Security requirement

Web Service Gateway

Deutsche Telekom Group

Version	6.0
Date	Dec 1, 2023
Status	Released

Publication Details

Published by
Deutsche Telekom AG
Vorstandsbereich Technology & Innovation
Chief Security Officer

Reuterstrasse 65, 53315 Bonn
Germany

File name	Document number 3.13	Document type Security requirement
Version 6.0	State Dec 1, 2023	Status Released
Contact Telekom Security psa.telekom.de	Validity Dec 1, 2023 - Nov 30, 2028	Released by Stefan Pütz, Leiter SEC-T-TST

Summary
This document was created on the basis of the specifications from the security guidelines valid in the Group and is aimed at all systems that provide gateways for Web service services.

Table of Contents

1. 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 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.

If compliance with the described requirements can not be achieved or is only partially feasible in individual cases, a risk assessment must be carried out together with a Security- and/or Data Privacy Expert (in accordance with the relevant requirement) and possible alternative protective measures agreed.

2. General

2.1. System hardening

Req 1 Unnecessary services must be disabled.

After the installation of systems and software products, supplier-preset, local or network-accessible services are often active that are not required for the operation and functionality of the specific system in the intended operating environment.

However, in principle only the services actually required may be active on a system.

Accordingly, all services that are not required on a system must be completely disabled immediately after installation. It must be ensured that these services remain disabled even after the system is restarted.

Motivation: Active services that are not required unnecessarily increase the attack surface of a system and, as a direct consequence, the risk of a successful compromise. This risk can be further increased if - as is often observed with services that are not required - a targeted examination and optimization of the configuration with regard to security does not take place sufficiently.

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

For this requirement the following warranty objectives are relevant:

ID: 3.01-5/7.0

Req 2 Only required software may be used on the system.

In the installation routines for software provided by the supplier, individual components of the software are often preselected as standard installations, which are not necessary for the operation and function of a specific system. This also includes parts of software that are installed as application examples (e.g. default web pages, sample databases, test data), but are typically not used afterwards.

Such components must be specifically deselected (not installed) during the installation of the system or - if deselection during installation is not possible - removed immediately afterwards.

In principle, no software may be used that is not required for the operation, maintenance or function of the system.

Motivation: Vulnerabilities in a system's software are gateways for attackers. By uninstalling unnecessary components, the potential attack surfaces can be significantly reduced.

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

For this requirement the following warranty objectives are relevant:

ID: 3.01-3/7.0

Req 3 The software used must be obtained from trusted sources and checked for integrity.

The software used on the system must be obtained from trusted sources and checked for integrity before installation.

This requirement applies to all types of software:

- Firmware and microcode for hardware components
- Operating systems
- Software Libraries
- Application Software
- Pre-integrated application solutions, such as software appliances or containers

as well as other software that may be used.

Trusted Sources

Trusted sources are generally considered to be:

- the official distribution and supply channels of the supplier
- third party distributors, provided they are authorized by the supplier and are a legitimate part of the supplier´s delivery channels
- internet downloads, if they are made from official provisioning servers of the supplier or authorized distributors
  (1) If the provisioning server offers various forms of downloads, those protected by encryption or cryptographic signatures must be preferred to those without such protection.
  (2) If the provisioning server secures the transport layer using cryptographic protocols (e.g. https, sftp), the associated server certificates or server keys/fingerprints must be validated with each download to confirm the identity of the provisioning server; if validation fails, the download must be cancelled and the provisioning server has to be considered an untrusted source.

Integrity Check

The integrity check is intended to ensure that the received software is free of manipulation and malware infection. If available, the mechanisms implemented by the supplier must be used for checking.
Valid mechanisms are:

- physical seals or permanently applied certificates of authenticity (if the software is provided on physical media)
- comparison of cryptographic hash values (e.g. SHA256, SHA512) of the received software against target values, which the supplier provides separately
- verification of cryptographic signatures (e.g. GPG, certificates) with which the supplier provides its software

In addition, a check of the software using an anti-virus or anti-malware scanner is recommended (if the vendor has not implemented any of the aforementioned integrity protection mechanisms for its software, this verification is mandatory).

Extended integrity checking when pulling software from public registries

Public registries allow developers to make any of their own software projects available for use. The range includes projects from well-known companies with controlled development processes, as well as from smaller providers or amateur developers.
Examples of such registries are:

- Code registries (e.g. GitHub, Bitbucket, SourceForge, Python Package Index)
- Container registries (e.g. Docker Hub)

Software from public registries must undergo an extended integrity check before deployment.
In addition to the integrity check components described in the previous section, the extended check is intended to explicitly ensure that the software actually performs its function as described, does not contain inherent security risks such as intentionally implemented malware features, and is not affected by known security vulnerabilities. If the software has direct dependencies on third-party software projects (dependencies are very typical in open source software), which must also be obtained and installed for the use of the software, these must be included in the extended integrity check.

Suitable methods for an extended integrity check can be, for example:

- Strict validation of project/package names (avoidance of confusion with deliberately imitated malicious software projects)
- dynamic code analysis / structured functional checks in a test environment
- static code analysis using a linter (e.g. Splint, JSLint, pylint)
- Examination using a security vulnerability scanner (e.g. Qualys, Nessus)
- Examination using a container security scanner (e.g. JFrog Xray, Harbor, Clair, Docker Scan)
- Examination using an SCA (Software Composition Analysis) tool or dependency scanner (e.g. OWASP Dependency Check, Snyk)

The test methods must be selected and appropriately combined according to the exact form of software delivery (source code, binaries/artifacts, containers).

Motivation: Software supply chains contain various attack vectors. An attacker can start at various points to manipulate software or introduce his own routines and damage or control the target environment in which the software is subsequently used. The attack can occur on the transport or transmission path or on the provisioning source itself. Accordingly, an attack is facilitated if software is not obtained from official and controlled sources or if an integrity check is omitted.
There is a particular risk for software obtained from public registries, as these are open to anyone for the provision of software projects. Perfidious attack methods are known, in which the attacker first provides completely inconspicuous, functional software for a while and as soon as it has established itself and found a certain spread, deliberately hidden malicious code is integrated in future versions. Other methods rely on similar-sounding project names for widely used existing projects or overruling version numbers to inject manipulated software into any solutions based on them.

Implementation example: Obtain the software via the official delivery channels of the supplier. Upon receipt of the software, immediately check for integrity using cryptographic checksums, as provided by the supplier, as well as scan for any infections by known malware using anti-malware / anti-virus scanners. Storage of the tested software on an internal, protected file storage and further use (e.g. rollout to the target systems) only from there.

For this requirement the following threats are relevant:
- Unauthorized modification of data
- Unnoticeable feasible attacks
- Attacks motivated and facilitated by information disclosure or visible security weaknesses

For this requirement the following warranty objectives are relevant:

ID: 3.01-2/7.0

Req 4	Features that are not required in the software and hardware used must be deactivated.

During the initial installation of software, features may have been activated by default that are not necessary for the operation and functionality of the specific system. Features are usually an integral part of the software that cannot be deleted or uninstalled individually.

Such features must be disabled immediately after the initial installation through the software's configuration settings, so that they remain permanently disabled even after the system is rebooted.

Even before delivery or during initial commissioning, features may have been activated by default in the hardware that are not required for the purpose of the specific system. Such functions, for example unnecessary interfaces, must also be permanently deactivated immediately after initial commissioning.

Motivation: A system's hardware or software often contains enabled features that are not being used. Such features can be an unnecessary target for manipulation. Furthermore, there is a potential that unauthorized access to areas or data of the system can be created.

Implementation example: [Example 1]
Deactivation of debugging functions in the software that are used in the event of fault analysis, but do not have to be active during normal operation.

[Example 2]
Disabling unused network interfaces of a server.

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

For this requirement the following warranty objectives are relevant:

ID: 3.01-4/7.0

Req 5	Outputs and messages must not disclose information on internal structures of the system.

Information about the internal structures of a system, including the components used there, and corresponding implementation details are generally considered to be in need of protection.

In general, this concerns information on
- Product names and product identifiers of implemented system components
- Operating systems, middleware, backend software, software libraries and internal applications as well as their software versions
- installed service packs, patches, hotfixes
- Serial numbers of components as well as stored product licenses
- Database Structures

Typical examples of outputs and messages in which disclosure of such system information can potentially occur:
- Login windows and dialogs
- Error messages
- Status messages
- Banners of active network services
- System logs and log files
- Debug logs, stack traces

As far as it is technically feasible without impairing the function and operation of the system, the output of affected system information must always be deactivated.

Access to affected system information must only be possible for authorized users of the system. As a rule, this circle of authorized users is to be limited to administrators and operators of the system. Access for authorized monitoring and inventory systems within the operating environment is also permitted.

A permissible exception to these restrictions exists for specific individual system information, the disclosure of which is technically mandatory for the intended function of the system in conjunction with third-party systems; For example, the presentation of supported protocols and their versions during the initial parameter negotiation in session setups between a client and a server.

Motivation: Information about the internal structures of a system can be used by an attacker to prepare attacks on the system extremely effective. For example, an attacker can derive any known vulnerabilities of a product from the software version in order to exploit them specifically during the attack on the system.

Implementation example: [Example 1]
Deactivation of the display of the product name and the installed version of a Web server in its delivered error web pages.

[Example 2]

Removal of the product name and the corresponding version string from the login banner of a deployed SSH server.

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

For this requirement the following warranty objectives are relevant:

ID: 3.01-9/7.0

2.2. System update

Req 6	Software and hardware of the system must be covered by security vulnerability support from the supplier.

Only software and hardware products for which there is security vulnerability support by the supplier may be used in a system.

Such support must include that the supplier
- continuously monitors and analyzes the product for whether it has been affected by security vulnerabilities,
- informs immediately about the type, severity and exploitability of vulnerabilities discovered in the product
- timely provides product updates or effective workarounds to remedy the vulnerabilities.

The security vulnerability support must be in place for the entire period in which the affected product remains in use.

Support phases with limited scope of services
Many suppliers optionally offer time-extended support for their products, which goes beyond the support phase intended for the general market, but is often associated with limitations. Some suppliers define their support fundamentally in increments, which may include limitations even during the final phase before the absolute end date of regular support.
If a product is used within support phases that are subject to limitations, it must be explicitly ensured that these restrictions do not affect the availability of security vulnerability support.

Open Source Software and Hardware
Open Source products are often developed by free organizations or communities; accordingly, contractually agreed security vulnerability support may not be available. In principle, it must also be ensured here that the organization/community (or a third party officially commissioned by them) operates a comprehensive security vulnerability management for the affected product, which meets the above-mentioned criteria and is considered to be reliably established.

Motivation: Hardware and software products for which there is no comprehensive security vulnerability support from the supplier pose a risk. This means that a product is not adequately checked to determine whether it is affected by further developed forms of attack or newly discovered vulnerabilities in technical implementations. Likewise, if there are existing security vulnerabilities in a product, no improvements (e.g. updates, patches) are provided. This results in a system whose weak points cannot be remedied, so that they remain exploitable by an attacker in order to compromise the system or to adversely affect 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
- Disruption of availability
- Denial of executed activities
- Unnoticeable feasible attacks
- Attacks motivated and facilitated by information disclosure or visible security weaknesses

For this requirement the following warranty objectives are relevant:

ID: 3.01-1/7.0

Req 7	Known vulnerabilities in the software or hardware of the system must be fixed or protected against misuse.

Known vulnerabilities in software and hardware components must be fixed by installing available system updates from the supplier (e.g. patches, updates/upgrades). Alternatively, the use of workarounds (acute solutions that do not fix the vulnerability, but effectively prevent exploitation) is permissible. Workarounds should only be used temporarily and should be replaced by a regular system update as soon as possible in order to completely close the vulnerabilities.

Components that contain known, unrecoverable vulnerabilities must not be used in a system.

The treatment of newly discovered vulnerabilities must also be continuously ensured for the entire deployment phase of the system and implemented in the continuous operating processes of security patch management.

Motivation: The use of components without fixing contained vulnerabilities significantly increases the risk of a successful compromise. The attacker is additionally favored by the fact that, as a rule, not only detailed information on vulnerabilities that have already become known is openly available, but often also already adapted attack tools that facilitate active exploitation.

Implementation example: Following the initial installation of an operating system from an official installation medium, all currently available patches and security updates are installed.

Additional information:
The primary sources of known vulnerabilities in software/hardware are lists in the release notes as well as the security advisories from the official reporting channels of the supplier or independent CERTs. In particular, the reporting channels are sensibly integrated into continuous processes of security patch management for a system, so that newly discovered vulnerabilities can be registered promptly and led into operational remedial measures.
As a complementary measure to the detection of potentially still contained types of vulnerabilities that have in principle already become known, targeted vulnerability investigations of the system can be carried out. Particularly specialized tools such as automated vulnerability scanners are suitable for this purpose. Examples include: Tenable Nessus, Qualys Scanner Appliance.

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
- Denial of executed activities
- Unnoticeable feasible attacks
- Attacks motivated and facilitated by information disclosure or visible security weaknesses

For this requirement the following warranty objectives are relevant:

ID: 3.01-10/7.0

3. Access Control

3.1. Consumers and providers

Req 8	The Web Service Gateway must uniquely authenticate and authorize consumers and providers.

The Web services gateway must ensure that the consumer is authorized to access this Web service and to receive this data or use these functions.
The web service gateway must also authenticate the providers of a web service.

Motivation: Only proper authentication can ensure that the information provided is read exclusively by the designated user or generated by the authentic provider.

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

For this requirement the following warranty objectives are relevant:

ID: 3.13-8/6.0

Req 9	The Web service gateway must authenticate itself to the Web service provider.

The provider of a web service must authenticate the web service gateway. To do this, the Web service gateway must provide suitable authentication features.

Motivation: In order for the Web service provider to be sure that the communication is running via the Web service gateway and that all security functions are guaranteed, the Web service gateway must provide the Web service provider with suitable authentication features.

Implementation example: One common method is gateway access tokens or a transmitted client certificate (mTLS).

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

For this requirement the following warranty objectives are relevant:

ID: 3.13-9/6.0

Req 10	The mechanism for authentication and authorization must be based on strong cryptographic algorithms / frameworks.

Strong cryptographic frameworks/algorithms in this sense include XML signatures, TLS client certificates, or tokens with suitable algorithms such as ECDSA/RSA signatures.

Motivation: Weak algorithms can be compromised by attackers and identities can be spoofed.

Implementation example: Certificates according to the certificate requirements for HTTPS.
Examples of tokens include JW, oAuth, and STS tokens, if transmitted over TLS.

Tokens:

- Only signature tokens based on asymmetric cryptography (ES,RS).

- Lifetimes as short as possible

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

For this requirement the following warranty objectives are relevant:

ID: 3.13-10/6.0

Req 11	The Web service gateway must validate consumer access to providers using downstream service repositories or authorizers.

Consumers of a Web service gateway must be checked for their authorization. This information is usually stored in a service directory (service repository or authorizer).

Motivation: To prevent overloads and malfunctions in the perimeter layer that affect access control, components that grant or verify permissions should be kept separate from the Web Service Gateway.

Implementation example: One implementation example is a keycloak that issues tokens based on the role and authorization concept contained in the keycloak and that is processed by the gateway.

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

For this requirement the following warranty objectives are relevant:

ID: 3.13-11/6.0

3.2. Components of the Web Service Gateway

Req 12	Predefined user accounts that are not required must be deleted or at least disabled.

On many systems, there are predefined but unused user accounts (e.g. "guest") after the initial installation.

These predefined user accounts must be deleted or at least disabled immedately after the initial installation; if these measures are not feasible, the corresponding user accounts must be blocked for remote access. In any case, disabled or blocked user accounts must also be provided with an authentication attribute (e.g. a password or an SSH key) so that unauthorized use of such a user account is prevented in the event of a misconfiguration.

Excempt from the requirement to delete or disable predefined user accounts are user accounts that are used exclusively for internal use on the corresponding system and that are required for the functionality of one or more applications of the system. Even for such a user account, it must be ensured that remote access or local login is not possible and that a user of the system cannot misuse such a user account.

Motivation: User accounts that are predefined by default in a product are typically common knowledge and can be targeted by an attacker for brute force and dictionary attacks. If these user accounts are not needed in a specific system, their existence represents an unnecessary attack surface. A particular risk is posed by predefined user accounts that are preconfigured without a password or with a well-known standard password. Such user accounts can be misused directly by an attacker if their security hardening was missed due to the unplanned use in the specific system.

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

For this requirement the following warranty objectives are relevant:

ID: 3.01-7/7.0

Req 13 User accounts must ensure the unique identification of the user.

Users must be identified unambiguously by the system.

This can typically be reached by using a unique user account per user.

So-called group accounts, which are characterized by the fact that they are used jointly by several people, must not be used. This also applies without restriction to privileged user accounts. Most systems initially have only a single user account with administrative privileges after the basic installation. If the system is to be administered by several persons, each of these persons must use a personal, individual user account to which appropriate administrative authorizations or roles are assigned

A special feature are so named technical user accounts. These are used for the authentication and authorization of systems among themselves or of applications on a system and can therefore not be assigned to a specific person. Such user accounts must be assigned on a per system or per application basis. In this connection, it has to be ensured that these user accounts can't be misused.
Ways to prevent misuse of such user accounts by individuals include:

- Configuration of a password that meets the security requirements and is known to as few administrators as possible.
- Configuring the user account that only a local use is possible and a interactive login isn't possible.
- Use of a technique for authentication of the specific user account with public and private key or certificates.
- Limiting the access over the network to legitimate systems.

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

Motivation: Unambiguous user identification is mandatory to assign a user permissions that are necessary to perform the required 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 access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources
- Denial of executed activities

For this requirement the following warranty objectives are relevant:

ID: 3.01-22/7.0

Req 14 Predefined authentication attributes must be changed.

After the takeover or initial installation of a system, there are usually predefined authentication attributes (e.g. passwords, SSH keys, SSL/TLS Certificates) in the system, as assigned by manufacturers, developers, suppliers or automated installation routines.

Such predefined authentication attributes must be changed to new, individual values immediately after the takeover or installation of the system.

Motivation: Values predefined by third parties in authentication attributes cannot be trusted because they do not represent a controlled secret. Affected authentication attributes can be misused by unauthorized persons to access and compromise systems. This risk is significantly increased if commonly known default values are used for authentication attributes (e.g. a default password for the administrator user account in a particular software product).

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

For this requirement the following warranty objectives are relevant:

ID: 3.01-8/7.0

Req 15	The permissions for users and applications must be limited to the extent necessary to fulfill their tasks.

The permissions on a system must be restricted to such an extent that a user can only access data and use functions that he needs in the context of his work. Appropriate permissions must also be assigned for access to files that are part of the operating system or applications or that are generated by the same (e.g. configuration and logging files).

In addition to access to data, applications and their components must also be executed with the lowest possible permissions. Applications should not be run with administrator or system privileges.

Motivation: If a user is granted too far-reaching permissions on a system, he can access data and applications to an extent that is not necessary for the fulfillment of the assigned tasks. This creates an unnecessarily increased risk in the event of abuse, in particular if the user or his user account is compromised by an attacker.
Applications with too far-reaching permissions can be misused by an attacker to gain or expand unauthorized access to sensitive data and system areas.

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

For this requirement the following warranty objectives are relevant:

ID: 3.01-23/7.0

Req 16	User accounts must be protected with a minimum of two authentication attributes.

To get a higher protection level it is necessary to use two or more independent authentication attributes.
This approach is commonly referred to as MFA (Multi-Factor Authentication).
A specific form is 2FA (2-Factor Authentication), which combines exactly two authentication features.

Motivation: User 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 user account to get access to wide parts of the system and stored data.

Implementation example: Very popular is 2FA in a variant consisting of an attribute that the user knows and an attribute that the user possesses.
Examples of such a 2FA are:

- Smartcard (e.g. MyCard) with PIN
- Private key with passphrase
- Hardware Token for one-time passwords

In companies of Deutsche Telekom group where the MyCard or a comparable smartcard is available this solution should be preferred.

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

For this requirement the following warranty objectives are relevant:

ID: 3.13-16/6.0

Req 17	Administrative access to the Web Service Gateway components must be provided via jump hosts or dedicated networks

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

For this requirement the following warranty objectives are relevant:

ID: 3.13-17/6.0

4. Confidentiality

4.1. Transport encryption

Req 18 TLS version 1.2 or 1.3 must be used.

User roles: Operation, Development, Integration

TLS (Transport Layer Security) is a protocol for the secure transmission of information over TCP/IP based connections and is the successor of SSL (Secure Socket Layer). TLS ensures the confidentiality, integrity and authenticity of the information or the communication partners.

TLS in version 1.2 [RFC 5246] and version 1.3 [RFC 8446] provides cipher suites with Authenticated Encryption Associated Data (AEAD). AEAD ensures the confidentiality as well as the integrity and authenticity of the transmitted information.

References:
[RFC 5246] T. Dierks, E. Rescorla: RFC 5246: The Transport Layer Security (TLS) Protocol Version 1.2, 2008
[RFC 8446] E. Rescorla: RFC 8446: The Transport Layer Security (TLS) Protocol Version 1.3, 2018

Motivation: The current version of TLS fixes previous known security vulnerabilities and attack surfaces on the TLS protocol handshake.

Implementation example: OpenSSL> protocol = tlsv1_3

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
- Denial of executed activities
- Unnoticeable feasible attacks
- Attacks motivated and facilitated by information disclosure or visible security weaknesses

For this requirement the following warranty objectives are relevant:

ID: 3.50-40/7.0

Req 19 For TLS, Diffie Hellman groups according to the table below must be used.

User roles: Operation, Development, Integration

The Diffie Hellman groups is used for key exchange with Perfect Forward Secrecy (PFS). Generally, a distinction is made between elliptic curve groups and finite field groups (mod p).

The following table contains the allowed Diffie Hellman groups.
Allowed Diffie Helman groups for use in TLS:

Diffie Hellman group	**IANA-No.**	**Referenzspezifikation**
brainpoolP512r1	33	RFC 7027
secp521r1	25	RFC 8422
x448	30	RFC 8422
brainpoolP384r1	27	RFC 7027
secp384r1	24	RFC 8422

brainpoolP256r1	26	RFC 7027
secp256r1	23	RFC 8422
x25519	29	RFC 8422
ffdhe4096	258	RFC 7919
ffdhe3072	257	RFC 7919

Remark on x448 and x25519:
x448 und x25519 are not (explicitly) recommended by BSI, but no weaknesses are known so far, and therefore they are to be classified as secure.

Remark on group 256:
Diffie Hellman group 256 (IANA-No.256) has a key length of 2048 bit [1] [2] and may only be used in legacy systems until the end of the **year 2025 [2]**. The group must be substituted by a stronger method (according to the enumeration above).

Remark on groups 256, 258 and 257:
Those groups are more vulnerable against the DHeater (DoS) attacks on server side than elliptic curve DH groups. Therefore, the brainpool and NIST (secp) groups should be preferred.

References:
[1] Bundesamt für Sicherheit in der Informationstechnik, TR-02102-2, Version 2023-01
[2] SOG-IS Crypto Evaluation Scheme: Agreed Cryptographic Mechanisms, v1.3, February 2023

Motivation: Standardized Diffie Hellman groups use secure parameters and speed up the key exchange.

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
- Denial of executed activities
- Unnoticeable feasible attacks
- Attacks motivated and facilitated by information disclosure or visible security weaknesses

For this requirement the following warranty objectives are relevant:

ID: 3.50-42/7.0

Req 20 Only Perfect Forward Secrecy (PFS) TLS-cipher suites must be used according to the tables below.

User roles: Operation, Development, Integration

Cipher suites specify the cryptographic methods of a connection.

Perfect Forward Secrecy (short PFS, also Forward Secrecy) means that transmitted information cannot be decrypted afterwards, even if the long-term key of the communication partners is known.

In TLS v1.2 cipher suites are defined as follows: *TLS_AKE_WITH_Enc_Hash.*
Following, the meaning of the individual components is explained:

- *AKE (Authenticated Key Exchange)*: Key agreement mechanism with authentication for the handshake protocol.
- *Enc (Encryption)*: Encryption algorithm with mode of operation for the record protocol.
- *Hash*:Hash algorithm for HMAC used for key derivation. If*Enc*is not an AEAD encryption mechanism, HMAC is also used for integrity protection.

The following table lists the allowed cipher suites with PFS in TLS v1.2 as well as the reference specifications. The design philosophy of TLS v1.2 was followed, which is why the table contains only AEAD constructions.
Allowed cipher suites with PFS in TLS v1.2:

Priority	**Cipher Suite**	**Reference specification**
HIGH	TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384	RFC 5289
HIGH	TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384	RFC 5289
LOW	TLS_DHE_DSS_ WITH_AES_256_GCM_SHA384	RFC 5288
LOW	TLS_DHE_RSA_ WITH_AES_256_GCM_SHA384	RFC 5288
HIGH	TLS_ECDHE_ECDSA_WITH_CHACHA20POLY1305_SHA256	RFC 7905
HIGH	TLS_ECDHE_RSA_WITH_CHACHA20POLY1305_SHA256	RFC 7905
LOW	TLS_DHE_RSA_WITH_CHACHA20POLY1305_SHA256	RFC 7905
HIGH	TLS_ECDHE_ECDSA_WITH_AES_256_CCM	RFC 7251
LOW	TLS_DHE_RSA_ WITH_AES_256_CCM	RFC 6655
HIGH	TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256	RFC 5289
HIGH	TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256	RFC 5289
LOW	TLS_DHE_DSS_ WITH_AES_128_GCM_SHA256	RFC 5288
LOW	TLS_DHE_RSA_ WITH_AES_128_GCM_SHA256	RFC 5288
HIGH	TLS_ECDHE_ECDSA_WITH_AES_128_CCM	RFC 7251
LOW	TLS_DHE_RSA_ WITH_AES_128_CCM	RFC 6655

Furthermore, in legacy systems the cipher suites of the following table are allowed.
Additional cipher suites with PFS in TLS v1.2 with AES-CBC:

Priority	**Cipher Suite**	**Reference specification**
HIGH	TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384	RFC 5289

HIGH	TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384	RFC 5289
LOW	TLS_DHE_DSS_WITH_AES_256_CBC_SHA256	RFC 5246
LOW	TLS_DHE_RSA_WITH_AES_256_CBC_SHA256	RFC 5246
HIGH	TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256	RFC 5289
HIGH	TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256	RFC 5289
LOW	TLS_DHE_DSS_WITH_AES_128_CBC_SHA256	RFC 5246
LOW	TLS_DHE_RSA_WITH_AES_128_CBC_SHA256	RFC 5246

Remark on the cipher suites for TLS v1.2:
The table entries are sorted by the symmetric encryption mechanism (Enc). For the authenticated key agreement methods (AKE), mechanism based on elliptic curves (ECDHE_ECDSA) are preferred. DHE (discrete logarithm) key establishment ciphers are more vulnerable against DoS (DHeater) than ECDHE, thus ECDHE should be preferred. The "Priority" column defines which cipher suites are preferred, i.e. cipher suites with a priority of "HIGH" are preferable to those with "LOW".

In TLS v1.3 cipher suites are defined as follows: *TLS_AEAD_Hash.*
Following, the meaning of the individual components is explained:

- AEAD: Authenticated encryption mechanism for the record protocol.
- Hash: Hash algorithm for HMAC and HKDF in the handshake protocol.

The following table lists the allowed cipher suites with PFS in TLS v1.3 as well as the reference specifications.
Allowed cipher suites with PFS in TLS v1.3:

Cipher suites	**Reference specification**
TLS_AES_256_GCM_SHA384	RFC 8446
TLS_CHACHA20_POLY1305_SHA256	RFC 8446
TLS_AES_128_GCM_SHA256	RFC 8446
TLS_AES_128_CCM_SHA256	RFC 8446

Any cipher suites specified in the future that correspond to the requirements defined in this document can be used as well. For example, this applies for cipher suites that use a hash function from the SHA-3 family.

References:
[1] Bundesamt für Sicherheit in der Informationstechnik, TR-02102-2, version 2023-01
[2] https://ciphersuite.info/cs/?security=secure&sort=asc
[3] https://ciphersuite.info/cs/?singlepage=true&security=recommended#

Motivation: The usage of modern cipher suites with Perfect Forward Secrecy protects the transport security in TLS.

Implementation example: TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256

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
- Denial of executed activities
- Unnoticeable feasible attacks
- Attacks motivated and facilitated by information disclosure or visible security weaknesses

For this requirement the following warranty objectives are relevant:

ID: 3.50-41/7.0

Req 21	For TLS, digital certificates from an appropriate certification authority with a sufficient key length and limited validity must be used.

User roles: Operation, Development, Integration

In TLS, digital certificates are used during the TLS handshake. TLS servers must use an appropriate TLS certificate. Clients need a certificate if mutual authentication is required.

TLS certificates for web servers that are accessible from the internet must be issued by public certification authorities that are classified as trustworthy by browsers and operating systems. These can be ordered, for example, via the service “TeleSec ServerPass” (please refer https://www.telesec.de/de/serverpass).

Regarding key lengths, validity and further configuration options, the Certificate Policy of the Certification Authority must be considered.

For web servers that are used exclusively for internal applications and are not accessible from the internet, digital certificates from a private (internal) Certification Authority can be used.

The minimal requirements according to the following table must be considered for each type of TLS certificates, this means also for client certificates:

Algorithm family	**Key length**	**Hash algorithm**
Elliptic Curve	250 bit	SHA-3, SHA-2 with an output length 256 bit
Digital Signature Algorithm (DSA)	3000 bit	SHA-3, SHA-2 with an output length 256 bit
RSA	3000 bit	SHA-3, SHA-2 with an output length 256 bit

Remarks on DSA and RSA certificates:
For DSA and RSA, key lengths smaller than 3000 bits may only be used in legacy systems [BSI TR 02102-1] until **end of the year 2025** [2] and should be substituted at the next opportunity. Because of the better performance, elliptic curve (EC-DSA) certificates shall be preferred (if supported and technically doable).
RSA-PKCS#1 v1.5 may only be used in legacy systems and should be (if feasible) substituted at the earliest opportunity [BSI TR 02102-1].

Restrictions on SHA-224/SHA-3-224:
SHA-224/SHA-3-224 may only be used in legacy systems and must be substituted by a stronger hash algorithm with an output length of at least 256 bits at the next opportunity.

The validity period of public TLS server certificates (issued by a certification authority, which issues certificates according to the specifications of the [CA/Browser Forum]) must not exceed 397 days. For other, internal TLS certificates, a validity period of 3 years should not be exceeded.

References:
[BSI TR 02102-1] Bundesamt für Sicherheit in der Informationstechnik: Cryptographic Mechanisms: Recommenda-

tions and Key Lengths, TR-02102-1, Version 2023-01
[1] Bundesamt für Sicherheit in der Informationstechnik, TR-02102-2, Version 2023-01
[2] SOG-IS Crypto Evaluation Scheme: Agreed Cryptographic Mechanisms, v1.3, February 2023
[CA/Browser Forum] https://cabforum.org/baseline-requirements-documents/

Motivation: Digital certificates form the basis of the authentication and build up trust. Without sufficiently strong authentication, man-in-the-middle attacks are possible.

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
- Denial of executed activities
- Unnoticeable feasible attacks
- Attacks motivated and facilitated by information disclosure or visible security weaknesses

For this requirement the following warranty objectives are relevant:

ID: 3.50-43/7.0

4.2. Caching of web service requests and responses

Req 22	If Web service requests or responses are cached, the Web service gateway must ensure the integrity of the messages.

Motivation: To ensure the integrity of the stored messages, a checksum must be formed over the messages.

Implementation example: To ensure the integrity of the communication between consumer and provider, end-to-end procedures such as XML signatures or "JSON web signatures" (RFC 7515) are recommended.
Alternatively, simple checksums such as CRC32 can be generated by the Web service gateway.

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

For this requirement the following warranty objectives are relevant:

ID: 3.13-22/6.0

Req 23	If messages are stored temporarily on persistent data storage media, the storage media must be encrypted.

Encryption must be in accordance with the current state of the art.

Motivation: On non-encrypted storage media, data residues can arise which can become problematic when disposing of the data media.
Also, in the event of a compromise of the system, data from previous web service communication data can be recovered.

Implementation example: Database encryption or complete file system encryption is recommended for this purpose.

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

For this requirement the following warranty objectives are relevant:

ID: 3.13-23/6.0

Req 24	If web service responses or requests are cached, the validity period must be selectable by the web service provider.

For multiple use of web service responses, no validity period may be provided in the default.
The maximum validity time must be limited by the gateway.

Motivation: To prevent unintentional multiple use of web service responses, these must not be cached in a standard manner.
The provider must ensure that its Web service responses are suitable for multiple use and specifically activate this functionality.
The maximum validity time must be limited by the gateway to prevent overflowing memories.

Implementation example: The use of the time-to-live (TTL) header is recommended for implementation.

ID: 3.13-24/6.0

5. Security and compliance functionalities

5.1. Publisher and consumers

Req 25	All Web Service requests must be validated by the Web Service Gateway against a detailed specification.

Each dataset submitted through a Web Service must match the expected data elements, expected length and/or expected range and whenever appropriate a formal specification describing the acceptable data.

Motivation: The detailed specification (including regular definitions) allows a much better description of the data that is expected as data elements in the web service.

Implementation example: A complete web service description can be achieved using e.g. WSDL 2.0 or OpenAPI.

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

For this requirement the following warranty objectives are relevant:

ID: 3.13-25/6.0

Req 26	The Web Service Gateway must be able to limit the maximum number of connections to Web Service Providers.

The limits should be freely selectable by the provider and may provide for a burst mode.

Motivation: To protect the Web service providers from overload, the Web service gateway must provide connection limiting functionalities.

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

For this requirement the following warranty objectives are relevant:

ID: 3.13-26/6.0

Req 27	Validation errors and exceeded connection limits must be logged by the Web Service Gateway.

Motivation: To troubleshoot and detect attacks on a Web service provider, the gateway must log validation errors and exceeded connection limits and provide them to the providers of a Web service.

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

For this requirement the following warranty objectives are relevant:

ID: 3.13-27/6.0

Req 28	Configured providers and consumers must be visible at all times.

Providers of web service interfaces must be able to provide a list of all consumers at any time.
It should be possible to provide this list automatically.

Motivation: To ensure auditing and compliance validation, all consumers and providers must be visible at all times.

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

For this requirement the following warranty objectives are relevant:

ID: 3.13-28/6.0

Req 29	Web service consumers and providers must be provided with a list of data which is evaluated and logged by the gateway.

Motivation: To ensure data security and data protection, the users of the Web Service Gateway must be made aware of which data is evaluated and logged.

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

For this requirement the following warranty objectives are relevant:

ID: 3.13-29/6.0

5.2. Web Service Gateway

Req 30	In overload situations, the system must behave in a predictable manner.

Even comprehensive native protections may not be able to prevent a system from becoming overloaded in extreme situations.

It must therefore be ensured that, in overload situations, the system does not switch to a state that overrides security-relevant functions or properties of the system. Performance losses (e.g. the reduction of the throughput of legitimate network packets or the number of answered server requests per period) are usually unavoidable in overload situations, but the regular functional behavior of the system must be fundamentally preserved.

In extreme cases, this can mean that a controlled shutdown of the system is more acceptable than continued operation in the event of uncontrolled failure of the security functions and thus the loss of system protection.

Motivation: By means of a denial-of-service attack, an attacker can try to overload a system in a targeted manner. If such a system then reacts unpredictably or fails its regular behavior, especially with regard to its security functions, this can open up an extended attack surface for the attacker on functions and data of the system and potentially endanger other linked systems.

Implementation example: A firewall that discards its filter rules in overload situations and forwards all packets without checking would not meet the requirement. In this case, blocking all packets by shutting down the firewall would be more acceptable than failing their regular task of protecting downstream systems.

For this requirement the following threats are relevant:
- Unauthorized use of services or resources
- Disruption of availability
- Unnoticeable feasible attacks

For this requirement the following warranty objectives are relevant:

ID: 3.01-13/7.0

Req 31	The components of the Web Service Gateway must be protected against unauthorized modification.

The protection should not be limited to the software of the Web Service Gateway, but should cover the entire operating system.

Motivation: To make the software and operating system tamper-resistant, mechanisms that ensure file integrity are the only way to prevent tampering.

Implementation example: In container architectures, the complete containers should be signed and verified before execution.
For example, in Tomcat applications, the JAR file can be signed and verified by the Tomcat server.

For this requirement the following threats are relevant:
- Unauthorized modification of data
- Unnoticeable feasible attacks

For this requirement the following warranty objectives are relevant:

ID: 3.13-31/6.0

Req 32	The configuration of the Web Service Gateway must be validated regularly.

To protect the configuration from unauthorized and unwanted changes, the rolled-out configuration must be regularly checked against the intended state.
In addition to the service configuration, this also applies to the interface configuration of consumers and providers.

Motivation: The configurations of the Web service gateways are provided by among other service repertoires and rolled out to decentralized Web service gateways by automatic mechanisms.
To ensure that these mechanisms function properly and to prevent unintentional changes to these configurations, the actual state of the configurations must be regularly compared with the target state.
This prevents malfunctions by guaranteeing that the configuration is up-to-date.

Implementation example: Configuration files can be signed and versioned.
The version used can be provided as a web service and checked automatically by the distribution service.

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

For this requirement the following warranty objectives are relevant:

ID: 3.13-32/6.0

Req 33	Changes to the configuration of the Web Service Gateway must be logged continuously.

Motivation: In order to be able to track changes, especially to the interface configuration, a continuous log of the changes and the person who made them must be kept.

For this requirement the following threats are relevant:
- Unauthorized modification of data
- Denial of executed activities

For this requirement the following warranty objectives are relevant:

ID: 3.13-33/6.0

Req 34	If a web service request is protected with signatures, the signature data must not be removed from the request by an intermediate processor.

The signature must be validated over the entire transmission path to guarantee the integrity and authenticity of the web service requests / responses.

Motivation: If this signature is removed, the integrity and authenticity of the web service is no longer guaranteed.

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

For this requirement the following warranty objectives are relevant:

ID: 3.13-34/6.0

6. Enrollment of suppliers and consumers

Req 35 For each provider and consumer of a Web service, a primary responsible party must be designated.

The primary owner is allowed to change the interface configuration. For example, accept URL endpoints, consumers, or even name or remove additional responsible parties.

Motivation: In order to have clear responsibilities, there must be one main person responsible for each provider and consumer of a web service.
This person is primarily responsible for all issues that arise, e.g. payload logging, changes to the encryption, inclusion of new consumers, recertification of consumers, etc.

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

For this requirement the following warranty objectives are relevant:

ID: 3.13-35/6.0

Req 36 Authorization of new consumers must be approved by the web service provider.

Changes to the authorized consumers must be logged continuously for each provider. The log must show who made which changes and when.

Motivation: Only the provider of a web service is responsible for who may use this service or receive this data, therefore each new consumer must be approved by the provider.

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

For this requirement the following warranty objectives are relevant:

ID: 3.13-36/6.0

Req 37 The default policy for new web service providers must reject all consumers.

Access control must follow the principle of "Default Deny", i.e. access to a service must not be granted without explicit permission.

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

For this requirement the following warranty objectives are relevant:

ID: 3.13-37/6.0

7. Architecture

Req 38	Gateways that are responsible for internal networks must not be directly accessible from the Internet.

On gateways that are directly accessible on the Internet, only providers that should also be accessible from the Internet may be provisioned.

Motivation: To reduce the number of externally compromisable web services and to reduce the impact of denial of service attacks, internal components must be separated from external ones.

Implementation example: An edge gateway should be used for gateways that are available in internal and external networks. The edge gateway forwards the requests from the Internet to the internal gateway.

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

For this requirement the following warranty objectives are relevant:

ID: 3.13-38/6.0

Req 39	Network traffic for Web Service must be separated from traffic for Web Applications.

Motivation: The gateways are usually not designed to deliver web applications, even if they are basically capable of doing so. Web applications are usually used to provide configuration and error analysis tools that must remain accessible in the event of an error.

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

For this requirement the following warranty objectives are relevant:

ID: 3.13-39/6.0

8. Logging

Req 40 The system clock must be synchronized to an accurate reference time (Time Standard).

A time reference source must be used which provides a time signal based on the Coordinated Universal Time ("UTC" = "**U**niversal **T**ime **C**oordinated").

Please Note: The UTC-synchronized system time may be transformed to local time using a corresponding timezone configuration setup for any output of time information, as long as this timezone adjustment is fully accountable.

Systems belonging to the same security domain must synchronize to one and the same time reference source.

Motivation: Reference time synchronization may be a technical prerequisite for many time-dependent mechanisms, for example: Validation of Certificates; Authentication. It is also much-needed to generate exact timestamps for logged events, since without the often required time-related correlation in case of a Security Incident or during a Problem Analysis cannot be achieved.

Implementation example: some valid time reference sources:

- trustworthy NTP ("**N**etwork**T**ime**P**rotocol") Server on the IP network
- DCF77 radio signal received via a physically connected receiver
- GPS radio signal received via a physically connected receiver

For this requirement the following threats are relevant:

- Disruption of availability
- Denial of executed activities
- Unnoticeable feasible attacks

For this requirement the following warranty objectives are relevant:

ID: 3.01-32/7.0

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 ("Timestamp").

Exceptions of this requirement are systems for which logging cannot be implemented because of building techniques, use case or operation area. Examples for these kind of systems are customer devices such as Smartphones or IADs/home gateways (e.g. Speedport).

The Timestamp of a logged event must contain at least the following information:

- date of the event (Year, Month, Day)
- time of the event (Hours, Minutes, Seconds)
- Timezone, those information belongs to

When logging, the applicable legal and operational regulations must be observed. The latter also include agreements that have been made with the company's social partners. Following these regulations logging of events is only allowed for a defined use case. Logging of events for doing a work control of employees is not allowed.

In addition - as for any data that is processed by a system - an appropriate protection requirement must also be taken into account and implemented for logging data; this applies to storage, transmission and access. In particular, if the logging data contains real data, the same protection requirements must be taken into account that is also used for the regular processing of this real data within the source system.

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

Event	**Event data to be logged**
Incorrect login attempts	• User account, • Number of failed attempts, • Source (IP address, client ID / client name) of remote access
System access from user accounts with administrator permissions	• User account, • Access timestamp, • Length of session, • Source (IP address) of remote access
Account administration	• Administrator account, • Administered user account, • Activity performed (configure, delete, enable and disable)
Change of group membership for accounts	• Administrator account, • Administered user account, • Activity performed (group added or removed)
Critical rise in system values such as disk space, CPU load over a longer period	• Value exceeded, • Value reached (Here suitable threshold values must be defined depending on the individual system.)

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. Logging data could be used as evidence to take legal steps against attackers.

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

For this requirement the following warranty objectives are relevant:

ID: 3.01-33/7.0

Req 42	Applicable retention and deletion periods must be observed for security-relevant logging data that is recorded locally.

From an IT security perspective, local storage of security-relevant logging data on a system is not mandatory. Since the local storage can be damaged in the event of system malfunctions or manipulated by a successful attacker, it can only be used to a limited extent for security-related or forensic analyses. Accordingly, it is relevant for IT security that logging data is forwarded to a separate log server.

Local storage can nevertheless take place; for example, if local storage is initially indispensable when generating the logging data due to technical processes or if there are justified operational interests in also keeping logging data available locally.

The following basic rules must be taken into account when storing logging data locally:

- Security-related logging data must be retained for a period of 90 days.
  (*This requirement only applies if no additional forwarding to a separate log server is implemented on the system and the logging data is therefore only recorded locally.*)
- After 90 days, stored logging data must be deleted immediately.

Deviances
Different retention periods and deletion periods may exist due to legal or regulatory requirements (especially in connection with personal data) or may be defined by contractual agreements. In these cases, the applicable periods must be agreed individually with a Project Security Manager (PSM) / Data Privacy Advisor (DPA) or are specified by them.

Motivation: Logging data is an immensely important IT security tool for preventing, detecting and clearing up system faults, security and data privacy incidents. On the other hand, the recording of logging data, like any other data processing, is also subject to legal and regulatory requirements. Accordingly, guidelines must be adhered to that reconcile the two.

Implementation example: Taking into account the current legal situation and applicable data privacy regulations, the following deletion periods for locally stored security-relevant logging data are implemented on an exemplary telecommunications system:

- Standard System Logs: Deletion after 90 days at the latest
- Logging of public IP addresses: Deletion (or anonymization) after 7 days at the latest
- Logging of the assignment of dynamic public IP addresses by the telecommunication solution: Deletion after 7 days at the latest
- Logging of non-billing-relevant call detail records: Deletion after 7 days at the latest
- Logging of the content of e-mail and SMS: Deletion after 24 hours at the latest
- Logging of the domain queries handled by the DNS server of the telecommunications solution: Deletion after 24 hours at the latest

For this requirement the following threats are relevant:

- Unauthorized access or tapping of data
- Denial of executed activities
- Unnoticeable feasible attacks

For this requirement the following warranty objectives are relevant:

ID: 3.01-34/7.0

Req 43	Security-relevant logging data must be forwarded to a separate log server immediately after it has been generated.

Logging data must be forwarded to a separate log server immediately after it has been generated. Standardized protocols such as Syslog, SNMPv3 should 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:

- Unauthorized modification of data
- Disruption of availability
- Denial of executed activities
- Unnoticeable feasible attacks

For this requirement the following warranty objectives are relevant:

ID: 3.01-35/7.0

Req 44	For security-relevant logging data that is forwarded to the separate log server, compliance with the applicable retention and deletion periods must be ensured.

The following basic rules must be taken into account:

- security-related logging data must be retained for a period of 90 days on the separate log server.
- after 90 days, stored logging data must be deleted immediately on the separate log server.

Deviances

Different retention periods and deletion periods may exist due to legal or regulatory requirements (especially in connection with personal data) or may be defined by contractual agreements. In these cases, the applicable periods must be agreed individually with a Project Security Manager (PSM) / Data Privacy Advisor (DSB) or are specified by them.

Log server under the responsibility of a third party

If the selected separate log server is not within the same operational responsibility as the source system of the loggin data, it must be ensured that the responsible operator of the log server is aware of the valid parameters for the logging data to be received and that they are adhered to in accordance with the regulations mentioned here.

Motivation: Logging data is an immensely important IT security tool for preventing, detecting and clearing up system faults, security and data privacy incidents. On the other hand, the recording of logging data, like any other data processing, is also subject to legal and regulatory requirements. Accordingly, guidelines must be adhered to that reconcile the two.

Implementation example: Taking into account the current legal situation and applicable data privacy regulations, the following deletion periods for forwarded security-relevant logging data from an exemplary telecommunications system are implemented on the separate log server:

- Standard System Logs: Deletion after 90 days at the latest
- Logging of public IP addresses: Deletion (or anonymization) after 7 days at the latest
- Logging of the assignment of dynamic public IP addresses by the telecommunication solution: Deletion after 7 days at the latest
- Logging of non-billing-relevant call detail records: Deletion after 7 days at the latest
- Logging of the content of e-mail and SMS: Deletion after 24 hours at the latest
- Logging of the domain queries handled by the DNS server of the telecommunications solution: Deletion after 24 hours at the latest

For this requirement the following threats are relevant:

- Unauthorized access or tapping of data
- Denial of executed activities
- Unnoticeable feasible attacks

For this requirement the following warranty objectives are relevant:

ID: 3.01-36/7.0

Req 45	The system must provide logging data that is required to detect the system-specific relevant forms of attack in a SIEM.

The forms of attack that are typically to be expected for the present system must be systematically analyzed and identified.
The MITRE Attack Matrix (https://attack.mitre.org) can be used as a structured guide during such an identification.

It must be ensured that the system generates appropriate logging data on events that are or may be related to these identified forms of attack and that can be used to detect an attack that is taking place.

The logging data must be sent to a SIEM immediately after the system event occurs.
SIEM (Security Information & Event Management) solutions collect event log data from various source systems, correlate it and evaluate it automatically in real time in order to detect anomalous activities such as ongoing attacks on IT/NT systems and to be able to initiate alarms or countermeasures.
The immediate receipt of system events is therefore absolutely crucial for the SIEM to fulfill its protective functions.

Note:
The immediate need to connect a system to a SIEM is specifically regulated by the separate "Operation" security requirements catalogs.
If the present system does not fall under this need, the requirement may be answered as "not applicable".

Motivation: A SIEM as an automated detection system for attacks can only be effective if it continuously receives sufficient and, above all, system-specific relevant event messages from the infrastructures and systems to be monitored. General standard event messages may not be sufficient to achieve an adequate level of detection and only allow rudimentary attack detections.

Implementation example: An example system allows end users to log in using a username and password. One of the typical forms of attack for this system would be to try to discover and take over user accounts with weak or frequently used passwords by means of automated password testing (dictionary or brute force attack). The example system is configured to record every failed login event in system protocols ("logs"). By routing this logging data in parallel to a SIEM, the SIEM can detect in real time that an attack is obviously taking place, alert it and thus enable immediate countermeasures.

ID: 3.01-37/7.0