clear separation between security functions and services and
• supports a distinction between administrative tasks and responsibilities.

For this requirement the following threats are relevant:
• Unauthorized access to the system
• Unauthorized use of services or resources

Req 16 Filter elements like firewalls must not contain services which have negative influence to the reliability of the firewall functionality.

Network functions which can affect each other must be implemented on different systems:
• Router using dynamic routing protocols,
• Firewalls if changes of the ruleset often occur or in case of deep packet inspection.

However, static IPSec tunnels can be terminated on a firewall if all access rules still apply and no rules are circumvented.

Motivation: More features in a single system lead to more risks of a disruption of the main functions.

For this requirement the following threats are relevant:
• Disruption of availability

Req 17 The default configuration of a filter element (firewall) must be "deny any any", that is every needed communication has explicitly to be allowed.

Work must be based on whitelists and must be configured in such a way that, where the set of rules is processed sequentially, there is always a general deny at the end (default deny). Incoming as well as outgoing traffic must be filtered where in individual cases this may be omitted. Logically incorrect IP packets, e.g., with clearly forged sender addresses (“spoofing”) must be discarded. The configured rules must correspond to a documented communication matrix.

To avoid activations for huge TCP/UDP port ranges, necessary services/protocols with dynamic port assignment should be investigated for alternatives, as far as a solution by using an appropriate firewall system with corresponding protocol support is not available. Otherwise, services might be accessible via the enabled ports which should actually be blocked.

Not every source IP address must individually configured if the protection of accessible destination services is the main goal. With relation to the criticality of the systems and their need of protection IP ranges instead of single IP addresses can be configured if the maintenance overhead and the number of possible configuration errors can be significantly reduced and no higher risk to the destination systems arises.

Motivation: Despite taking great care, the use of blacklists may cause communication links to be released unintentionally. This measure is used to directly protect the systems and also to protect the entire network and ensure its stability.
For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized use of services or resources
- Unnoticeable feasible attacks
- Attacks motivated and facilitated by information disclosure or visible security weaknesses

Req 18 Filter elements (firewalls) may forward to internal systems only packets from allowed internal net-works.

IP packets with addresses (source or destination) not used in internal networks have to be dropped.

Motivation: Protection against malicious packets which could influence the availability and integrity of the destination system.

Implementation example: No default route on internal systems (network elements just as servers). Modern IP stacks drop packets with no returning route (RP filter).

For this requirement the following threats are relevant:
- Disruption of availability

Req 19 Access rules must be configured for IPv4 and IPv6 similarly.

Packet filters, firewall rules etc. must be set up so that the same protection measures apply to IPv4 and IPv6. Services that are not accessible via IPv4 from a particular network must also not be accessible via IPv6.

Motivation: The use of IPv6 should not enable protection measures implemented for IPv4 to be circumvented.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized use of services or resources
- Disruption of availability

System management

Req 20 The management of the datacenter or cloud infrastructure must take place via an own manage-ment network which is strictly separated from other networks.

The access for administrative tasks must be configured in such a way that only administrators are able to access the system. The access must be configured so that it is only possible via dedicated jumphosts.

It is recommended that all manageable ("active") components are always reachable within their management network. Servers via ILO, network elements like router and switch over a dedicated management port. To ensure that access is possible all the time, the management network should be part of an own network infrastructure independent from data and customer traffic. Furthermore it is recommended to use a dedicated management network (VLAN) for every kind of technology and this network is reachable only from the designated management systems.

Motivation: Restricting possible access to a unique defined path and number of options makes it possible to clearly control access. Implemented protection mechanisms of the systems cannot be circumvented this way. The operator must not loose control over the infrastructure.
3.58 Architecture of datacenter and cloud infrastructures


For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unnoticeable feasible attacks

Req 21 Systems for system management must be used in a controlled manner.

Management systems, backup as well as monitoring systems are to be planned, installed and operated carefully. Usually they have privileged access to many systems by default. It if is not possible to do without this privileged access (e.g., with administrator rights), measures must be implemented to minimize any involved risks regarding the managed systems (“controlled”). The following applies:

- Communication between management and managed systems must be unidirectional with only one TCP/IP port.
- Central systems should be able to do without any privileged access to the managed systems, if possible.
- No central component should have privileged access to a critical mass of systems.
- Automated installation and configuration mechanisms should be secured in such a way that they cannot be misused for unwanted changes to systems; e.g., they should be deactivated after installation and initial configuration. Especially for installation of software packages and configuration changes, a signature mechanism could be used to ensure that only authorized changes are applied.
- Backup data must be accessible from authorized systems only.

Motivation: If a central system is compromised, the possibility to directly compromise further systems with privileged rights must be restricted.

Implementation example: To configure a system or to collect configuration parameters, a way has been chosen where the target system establishes the communication. Access in near-real-time can take place by regularly triggering. HT-TPS is recommended because only one port is needed and confidentiality and authentication is included.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Disruption of availability

Req 22 System management software for managing different types of infrastructure elements or application systems must have a role based model that is oriented to the responsibilities of the different managing tasks.

Besides a different management for different kinds of physical systems, e.g., storage or server, the management software should also separate the management functions for server and network administration in virtual environments.
Motivation: A differentiation of network and system administration supports the principle of "separation-of-duties".

For this requirement the following threats are relevant:
• Unauthorized access to the system
• Unauthorized modification of data
3.59 Load Balancer as Web Server

Introduction

The security requirement is used as a basis for an approval in the PSA process, among other things. It also serves as an implementation standard for provisions of the Group Policy on IT/NT Security in units which do not participate in the PSA process. These requirements shall be taken into account from the very beginning, including during the planning and decision-making processes.

When implementing these security requirements, the precedence of national, international and supranational law shall be observed.

Comment on the term "web server":
A load balancer does not implement the full functionality of a web server, but basically the functionality of a reverse HTTP(S) proxy. Therefore the load balancer can replace a web server only if merely the functionality of a proxy is required.

Requirements on web server software

Req 1 Software and hardware components that are no longer supported by vendor, producer or developer must not be used.

Components that have reached end-of-life or end-of-support must not be used. Excluded are components that have a special support contract. This contract must guarantee the correction of vulnerabilities over components life-time.

Motivation: Hardware and software components that have reached end of life or end of support represent a risk for a system. This means that a vendor does not supply remedial updates or patches for a component should errors or vulnerabilities occur. This means that vulnerabilities cannot be fixed when they occur and could be exploited to compromise the system or to impair its availability.

For this requirement the following threats are relevant:
• Unauthorized access to the system
• Unauthorized access or tapping of data
• Unauthorized modification of data
• Disruption of availability

Req 2 Known vulnerabilities in software and hardware of the system must be fixed or protected.

Prior to installation of a software or hardware component, users must check whether any vulnerability has been disclosed and published for the version they are installing. Any component that proves to have a vulnerability must not be installed or used. Excepted from this rule are components for which the vendor has already provided a measure to remedy the vulnerability, e.g. a patch, update or workaround. In this case, the additional measure must be implemented on the system. Furthermore it is a ongoing process during the complete life cycle of the system to fix upcoming vulnerabilities promptly.

Motivation: Publication of vulnerabilities increases the risk of successful exploitation by an attacker, especially since the published information usually includes details on how to exploit the vulnerability and tools that make exploitation possible. One example of this is when Web servers are compromised due to not patched vulnerabilities.

For this requirement the following threats are relevant:
• Unauthorized access to the system
• Unauthorized access or tapping of data
## Configuration Requirements

<table>
<thead>
<tr>
<th>Req 3</th>
<th>The reachability of services must be restricted.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Typically services that are enabled in the basic configuration are accessible over all interfaces of the system and can be reached from systems in connected networks. This availability is often not needed or meaningful for system functioning. For this reason, services should only be enabled on interfaces where their usage is required. On interfaces were services are active, the reachability must be limited to legitimate communication peers. This limitation must be realized on the system itself (without measures (e.g. firewall) at network side).</td>
<td></td>
</tr>
</tbody>
</table>

**Motivation:** Disabling services on interfaces which do not require system accessibility or by limiting the reachability can greatly reduce the potential vulnerabilities offered to an attacker. For example, access to a system via SSH from the Internet is not necessary. If this service could be accessed from the interface connected to the Internet, this would greatly increase the risk of attacks on the service.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized use of services or resources
- Disruption of availability
- Attacks motivated and facilitated by information disclosure or visible security weaknesses

<table>
<thead>
<tr>
<th>Req 4</th>
<th>Unused functions of the operated software and hardware must be deactivated.</th>
</tr>
</thead>
<tbody>
<tr>
<td>During installation of software and hardware often functions will be activated that are not necessarily needed for operation or function of the system. Functions of software are currently inherent part which could not be deleted or deinstalled individually. Such functions must be deactivated in the configuration of the system permanently.</td>
<td></td>
</tr>
</tbody>
</table>

Beside the functions of the software also hardware functions are active which are not necessary for a system. Functions like unused interfaces must permanently deactivated. Permanent means that they must not be reactivated again after system reboot.

**Motivation:** The hardware or software of a system often contains functions which are not used and so will be a risk for system security. Such functions give an attacker the possibility to manipulate the system. Furthermore it is possible to get unauthorized access other areas or data of the system. An example is a debugging function in software which can be used for troubleshooting but must not be activated during normal operation. Or a hardware interface that will not be used and so is unsecured an allows possibly unauthorized access to the system.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Disruption of availability

<table>
<thead>
<tr>
<th>Req 5</th>
<th>HTTP requests, that do not comply with appropriate protocol specifications, must be rejected.</th>
</tr>
</thead>
<tbody>
<tr>
<td>HTTP requests must comply with RFC2616. An HTTP request, for example, containing an incorrect content length header must be rejected with an appropriate HTTP status code. This requirement is applicable only if the load balancer supports an HTTP compliancy check (e.g. &quot;Strict HTTP inspection&quot;).</td>
<td></td>
</tr>
</tbody>
</table>
Motivation: Any request that does not satisfy the RFC specification could indicate an attempted attack.

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources

Req 6 HTTP methods that are not required must be deactivated.

Standard requests to web servers only use GET and POST. If other methods are required, they must be processed securely.

Motivation: HTTP TRACE could be misused by an attacker. This method allows for debugging and trace analysis of connections between the client and the web server. The Microsoft IIS web server uses the TRACK alias for this method. Other HTTP methods could also be used to obtain information about the server, or they could be directly misused by an attacker.

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Attacks motivated and facilitated by information disclosure or visible security weaknesses

Req 7 The server string in the HTTP header must be replaced.

The HTTP header must not include information on the version of the load balancer and the modules/add-ons used.

Motivation: Any information about the software could allow conclusions to be drawn about security vulnerabilities. If, for example, the server string is "Server: Apache/2.4.1 (Unix)", then an attacker could look specifically for vulnerabilities of this version.

Implementation example: Change the configuration to deliver

Server:
Webserver as HTTP header.

For this requirement the following threats are relevant:
- Attacks motivated and facilitated by information disclosure or visible security weaknesses

Req 8 Long runtimes of HTTP requests must be prevented.

The configuration of timeouts, which control the response to HTTP requests, must prevent long runtimes. The technical options may differ for different load balancer products.

Motivation: The load balancer performance can be reduced significantly by long runtimes of HTTP requests - up to a denial-of-service. This can be misused for an attack (example: Slowloris).

For this requirement the following threats are relevant:
- Disruption of availability

Req 9 The load balancer must protect itself and downstream systems against attacks and faulty requests.

Load balancers provide several configuration options as protection measures, e.g.
- Limit the length of HTTP requests
- Activate "Slow Start"
- Set "maximum connections per real server"
• Protect SynFlood

The specific configuration measures depend on the options of the load balancer (see the
documentation of the vendor) and the requirements of the application behind the load balancer.

Motivation: Attacks and faulty requests should be recognised and repelled as early as possible.

For this requirement the following threats are relevant:
• Disruption of availability

HTTPS requirements

<table>
<thead>
<tr>
<th>Req 10</th>
<th>For encryption with HTTPS the TLS protocol must be used.</th>
</tr>
</thead>
</table>
SSL must be considered outdated and thus may not be activated or must be deactivated, respectively.

Motivation: Particularly SSLv2 has a number of weaknesses that make it impossible to use from a security point of view. TLS is the further development of SSL. It is already established for years so there is no need for further use of SSL.

For this requirement the following threats are relevant:
• Unauthorized access or tapping of data
• Unauthorized modification of data

<table>
<thead>
<tr>
<th>Req 11</th>
<th>The web server must be configured in such a way that the use of the latest version of the TLS protocol is enabled.</th>
</tr>
</thead>
</table>
In particular, the web server must be configured for the use of TLS 1.2.

Motivation: The latest version of the protocol offers the best possible protection and contains fixes to known vulnerabilities in previous versions of the protocol.

For this requirement the following threats are relevant:
• Unauthorized access or tapping of data
• Unauthorized modification of data

<table>
<thead>
<tr>
<th>Req 12</th>
<th>The TLS configuration must not use any unsecure cipher suites.</th>
</tr>
</thead>
</table>
Cipher suites with one of the following characteristics are not secure
• no server authentication,
• no encryption,
• encryption with DES, RC4 or any other algorithm with a key length of less than 128 bits
• use of MD5 as hash function

Motivation: Cipher suites known to be unsecure do not offer sufficient protection.

For this requirement the following threats are relevant:
• Unauthorized access or tapping of data
• Unauthorized modification of data

<table>
<thead>
<tr>
<th>Req 13</th>
<th>The TLS configuration must provide that the cipher suite considered most secure is being chosen with highest priority.</th>
</tr>
</thead>
</table>
A cipher suite contains the definition of four algorithms. These are used for key exchange, authentication, encryption and as a hash function. General guidelines for the prioritization are
• For the key exchange the Diffie-Hellman method must be preferred because it offers perfect forward secrecy. Cipher suites using the Diffie-Hellman method usually may be identified by the strings DHE or ECDHE. ECDHE has higher priority than DHE.
3.59 Load Balancer as Web Server

- For encryption the Advanced Encryption Standard (AES) or Camellia with a key length as big as possible has to be used.
- As a hash function SHA-2 has to be used. This function usually may be identified by the string SHA followed by a number (256, 384 or 512). Warning: if the string SHA is not followed by a number this identifies the SHA-1 function which is significantly less secure.

*Motivation: When a TLS connection is being established a cipher suite is selected based on the cipher suites available both on client and on server side. In order to ensure a high compatibility to all kinds of client systems the web server must not only allow for the cipher suites considered most secure. To make sure that nevertheless for each client the best possible cipher suite is selected and thus the connection is best protected the configuration must contain an according prioritization.*

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data
- Unauthorized modification of data

**HTTPS requirements**

**Req 14** Certificates must be issued by a certification authority whose certificates are recognized by the commonly used web browsers.

For critical applications that can be used via the Internet, use of an extended validation certificate (EV certificate) is recommended.

*Motivation: Only if the certificate authority (CA) is contained in the CA list of the browser being used the browser can verify the authenticity of the server or web application.*

Stricter issuing criteria apply to EV certificates. If an EV certificate is used, this is visualized in the browser. Even if EV certificates do not improve security, their use increases the trustworthiness of the server for the user.

For this requirement the following threats are relevant:
- Attacks motivated and facilitated by information disclosure or visible security weaknesses

**Req 15** Certificates must lose their validity after a maximum of 36 months.

*Motivation: The methods used for analysing and breaking cryptographic processes are improved continuously. Therefore the security of the certificates can be ensured for a limited period only. But, according to a general estimation, the security of the certificates is ensured for the required validity period of three years, if an appropriate key length is used.*

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data
- Unauthorized modification of data

**Req 16** Certificates must have a key length of at least 2048 bits when using RSA or 256 bits when using ECC.

*Motivation: In order to guarantee the security of certificates over the validity period, the cryptographic keys must have an appropriate length. According to a general estimation, a key length of 2048 bits provides sufficient protection for the next years. For ECC algorithms, shorter key lengths already provide the same level of security.*

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data
- Unauthorized modification of data
3.60 PostgreSQL databases

Introduction
This security requirement has been prepared based on the provisions of the Group Policy on IT/NT Security. The security requirement is used as a basis for an approval in the PSA process, among other things. It also serves as an implementation standard for provisions of the Group Policy on IT/NT Security in units which do not participate in the PSA process. These requirements shall be taken into account from the very beginning, including during the planning and decision-making processes. When implementing these security requirements, the precedence of national, international and supranational law shall be observed.

General security requirements for PostgreSQL DBMS

Req 1 In the production environment, community-supported or commercially supported version of the PostgreSQL software must be used.

Motivation: Developers doesn't provide security fixes for PostgreSQL versions which are in EOL (End Of Life) stage.

Implementation example: To check if your PostgreSQL server is supported by the community, you can execute

SELECT version();

query to obtain version of the software, then compare returned result with vendor information published on below web page:

http://www.postgresql.org/support/versioning/

To check the client, run psql -V command, then compare the result with the information from the webpage.

For this requirement the following threats are relevant:
• Unauthorized access or tapping of data
• Unauthorized modification of data
• Unauthorized use of services or resources
• Disruption of availability

Hardening

Req 2 On one operating system instance (both physical HW or virtualized) must only run one instance of a PostgreSQL system.

Motivation: If multiple database instances for different tasks (e.g., Internet and intranet) are running on one operating-system instance, the instances are not separated from one another. There is a risk that attackers could also corrupt the second database system. To separate the individual database instances deploy virtualization solutions.

Implementation example: To separate the individual database instances, administrators can use physical server hard-ware and virtualization solutions.

For this requirement the following threats are relevant:
• Unauthorized access or tapping of data
• Unauthorized modification of data
• Disruption of availability
Default databases

Users and roles

Service minimization

Secure transmission of passwords

Req 3 (Windows only) Default passwords on PostgreSQL database systems must be changed.

Motivation: Exploitation of default (easy guessable) passwords is easier way to hack into system.

Implementation example: You must especially check "postgres" account if it has not set password same as username.

To change the user password use below command:

Following command is recommended to change passwords: 

```
#password <username>
```

This statement will not be logged in the logfile itself.

For this requirement the following threats are relevant:

- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources
- Denial of executed activities

Req 4 Passwords in PostgreSQL must be stored in salted hash format (e.g. salted MD5) and transmitted in hashed form.

Motivation: Using hashing for securing the transmission of data and storage of data increases security.

Implementation example: To check if users have passwords stored as MD5 hashes execute following query from privileged account (e.g. "postgres"):

```
SELECT usename, passwd FROM pg_shadow;
```

Example of PostgreSQL MD5 hash:

```
md5f3c7d6d350e349efc7200ad5f781c7a5
```

To enforce using of MD5 hashes for the transmission you need to set "md5" authentication method on particular entry in "pg_hba.conf" file.

Below entries enforce of using "md5" authentication method for all users, databases on local and remote connections.

- CONNECTION_TYPE DATABASE USER ADDRESS METHOD
- "local" is for Unix domain socket connections only
- local all all md5
- IPv4 local connections: host all all
- 127.0.0.1/32 md5
• IPv6 local connections: host all all ::1/128 md5

Ensure that "password_encryption" parameter in "postgresql.conf" is set to "on".

To ensure encrypted password storage, you e.g. have to create the users in a dedicated way:

CREATE ROLE user1 ENCRYPTED PASSWORD '1234567'

For this requirement the following threats are relevant:
• Unauthorized access to the system
• Unauthorized access or tapping of data
• Unauthorized modification of data
• Unauthorized use of services or resources

---

**Req 5** The authentication method "peer authentication" must not be used for non administrative accounts, it may be only used for technical accounts.

If it's absolutely needed it must be used only on technical accounts and only for maintenance purposes (e.g. database backup)

**Motivation:** Administrative actions must be protected by strong authentication mechanisms.

Implementation example: Comment out entries in "pg_hba.conf" where "peer" method is used or change method to other (e.g. md5).

For this requirement the following threats are relevant:
• Unauthorized access or tapping of data
• Unauthorized modification of data
• Unauthorized use of services or resources

---

**Principle of least privilege and separation of privileges**

**Req 6** Only the Database Administrator (DBA) must have SUPERUSER, CREATEROLE or CREATEDB privileges.

**Motivation:** Granting extensive privileges to ordinary users can cause various security problems, such as: intentional/unintentional access, modification or destroy of data.

Implementation example: To identify roles with extensive rights execute following query:

```sql
SELECT rolname,rolsuper,rolcreaterole,rolcreatedb FROM pg_roles WHERE rolsuper IS TRUE OR rolcreaterole IS TRUE OR rolcreatedb IS TRUE;
```

Below commands can be useful for removing of unnecessary privileges:

```sql
ALTER ROLE user_name WITH NOSUPERUSER;
ALTER ROLE user_name WITH NOCREATEROLE;
ALTER ROLE user_name WITH NOCREATEDB;
```

For this requirement the following threats are relevant:
• Unauthorized access or tapping of data
• Unauthorized modification of data
• Unauthorized use of services or resources
Req 7 Only the DBA must have privileges on pg_catalog.pg_authid table.

Motivation: In pg_catalog.pg_authid table there are stored credentials such as username and password. If hacker has access to the table, then he can extract these credentials.

Implementation example: To check what users have access to pg_catalog.pg_authid, execute following command from psql:

```
\dp pg_catalog.pg_authid
```

Below command can be useful when removing unnecessary privileges on database objects:

```
REVOKE ALL PRIVILEGES ON pg_catalog.pg_authid FROM user_name;
```

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources

Req 8 The privilege rologin must be only set for dedicated users / roles (not "all"), which have an entry in the file pg_hba.conf.

Motivation: If role has host login right, then it can be used for remote network login, therefore creating a new attack vector.

Implementation example: To identify roles with login privileges execute following query:

```
SELECT rolname, rolcanlogin FROM pg_roles;
```

You can remove login privileges with below command.

```
ALTER ROLE user_name WITH NOLOGIN;
```

Additionally it's recommended to open "pg_hba.conf" and comment-out every unnecessary entry beginning with host (host, hostssl, hostnossl)

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data
- Unnoticeable feasible attacks

Req 9 The "trust" authentication method must NOT be used.

Motivation: If "trust" authentication is specified for particular host, you can log in from the host on every available role without providing a password.

Implementation example: From "pg_hba.conf" file comment out every entry with "trust" ending.

For this requirement the following threats are relevant:
- Unauthorized modification of data
- Unauthorized use of services or resources
- Unnoticeable feasible attacks

Req 10 Privileges on objects must not be granted to PUBLIC.

Motivation: If privilege is assigned to PUBLIC it means that it is assigned to every database user. This is a violation of segregation of privileges principle.

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data
- Unauthorized modification of data
• Unauthorized use of services or resources

---

### 3.60 PostgreSQL databases

**Req 11** Privileges on objects must not be granted with "GRANT" option.

*Motivation: If GRANT option is specified, the recipient of the grant may grant it to others. This can result in violation of segregation of privileges principle.*

Implementation example: From "psql" client after connection to specific database ("psql database_name") execute following shortcut to identify privileges on the tables:

\> dp (access privileges with GRANT OPTION are marked with the star "***" sign displayed after particular right) REVOKE GRANT OPTION FOR ALL ON table_name FROM user_name;

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unnoticeable feasible attacks

---

### SQL functions and packages

**Req 12** SQL functions with SECURITY DEFINER must be used in a restrictive manner.

*Motivation: The SECURITY DEFINER property of functions is similar to the setuid(2) feature in Unix Operating Systems. This property allows users to execute functions with the privileges of the owner of the functions rather than with the privileges of the user invoking the function.*

Implementation example: To find functions with SECURITY DEFINER you have to execute following query:

```sql
SELECT pg_proc.proname, pg_namespace.nspname, pg_user.usename FROM pg_proc JOIN pg_namespace ON pg_proc.pronamespace=pg_namespace.oid JOIN pg_user ON pg_proc.proowner=pg_user.usesysid WHERE prosecdef='t';
```

Then you can delete particular function:

```sql
DROP FUNCTION function_name(param_1 TEXT, param_2 TEXT);
```

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data
- Unauthorized modification of data

---

### SQL extensions with operating system or network access

**Req 13** If not otherwise necessary, only "c" and "internal" shall be used as non-trusted procedural languages.

*Motivation: If additional programming languages such as plperl are installed with non-trust mode, then it's possible to gain OS-level access permissions such as executing OS commands.*

Implementation example: To check what languages are non-trusted execute following query:

```sql
SELECT lanname AS language, lanpltrusted AS trusted FROM pg_language WHERE lanpltrusted = 'f';
```

Then you can remove language by below query:

```sql
DROP LANGUAGE proclang_name;
```
Additional background info: http://www.postgresql.org/docs/8.2/static/plperl-trusted.html

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources
- Unnoticeable feasible attacks
Req 14 Procedural languages which are not needed must be removed.

*Motivation:* Additional procedural languages installed on the database can create new attack vector for the attacker.

Implementation example: You can list all available procedural languages by below query:

```sql
SELECT lanname AS language FROM pg_language;
```

You can remove unneeded language by executing following command:

```sql
DROP LANGUAGE plang_name;
```

Name the defaults which can / should "stay":

```
language
--------
• internal
• c
• sql
• plpgsql
```

For this requirement the following threats are relevant:

- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unnoticeable feasible attacks

Access rights

Req 15 The PostgreSQL "data_directory" and config files must be assigned exclusively to the database system's operating-system account (such as "postgres"). Other systems users must not have access to the "data_directory" and config files.

*Motivation:* If file permissions on data and config files are not properly defined, other users may read, modify or delete those files.

For example: it may be possible to tamper log files or copy whole database with all data inside.

Implementation example: You can execute similar to below OS command to find PostgreSQL files with broken privileges:

```bash
find / -user postgres -group postgres -perm /go=rwx
```

Then you can correct privileges/ownership with "chmod" and "chown" OS command

For this requirement the following threats are relevant:

- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources
Data communication

Req 16 Access to the databases must be implemented on an 1:1 relationship between database user and database server. For this, the built in filtering function based on the pg_hba.conf must be used.

Motivation: This allows for privilege separation and therefore reduction of risk of misuse

Implementation example: Below entries from pg_hba.conf shows how this requirement can be handled:

```
hostssl database_1 user_sys_1 10.23.24.123/32 md5 hostssl database_1
user_sys_2 10.23.24.11/32 md5
```

For this requirement the following threats are relevant:

- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unnoticeable feasible attacks

Req 17 If the PostgreSQL server uses encrypted connections the usage of secure client ciphers must be enforced.

The following categories of SSL Ciphers must not be used: ADH, LOW, EXP and MD5.

Motivation: If SSL on the server is not enabled, there's no possibility for a client to establish secure SSL connection with them.

Implementation example: In the "postgresql.conf" file set "ssl" parameter to "true".

```
"postgresql.conf" set "ssl_ciphers" parameter as below:
ssl_ciphers = 'HIGH:MEDIUM:+3DES:!aNULLH'
```

A very good description for secure postgres installation / configuration can be found at: https://bettercrypto.org.

Every entry in the file "pg_hba.conf" should start with the "hostssl" praefix, as seen below:

```
hostssl database_1 user_sys_1
10.23.24.123/32 md5 hostssl database_1
user_sys_2 10.23.24.11/32 md5
```

Furthmore we recommend www.bettercrypto.org for additional crypto information.

For this requirement the following threats are relevant:

- Unauthorized access or tapping of data

System monitoring

Logging
Req 18  Logging functions must be turned on and properly configured according / compliant to local law.

Motivation: If logging functions are not enabled (and/or not properly configured) there will be no track about user activities in database, so diagnosis of security incidents may be limited or impossible.

Implementation example: In "postgresql.conf" you have to turn on "logging_collector" and set "log directory" as on ex-ample:
logging_collector =
on log_directory =
'pg_log'

It's recommended to turn on following features:
log_connections =
on
log_disconnections = on log_duration =
on log_hostname =
on
log_line_prefix = '%t %u %d %h'

For this requirement the following threats are relevant:
• Denial of executed activities
• Unnoticeable feasible attacks
• Attacks motivated and facilitated by information disclosure or visible security weaknesses

Auditing and monitoring

Req 19  Log files from PostgreSQL must be monitored continually for misuse scenarios.

The monitoring can be e.g. done fully automatically only triggering alarms for the operations unit. At any case, the compliance to local laws / workers council should be taken into account.

Motivation: If regular log review is not performed, then there's higher risk that malicious activity has been not dis-covered.

Implementation example: You can grep server log files looking for suspicious activity like access attempts to sensitive tables (e.g. "pg_shadow", "pg_authid").

Below example shows how this requirement can be handled:
grep 'permission denied' /var/lib/postgresql/8.4/main/pg_log/*.log

For this requirement the following threats are relevant:
• Unnoticeable feasible attacks
Hardening

This chapter lists the requirements for hardening database systems. The measures are comparable to those for hardening the operating system. They reduce the likelihood of successful attacks on database systems and/or the consequences thereof.

Default databases

Req 20 (Default) databases that are not required must be deleted on the database system.

Motivation: When installing database systems, test or practice databases are often installed which are not required once the database goes productive. In the past, vulnerabilities of these test databases have become known which allow an attacker to gain privileged rights on the database system. Such knowledge enables an attacker to access the database system. Therefore, all databases that are not required shall be deleted.

For this requirement the following threats are relevant:

- Unauthorized use of services or resources

Users and roles

Req 21 (Default) users and (default) roles that are not required must be deleted.

Motivation: When installing a database system, a large number of users and roles are automatically installed, which are not required to operate the database. These are envisaged, for instance, for practice and test databases. Knowledge of default users and roles allows an attacker to gain (privileged) access to the database system. Therefore, users and roles that are not needed for database functions shall be deleted.

For this requirement the following threats are relevant:

- Unauthorized access to the system

SQL functions and packages

This section defines requirements for extensions of the standard SQL command language through additional program packages. These program packages enable often access to operating system/file level or to external network services (e.g. HTTP servers).

Req 22 Extended SQL functions that are not required (e.g. T-SQL, PL/SQL, SQL PL, extended stored procedures) and/or packages from the database system must be deleted.

Motivation: In many instances an attacker exploits such functions to transfer malware onto the system or execute commands with privileged rights.

For this requirement the following threats are relevant:

- Unauthorized use of services or resources

SQL extensions with operating-system or network access

Req 23 Database functions which permit access to operating-system files must be deleted or deactivated.

Motivation: Some database systems allow access to the operating-system level via special stored procedures or SQL functions. A multitude of vulnerabilities of these procedures are known which allow an attacker to gain unauthorized access to a system or execute malware on
3.60 PostgreSQL databases

the target system. These procedures are often provided with unnecessarily high, mainly privil

eged rights or allow anyone to execute programs without authentication.

For example, malware exploits the procedures MS-SQL: xp_cmdshell and Oracle: utl_tcp.

For this requirement the following threats are relevant:
• Unauthorized use of services or resources

---

Req 24 Database functions which allow access to other network services (e.g., SMTP, HTTP, SNMP, FTP etc.), must be deleted or deactivated.

Motivation: Some database systems provide functions that are normally offered by an application server. For instance it is possible via special stored procedures to send e-mails or launch web queries to external systems.

For this requirement the following threats are relevant:
• Unauthorized use of services or resources

---

Req 25 Database functions which allow access to the operating system level or network services, must not be accessible by the roles / groups "Public" and / or "Everyone".

Motivation: Withdrawing rights can prevent the use of database functions to gain unauthorized access to network services and the operating-system level. Functions to execute operating-system commands and to use network services are thus only available to authorized users, and not to everyone.

For this requirement the following threats are relevant:
• Unauthorized access to the system
• Unauthorized access or tapping of data
• Unauthorized modification of data
• Unauthorized use of services or resources

Access rights

Req 26 The operating-system rights for the database files and directories (program, control, trace and log files) must be assigned exclusively to the database system’s operating-system account.

Some database systems store sensitive account data in plain text in trace or log files. Therefore, access rights to sensitive files and directories on the database system MUST be set so that non-administrative users do not have any read, write or execution rights.

Access to essential system files and directories on the database system must be reserved for the database system’s user account.

Motivation: By implementing restrictive access rights, the risk of manipulation can be substantially reduced.

For this requirement the following threats are relevant:
• Unauthorized access to the system
• Unauthorized access or tapping of data
• Unauthorized modification of data
• Unauthorized use of services or resources
System monitoring

Auditing and monitoring

---

Req 27  Important database services and instances must be monitored continually for misuse scenarios.

The monitoring of user actions for misuse shall comply with national legislation currently in force (for details see the “Security Requirement on Misuse Detection”).

Motivation: There are many conceivable ways to misuse database systems. Users generate an unusually high data usage rate or operate at unusual times of day. Attackers utilize unusual and critical commands for database queries, as well as tools and malware to extend their rights. To detect misuse, database systems should be continually monitored for misuse scenarios, e.g. by means of database triggers and log monitoring.

For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources
### 3.80 Nginx

#### Introduction

This security requirement has been prepared based on the provisions of the Group Policy on IT/NT Security.

The security requirement is used as a basis for an approval in the PSA process, among other things. It also serves as an implementation standard for provisions of the Group Policy on IT/NT Security in units which do not participate in the PSA process. These requirements shall be taken into account from the very beginning, including during the planning and decision-making processes. When implementing these security requirements, the precedence of national, international and supranational law shall be observed.

#### Nginx webserver platform requirements

**Req 1** Software and hardware components that are no longer supported by vendor, producer or developer must not be used.

Components that have reached end-of-life or end-of-support must not be used. Excluded are components that have a special support contract. This contract must guarantee the correction of vulnerabilities over components life-time.

**Motivation:** Hardware and software components that have reached end of life or end of support represent a risk for a system. This means that a vendor does not supply remedial updates or patches for a component should errors or vulnerabilities occur. This means that vulnerabilities cannot be fixed when they occur and could be exploited to compromise the system or to impair its availability.

For this requirement the following threats are relevant:

- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Disruption of availability

**Req 2** Data with need of protection must be secured against unauthorized viewing and manipulation during transmission and storage.

Adequate security measures for transmission and storage must be implemented of data with a need for protection that are classified as internal, confidential or strictly confidential. The chosen measure depends on the classification for the data and other factors such as the type of network used during transmission, the storage location for data, etc. Furthermore must be guaranteed that confidential and strictly confidential data will not be unprotected during temporary storage (e.g. in web cache, temporary folders).

All authentication data such as user names, passwords, PINs, etc. must be protected against unauthorized viewing and manipulation. This applies equally to permanent storage and transmission. Typical measures taken to protect authentication data or combinations of these are:

For storage:

- Client systems: encryption or obfuscation of authentication data, no persistent storage, limitation of access rights.
- Server systems: hashing of authentication data with PBKDF like scrypt or bcrypt or when this is not possible hashes with salt.
For transmission:

- Usage of cryptographically protected network protocols.
- Challenge response method (Disadvantage of this solution is that passwords needed in clear-text on server. If this solution is feasible depends on the individual threat scenario)

Files of a system that are needed for the functionality must also be protected against manipulation. This is necessary because system's integrity can be damaged when the system access this kind of files. An example is the use of check-sum or cryptographic methods to validate if e.g. firmware images, patches, drivers or kernel modules are free of manipulations.

For transmission of data with a need of protection network protocols that are insecure due to insufficient security measures shall not be used. Examples are: SSLv3, SSHv1, FTP, Telnet, SNMPv1 and 2c. In case of these protocols a newer version without vulnerabilities or a secure alternative must be used.

Motivation: If data with a need of protection will not be secured an attacker could record or manipulate the data during transmission over a network. An example is the recording of user names and passwords during system administration with the telnet clear-text protocol. Storing data on a system without adequate protection may mean that unauthorized users can copy or modify it. One example is when passwords can be read out when they stored in an inadequate secured way (e.g. usage of unsecure hashing algorithms like MD5 or SHA-1) or even without encryption on a system or manipulation of firmware to affected the system integrity.

For this requirement the following threats are relevant:

- Unauthorized access or tapping of data
- Unauthorized modification of data

---

**Req 3**

Prevent overload situation to make the system predictable.

A web server must be designed and configured to react in a predictable way to overload situations.

If a situation nevertheless arises where the web server measures do not work, the host system should initiate protective measures against overloads.

In such a case, steps must be taken to ensure that the system does not get into an undefined and possibly insecure state. In an extreme case this may mean that a controlled system shutdown is preferable to uncontrolled failure of the security functions and thus loss of system protection.

Motivation: An attacker may try to overload a system using denial-of-service attacks so that the system's availability or integrity is disrupted. If such a system then responds unpredictably, this constitutes a risk for the actual functions and data and, where applicable, for other systems.

Implementation example: One example in this respect is the usage of a firewall in the form of IPTABLES in order to limit the number of connections from a client within a time period to the web server.

```
/sbin/iptables -A INPUT -p tcp --dport 80 -i eth0 -m state --state NEW -m recent --set
/sbin/iptables -A INPUT -p tcp --dport 80 -i eth0 -m state --state NEW -m recent --update --seconds 60 --hitcount 10 -j
```

DROP

Furthermore, the use of a WAF (Web Application Firewall) is recommended in order to protect the web server and the host system.

For this requirement the following threats are relevant:

- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources
3.80 Nginx

• Disruption of availability
• Unnoticeable feasible attacks

Req 4 The web server must be the only externally accessible service of a system, unless the web server is exclusively used for an administration interface.

The web server must not be operated on the same system as a database, firewall or a system for the detection of network attacks (with the exception of host-based IDS).

There may be multiple web servers in a system. The “system” may be a virtual system.

If the web server only provides an administration interface to a network component (router, firewall, etc.), for example, the service to be administrated is permitted to be running in parallel to the web server.

Motivation: As soon as security is compromised on one of the components installed on the server system, the entire server system must be considered compromised.

For this requirement the following threats are relevant:
• Unauthorized access to the system

Nginx webserver installation requirements

Req 5 Access rights for web server configuration files must only be granted to the owner of the web server process or a user with system privileges.

Motivation: Configuration files may only be written by the owner of the web server process or a user with system privileges. Otherwise it would be possible for unauthorized users to change the configuration of the web server or to obtain configuration information which could be used for an attack.

Implementation example: Delete “read” and “write” access rights for “others.” Only grant “write” access to the user who configures the web server.

For this requirement the following threats are relevant:
• Unauthorized access or tapping of data
• Unauthorized modification of data
• Unauthorized use of services or resources
• Disruption of availability

Req 6 Default content must be removed.

Default content (examples, help files, documentation, aliases) that is provided with the standard installation must be removed.

Motivation: By using examples, information could be obtained about the installed software (version). Examples can include security vulnerabilities.

For this requirement the following threats are relevant:
• Unauthorized access or tapping of data
• Attacks motivated and facilitated by information disclosure or visible security weaknesses
Req 7 Components and modules of the Nginx web server, which are not required for operation or functionality, must not be installed.

All unnecessary web server components and modules must be removed during installation. During the default installation, components and modules are compiled which are not required for operation or functionality.

_Motivation: Enhancements, components or functions which cannot be used in the application may have security vulnerabilities that go undetected._

Implementation example: When installing Nginx all components that are required or removed must be explicitly stated during configuration. In addition, installing more modules than necessary impairs performance. Help can be accessed using the command: `./configure --help`.
The option is given of deinstalling, for instance, the autoindex module. (--without-http_autoindex_module)

`./configure without-xxx --with-xxx`


For this requirement the following threats are relevant:
- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources
- Disruption of availability
- Unnoticeable feasible attacks

Req 8 Information on the Nginx web server in error pages, which are supplied by the web server, must be removed.

Default error pages must be replaced with user-defined error pages. User-defined error pages must not include version information about the web server and the modules/enhancements used. Error messages must not include internal information such as internal server names, version numbers, error codes, etc.

_Motivation: Any information about the web server could allow conclusions to be drawn about security vulnerabilities._

Implementation example: The status information supplied by the Nginx web server must be disabled in the configuration using the command "server_tokens off". The separate error pages are also defined in the configuration:

```bash
location / {
    error_page 404 /mein_error_404.html;
    error_page 500 502 503 504 /mein_error_50x.html;
}
```

Furthermore, error messages can also be summarized here or refer to a URL. In order to conceal the web server itself as well, the name of the web server can be changed in the file "src/http/ ngx_http_header_filter_module.c". Nginx must then be recompiled.
Line:
static char ngx_http_server_string[] = "Server: nginx"
CRLF;
static char ngx_http_server_full_string[] =
"Server: " NGXINX_VER CRLF;
change in:
static char ngx_http_server_string[] = "Server: my
Web Server" CRLF;
static char ngx_http_server_full_string[] =
"Server: my Web Server" CRLF;

For this requirement the following threats are relevant:
• Disruption of availability
• Attacks motivated and facilitated by information disclosure or visible security weaknesses

Nginx webserver configuration requirements

Req 9        All web server processes must not run with system privileges.

If the process is started by a user with system privileges, a different user ID without system
privileges must be used after the start.

Motivation: If the web server process runs with administrative access rights, an attacker who
obtains control over this process would be able to control the entire system.

Implementation example: ...
• nginx.conf
  •
  user www-data nogroup;
  • User and group without system authorization to execute the web server.
  ...

For this requirement the following threats are relevant:
• Unauthorized access to the system
• Unauthorized access or tapping of data
• Unauthorized modification of data
• Unauthorized use of services or resources
• Disruption of availability
• Unnoticeable feasible attacks

Req 10       Different instances of the web server must run in separate environments.

The environments must be separated to such an extent that an environment which has been
compromised by an attacker cannot influence the other environment.

Motivation: If an attacker obtains unauthorized access to an instance of the Nginx web server,
he must be prevented from accessing other server instances as well.

Implementation example: ...
• Call environment 1 with own configuration:
  • /usr/local/sbin/nginx -c /usr/local/nginx/conf/conf1.conf
  • Configuration file 1 "conf1.conf"
  #
  user www1-data nogroup1;
  # User1 / Group1 without system authorization

events {
  • Configuration of global events
}
http {
  • HTTP-specific configuration for all virtual servers server {
    • Configuration of 1st environment
      listen 127.0.0.1:8080;
      server_name
      environment1.org;
      location /environment1 {
        root /usr/local/environment1;
        # ...
      }
  }
}

########## End environment 1
#
#
• Call environment 2 with own configuration:
  • /usr/local/sbin/nginx -c /usr/local/nginx/conf/conf2.conf
  • Configuration file 2 "conf2.conf"
#
user www2-data nogroup2;
# User2 / Group2 without system authorization

events {
  • Configuration of global events
}
http {
  • ...
  server {
    • Configuration
      environment 2 listen
      127.0.0.1:80;
      server_name
      environment2.org;
      location /environment2 {
        root /usr/local/environment2;
        • ...
      }
  }
}

########## End environment 2

For this requirement the following threats are relevant:
• Unauthorized access to the system
• Unauthorized access or tapping of data
• Unauthorized modification of data
• Unauthorized use of services or resources
• Disruption of availability

Req 11 The Nginx web server must protect itself against overload situations.

A system must feature protective mechanisms which prevent overload situations as far as possible. In particular, par-tial or complete impairment of system availability must be prevented. Examples of possible protective measures in-clude:- Restriction of the main memory available for each application- Restriction of the maximum sessions of a web application- Stipulation of the maximum size of a data record- Restriction of CPU resources per process- Prioritization of
processes - Restriction of the number or the size of transactions of a user or from an IP address over a certain period.

Motivation: Attackers often try to bring about a system overload situation by means of denial-of-service attacks.
If such an attack is successful, this may impair the availability and also integrity of parts of or even the entire system.

Implementation example:
...
• nginx.conf
  
  user www-data nogroup; # user
  / group worker_processes 2;
  • 2 processors are used.
  • A configuration with "auto" should not be used, as here the number of installed processors in the system is used.
  
  • To prevent buffer overflows, the following default values must be reduced. It should be noted that the web server also continues to execute all functions with the respective application.
    client_max_body_size 1k;
    large_client_header_buffer_size 4 8k;
    client_body_buffer_size 1k;
    client_header_buffer_size 1k; events {
      worker_connections 1024;
      # Restricts the number of simultaneously opened connections of a "worker_process"
    }
  ...

For this requirement the following threats are relevant:
• Disruption of availability

<table>
<thead>
<tr>
<th>Req 12</th>
<th>Prevent long runtimes of web server queries.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Long runtimes of web server queries by clients should be prevented with the aid of defined timeouts.</td>
<td></td>
</tr>
</tbody>
</table>

Motivation: The web server's performance can be substantially impaired through long timeouts. This may occur deliberately through an attack. In order to prevent such attacks, fixed timeout periods must be defined in the configuration. Nginx offers a timeout directive in this respect virtually for every function.

Implementation example:
...
  # nginx.conf
  #
  client_body_timeout 8;
  client_header_timeout 8;
  keepalive_timeout 5
  5; send_timeout 8;

Since the used directives are deployed individually for each web server, this can only serve as an example and apply as a reference.
Other directives: http://nginx.org/en/docs/dirindex.html

...
WebDAV

Req 16  If WebDAV is used for writing files, access must not be granted without successful authentication.

Motivation: WebDav makes it possible to update content online which has been made available by the web server. This function could therefore be misused to change website content.

For this requirement the following threats are relevant:
• Unauthorized modification of data

Req 17  If WebDAV is used, access rights must be configured restrictively.

Access rights to all files accessible by WebDAV must be configured as restrictively as possible. Additionally, if Web-DAV is used, WebDAV access must be restricted to the directories required.

Motivation: WebDav makes it possible to update content online which has been made available by the web server. This function could therefore be misused to change website content.

For this requirement the following threats are relevant:
• Unauthorized access or tapping of data
• Unauthorized modification of data
FastCGI

Req 18  The usage of FastCGI requires a separate configuration.

In order to map dynamic content such as PHP via the Nginx web server, the client must be connected to the interpreter (e.g. PHP) via separate middleware. The "fastCGI" module is ideal for this purpose. The configuration parameters are stored in a separate file.

The advantage of this middleware relates, among other things, to rights assignment. Thus it is possible to control access to the web server and the rights for executing program code. It is, in turn, possible to grant different execution rights to different clients or locations (IP addresses).

Motivation: Without this explicit rights assignment, it would be possible to store a cgi or PHP file in an upload directory and to execute the file by calling it using the rights of the Nginx web server. The example provides another possibility. Here a check is still being run to determine whether the file to be handed over comes from an upload directory. Thus no handover to the interpreter would take place at this point.

Implementation example: ...

# nginx.conf
#
location ~ \.php$ {  
include /etc/nginx/fastcgi_params; if
($uri !~ "^/upload") {
fastcgi_pass 127.0.0.1:9000;
}
}
...

For this requirement the following threats are relevant:
• Unauthorized access to the system
• Unauthorized access or tapping of data
• Unauthorized modification of data
• Unnoticeable feasible attacks

HTTPS

Req 19  The TLS protocol must be used for encryption with HTTPS.

SSL must be regarded as outdated and must therefore not be enabled or must be disabled. Instead, the TLS protocol must be used.

Motivation: A host of vulnerabilities are known especially with regard to SSLv2, which makes usage impossible from a security perspective. TLS has now been established as a further development of SSL for many years so that there is no reason to continue using SSL.

Implementation example: The current version of Nginx uses "ssl_protocols TLSv1 TLSv1.1 TLSv1.2" as default.

...  
# nginx.conf
#
worker_process es 2; http {
ssl_session_cache shared:SSL:5m;
ssl_session_timeout 5m;
}
server {  
listen 443 ssl;  
server_name www.server1.com;  
ssl_certificate www.server1.com.crt;  
ssl_certificate_key  
www.server1.com.key; ssl_protocols  
TLSv1 TLSv1.1 TLSv1.2;  
...  
ssl_prefer_server_ciphers on; 

For this requirement the following threats are relevant:  
• Unauthorized access or tapping of data  
• Unauthorized modification of data

---

Req 20 The web server must be configured in such a way that the use of the latest version of the 
TLS protocol is enabled.  
In particular, the web server must be configured for the use of TLS 1.2.  
Motivation: The latest version of the protocol offers the best possible protection and contains fixes to 
known vulnerabilities in previous versions of the protocol.  
For this requirement the following threats are relevant:  
• Unauthorized access or tapping of data  
• Unauthorized modification of data

---

Req 21 The TLS configuration must not use any unsecure cipher suites.  
Cipher suites with one of the following characteristics are not secure
• no server authentication,  
• no encryption,  
• encryption with DES, RC4 or any other algorithm with a key length of less than 128 bits  
• use of MD5 as hash function

Motivation: Cipher suites known to be unsecure do not offer sufficient protection.  
For this requirement the following threats are relevant:  
• Unauthorized access or tapping of data  
• Unauthorized modification of data

---

Req 22 The TLS configuration must provide that the cipher suite considered most secure is being 
chosen with highest priority.  
A cipher suite contains the definition of four algorithms. These are used for key exchange,  
authentication, encryption and as a hash function. General guidelines for the prioritization are
• For the key exchange the Diffie-Hellman method must be preferred because it offers perfect  
forward secrecy. Cipher suites using the Diffie-Hellman method usually may be identified by 
the strings DHE or ECDHE. ECDHE has higher priority than DHE.  
• For encryption the Advanced Encryption Standard (AES) or Camellia with a key length as  
big as possible has to be used
• As a hash function SHA-2 has to be used. This function usually may be identified by the string SHA followed by a number (256, 384 or 512). Warning: if the string SHA is not followed by a number this identifies the SHA-1 function which is significantly less secure.

Motivation: When a TLS connection is being established a cipher suite is selected based on the cipher suites available both on client and on server side. In order to ensure a high compatibility to all kinds of client systems the web server must not only allow for the cipher suites considered most secure. To make sure that nevertheless for each client the best possible cipher suite is selected and thus the connection is best protected the configuration must contain an according prioritization.

For this requirement the following threats are relevant:
• Unauthorized access or tapping of data
• Unauthorized modification of data

Req 23 Certificates must be issued by a certification authority whose certificates are recognized by the commonly used web browsers.

For critical applications that can be used via the Internet, use of an extended validation certificate (EV certificate) is recommended.

Motivation: Only if the certificate authority (CA) is contained in the CA list of the browser being used the browser can verify the authenticity of the server or web application

Stricter issuing criteria apply to EV certificates. If an EV certificate is used, this is visualized in the browser. Even if EV certificates do not improve security, their use increases the trustworthiness of the server for the user.

For this requirement the following threats are relevant:
• Attacks motivated and facilitated by information disclosure or visible security weaknesses

Req 24 Certificates must lose their validity after a maximum of 36 months.

Motivation: The methods used for analysing and breaking cryptographic processes are improved continuously. Therefore the security of the certificates can be ensured for a limited period only. But, according to a general estimation, the security of the certificates is ensured for the required validity period of three years, if an appropriate key length is used.

For this requirement the following threats are relevant:
• Unauthorized access or tapping of data
• Unauthorized modification of data

Req 25 Certificates must have a key length of at least 2048 bits when using RSA or 256 bits when using ECC.

Motivation: In order to guarantee the security of certificates over the validity period, the cryptographic keys must have an appropriate length. According to a general estimation, a key length of 2048 bits provides sufficient protection for the next years. For ECC algorithms, shorter key lengths already provide the same level of security.

For this requirement the following threats are relevant:
• Unauthorized access or tapping of data
• Unauthorized modification of data

Nginx webserver logging

Req 26 Access to the web server must be logged.

The logging format must include the following information:
• Access timestamp
• Source (IP address)
• User (if known)
• URL
• Status code of the response from the web server

The applicable statutory, rate-plan and commercial provisions must be taken into account during logging. These provisions state, among other things, that event logging must only take place for a specific purpose. The logging of events to monitor employee work is not permitted.

Motivation: To analyze security incidents, it is very important to have basic information on how an attack was conducted. Since a web server constitutes an external interface, certain information exists solely on the web server even with an attack on a downstream system and must therefore be logged on the web server.

Implementation example:

```bash
...
# # nginx.conf
#
http {
    log_format compression '$remote_addr - $remote_user
    [$time_local] "$request" $status $body_bytes_sent
    "$http_referer" "$http_user_agent" "$gzip_ratio"';
    server {
        gzip on;
        access_log /var/log/nginx-access.log compression;
        # only log critical errors
        error_log /var/log/nginx/error.log crit;
    ...
    }
}
```

For this requirement the following threats are relevant:
• Denial of executed activities
• Unnoticeable feasible attacks

Configuration requirements

---

Req 27 The web server must be robust against overload situations.

A web server must provide security measures to deal with overload situations. In particular, partial or complete impairment of web server availability must be avoided. Potential protective measures include:

• Restricting the maximum number of HTTP sessions per IP address
• Defining the maximum size of a HTTP request
• Defining a timeout for HTTP requests

Restrictions must be implemented in consideration of the application to be protected and its characteristics. The following values may be used as a guideline:
If the web server will not be used for uploads:

- Maximum number of HTTP sessions per IP address: 50
- Maximum size of a HTTP request: 20000 bytes
- Timeout for HTTP requests: 30 seconds

If the web server may also be used for uploads:

- Maximum number of HTTP sessions per IP address: 50
- Maximum size of a HTTP request: 10000000 bytes or, if known, maximum size of expected upload
- Timeout for HTTP requests: 60 seconds or, if known, time to complete maximum upload

Motivation: Attackers often try to bring a web server into an overload situation by using denial-of-service (DoS) attacks. If such an attack is successful the web server’s availability or integrity may be impaired.

For this requirement the following threats are relevant:

- Disruption of availability