Security requirement

SAP Netweaver areas

Deutsche Telekom Group

Version	373 (internal)
Date	Nov 29, 2023
Status	In work

Publication Details

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

Reuterstrasse 65, 53315 Bonn
Germany

File name	Document number	Document type
	3.22	Security requirement

Version	State	Status
373 (internal)	Nov 29, 2023	In work

Contact	Validity	Released by
Telekom Security psa.telekom.de		

Summary
The requirements are vaild for:

- Netweaver SAP-Systeme
- SAP-HANA
- SapRouter
- WebDispatcher
- Fiori-Applikation-Server,
- Fiori-Apps

Table of Contents

1. Introduction

This security document has been prepared based on the general security policies of the Group.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.

SAP specific:
For access to SAP notes and links within that document a account for SAPNET is partially required.

Notes about the "System Properties"

Area "Components"
SAP is a "Server Application".
The SAProuter and the SAP Web Dispatcher are not a "Network Element/Infrastructure" and / or "Network Element/Infrastructure with HTTP GUI".
Fiori is not a "Mobile Application".

Area "Applications"
If the system is exclusively a SAP Netweaver System, the application types "Web Application" and "Web Service Interface" no longer need to be checked in the system properties in the PSA portal after selecting "SAP" due to the requirements for these are integrated into this document.
When selecting the "SAP" application type, "none" must be selected in the "Data Security" area of the system properties when asking for the database and the protocol, due to the requirements for these are integrated into this document.

Area "Operator"
The SAP Support access does not include access to the system by external companies. In the case of SAP Support access, tick "No, no access to system from a 3rd party".

2. Requirements

2.1. General aspects

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.

Implementation example: A: Check of the running processes at OS level with administrative users with ...

- Unix: ps -ef
- Linux: ps -aux
- Windows: Taskmanager

B: Check of the reachable services with ...

- Unix: netstat -an and/or systemctl --type=service --state=active
- Linux: netstat -taupen and/or systemctl --type=service --state=active
- Windows: netstat -an

C: Check of the Config:

- To find clues as to whether the service is needed or not?
- Is the service customized by the administrator?
- Is it the default configuration and the service just started?

D: Disable unnecessary services and their ports.

ID: 3.22-1/i373

Req 2	The accessibility of activated services must be restricted.

In principle, a service provided must be completely deactivated on all interfaces of the system through which accessibility of the service is not required for the proper operation of the system. The deactivation is primarily to be implemented by a corresponding configuration of the service or operating system. In cases where the available configuration options do not allow deactivation on individual interfaces, a local filter ("Host Firewall") may instead be used on the system to block access to the service via unnecessary interfaces.

The accessibility of a service via the required interfaces must also be restricted to legitimate communication partners. The restriction must be implemented by a corresponding configuration of the service or operating system or by means of a local filter ("Host Firewall"). Alternatively, this task may be outsourced to a network-side filter element, provided that the system is located in a suitable separate network segment and communication with this segment is only possible via the network-side filter element.

Motivation: By deactivating services on interfaces through which accessibility is not necessary, as well as by restricting possible communication partners, the attack surface offered by a system can be greatly reduced.

Implementation example: An SNMP service used to monitor a system is enabled exclusively on the dedicated management network interface of the system. A firewall also regulates that only the legitimate monitoring system of the infrastructure environment can reach this service.

ID: 3.22-2/i373

Req 3	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.

ID: 3.22-3/i373

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.

ID: 3.22-4/i373

Req 5	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 soft-

ware 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 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.

ID: 3.22-7/i373

Req 8	Stored data in need of protection must be protected against unauthorized access, modification and deletion.

The need for protection of stored data depends on its classification (e.g. according to applicable legal data privacy requirements, regulatory requirements, contractual obligations), the potential damage in the event of its misuse, and other relevant factors (e.g. the location of storage). The nature and extent of protective measures must be appropriately chosen.
Stored authentication attributes such as passwords, private keys, tokens or certificates etc. are generally considered to be in need of protection. Data that determines the functionality and security-relevant behavior of a system (e.g. system configuration files, operating systems and kernels, drivers) are also considered to be fundamentally in need of protection.

Compliance with the protection objectives of confidentiality, integrity and availability must be consistently guaranteed for stored data in need of protection. This also applies during only short-term storage (e.g. when storing in a web cache or in a temporary folder within a data processing chain).

Basically, access to data in need of protection in a system must be fully regulated on the basis of technically implemented authorization assignments and controls.

If such technical access control alone is no longer sufficient to ensure the necessary protection requirements of stored data, or if its effectiveness cannot be consistently ensured, additional cryptographic methods (e.g. encryption, signing, hashing) must be implemented. Cryptographic methods used in the storage of data must be suitable for this purpose and must have no known vulnerabilities.

Motivation: The storage of data on a system without adequate protection enables an attacker to view, use, disseminate, modify or destroy it without authorization. This potentially opens up additional attack vectors on the immediate and connected other systems and can lead to significant failures, loss of control and damage as well as resulting penalties and loss of reputation towards customers and business partners.

Implementation example: [Example 1]
A system exports data for transport to mobile media. Since the system's technical access control at the file permission level no longer applies as soon as the mobile media is removed from the system, additional measures must be taken to protect the data. Before the system writes the data to the mobile media, it is encrypted accordingly using a suitable algorithm. The associated encryption key is exchanged on a separate channel so that the data can be decrypted and processed again in the legitimate target system. An attacker who takes possession of the mobile media, on the other hand, has no access to the data.

[Example 2]
Only cryptographic hashes of passwords generated with a secure password hashing method are stored in the local user database of a system. For the system, these hashes are sufficient to authenticate users when they log on to the system. However, if an attacker can copy the user database, he does not immediately come into possession of plaintext passwords with which he could log on to the system on behalf of the users.

[Example 3]
On a system, the configuration files of the Web server can only be written by the legitimate admin in which corresponding permissions have been set in the file system. The access control of the operating system kernel thus denies all other users of the system to make changes to the configuration files of the web server; including the web server service account itself, which also reduces the attack surface from the outside in case of vulnerabilities in the web server.

ID: 3.22-8/i373

Req 9	Data in need of protection must be protected against unauthorized access and modification during transmission.

The need for protection of data to be transmitted depends on its classification (e.g. according to applicable legal data

privacy requirements, regulatory requirements, contractual obligations), the potential damage in the event of its misuse, and other relevant factors (e.g. transmission via public networks). The nature and extent of the protective measures must be appropriately chosen.
Authentication attributes such as passwords or tokens etc. are generally considered to be in need of protection. Data that determines the functionality and security-relevant behavior of a system (e.g. updates & patches, configuration parameters, remote maintenance, control via APIs) are also considered to be fundamentally in need of protection.

Compliance with the protection objectives of confidentiality and integrity must be consistently guaranteed during the transmission of data in need of protection.

As a rule, this requires the implementation of cryptographic methods (e.g. encryption, signatures, Hashes).
Cryptographic methods may

- be applied directly to the data before transmission, which can make subsequent transmission acceptable even via insecure channels
- be used on the transmission channel to create a secure channel and protect any kind of data passing through it
- or be implemented as a combination of both.

Cryptographic methods used in the transmission of data must be suitable for this purpose and must have no known vulnerabilities.

Motivation: The transmission of data without adequate protection enables an attacker to intercept, use, disseminate, modify or remove it from transmission without authorization. This potentially opens up further attack vectors on the immediate target systems as well as connected other systems and can lead to significant failures, loss of control and damage as well as resulting penalty claims and reputational losses towards customers and business partners.

Implementation example: [Example 1]
Confidential documents are encrypted before they are sent by e-mail to the customer.

[Example 2]
An administrator configures a new cloud application over the Internet. Access is via a TLS-encrypted connection ("https").

[Example 3]
A system obtains automatic software updates from an update server. The update server delivers the software updates cryptographically signed. The system can thus validate the received software updates and reliably rule out that they have been manipulated during transmission.

ID: 3.22-9/i373

Req 10	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.

ID: 3.22-10/i373

Req 11	The system must be protected against overload situations.

A system must have protective mechanisms that prevent overload situations as far as possible.
In particular, a partial or complete impairment of the availability of the system must be avoided.

Examples of possible protective measures are:

- Limiting the amount of memory (RAM) available per application
- Limiting the maximum sessions of a web application
- Limiting the maximum size of a dataset
- Limiting CPU resources per process
- Prioritizing processes
- Limiting the number or size of transactions by a user or from an IP address over time

Note:
A system can usually not protect itself against network-based attacks with extremely high data or packet rates, the so-called "Distributed Denial of Service" (DDoS) attacks. To defend against DDoS attacks, an upstream solution in the network layer is required.

Motivation: Attackers can try to use up the resources of a system with targeted resource-intensive or large-volume requests, so that the system can no longer fulfill its regular tasks or intended task volumes and the availability of the services offered is effectively disrupted. Limiting the maximum resources that can be used per request made to the system is a fundamental measure to reduce the impact of such denial-of-service (DoS) attacks.

ID: 3.22-11/i373

Req 12	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 13	The system must be implemented robustly against unexpected inputs.

Data transferred to the system must first be validated before further processing to ensure that the data corresponds to the expected data type and format. This is intended to eliminate the risk of manipulation of system processes and states by appropriately constructed data content. Validation must be carried out for any data that is transferred to the system. Examples include user input, values in data fields, and log contents.

The following typical implementation mistakes must be avoided:
- lack of validation of the length of passed data
- Incorrect assumptions about the format of data
- lack of validation of received data for conformity with the specification
- Inadequate handling of protocol deviations in received data
- Insufficient limitation of recursion when parsing complex data formats
- Insufficient implementation of whitelisting or escaping to protect against inputs outside the valid value range

Motivation: An attacker can use specifically engineered data content to try to put a system that does not sufficiently validate received data before internal processing into an unstable state or to trigger unauthorized actions within the system. The damage potential of such attacks depends on the individual system, but has a theoretical range from uncontrolled system crashes to a controlled execution of specially injected code and the resulting complete compromise of a system.

ID: 3.22-13/i373

2.2. Securtiy of network and communication

Req 14	Direct access to an SAP system must always be protected by an upstream system separated from the network side, according to the protocols used.

The direct access on the SAP Systeme over the network must be restricted. Depending on the protocol different systems must be established infront of the SAP System (e. g. SAProuter, SAP Web Dispatcher, ...).

Motivation: Unauthorized access to server components may lead to the system being compromised, since several SAP backend components are not designed for direct access by clients and therefore do not offer adequate protection against non-authorized access.

Implementation example: Are DIAG- and RFC-protocols in use, the SAP system must be protected by a network-side separated upstream SAProuter.

Are HTTP-, HTTPS- and SMTP-protocols in use, the SAP system must be protected by a network-side separated upstream SAP Web Dispatcher.

Are DIAG-, RFC-, HTTP-, HTTPS- and SMTP-protocols in use, the SAP System must be protected by each a network-side separated upstream SAProuter and a network-side separated upstream SAP Web Dispatcher.

Requirement-ID: 011b57bf

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

For this requirement the following warranty objectives are relevant:

ID: 3.22-14/i373

Req 15	Direct external access to a SAP system must always be protected by a network-side separated upstream system in the DMZ, according to the protocols used.

It must be ensured, that all protocols (DIAG, RFC, HTTP, HTTPS, SMTP) are filtered by the SAProuter and/or the SAP Web Dispatcher within the DMZ. Protocols, which are not processed on the SAProuter and/or the SAP Web Dispatcher, are not allowed to reach these ones.

Different upstream systems must be used for internal and external access.

Motivation: The SAP System must be protected for unauthorized access and requests.

Implementation example: Are DIAG- and RFC-protocols in use, the SAP system must be protected by a SAProuter.

Are HTTP-, HTTPS- and SMTP-protocols in use, the SAP system must be protected by a SAP Web Dispatcher.

Are DIAG-, RFC-, HTTP-, HTTPS- and SMTP-protocols in use, the SAP system must be protected by a SAProuter and a SAP Web Dispatcher.

Requirement-ID: 74f42f33

ID: 3.22-15/i373

Req 16	Information classified as “confidential” must be transmitted in encrypted form.

Generally, confidential data must be transmitted encrypted, e.g. by using TLS (SSL) or SNC for CPIC/RFC/DIAG. The

SNC connection must be set up with the SNC mode 3 (encryption of the communication).

The encryption must established between SAP System and the client, between SAP Systems and between SAP Systems and Non-SAP Systems. If the communication between two systems cannot be ecrypted by the systems directly, at least the SAPRouter-to-SAPRouter communication must be encrypted, which are located between the two systems.

The only exception to the encryption requirement exists in cases where the transmission is solely routed locally within a DTAG-owned data center across systems, networks and lines owned by DTAG. An example of this is a layer-2 switch to which all involved systems are connected locally. It must be ensured that only authorized individuals have access to this network. 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).

Motivation: If an attacker succeeds in eavesdropping on communication, he will be able to access vulnerable data and steal authentication data (passwords, login tickets). This opens the way to unauthorized access to the system.

Implementation example: S. Chapter 2.3 Cryptographic

Requirement-ID: ae5950c0

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

For this requirement the following warranty objectives are relevant:

ID: 3.22-16/i373

Req 17	Non-SAP Systems are not allowed to use the same network area as SAP Systems.

The communication between SAP Systems is mostly unencrypted.

Motivation: A lot of the comunication between SAP Systems is unencrypted. To secure the data and the systems for unauthorized access, SAP Systems and Non-SAP Systems must be separated by the network area.

Implementation example: Requirement-ID: ec19682c

ID: 3.22-17/i373

Req 18	It must be ensured that a secure connection is established between SAP systems and SAProuters.

It must be ensured that a secure connection is established between SAP Systems and SAProuters as well between SAP Systems and SAProuters by themself.

Motivation: Ensure integrity, authenticity and confidentiality of the data by prevention of an unauthorized access on the data.

Implementation example: To take care about:
- Usage of a recognised and public Certificate Authority (CA) for the key administration.
- Usage of the SAP Cryptographic Library (*SAPCRYPTOLIB*).
- s. Chapter 2.3 Cryptographic for further details (e.g. key length, encryption algorithm).

Requirement-ID: c6cbb721

ID: 3.22-18/i373

2.3. Cryptographic ----- If requirements 19 to 44 are met, the security requirement "Cryptographic Algorithms and Security Protocols" (3.50) for SAP Netweaver environments is satisfied.

Req 19	Each deviation have to be aligned with the security organisation of Deutsche Telekom Group. All requirements fulfilling (based on the publication date of this document) the current standards of following organization

Each deviation have to be aligned with the security organisation of Deutsche Telekom Group. All requirements fulfilling (based on the publication date of this document) the current standards of following organizations:

- German Information Security Agency - Bundesamt für Sicherheit in der Informationstechnik (BSI) [BSI]
- European Telecommunication Standards Institute (ETSI) [ETSI]
- Senior Officials Group Information Systems Security (SOGIS) [SOGIS]
- National Institute of Standards and Technology (NIST) [NIST]
- International Organization for Standardization (ISO) [ISO]
- Internet Engineering Task Force (IETF) [IETF]

References:
[BSI] https://www.bsi.bund.de/DE/Home/home_node.html (last view: October 8th, 2019)
[ETSI] https://www.etsi.org/ (last view: October 8th, 2019)
[SOGIS] https://www.sogis.eu/ (last view: October 8th, 2019)
[NIST] https://www.nist.gov/ (last view: October 8th, 2019)
[ISO] https://www.iso.org/home.html (last view: October 8th, 2019)
[IETF] https://www.ietf.org/ (last view: October 8th, 2019)

Motivation: Only well analyzed cryptographic algorithms have a sufficient systematic protection against cryptographic attacks. It shall be ensured that the recommendations are based on a common functional consent.

Implementation example: The SAPcryptoLib / CommoncryptoLib is to install and to configure. Download the latest SAP Cryptographic Library from the SAP Software Download Center.

Please follow the SAP instructions (SAP Help, SAP Security Guides, SAP Notes) for setting up the SAP SAPcryptoLib / CommonCryptoLib and configuring SSL/HTTPS and SNC on the respective SAP product. The steps for the configuration are different for SSL/HTTPS and SNC communication and even for the different possible SNC communication types for AS ABAP (RFC, DIAG/SAPGUI, CPIC, printer, external programs, SAProuter, AS Java) and furthermore still depending on the respective version and the release.

Requirement-ID: 092a7f10

ID: 3.22-19/i373

Req 20	Well-established and up-to-date crypto libraries must be used to implement cryptographic algorithms.

User roles: Development, Integration

The use of self-implemented cryptographic systems shall be avoided. Instead, well-established crypto libraries which implement the approved algorithm set of this document must be used. If customized crypto implementations are required, they must follow best practices and must be supervised by the security organization of Deutsche Telekom group.

Motivation: Only well known crypto libaries have been analyzed by security researchers and providing a sufficient protection. Those libaries will be updated regurarly (e.g. in case of vulnerabilities).

Implementation example: The requirement is fulfilled when the SAP Cryptographic Library or T-Secure is used.

Requirement-ID: b9ba557a

ID: 3.22-20/i373

Req 21	Cryptographic systems must be implemented in replacable modules.

User roles: Development, Integration

Static implementations complicate corrections and replacements of faulty implementations. Especially in case of unexpected security incidents or future computer architectures (e.g. quantum computing) a fast replacement of cryptographic modules might be necessary.

Motivation: Cryptograhic implementation shall be fully replacebale without any dependencies to the software environment.

Implementation example: The requirement is fulfilled when the SAP Cryptographic Library or T-Secure is used

Requirement-ID: 176b1a12

ID: 3.22-21/i373

Req 22	The application must be able to configure cryptographic systems and must provide swap functions.

ser role: Development

Deactivation and modification functions (e.g. cipher suites modifications) must be implemented during the development of cryptographic systems.

Swap functions will only be applied on persistant data storages, but not on transport encryption.

Unexpected events like new attacks or future computing architectures (e.g. universal quantum computers), might cause an immediate substitution of broken schemes.

Motivation: Crypto management is a mandatory prerequisite for crypto agility. It allows to reduce risk and effort in case of urgent system changes in a running system

Implementation example: The requirement is fulfilled when the SAP Cryptographic Library or T-Secure is used as GSS-API V2-Lib

Requirement-ID: e11ce0f5

ID: 3.22-22/i373

Req 23	Following symmetric block ciphers must be used: AES-256, AES-192 or AES-128

User roles: Operation, Development, Integration

The "Advanced Encryption Standard" (AES) has been standardized in 2001 as a variant of the "Rijndael"-algorithm.
AES has been analysed intensively since 2001. The algorithm is standardized by ISO/IEC 18033-3 and by Federal Information Processing Standard (FIPS) 197. AES is a block cipher with a specified input block length of 128 bit and supports three key sizes (128 bit, 192 bit and 256 bit). AES-128, AES-192 and AES-256 denoting the respective key lengths. AES-256 can be used to prevent (theoretical) attacks by quantum computers (Grover's algorithm).

Remarks on data privacy:
All requirements in this document are limited to data and information security in terms of confidentiality and integrity. Requirements on cryptographic methods to fulfill requirements on anonymization and pseudonymization with regards

to data privacy are published in document “Anonymization and Pseudonymization”

Motivation: he AES security has been analyzed since almost two decades and provides quantum-safe encryption by using the 256 bit key variant of AES.

Implementation example: With the implementation example given here, the following requirements are met for all SAP software products: 23, 24, 25, 26, 27, 30, 38, 40, 41, 42 and 44.

Please follow the SAP instructions (SAP Help, SAP Security Guides, SAP Notes) for setting up the SAP SAPcryptoLib / CommonCryptoLib and configuring SSL/HTTPS and SNC on the respective SAP product. The steps for the configuration are different for SSL/HTTPS and SNC communication and even for the different possible SNC communication types for AS ABAP (RFC, DIAG/SAPGUI, CPIC, printer, external programs, SAProuter, AS Java) and furthermore still depending on the respective version and the release.

The specifications for encrypting the communication to the respective SAP product are described in the following sections sorted by SAP products.

AS ABAP und ICM (ab Release 6.20)

Please see the following SAP notes: 1848999, 510007, 2570499, 2065806, 2450794, 2384243, 2110020, 2384290, 1433874.
Set the TLS and Cipher settings with TA RZ10 with the profile parameters to the following values:

- ssl/ciphersuites = 544:PFS:+eAES128:-eAES_CBC::EC_HIGH:+EC_OPT
- ssl/client_ciphersuites = 544:PFS:+eAES128:-eAES_CBC::EC_HIGH:+EC_OPT
- icm/server_port_<xx> = ..., PROT=HTTPS, ..., SSLCONFIG=ssl_config_<yy>
- icm/ssl_config_<yy> = ..., CIPHERS=544:PFS:+eAES128:-eAES_CBC::EC_HIGH:+EC_OPT

Notes:

- If you have maintained the SSL configuration with the icm/ssl_config_ <yy> parameter, you must set the SSLCONFIG option from the icm/server_port_<xx> parameter to the "ssl_config_ <yy>" value (<yy> corresponding to the icm/ssl_config_ <xx> parameter).
- Parameter icm/ssl_config_<yy>: The CRED option must be specified e.g. CRED=SAPSSLSsapext.pse.
- Parameter icm/ssl_config_<yy>: The contents of the CIPHERS option are subject to the same rules and syntax that apply to the ssl/ciphersuites profile parameter.

The following must be configured for SNC:
The certificate/key for SNC must have at least the following properties when generated with sapgenpse or the TA STRUST:

- minimum sha256 (option -a)
- Bit length at least 2048 (option -s)

Set the following SNC profile parameters (TA RZ10):

- snc/enable = 1
- snc/gssapi_lib = <path and file name of SAP Cryptographic Library> e. g. /usr/sap/<SID>/SYS/exe/run/libsapcrypto.so
- snc/identity/as = p:<distinguished_name>
- snc/data_protection/min = 3
- snc/data_protection/max = 3
- snc/data_protection/use = 9
- snc/accept_insecure_gui = 0
- snc/accept_insecure_cpic = 0
- snc/accept_insecure_rfc = 0
- snc/accept_insecure_r3int_rfc = 0

When configuring SNC between the AS ABAP and other server components, you must make Access Control List Entries on the application server. To do this, you must know the SNC name used by the communication partner. SNC system access control list maintenance is done with TA SM30 in the table SNCSYSACL (view VSNCSYSACL and type=E).

AS Java und ICM

Please see the following SAP notes: 1848999, 510007, 2065806, 2384243, 2110020, 2284059, 2540433, 2384290, 2616423, 1240081, 2708581.

Set the TLS and Cipher settings to the following values:

Incomming connections with SAP CommonCryptoLib (setting of parameters e.g. in DEFAULT.PFL):

- ssl/ciphersuites = 544:PFS:+eAES128:-eAES_CBC::EC_HIGH:+EC_OPT
- ssl/client_ciphersuites = 544:PFS:+eAES128:-eAES_CBC::EC_HIGH:+EC_OPT
- icm/server_port_<xx> = ..., PROT=HTTPS, ..., SSLCONFIG=ssl_config_<yy>
- icm/ssl_config_<yy> = ..., CIPHERS=544:PFS:+eAES128:-eAES_CBC::EC_HIGH:+EC_OPT

Notes:

- If you have maintained the SSL configuration with the icm/ssl_config_ <yy> parameter, you must set the SSLCONFIG option from the icm/server_port_<xx> parameter to the "ssl_config_ <yy>" value (<yy> corresponding to the icm/ssl_config_ <xx> parameter).
- Parameter icm/ssl_config_<yy>: The CRED option must be specified e.g. CRED=SAPSSLSsapext.pse.
- Parameter icm/ssl_config_<yy>: The contents of the CIPHERS option are subject to the same rules and syntax that apply to the ssl/ciphersuites profile parameter.

Outgoing Connections with IAIK JCE
Please refer to SAP Note 2284059 - Updating the SSL Library in the NW Java Server. The following list lists the secure ciphers:

- cipherSuite=TLS_DHE_RSA_WITH_AES_256_GCM_SHA384
- cipherSuite=TLS_DHE_RSA_WITH_AES_256_CBC_SHA256
- cipherSuite=TLS_DHE_RSA_WITH_AES_128_GCM_SHA256
- cipherSuite=TLS_DHE_RSA_WITH_AES_128_CBC_SHA256

All other ciphers present in the default should be deleted (i.e. all with TLS_RSA) e.g.:

- TLS_RSA_WITH_AES_128_CBC_SHA256
- TLS_RSA_WITH_CAMELLIA_128_CBC_SHA256

Supported SSL/TLS version change (=> TLS 1.2 required). The following setting must be made:

- client.minProtocolVersion=TLS12

The certificate/key for SNC must have at least the following properties when generated with sapgenpse:

- minimum sha256 (option -a)
- Bit length at least 2048 (option -s)

SAP HANA, SAP S/4HANA, ABAP, HANA internal SAP Web Dispatcher

Please refer to the HANA Security Guide and the following SAP Notes: 1848999, 2829919, 2159014, 3201581, 2110020, 2256091, 1761693, 2487639, 2036111 including attachment "Frequently_Used_Config_Parameters _in_SAP_HANA_1_3.pdf", 2186744.

ABAP
Set the TLS and Cipher settings with TA RZ10 with the profile parameters to the following values:

- ssl/ciphersuites = 544:PFS:+eAES128:-eAES_CBC::EC_HIGH:+EC_OPT

- ssl/client_ciphersuites = 544:PFS:+eAES128:-eAES_CBC::EC_HIGH:+EC_OPT
- icm/server_port_<xx> = ..., PROT=HTTPS, ..., SSLCONFIG=ssl_config_<yy>
- icm/ssl_config_<yy> = ..., CIPHERS=544:PFS:+eAES128:-eAES_CBC::EC_HIGH:+EC_OPT

Notes:

- If you have maintained the SSL configuration with the icm/ssl_config_ <yy> parameter, you must set the SSLCONFIG option from the icm/server_port_<xx> parameter to the "ssl_config_ <yy>" value (<yy> corresponding to the icm/ssl_config_ <xx> parameter).
- Parameter icm/ssl_config_<yy>: The CRED option must be specified e.g. CRED=SAPSSLSsapext.pse.
- Parameter icm/ssl_config_<yy>: The contents of the CIPHERS option are subject to the same rules and syntax that apply to the ssl/ciphersuites profile parameter.

The following must be configured for SNC:
The certificate/key for SNC must have at least the following properties when generated with sapgenpse or the TA STRUST:

- minimum sha256 (option -a)
- Bit length at least 2048 (option -s)

Set the following SNC profile parameters (TA RZ10):

- snc/enable = 1
- snc/gssapi_lib = <path and file name of SAP Cryptographic Library> e. g. /usr/sap/<SID>/SYS/exe/run/libsapcrypto.so
- snc/identity/as = p:<distinguished_name>
- snc/data_protection/min = 3
- snc/data_protection/max = 3
- snc/data_protection/use = 9
- snc/accept_insecure_gui = 0
- snc/accept_insecure_cpic = 0
- snc/accept_insecure_rfc = 0
- snc/accept_insecure_r3int_rfc = 0

When configuring SNC between the AS ABAP and other server components, you must make Access Control List Entries on the application server. To do this, you must know the SNC name used by the communication partner. SNC system access control list maintenance is done with TA SM30 in the table SNCSYSACL (view VSNCSYSACL and type=E).

SAP Web Dispatcher (internal HANA), see note 3201581 and for HANA XS Classic. Maintain the configuration file (webdispatcher.ini):

- ssl/ciphersuites = 544:PFS:+eAES128:-eAES_CBC::EC_HIGH:+EC_OPT

Please note that ssl/client_ciphersuites is not relevant for the embedded web dispatcher because the embedded web dispatcher never acts as an HTTPS client. If ssl/ciphersuites is set, sslminprotocolversion and sslciphersuites are ignored in global.ini (see below):

section [communication]

- sslminprotocolversion = TLS12
- sslCipherSuites = 544:PFS:+eAES128:-eAES_CBC::EC_HIGH:+EC_OPT

SAP HANA, SAP S/4HANA
SAP recommends using encrypted communication channels where possible (s. SAP Sec. Guide HANA)!
Configuration of the SAP-HANA-System can be done using the DBA Cockpit for SAP HANA (TA ST04 or DBACOCKPIT) within the view "Configuration -> Ini files" or within the SAP HANA Studio below of "Administration" within in the

view of configuration. Set the parameter within the according section of the global.ini (and or multidb.ini) file:

section [communication]

- sslcryptoprovider = commoncrypto (for SAP CommonCryptoLib)
- ssl = on or systempki (internal communication between hosts)
- ssl_local = on (internal communication within a host)
- sslCipherSuites = PFS:+eAES128:-eAES_CBC::EC_HIGH:+EC_OPT (Set of valid cipher for the communication)
- sslEnforce = true (external communication between hosts)
- sslminprotocolversion = TLS12
- sslmaxprotocolversion = MAX

section [cryptography] (when FIPS 140-2 compliance must be set)

- ccl_fips_enabled = true

Sektion [ldap] (for authentication via IAM)

- sslminprotocolversion = TLS12
- sslmaxprotocolversion = MAX
- sslciphersuites= PFS:+eAES128:-eAES_CBC::EC_HIGH:+EC_OPT (Set of valid cipher for the communication)

section [multidb]

- enforce_ssl_database_replication = true

section [system_replication_communication]

- ssl = on (ab Support Package 10 nicht mehr nötig (Note 2256091))
- enable_ssl = on (SSL-Kommunikation für Quell- und/oder Ziel-Replikationskanäle in einem System-Replikationsszenario)

Attention see note 2159014:
Internal communication between ABAP and HANA must also be configured on ABAP side.
Concerning for: For global.ini -> [communication] -> ssl = on (SAP HANA internal communication uses SSL).
On SAP ABAP side, encryption is enabled using the following parameters:

- dbs/hdb/connect_property = ENCRYPT=TRUE

Attention: The clients must also have encrypted access, otherwise the communication will be rejected by HANA.
This must be configured separately. Please refer to chapter 4.3.1.3 TLS/SSL Configuration on the Client of the HANA Security Guide and also the "SAP HANA Client Interface Programming Reference" in the SAPHelp.

The above will enable TLS 1.2 for all HANA ports except the following two:

- SAP HANA database lifecycle manager via SAP Host Agent (port 1128/1129)
- SAP start service (sapstartsrv) (port 5xx13/5xx14)

The settings here are the same as would be needed for ABAP and so this note should be followed: 2384290 - SapSSL update to facilitate TLSv1.2-only configurations
As mentioned in the note, you will need to set the ssl/ciphersuites and ssl/client_ciphersuites from "Profile parameter values for limiting protocol versions to strict TLSv1.2-only" in both of the above files and restart both the host agent and sapstartsrv.

- ssl/ciphersuites = 544:PFS:+eAES128:-eAES_CBC::EC_HIGH:+EC_OPT
- ssl/client_ciphersuites = 544:PFS:+eAES128:-eAES_CBC::EC_HIGH:+EC_OPT
- icm/server_port_<xx> = ..., PROT=HTTPS, ..., SSLCONFIG=ssl_config_<yy>
- icm/ssl_config_<yy> = ..., CIPHERS=544:PFS:+eAES128:-eAES_CBC::EC_HIGH:+EC_OPT

Notes:

- If you have maintained the SSL configuration with the icm/ssl_config_ <yy> parameter, you must set the

SSLCONFIG option from the icm/server_port_<xx> parameter to the "ssl_config_ <yy>" value (<yy> corresponding to the icm/ssl_config_ <xx> parameter).
- Parameter icm/ssl_config_<yy>: The CRED option must be specified e.g. CRED=SAPSSLSsapext.pse.
- Parameter icm/ssl_config_<yy>: The contents of the CIPHERS option are subject to the same rules and syntax that apply to the ssl/ciphersuites profile parameter.

SAP Web Dispatcher

If SAP Web Dispatcher is to terminate the SSL connection, it must have security information that it can use for the SSL communication(s). This information is stored in the SAP Web Dispatcher's Personal Security Environments (PSEs) as follows:

- For the connection where it is the server component, this information is stored in its SSL server PSE. The default SSL server PSE uses the file name SAPSSLS.pse and is located in /usr/sap/<SID>/<webdisp>/sec (Unix).
- If the SAP Web dispatcher also establishes an SSL connection to the AS Java, its security information to be used for that connection is stored in the SSL client PSE. The default SSL client PSE uses the file name SAPSSLC.pse and is also located in /usr/sap/<SID>/<webdisp>/sec (Unix).Install the SAP Cryptographic Library and sapgenpse executable on the SAP Web Dispatcher's host (Unix: /usr/sap/<SID>/<webdisp>).

Please see the following SAP notes: 1848999, 2110020, 2384290
Set the TLS and Cipher settings to the following values in the profile file / configuration file.

- ssl/ciphersuites = 544:PFS:+eAES128:-eAES_CBC::EC_HIGH:+EC_OPT
- ssl/client_ciphersuites = 544:PFS:+eAES128:-eAES_CBC::EC_HIGH:+EC_OPT
- icm/server_port_<xx> = ..., PROT=HTTPS, ..., SSLCONFIG=ssl_config_<yy>
- icm/ssl_config_<yy> = ..., CIPHERS=544:PFS:+eAES128:-eAES_CBC::EC_HIGH:+EC_OPT
- wdisp/ssl_encrypt = 2 (in any case pass on encrypted)

Notes:

- If you have maintained the SSL configuration with the icm/ssl_config_ <yy> parameter, you must set the SSLCONFIG option from the icm/server_port_<xx> parameter to the "ssl_config_ <yy>" value (<yy> corresponding to the icm/ssl_config_ <xx> parameter).
- Parameter icm/ssl_config_<yy>: The CRED option must be specified e.g. CRED=SAPSSLSsapext.pse.
- Parameter icm/ssl_config_<yy>: The contents of the CIPHERS option are subject to the same rules and syntax that apply to the ssl/ciphersuites profile parameter.

When you generate a certificate / key with sapgenpse at least the following properties must be present:

- minimum sha256 (option -a)
- Bit length at least 2048 (option -s)
- CA must sign certificate/key

SAProuter

The connection between the neighboring SAProuters is protected with SNC. The SAProuter can also establish SNC-encrypted communication to / between / from other components (AS ABAP, DIAG/SAPGUI, printers, external programs, AS Java).
The certificate / key for SNC must have at least the following properties when generated with sapgenpse:

- minimum sha256 (option -a)
- Bit length at least 2048 (option -s)

Check the trace files to determine which SAPcryptoLib / CommonCryptoLib and Cipher are enabled.

- AS ABAP: dev_w0
- AS Java: dev_server<n>
- ICM: dev_icm
- HANA: dev_w0 and dev_webdisp
- SAP Web Dispatcher: dev_webdisp
- SAPRouter: dev_rout

The log and trace files generated by the AS Java processes and the applications executed on AS Java are stored in the directory usr\sap\<SID>\<instance name>\j2ee\cluster\server<n>\log.The HANA log and trace files are located at /usr/sap/<SID>/HDB<instance number>/<hostname>/trace.

Requirement-ID: 39a81cec

ID: 3.22-23/i373

Req 24	Following Authenticated Encryption with Associated Data (AEAD) must be used as symmetric block cipher mode of operation: Galois-Counter-Mode (GCM), Counter with CBC-MAC (CCM).

User roles: Operation, Development, Integration

If the input data is longer than the AES input block length of 128 bit, an operation mode must be used to encrypt all the data.
GCM and CCM are Authentication Encryption with Associated Data (AEAD) modes of operation and providing confidentiality and integrity to the plain text. AEAD also provides integrity of associated data like protocol header information. GCM and CCM are specified by ISO/IEC 19772, NIST SP 800-38C and NIST SP 800-38D standards. AEAD ciphers should be preferred, as far as possible.
The classical modes CBC, CTR and OFB only provide confidentiality. The following table represents the valid applications and operation conditions for the operation modes:

Operation Mode	Confidentiality	Integrity
GCM	Yes	Yes
CCM	Yes	Yes
CBC	Yes	No
CTR	Yes	No
OFB	Yes	No

Requirements on AEAD modes:

- Requirements on Galois-Counter-Mode (GCM):
  - The authentication tag must be at least 96 bit long
  - The initialization vector (nonce) must not repeat and shall be at least 96 bit long
- Requirements on Counter Mode with CBC-MAC (CCM):
  - The authentication tag must be at least 96 bit long
  - The initialization vector (nonce) must not repeat with a same key

Remarks on data privacy:
All requirements in this document are limited to data and information security in terms of confidentiality and integrity.

Requirements on cryptographic methods to fulfill requirements on anonymization and pseudonymization with regards to data privacy are published in document “Anonymization and Pseudonymization”.

Motivation: Wrong implemented mode of operation may cause a weakening of cryptographic systems.

Implementation example: The implementation of the requirement is described in requirement 23 under the item "Implementation example".

ID: 3.22-24/i373

Req 25	If Authenticated Encryption with AEAD modes of operation is not available, one of the following confidentiality modes of operation must be used: Counter Mode (CTR), Cipher-Block Chaining (CBC) or Output Feedback (OFB).

User roles: Operation, Development, Integration

If the input data is longer than the AES input block length of 128 bit, an operation mode must be used to encrypt all the data.

AEAD ciphers should be preferred, as far as possible.
The classical modes CBC, CTR and OFB only provide confidentiality.
Therefore, they must generally be combined with mechanisms for data authentication (e.g. message authentication codes (MAC)). The mentioned "classical" confidentiality modes are specified by ISO/IEC 10116, or NIST SP 800-38A.

Requirements on confidentiality modes (refer to annex B of ISO/IEC 10116):

- Requirements on Counter Mode (CTR):
    - The counter and initialization vector (IV) are 128 bit long
    - The counter must not repeat with the same key
    - The chosen initialization vector must ensure that the counter will not repeat with the same key
- Requirements on Cipher Block Chaining Mode (CBC):
    - The initialization vector (IV) is 128 bit long and must be randomly chosen
    - The initialization vector (IV) must not be impactable or derivable
    - A secure padding scheme have to be used
- Requirements on Output Feedback Modus (OFB):
    - The initialization vector (IV) is 128 bit long and must be randomly chosen

Remarks on data privacy:
All requirements in this document are limited to data and information security in terms of confidentiality and integrity. Requirements on cryptographic methods to fulfill requirements on anonymization and pseudonymization with regards to data privacy are published in document “Anonymization and Pseudonymization”.

Motivation: The classical modes CBC, CTR and OFB only provide confidentiality, Therefore they must generally be combined with mechanisms for data authentication

Implementation example: The implementation of the requirement is described in requirement 23 under the item "Implementation example".

ID: 3.22-25/i373

Req 26	Cryptographic hash functions of the SHA-3 and SHA-2 family must be used with a minimum output length of 256 bit.

User roles: Operation, Development, Integration

The security of hash functions is driven by their output lengths, which are denoted as suffix (e.g. SHA-3-512 has an out-

put length of 512 bit)

- Standardized variants of the SHA-2 family are [ISO/IEC 10118-3, FIPS 180-4]: SHA-512, SHA-384, SHA-256.
- Standardized variants of the SHA-3 family are [ISO/IEC 10118-3, FIPS 202]: SHA-3-512, SHA-3-384, SHA-3-256.

Following hash applications are valid:

- Digital signatures
- Message Authentication Codes (MAC)
- Mask generation functions (MGF)
- Key Derivation Function (KDF) and Password-Based Key Derivation Functions (PBKDF)
- Pseudo Random Number Generators (PRNG)
- Integrity protection

Remark on authenticiy and confidentiality:
Hash function don't provide confidentiality in terms of encryption and also no protection against manipulation (active attacks). Hash functions are public known keyless one way functions. An attacker easily manipulate messages and calculate a valid hash of the modified message. If authenticity of a message must be guaranteed, a keyed Message Authentication Code (e.g. HMAC) must be used.

Remark on SHA-3-224/SHA-224:
SHA-224/SHA-3-224 may only be used for Message Authentication Codes (MAC) and as primitive of Key Derivation Functions (KDF) for legacy systems and should be replaced by stronger hash functions with a longer output length (>=256 Bit).

References:
[TR2102-1] BSI-Technical Guideline: Cryptographic Mechanisms - Recommendations and Key Lengths, page 37, February 20

Motivation: Cryptographic hash functions are used in many use cases and are basis of several cryptographic schemes, like digital signatures.

Implementation example: The implementation of the requirement is described in requirement 23 under the item "Implementation example".

ID: 3.22-26/i373

Req 27	The following Message Authentication Codes (MAC) must be used: HMAC (based on SHA-2/3 hash functions) or CMAC (based on AES).

User roles: Operation, Development, Integration

Message Authentication Codes (MAC) protect the integrity of messages in security protocols as IPSec or TLS/SSL. Moreover MACs are used in other cryptographic applications, such as Key Derivation Functions (KDF).

The following table represents the valid algorithms and parameter:

MAC	Approved Algroithms	Special Operation Conditions / Remarks

HMAC [ISO/IEC 9797-2, FIPS 198]	SHA-512 SHA-3-512 SHA-384 SHA-3-384 SHA-256 SHA-3-256	The key lengths must be at least 128 bit long. The length of the MAC must be at least 64 bit long, however it is recommended to use MACs not shorter than 96 bit.
CMAC [ISO/IEC 9797-2, NIST SP 800-38B]	AES-256 AES-192 AES-128	The key length will be defined by the choosen AES variant (128/192/256 bit). The length of the MAC must be at least 64 bit long, however it is recommended to use MACs not shorter than 96 bit.

Remark on CMAC:
Simple implementations of CBC-MAC are not resistant against extenstion attacks. One in [ISO/IEC 9797] or [NIST SP 800-38B] standardized secure variant of CBC-MAC is denoted as CMAC (MAC Algorithm 5 in [ISO/IEC 9797]).

Remark on GMAC:
The authenticated encryption mode GCM is the underlying basis of GMAC contructions. GMAC is a standalone MAC (not part of GCM) and only has a similar security as HMAC and CMAC in particular cases [BSI TR-02102-1]. Thus GMAC is not recommended as MAC algorithm.

Restrictions on SHA-224/SHA-3-224/SHA-1:
The usage of SHA-224/SHA-3-224/SHA-1 as HMAC is only permitted for legacy systems and should be replaced by stronger hash algorithms which creating outputs not less than 256 bit.

Remarks on data privacy:
All requirements in this document are limited to data and information security in terms of confidentiality and integrity. Requirements on cryptographic methods to fulfill requirements on anonymization and pseudonymization with regards to data privacy are published in document “Anonymization and Pseudonymization”.

Motivation: Message Authentication Codes (MAC) protect the integrity of messages in security protocols as IPSec or TLS/SSL. Moreover MACs are used in other cryptographic applications, such as Key Derivation Functions (KDF).

Implementation example: The implementation of the requirement is described in requirement 23 under the item "Implementation example".

ID: 3.22-27/i373

Req 28	Physical random number generators or following strong pseudo-random number generators (deterministic sources) with sufficient entropy must be used to generate cryptographic keys for algorithms: HMAC-DRBG, Hash-DRBG or CTR-DRBG.

HMAC-DRBG, Hash-DRBG or CTR-DRBG are based on the specifications of NIST SP800-90A and ISO18031. If a pseudo-random number generator (deterministic random bit generator = DRBG) is used to generate a cryptographic key, a non-deterministic value must be used to calculate the seed value. This non-deterministic source must have a sufficient high entropy.

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 (stochastic analysis).

Implementation example: The SAPcryptoLib / CommoncryptoLib is to install and to configure. Download the latest SAP Cryptographic Library from the SAP Software Download Center.

ID: 3.22-28/i373

Req 29	Symmetric keys must be protected against unauthorized access and must neither transferred nor stored unencrypted outside the system boundaries.

User role: Development, Operations

Basic requirements on data access:
The data access on key material must be restricted to necessary system services and user groups inside and outside of the system boundaries.

Requirements on transport and storage of symmetric keys:
AES-Synthetic Initialization Vector (AES-SIV) or AES-Kexwrap must be used to transport and storage of symmetric keys (e.g. Pre-shared keys) outside the system boundaries.

This requirement is applicable for the transport of cryptographic keys over an unsecure channel or storage of key material on an unprotected storage medium. The methods above ensuring confidentiality of the key and often called “key-wrap”. Both a specified in [RFC5297] and [SP800-38F]. Keys may also be stored and transported via secure hardware (e.g. HSM) or on secure elements (e.g. Smart Card).

Motivation: Symmetric keys must be protected against unauthorized access and must neither transferred nor stored unencrypted outside the system boundaries.

Implementation example: Limitation of access to the key file and the directory in which the key(s) are located:

- Directory (Unix): 700
- Key file (Unix): 600

The exchange of the key and the key password must be carried out separately and secured, e.g. PGP/SMIME, E2E encrypted messenger, ...

ID: 3.22-29/i373

Req 30	Key agreement protocols must fulfill Perfect Forward Secrecy (PFS) properties and must be protected against Man-in-the-middle attacks (e.g. Authenticated Ephemeral Elliptic Curve Diffie Hellman).

Asymmetric key agreements basically are based on (Elliptic Curve) Diffie-Hellman key exchange (e.g. Ephemeral EC-DH or MQV). The advantage of this type of key exchange is, that the complete key will never be transferred over the communication channel. Ephemeral key exchange means, that the session secrets will be deleted after usage. Perfect Forward Secrecy (PFS) means that a recorded ciphertext can’t be decrypted afterwards, regardless whether the long-term key (e.g. private RSA key) is known by an attacker.
There are different key exchange protocols standardized in ISO/IEC11770-3. Each development of PFS key exchange protocols shall follow BSI requirements [BSI TR 02102-1].

Remarks on standardized DH groups:
The generation of secure system parameter (generators, and finite groups) is a complex, sensitive and sophisticated task. As a result of this, developers often using standardized system parameters [RFC5114]. The minimum lengths or groups and parameter must be considered, as they are not part of the IANA numbering of the DH groups.

Motivation: The key agreement solves the key exchange problem of symmetric cryptography. With Perfect Forward Secrecy, the attacker is not able to break the confidentiality of transferred cryptograms if the long-term key (RSA private key) has been compromised. Man-in-the-middle attacks will be avoided by using certificate based authentication of the peers.

Implementation example: The implementation of the requirement is described in requirement 23 under the item "Implementation example".

ID: 3.22-30/i373

Req 31	Private keys of asymmetric cryptographic algorithms must be protected against unauthorized access and must not transferred or stored unencrypted outside system boundaries.

Basic requirements on data access:
The data access on key material must be restricted to necessary system services and user groups inside and outside of the system boundaries.
Private keys are used for authentication, decryption of encrypted messages and for digital signatures of messages. A digital signature (= comparable with a personal signature) can only be generated 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 of individual persons must be protected against any misuse via a personal secret (PIN or Password) as well.

Remarks on remote signatures:
Remote signatures may only be created by certified trust provided (trust center / trust services like www.telesec.de). They must reveal its compliance with the eIDAS standards and the certificate must be verified and renewed on a regular basis.

Motivation: If an attacker gets access to a private key, he is able to impersonate himself as the original user / system.

Implementation example: Limitation of access to the key file and the directory in which the key(s) are located:

- Directory (Unix): 700
- Key file (Unix): 600

The exchange of the key and the key password must be carried out separately and secured, e.g. PGP/SMIME, E2E encrypted messenger, ...

ID: 3.22-31/i373

Req 32	Kryptographische Langzeitschlüssel müssen eine der Anwendung angemessene begrenzte Gültigkeit besitzen.

Nutzerrolle: Betrieb, Entwicklung, Integration
Generell wird zwischen Langzeitschlüsseln und kurzlebigem Schlüsselmaterial, wie Session Keys unterschieden. Langzeitschlüssel werden über einen längeren Zeitraum für kryptographische Operationen verwendet (z.B. TLS Serverzertifikate oder SSH Schlüssel), wohingegen Session Keys nur für eine Protokollsitzung verwendet und danach gelöscht werden. Der Begriff Langzeitschlüssel wird nicht über eine konkrete Zeitspanne definiert, sondern über die mehrfache Verwendung desselben Schlüsselmaterials.

Die Begrenzung der Gültigkeit kryptographischer Langzeitschlüssel beschränkt die Möglichkeiten der Kryptoanalyse, die Auswirkungen im Fall einer K ompromittierung des privaten Schlüssels sowie das Risiko, dass der zugrundeliegende Algorithmus oder die Schlüssellänge während des Gültigkeitszeitraums des Schlüssels als unsicher klassifiziert werden.

Die Gültigkeit asymmetrischer Langzeitschlüssel kann z.B. durch den Gültigkeitszeitraum des zugehörigen digitalen Zertifikats definiert werden. Dabei ist zu beachten, dass bei einer Zertifikatserneuerung nicht zwingend der Schlüssel erneuert werden muss (hierbei ist die Policy der ausstellenden Certificate Authority zu beachten). Grundsätzlich sollte die gesamte Gültigkeit von Schlüsseln (ggf. verteilt auf mehrere Zertifikate), die für Endanwendungen und damit häufig verwendet werden, zwischen einem und drei Jahren liegen.
In einzelnen, kontrollierbaren Fällen mit geringer Verwendung wie beispielsweise dem Schlüssel einer Certification Authority dürfen längere Gültigkeiten gewählt werden. Im Notfall (z.B. einer Kompromittierung) ist der entsprechende Schlüssel vorzeitig zu sperren.

Die folgenden Gültigkeitszeiträume werden empfohlen:

- Endentitäten (z.B. Anwender, TLS-Server): 1-3 Jahre
- Zwischenzertifizierungsstellen (Subordinate/ Intermediate CA): 8-15 Jahre
- Wurzelzertifizierungsstellen (Root CA): 20-30 Jahre

Es kann erforderlich sein, dass Schlüssel auch über ihre Gültigkeit hinaus verwendet werden müssen. Dies kann für öffentliche Schlüssel zur Prüfung von Signaturen in Langzeitanwendungen oder für private Schlüssel zur Entschlüsselung von Langzeitdaten wie beispielsweise E-Mails der Fall sein.

Motivation: Durch beschränkte Gültigkeitszeiträume von asymmetrischen Langzeitschlüsseln kann die Sicherheit der verwendeten Verfahren/Schlüssellängen in regelmäßigen Abständen angepasst werden.

Implementation example: TLS-Web Server certifikat with valid time from one to three years.

Client -> SAP Web Dispatcher => valid CA neccessary

SAP Web Dispatcher -> Applikation (OnPremise) -> self signed possible

SAProuter -> SAProuter => self signed possible due to password alternative - no request by CA.

runtimes:
2048 = 1 year
3072 = 3 years
4096 = 5 years (only self signed ones)

ID: 3.22-32/i373

Req 33	The security of cryptographic long-term keys must be checked regularly and adapted if necessary.

User roles: Operation, Development, Integration

Due to the increase in computing power and new attacks, the key lengths of the used cryptographic methods must be adjusted from time to time. In case of special progress in cryptanalysis (e.g. through universal quantum computers) even an adaption the cryptographic method may be necessary. This concerns in particular cryptographic long-term keys such as key pairs that were issued for digital certificates or SSH keys. At the latest at the renewal of the certificates it must be checked whether the used method and key length are still considered secure. For example, in BSI TR-02120-1 (as of 2019) the usage of 3072-bit keys instead of 2048-bit keys as from 2023 is recommended.

Motivation: Advances in computing might require the adaptation of cryptographic key sizes.

Implementation example: Checking whether the key length of long-term keys is still sufficient when a certificate is renewed.

ID: 3.22-33/i373

Req 34	Cryptographic keys (symmetric and asymmetric) must be renewable

User role: Operation, Development

Active attacks and technical developments in computing technology are the main reasons to substitute key material, e.g. if the encryption system is upgraded from AES-128 to AES-256 in order to be quantum-resistant. Moreover, the security of cryptographic systems relies on the secrecy of the key. Thus, the usage of static keys might be a high risk if the static keys become compromised. Rekeying is a mandatory counter-measure which is implemented in modern encryption systems. Further information can be found in [ISO/IEC JTC1 SC27 SD 12].

Renewed keys must not be derived from active or previously used key material (e.g. session key). The operator of a system must to define a suitable lifetime of the keys.

Motivation: Definition of a key life time paramter of IPSec session keys in the system configuration of an IPSec gateway.

Implementation example: The requirement is fulfilled when the SAP Cryptographic Library or T-Secure is used.

ID: 3.22-34/i373

Req 35	Compromised keys or certificates must be deactivatable / revocable.

User roles: Operation, Development, Integration

If a private key compromise is detected, the related digital certificate must be revoked immediately. It must no longer be possible to authenticate with the private key. The revocation procedures must be integrated into other regulating processes.

Motivation: If attackers get possession of a private key (e.g. "private RSA key"), they can assume the identity of the victim and thus use applications or functions without authorization, for example.

Implementation example: Use of a CA and the associated options for certificate management certificates (issuing, exchanging and withdrawing).

- Automated process for blocking/revoking unused certificates using the Certificate Revocation List (CRL) as part of the Certificate Management Protocol (CMP)

ID: 3.22-35/i373

Req 36	Key material which is no more used must be destroyed, or rendered useless immediately and irrecoverable by secure methods.

User role: Operation

Following electronic data ereasure procedures are recommended:

- DoD-5220.22-M (ECE)
- Bruce-Schneier-algorithm
- Peter-Gutmann-algorithm
- ATA-"Secure-Erase" (SSD or SSHD disks)- all bits must be set to logical 1

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" AE

Implementation example: - Overwriting of the key storage using DoD-5220.22-M (ECE) algorithm
- Withdrawal of a web server certificate
- Automated process for blocking/revoking unused certificates using the Certificate Revocation List (CRL) as part of the Certificate Management Protocol (CMP)

ID: 3.22-36/i373

Req 37	Following cryptographic algorithms for storage and verification of password hashes may only be used: scrypt [RFC 7914], bcrypt, Argon2, SHA-512-crypt / Crypt(3), SHA-256-crypt / Crypt(3), or PBKDF2 [RFC8018].

Passwords used for authentication must always be stored as a password hash.
Secure algorithms for password hash storage and verification:

Algorithm	Maximum Password Length	Salt Length	Security Parameter / Work Factor	Additional Parameter
scrypt	80 characters	128 bit	N=2^18; r=4; p=1	none
bcrypt	50 characters	128 bit	2^n with n=13	none

Argon2	80 characters	128 bit	n=256; 2^m with m=10, p=1	none
SHA-512-crypt / Crypt(3)	80 characters	96 bit (16 characters)	n=640.000	id=6
SHA-256-crypt / Crypt(3)	80 characters	96 Bit (16 characters)	n=640.000	id=5
PBKDF2	80 characters	128 bit	n=640.000	none

Remarks on passwords:
The current rule set on passwords can be found in the document “3.01 Technical Baseline NT/IT”.

Remarks on the security parameter:
All parameter sets are defined according to security best practices in order to be resistant against offline attacks by speed down the hash generation rate.

Remark on SHA-512/256-crypt:
Generic hash function must not be used for password hashing, due to the fact that those are not resistant against special attack methods (GPU, ASIC/FPGA) which can can calculate a large number of hash values per second. Thus SHA-512-crypt is a specific algorithm based on SHA-512 in order to compute and verify passwort hashes.

Remark on key derivation:
Recommendations on password-based key derivations can be found in section "Key Management"

References:
[1] https://pthree.org/2016/06/28/lets-talk-password-hashing/ (last view: 04.10.2018)
[2] https://www.owasp.org/index.php/Password_Storage_Cheat_Sheet (last view: 04.10.2018)
[3] https://veggiespam.com/painless-password-hash-upgrades/ (last view: 04.10.2018)
[4] https://arstechnica.com/information-technology/2013/09/long-passwords-are-good-but-too-much-length-can-be-bad-for-security/ (last view: 04.10.2018)
[RFC7914] C. Percival, Tarsnap, S. Josefsson, SJD AB: RFC 7914, The scrypt Password-Based Key Derivation Function, 2016
[RFC8018] K. Moriarty, Ed., Dell EMC, B. Kaliski, Verisign, A. Rusch, RSA: RFC 8018, PKCS #5: Password-Based Cryptography Specification Version 2.1, 2017
[TBS] Deutsche Telekom AG: Technical Baseline Security for IT-/NT-Systems, version 2.2, 2016

Motivation: Only special password hash algorithms are resistant against online and offline attacks on password hash tables.

Implementation example: Example for the ABAP-Stack:
For local login, the password is saved as a HASH. The SAP default is not sufficiently secure. The requirement is therefore not met. The setting is made using the profile parameter **login/password_hash_algorithm** (via TA RZ10).

SAP-Default: encoding=RFC2307, algorithm=iSSHA-1, iterations=1024, saltsize=96

In order to fulfill the requirement, the parameter must be set as follows: "algorithm=iSSHA-256, iterations=15000 (=max) saltsize=128"
Note: Depending on the performance of the system, the value of the "iterations" can be chosen to be less than 1500.

ID: 3.22-37/i373

Req 38	The following signature algorithms must be used: EC-DSA, RSA-PSS, DSA/DSS.

User roles: Development, Integration
Signature algorithms are used to verify the authenticity of the sender and the integrity of a signed message. The verification is performed by means of the public key.

Signature algorithms with appendix (i.e. signature is attached to the message) always need a cryptographic hash

function.

The following table represents the valid parameters for digital signatures:

Signature Algorithm	Minimum Length of the security parameter	Permitted hash algorithms
EC-DSA [ISO/IEC 14888-3], [FIPS 186-4]	250-bit (the order of the generator base point on the elliptic curve shall not be shorter than 250)	SHA-3, SHA-2 with an output length 256-bit
RSA-PSS [ISO/IEC 14888-2], [FIPS 186-4]	3000-bit RSA modulus (n) with a public key e > 2^16	SHA-3, SHA-2 with an output length 256-bit
DSA / DSS [ISO/IEC 14888-3], [FIPS 186-4]	3000-bit prime p (defined on finite field); 250-bit prime q	SHA-3, SHA-2 with an output length 256-bit

Further Information on digital signatures can be found in [BSI TR-02102-1].

Restrictions on SHA-224/SHA-3-224:
SHA-224/SHA-3-224 may only be used in legacy Systems and have to be replaced by a stronger hash algorithm which has a Output length not less than 256 bit.

Restrictions on RSA PKCS#1 v1.5:
RSA PKCS#1 v1.5 may only be permitted on legacy Systems and should be replaced by state of the art signature algorithms.

Approved elliptic curves:

- brainpoolP512r1
- NIST P-521
- brainpoolP384r1
- NIST P-384
- brainpoolP320r1
- brainpoolP256r1
- NIST P-256
- Curve25519/Ed25519

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: The implementation of the requirement is described in requirement 23 under the item "Implementation example".

ID: 3.22-38/i373

Req 39	Certificates must be from a trustworthy certification authority.

In the context of PSM consideration, it may be possible to use self-signed certificates/keys.

Motivation: The authenticity of a server (web service) can only be guaranteed by using certificates by a valid and trustworthy certification authority.

Implementation example: A trustfull CA is e. g. the TeleSec.

ID: 3.22-39/i373

2.4. Transport Layer Security (Sample for requirements 36 - 40 in req. 17 inserted)

Req 40	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: TLS version 1.2 or 1.3 must be used.

Implementation example: The implementation of the requirement is described in requirement 23 under the item "Implementation example".

ID: 3.22-40/i373

Req 41	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

Prio	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

Prio	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.

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_SHA384TLS_CHACHA20_POLY1305_SHA256TLS_AES_128_GCM_SHA256TLS_AES_128_CCM_SHA256	RFC 8446RFC 8446RFC 8446RFC 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 2022-01, 24.01.2022
[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: The implementation of the requirement is described in requirement 23 under the item "Implementation example".

ID: 3.22-41/i373

Req 42	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
brainpoolP384r1	27	RFC 7027
secp384r1	24	RFC 8422
brainpoolP256r1	26	RFC 7027
secp256r1	23	RFC 8422
ffdhe4096	258	RFC 7919
ffdhe3072	257	RFC 7919

Remark on group 256:
Diffie Hellman group 256 (IANA-No.256) has a ley 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).

Remarks 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 2022-01, 24.01.2022
[2] SOG-IS Crypto Evaluation Scheme: Agreed Cryptographic Mechanisms, v1.2, January 2020

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

Implementation example: The implementation of the requirement is described in requirement 23 under the item "Implementation example".

ID: 3.22-42/i373

Req 43	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 have to 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 until the year 2023 [BSI TR 02102-1] 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 TLS certificates should not exceed 3 years.

References:
[BSI TR 02102-1] Bundesamt für Sicherheit in der Informationstechnik: Cryptographic Mechanisms: Recommendations and Key Lengths, TR-02102-1, Version 2022-01, 24.01.2022
[1] Bundesamt für Sicherheit in der Informationstechnik, TR-02102-2, Version 2022-01, 24.01.2022
[2] SOG-IS Crypto Evaluation Scheme: Agreed Cryptographic Mechanisms, v1.2, January 2020

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

Implementation example: Use of certificates from a recognized CA

ID: 3.22-43/i373

Req 44	The security of TLS webserver implementations and configurations must be verified regularly.

User roles: Operation, Development, Integration

Secure cipher suites consisting all essential parts of a modern security protocol:

- Authentication scheme
- Key Exchange algorithm
- Encryption algorthm
- MAC algorithm

The level of security is mainly driven by the concrete implementation and applied software. Furthermore the security depends on the used protocol version. Therefore it is mandatory to select applied cipher suites based on security best practices.

Recommended sources and test tools are:

- Cipher Suite Info (https://ciphersuite.info/)
- SSL Labs (https://www.ssllabs.com/ssltest/)
- Testssl (https://testssl.sh/)
- DTSP Internal Scan Platform (https://dtsp.telekom-dienste.de/)

Motivation: Only secure cipher suites reduces the attack surface and protect against downgrade attacks.

Implementation example: The implementation of the requirement is described in requirement 23 under the item "Implementation example".

ID: 3.22-44/i373

2.5. Logging

Req 45	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

ID: 3.22-45/i373

Req 46	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.

ID: 3.22-46/i373

Req 47	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

ID: 3.22-47/i373

Req 48	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.

Implementation example: For SAP the corresponding log data must be forwarded to an external system immediately after they are generated (real time).

Following security-relevant logs (as an example for ABAP) must be stored on a separated and secured system:

- SAP Security Audit Log
- System Log
- Gateway Log
- HTTP Server and Client Log
- Business Transaction Log

The SAP Enterprise Thread Detection (ETD) could be used for storing and administration of the log files.

ID: 3.22-48/i373

Req 49	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

ID: 3.22-49/i373

Req 50	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

2.6. Authorization Management

Req 51	The use of system functions that require protection as well as access to internal or confidential data must not be possible without prior authentication and authorization.

The use of functions of the system that require protection as well as access to data classified as internal or confidential must only be possible after the user has been uniquely identified and successfully authenticated by means of the user name and at least one authentication attribute. In addition, it must be verified that the user is authorized to access the affected functions and data within the user role assigned to him or her in the system.

An exception to this are functions and data that may be used publicly without restriction; for example, the area of a website on the Internet where only public information is provided.

Examples of features that require prior authentication include:

- Remote access to network services (such as SSH, SFTP, web services)
- Local access to the management console
- Local use of operating system and applications

Examples of authentication features that can be used:

- Passwords
- cryptographic keys or certificates (e.g., in the form of smart cards)

This requirement also applies without restriction to any machine access to the system (here the implementation is usually carried out by using so-called M2M - "Machine-to-Machine" - user accounts).

Motivation: The unambiguous authentication and authorization of access to a system are elementary to protect functions and data from misuse.

ID: 3.22-51/i373

Req 52	The roles for all SAP users must be implemented via GRC.

The roles for all SAP users must be implemented via GRC.

Motivation: Legal requirements and guidelines can be implemented centrally with little effort.

ID: 3.22-52/i373

Req 53	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.

ID: 3.22-53/i373

Req 54	User accounts must be protected with at least one authentication attribute.

All user accounts in a system must be protected against unauthorized use.

For this purpose, the user account must be secured with an authentication attribute that enables the accessing user to be unambiguously authenticated. Common authentication attributes are e.g.:

- passwords, passphrases, PINs (factor KNOWLEDGE: "something that only the legitimate user knows")
- cryptographic keys, tokens, smart cards, OTP (factor OWNERSHIP: "something that only the legitimate user has")
- biometric features such as fingerprints or hand geometry (factor INHERENCE: "something that only the legitimate user is")

The authentication of users by means of an authentication attribute that can be faked or spoofed by an attacker (e.g. telephone numbers, IP addresses, VPN affiliation) is generally not permitted.

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

If the system and the application scenario support it, multiple independent authentication attributes should be combined if possible in order to achieve an additional increase in security (so-called MFA or Multi-Factor-Authentication).

Motivation: User accounts that are not protected by appropriate authentication attributes can be abused by an attacker to gain unauthorized access to a system and the data and applications stored on it.

ID: 3.22-54/i373

Req 55	Privileged user accounts must be protected with at least two authentication attributes from different factors.

A privileged user account is a user account with extended authorizations within a system. Extended authorizations enable access to configuration settings, functions or data that are not available to regular users of the system. In direct dependence on the special tasks that are carried out via a privileged user account within a system, the assigned extended authorizations can be specifically restricted or include completely unrestricted system access.

Examples of privileged user accounts:

- Accounts for administration, maintenance or troubleshooting tasks
- Accounts for user administration tasks (e.g. creating/deleting users; assigning permissions or roles; resetting passwords)
- Accounts that are authorized to legitimize, initiate or prevent business-critical processes
- Accounts that have access to data classified as SCD (Sensitive Customer Data) in the interests of Group Deutsche Telekom, its customers or the public
- Accounts that have extensive access to data defined as "personal" according to the EU-GDPR (e.g. mass retrieval of larger parts or the complete database)

A single authentication attribute for privileged user accounts with their extended authorizations is usually no longer sufficient.

In order to achieve an adequate level of protection, at least two mutually independent authentication attributes must be used. The authentication attributes must come from various factors (knowledge, ownership, inherence). A combination of authentication attributes of the same factor (e.g. two different passwords) is not permitted

This approach is commonly referred to as MFA (Multi-Factor Authentication).
A specific form of MFA is 2FA (2-factor authentication), which combines exactly two authentication attributes.

Motivation: Privileged user accounts represent an increased risk to the security of a system. If an attacker successfully compromises such a user account, he receives extensive authorizations with which he can bring the system or system parts under his control, disrupt system functions, view/manipulate processed data or influence business-critical processes. The combination of multiple authentication attributes of different types significantly minimizes the risk of a user account being compromised.

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

- smartcard (e.g. MyCard) plus PIN
- private key plus passphrase
- classic password plus hardware token for the generation of OTPs

ID: 3.22-55/i373

Req 56	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.

ID: 3.22-56/i373

Req 57	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).

ID: 3.22-57/i373

Req 58	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.

ID: 3.22-58/i373

Req 59	Sessions must be protected against unauthorized takeover ("session hijacking").

Interfaces that provide session functionality to the system must implement technical measures to prevent a legitimate user's session from being taken over and continued by an unauthorized third party.

Such protection can be achieved, for example, by implementing a combination of the following options that makes sense for the specific system:

- At the transport layer: Use of the TCP protocol (with its sequence numbers) and corresponding filter lists
- At the session layer: Use of the TLS Protocol
- At the application layer: Negotiation of a random secret session key between sender and receiver to authorize

all session traffic (e.g. session ID, session cookie, session token)

- Use of cryptographic methods to protect session keys from eavesdropping or modification attacks

Motivation: Unprotected sessions can potentially be hijacked and continued by an attacker in order to exercise unauthorized access to the system in the context of the affected user.

ID: 3.22-59/i373

Req 60	The system must allow users to log out of their current session.

The system must have a feature that enables the logged-in user to log out at any time. It must not be possible to resume a logged-out session without re-authenticating the user.

Motivation: A user must retain complete control over the sessions he has established in order to be able to terminate his access to a system at any time according to the situation and thus protect data and functions exposed via this access. In addition, the user must be able to assume that sessions specifically terminated by him cannot subsequently be resumed and continued by unauthorized third parties.

ID: 3.22-60/i373

Req 61	Sessions must be automatically terminated after a period of inactivity adapted to the intended use.

It is necessary that sessions on a system are automatically terminated after a specified period of inactivity.

For this reason, a time-out for sessions must be set. The time period to be selected here depends on the use of the system and, if applicable, the physical environment. For example, the time-out for an application in an unsecured environment must be shorter (a few minutes) than the time-out for an application used by operations personnel for system monitoring tasks in an access-protected area (60 minutes or more).

Motivation: For an open but unused session, there is a risk that an illegitimate user may take over and continue it unnoticed in order to exercise unauthorized access to the system and the data contained therein on behalf of the affected user.

ID: 3.22-61/i373

Req 62	If a password is used as an authentication attribute, it must have at least 12 characters and contain three of the following categories: lower-case letters, upper-case letters, digits and special characters.

A system may only accept passwords that comply with the following complexity rules:

- Minimum length of 12 characters.
- Comprising at least three of the following four character categories:
    - lower-case letters
    - upper-case letters
    - digits
    - special characters

The usable maximum length of passwords shall not be limited to less then 25 characters. This will provide more freedom to End Users when composing individual memorizable passwords and helps to prevent undesired behavior in password handling.

When a password is assigned, the system must ensure that the password meets these policies. This must be preferably enforced by technical measures; if such cannot be implemented, organizational measures must be established.
If a central system is used for user authentication [see also Root Security Requirements Document[i] "3.69 IAM

(Identity Access Management) - Framework"], it is valid to forward or delegate this task to that central system.

Permissible deviation in the password minimum length

Under suitable security-related criteria, conditions can potentially be identified for a system that enable the minimum password length to be reduced:

- It is generally permissible to reduce the minimum password length for systems that use additional independent authentication attributes within the authentication process in addition to the password (implementation of 2-Factor or Multi-Factor Authentication).
- Any reduction in the minimum password length must be assessed individually by a suitable technical security advisor (e. g. a PSM from Telekom Security) and confirmed as permissible. In the assessment, the surrounding technical, organizational and legal framework parameters must be taken into account, as well as the system-specific protection requirements and the potential amount of damage in the event of security incidents.
- The absolute minimum value of 8 characters length for passwords must not be undercut.

Motivation: Passwords with the above complexity offer contemporary robustness against attacks coupled with acceptable user friendliness. Passwords with this level of complexity have proven their efficiency in practice. Trivial and short passwords 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.

Implementation example: Example for ABAP:
login/min_password_lng = 12
login/min_password_lowercase = 1
login/min_password_uppercase = 1
login/min_password_digits = 1
login/min_password_specials = 1
login/password_compliance_to_current_policy = 1

ID: 3.22-62/i373

Req 63	If a password is used as an authentication attribute for technical accounts, it must have at least 30 characters and contain three of the following categories: lower-case letters, upper-case letters, digits and special characters.

Technical user accounts are characterized by the fact that they are not used by people. Instead, they are used to authenticate and authorize systems to each other or applications on a system.

A system must only use passwords for technical user accounts that meet the following complexity:

- Minimum length of 30 characters
- Comprising at least three of the following four character categories:
    - lower-case letters
    - upper-case letters
    - digits
    - special characters

Motivation: Due to their use in machine-to-machine (M2M) communication scenarios, technical user accounts are often equipped with privileges that can be of high interest to an attacker to compromise infrastructures. Without mechanisms of extensive compromise detection, the risk of a password being determined or broken by an attacker can increase significantly over time. A significant increase in password length counteracts these risks and can also be implemented particularly easily in M2M scenarios, since handling a very long password is not a particular challenge for a machine (as opposed to a person).

ID: 3.22-63/i373

Req 64	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 is used, it is valid 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.

ID: 3.22-64/i373

Req 65	If a password is used as an authentication attribute, it must be changed after 12 months at the latest.

The maximum permitted usage period for passwords is 12 months.
If a password reaches the maximum permitted usage period, it must be changed.

For this purpose, the system must automatically inform the user about the expired usage period the next time he logs on to the system and immediately guide him through a dialog to change the password. Access to the system must no longer be permitted without a successfully completed password change.
For technical user accounts (M2M or Machine-2-Machine), which are used for the authentication and authorization of systems among themselves or by applications on a system, automated solutions must also be implemented to comply with the permitted usage period for passwords.

Alternatively, if such an automatic mapping of the process for changing the password cannot be implemented, an effective organizational measure must be applied instead, wich ensures a binding manual password change at the end of the permissible period of use.

Motivation: Unlike more modern authentication attributes, passwords are easier to attack. Without specific measures for reliable, technically automated detection of compromises, the risk of a password being discovered or broken by an attacker can increase considerably over time.

Implementation example: Example ABAP setting the parameter:

- login/password_expiration_time = 365
- rfc/reject_expired_passwd = 1

ID: 3.22-65/i373

Req 66	If a password is used as an authentication attribute, the reuse of previous passwords must be prevented.

A history of the previously used passwords must be recorded for each user account. When a password change is initiated for a user account, the new password must be compared with this password history. If the reuse of a password is detected, the password change must be rejected. This validation process must be implemented in the system on the basis of technical measures. If a central IAM system is used for user authentication, the implementation can be forwarded to the central IAM system or outsourced there [see also Root Security Requirements Document[i] "3.69 IAM (Identity Access Management) - Framework"].

In general, the password history should ensure that a password that has already been used can never be used again.

However, due to technical limitations, a password history cannot be recorded indefinitely in many IT/NT products. In this case, the following basic rules must be observed:

- a password that has already been used must not be reusable for a period of at least 60 days (measured from the point in time at which the affected password was replaced by another)
- in systems in which the period of at least 60 days cannot be implemented, the longest possible period must be configured. In addition, it must be confirmed by a Project Security Manager (PSM) that the configured period is still sufficient in the overall context of the system with regard to the security requirement.

Annotation:
Some IT/NT products do not offer any technical configuration parameters with which the password history can be linked directly to a time period, but only allow the definition of the number of passwords to be recorded. In such cases, the time period can alternatively be ensured by linking the following, usually generally available configuration parameters. Within the resulting policy, a user can only change his password once a day and, due to the number of passwords recorded, can reuse an old password effectively after 60 days at the earliest.

- Minimum Password Age: 1 day
- Password History: Record of the last 60 passwords used

With this implementation variant, it should be noted that the minimum age for the password should not be more than one day in order not to inappropriately restrict the user with regard to the fundamental need to be able to change the password independently at any time.

Motivation: Users prefer passwords that are easy to remember and often use them repeatedly over long periods of time when the system allows. From the user's point of view, the behavior is understandable, but effectively leads to a considerable reduction in the protective effect of this authentication parameter. With adequate knowledge of the user or information obtained from previous system compromises, an attacker can gain access to supposedly protected user accounts. Particularly in situations in which new initial passwords are assigned centrally as part of an acute risk treatment, but users change them immediately to a previous password for the sake of simplicity, there is a high risk that an attacker will resume illegal access. It is therefore important to prevent users from reusing old passwords.

Implementation example:

Example sets for Windows password-History = 24 , Linux Passwort-History = 60, SAP-Netweaver ABAP = login/password_history_size = 5

ID: 3.22-66/i373

Req 67	If passwords are used as an authentication attribute, those must be stored using a suitable and approved "Password Hashing" method to protect against offline-attacks like brute force or dictionary attacks.

This requirement relates to the storage of passwords in all types of user databases, as used in this system, in order to authenticate incoming access (local or remote) by users or other systems.

If an attacker obtains the copy of a user database of the system, he is able to bring it into a fully independent environment and utilize automatized dictionary or brute force attacks to determine contained passwords. Specialized tools in combination with high computing power allow for high cracking rates in a relatively short period of time, if protective measures are insufficient. Due to the independency from the source system, such an offline attack happens unnoticed.

The following countermeasure must be implemented, since this ensures best possible protection against offline attacks:

- passwords must be stored using a cryptographic one-way function ("Password Hashing") which is suitable for that purpose and verifiably secure as matters stand

Please Note:
valid password hashing algorithms are described in Security Requirement Catalog "3.50 Cryptographic Algorithms and Security Protocols".

Explicitly NOT PERMISSIBLE is:

- to store passwords in cleartext
- to store passwords in any format which can be directly backcalculated
- to store passwords using reversible encryption

Please Note:
In this context, "directly backcalculatable formats" refers to those that simply encode the password, without involving a secret key in the transformation process. Since the password will no longer show up as original cleartext after it has been processed, those formats may easily be mistaken to provide confidentiality. Effectively, they do not offer any protection. The enconding is fixed and therefore an attacker can easily make use of it to compute the original cleartext password from the encoded string.
Examples for directly backcalculatable formats are: "base64", "rot13"
"Reversible" are all encryption methods which, using the appropriate key, enable encrypted content to be transformed back into the original content. Accordingly, with reversible encryption there is always the challenge of keeping the key secure and protecting it from unauthorized access. Reversibility is a required fundamental property in many areas of encryption applications, e.g. for transferring confidential messages, but it is counterproductive for storing passwords: a stored password must remain comparable by means of technical methods, but it must no longer be possible to convert it back into plain text in order to protect it as well as possible from unauthorized viewing.
Examples for reversible encryption are: "AES", "CHACHA20", "3DES", "RSA"

Motivation: Without protective measures, an attacker in possession of a user database copy is able to determine masses of contained passwords in short time by merely trying out character string combinations or making use of dictionaries. Passwords stored in cleartext or any backcalculatable format are fully defenseless to an offline attack. Once a password has been ascertained it can be used by an attacker for unauthorized access to the system and the data on it.

Implementation example: Example for ABAP (s. SAP note 2140269). Setting of parameters:

- login/password_hash_algorithm = encoding=RFC2307, algorithm=iSSHA-256, iterations=10000, salt-size=128
- login/password_charset = 2
    - Attention: With login/password_charset = 2, the system stores passwords in a format that systems with older kernels cannot interpret. Therefore, ensure that all systems involved support the new password coding before setting the profile parameter to the value 2.

ID: 3.22-67/i373

Req 68	If a password is used as an authentication attribute, a protection against online attacks like brute force and dictionary attacks that hinder password guessing must be implemented.

Online brute force and dictionary attacks aim for a regular access interface of the system while making use of automated guessing to ascertain passwords for user accounts.

To prevent this, a countermeasure or a combination of countermeasures from the following list must be implemented:

- technical enforcement of a waiting period after a login failed, right before another login attempt will be granted. The waiting period shall increase significantly with any further successive failed login attempt (for example, by doubling the waiting time after each failed attempt)
- automatic disabling of the user account after a defined quantity of successive failed login attempts (usually 5). However, it has to be taken into account that this solution needs a process for unlocking user accounts and an attacker can abuse this to deactivate accounts and make them temporarily unusable

- Using CAPTCHA ("**C**ompletely**A**utomated**P**ublic**T**uring test to tell**C**omputers and**H**umans**A**part") to prevent automated login attempts by machines ("robots" or "bots") as much as possible. A CAPTCHA is a small task that is usually based on graphical or acoustic elements and is difficult to solve by a machine. It must be taken into account that CAPTCHA are usually not barrier-free.

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: Without any protection mechanism an attacker can possibly determine a password by executing dictionary lists or automated creation of character combinations. With the guessed password than the misuse of the according user account is possible.

Implementation example: Example ABAP:
login/fails_to_user_lock = 5
login/failed_user_auto_unlock = 0

ID: 3.22-68/i373

Req 69	If passwords are used as an authentication attribute, they must not be displayed in plain text during input.

Passwords must not be displayed in legible plain text on screens or other output devices while they are entered. A display while entering must not allow any conclusions to be drawn about the characters actually used in the password.

This requirement applies to all types of password input masks and fields.
Examples of this are dialogs for password assignment, password-based login to systems or changing existing passwords.

Exceptions:

- Within an input field, an optional plain text representation of a password is permitted, provided that this plain-text representation serves a valid purpose, exists only temporarily, has to be explicitly activated by the legitimate user on a case-by-case basis and can also be deactivated again immediately by the latter.
  A valid purpose would be, for example, to allow the legitimate user an uncomplicated visual check, if necessary, that he has entered the password correctly in a login dialog before finally completing the login.
  Such an optional plain text representation of a password must remain fully in the control of the legitimate user so that he can decide on its activation/deactivation according to the situation. In the default setting of the system, the plain text representation must be deactivated.
- The typical behavior on many mobile devices (smartphones) of displaying each individual character very briefly in plain text when entering a password - in order to make it easier for the user to control input - is fundamentally permissible there. However, the full password must never be displayed in plain text on the screen.

Motivation: In the case of a plain text display, there is a risk that third parties can randomly or deliberately spy on a password via the screen output while typing.

Implementation example: When displayed on the screen, each individual character is uniformly replaced by a "*" while entering a password.

ID: 3.22-69/i373

Req 70	If passwords are used as an authentication feature, the new password must differ from the old password by at least 1 characters.

If passwords are used as an authentication feature, the new password must differ from the old password by at least 1 characters.

Motivation: Make sure that the passwords are not too identical.

Implementation example: If passwords are used as an authentication feature, the new password must differ from the old password by at least 1 characters.

ID: 3.22-70/i373

2.7. General parameter for Netweaver (ABAP and Java)

Req 71	On Netweaver systems (ABAP-Stack), the JAVA-Stack must be deactivated.

There is not allowed to operate the abap and the java stack on the same application. That was a change by SAP, now dual-stack-systems aren't supported.

Motivation: Dual-Stack-Systems are not supported by SAP.

Implementation example: Check Parameter "J2EE Engine = 0

ID: 3.22-71/i373

Req 72	The SAP-Module "Software-Manager" must be active in the System.

It must be ensured that every system has the SAP-Software-Manager-Service. The service is connected to the global Solution-Manager. It is not allowed, that systems use an local unmanaged software manager.

Motivation: With using the Software-Manager it is ensure that the System are patched by SAP.

ID: 3.22-72/i373

Req 73	Once the standard installation is complete, suitable encryption libraries must be installed.

Once the standard installation is complete, suitable encryption libraries must be installed. It can be used the implementations for SAP or T-Systems.

Motivation: In older releases, the SAP Java stack (JVM) does not provide any strong encryption mechanisms by default. No strong encryption algorithms can be used without downloading and installing the encryption library. This therefore gives attackers the chance to attack the system, since the mechanisms it uses are too weak. Furthermore, the JRE requires “unlimited encryption capabilities”, as they are “limited” in the default setting. They are controlled by local_policy.jar and MUST be manually updated with SAP’s policy files.

Implementation example:

1. JVM – Java Virtual Machine Query which library is loaded:
   Visual administrator (VA): Dispatcher | Libraries | core_lib. The iaik_jce.jar file muss be included in the list of loaded jar files. The Iaik_jce export version must not be listed. As of SAP NetWeaver 7.1 the NetWeaver Administrator is used for this purpose.
2. JRE – Java Runtime Environment
   As of Release 7.0 strong JRE encryption comes with the delivery package. It is necessary to replace the limited policy files of the 1.4 J2SE platform. See note 739043:https://service.sap.com/sap/support/notes/739043For release 6.40 the steps involved in “Enable StrongEncryption” are to be executed, or verified.

See note 85833:http://https://service.sap.com/sap/support/notes/853833

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: 3.22-73/i373

2.8. Securing the ABAP-Stack

2.8.1. ABAP-Stack: OS Directories

Req 74	Access to the SAP installation logs must be severely restricted.

For the ABAP and Java stack, the installation files (logs) are stored on the standard SAP operating directory /usr/sap/<SAPSID>/SYS. This is the default value of the DIR_INSTALL variable, which specifies the location for the SAP installation directory.

Motivation: Installation logs contain data with a high protection requirement. After the installation is complete, only authorized users may access it.

Implementation example: Changing the access rights for the installation logfiles with the following commands:
```
chown root <file name>
chmod 700 <file name>
```
If possible, the installation logs can also be deleted.

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

For this requirement the following warranty objectives are relevant:

ID: 3.22-74/i373

Req 75	The directory "SIDADM-Home" and its content must be protected from non-authorized access.

On operation system, the home directory from "SIDADM-Home" must be protected from non-authorized access.

Motivation: The content under the HOME-directory must be protect, there are useful information's about the application.

Implementation example:

OS-Verzeichnisname	OS-Zugriffsberechtigung	OS-User/OS-Gruppe
<home Verzeichnis von <sapsid>adm>	700	<sapsid>adm/sapsys
<home Verzeichnis von <sapsid>adm>/*	700	<sapsid>adm/sapsys

OS-Commands for setting user and group permissions:
```
chown <OS-User>:<OS-Gruppe> <Verzeichnisname>
```

```
chmod <OS-Berechtigung> <Verzeichnisname>
```

Beispiel:

```
chown <sapsid>adm:sapsys <home Verzeichnis von <sapsid>adm>
chmod 700 <home Verzeichnis von <sapsid>adm>
```

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
- Denial of executed activities

For this requirement the following warranty objectives are relevant:

ID: 3.22-75/i373

Req 76	On operation system the directory "/spamnt/<SAPSID>" and their sub-directory's with its contents must be protected from non-authorized access.

On operation system, the directory "/spamnt/<SAPSID>" and there sub-directory's must be protected from non-authorized access, Under the directory "/sapmnt/<SAPSID>" are very importent subdircotrys and content for the applikation.

Motivation: Usually the files are on a global host and are distributed via NFS, SMB. If is possible to access to directory and files, a manipulation for this data-file can be change the working of the application.

Implementation example:

OS directory name	OS access	OS-user/OS-group
/sapmnt/<SAPSID>/exe	775	<sapsid>adm/sapsys
/sapmnt/<SAPSID>/global	700	<sapsid>adm/sapsys
/sapmnt/<SAPSID>/profile	755	<sapsid>adm/sapsys

OS command for setting access and the OS-users with the relevant OS-group:

```
chown <OS-user>:<OS-group> <OS directory>
chmod <OS access> <OS directory>
```

Example:

```
chown <sapsid>adm:sapsys /sapmnt/<SAPSID>
chmod 751 /sapmnt/<SAPSID>
```

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

For this requirement the following warranty objectives are relevant:

ID: 3.22-76/i373

Req 77	The directory "/usr/sap/>SAPSID> und its content must be protected for non authorized access of the operation system.

The content in the directory "usr/sap/<SAPSID>" is for the application essential. There must be protected for access from operations system.

Motivation: Unauthorized access to directory's and files can be change Information, at worst case the application is stop working.

Implementation example:

OS directory name	OS access	OS user/OS group
/usr/sap/<SAPSID>	751	<sapsid>adm/sapsys
/usr/sap/<SAPSID>/<Instance ID>	755	<sapsid>adm/sapsys
/usr/sap/<SAPSID>/<Instance ID>/sec	700	<sapsid>adm/sapsys
/usr/sap/<SAPSID>/SYS	755	<sapsid>adm/sapsys
/usr/sap/<SAPSID>/SYS/*	755	<sapsid>adm/sapsys

OS command for setting access and the OS-users with the relevant OS-group:

```
chown <OS-user>:<OS-group> <OS directory>
chmod <OS access> <OS directory>
```

Example:

```
chown <sapsid>adm:sapsys /usr/sap/<SAPSID>
chmod 751 /usr/sap/<SAPSID>
```

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

For this requirement the following warranty objectives are relevant:

ID: 3.22-77/i373

Req 78	The SAP transport directory and its content must be protected for non-authorized access.

The content of the transport directory must be protected for non-authorized access. Usually the directory is deployed on a host and distributed via SMB or NFS protocol.

Motivation: The content of the transport directory must be protected for non authorized access. The directorys are importend for proper work.

Implementation example:

OS-Verzeichnisname/OS-Dateien	OS-Zugriffsberechtigung	OS-User/OS-Gruppe
/usr/sap/trans	775	<sapsid>adm/sapsys
/usr/sap/trans/*	770	<sapsid>adm/sapsys
/usr/sap/trans/.sapconf	775	<sapsid>adm/sapsys

OS-Kommandos zum Setzen der OS-Zugriffsberechtigungen und des OS-Users und der OS-Gruppe:

```
chown <OS-User>:<OS-Gruppe> <Verzeichnisname>
chmod <OS-Berechtigung> <Verzeichnisname>
```

Beispiel:

```
chown <sapsid>adm:sapsys /usr/sap/trans
chmod 775 /usr/sap/trans
```

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

For this requirement the following warranty objectives are relevant:

ID: 3.22-78/i373

Req 79	On operating system level SAP tools must be secured from unauthorized access.

Ensure that the Sap-Tools have his own users, groups and directory's on operation system.

Motivation: Having unauthorized access to the files the file content could be changed. Changes on the file content could lead to system failures and/or to an outage.

Implementation example: For minimum following tools access must be restricted:

Tool name	Directory	Access	User/Group	Function
tp		770	<sapsid>adm/sapsys	Perform transports
sapevt	/usr/sap/<SAPSID>/SYS/exe/run	750	<sapsid>adm/sapsys	Netweaver events starting on operating system level
saposcol	/usr/sap/<SAPSID>/SYS/exe/run	4710	root/sapsys	Collects information-en from 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
- Disruption of availability
- Unnoticeable feasible attacks

For this requirement the following warranty objectives are relevant:

ID: 3.22-79/i373

2.8.2. ABAP-Stack: Standard User

Req 80	The SAP* user must be protected.

By default the SAP system generates accounts. One of them is SAP*. The account as high permissions on SAP. It must

be ensured for unauthorized access and it is only used in case of extreme exception.

It must be differentiated between the user SAP* within the SAP application and the initial user SAP*. The user SAP* within the SAP application can be maintained with transaction SU01. While the initial user SAP* with the password "PASS" is defined in the SAP kernel and it cannot be changed. So this user cannot be locked and the password and the authorizations cannot be changed. Furthermore, the initial user SAP* has high authorizations. The access on this powerful user is controlled by a SAP parameter and the existence of the user SAP* within the SAP application. The user SAP* within the SAP application must be setup strongly restricted as well.

Motivation: Unauthorized access with that user can lead to the possibility to read and change the data and or of a system outage.

Implementation example: Get an overview about all existing clients and check if they are valid. For that use transaction SCC4.

Use transaction SU01 (user maintenance) wihtin the related client and check respectively set the following for the user master record of the SAP*:

- Exists for the client.
- Must be locked.
- The password must be deactivated (due to this the change of the standard password is not necessary).
- Belongs to group SUPER.
- All rights must be deleted.

Additionally the system parameter “login / no_automatic_user_sapstar = 1” MUST be set.

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

For this requirement the following warranty objectives are relevant:

ID: 3.22-80/i373

Req 81	The user DDIC must be protected from unauthorized access

By default the SAP-System generates accounts. One of them is DDIC. The account as high permissions on SAP. It must be protected from unauthorized access and it is only used in case of exception.

Motivation: Unauthorized access with that user can lead to the possibility to read and change the data and or of a system outage.

Implementation example: Get an overview about all existing clients and check if they are valid. For that use transaction SCC4.

Use transaction SU01 (user maintenance) within the related client and set the following for the user master record of the SAP*:

- Must be locked.
- The password must be deactivated (due to this the change of the standard password is not necessary).
- Belongs to group SUPER.

By default, the user DDIC is set up to be used for the transport background job (RDDIMPDP). A new user must be defined for this job so that DDIC user can be locked:

- The new user needs SAP_ALL and S_A.SYSTEM authorizations because the job calls function modules for various applications that cannot be determined for all cases ahead of time.
- Set up the new user as a system user so that no one can use it as a dialog user.
- Set up the new user in all clients that are used for import.
- Adjust the job RDDIMPDP so that the new user is the owner (in transaction SM37).

ID: 3.22-81/i373

Req 82	The User EARLYWATC only exists in Client 066. If it isn't used, it must be inactive.

The EARLYWATCH user is needed for the Early-Watch-Service in client 066. It exists only in client 066. If it isn't used, he must be inactive.

Motivation: With client 066 and the user EARLYWATCH system information is collected that is used for e. g. the Going-Life-Check or the EARLYWATCH-Report.

Implementation example: Use transaction SU01 (user maintenance) in client 066 and set the following for the user master record of the SAP*:

- Must be locked.
- The password must be deactivated (due to this the change of the standard password is not necessary).
- Belongs to group SUPER.

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: 3.22-82/i373

Req 83	The Account SAPCPIC must be protected from unauthorized access. Also it's not allowed the use the account for RFC-Connections.

The account SAPCPIC is not used by (RFC) Services to communicate to destination. Create an own user specific rights.

Motivation: The user SAPCPIC is a unspecified system user, which can be used for programms and function modules within the SAP system. But it isn't visible, who is using it. Therefore this user should not be used. Instead of this own specified users should be created. Using a naming convention for the user-ID the user is attributable and it can be customized to the needs.

Implementation example: Execute transaction SU01 (user maintenance) on the relevant client:

1. Recommended option: Delete the user SAPCPIC, when it is not needed!
2. Option: When the user cannot be deleted, ...
    - ... change the standard password
    - ... put it to the group SUPER
    - ... set user type "system"

ID: 3.22-83/i373

Req 84	The account TMSADM is a Systemuser by SAP. He is only to allow for the communication by TMS (Transport Management System).

The account TMSADM is needed for the TMS communication. During initialization and configuration of the TMS in Client 000 the user TMSADM is created only in client 000 with its standard password. Due to the initialization of the TMS RFC destinations to the systems within the transport composite regarding the rules of the configured transport routes are created. Using this RFC destinations the user TMSADM communicate with the other systems in the composite and gets information e. g. the content of the transport directory.

Motivation: It is to be prevented that transports can only be used by authorized persons. The transport contents are only visible to them. So securing the TMSADM user is very important.

Implementation example:

1. The Account existonly in client000.
2. The TMSADM must be member of group SUPER.
3. Change the default password for TMSADM

The account TMSADM can't be deactivated! Further information about the setting of the password of TMSADM user you will get with following SAP notes: 761637, 1488406, 1610103, 1414256, 1552894, 1726102

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: 3.22-84/i373

2.8.3. ABAP-Stack: User

Req 85	All users must have a suitable user type, based on the kind of use involved.

There are different user types in a SAP system: dialog -, system -, communication - and service users. Depending on the type of use, a suitable user type must be selected.

Motivation: Use of a false user type can open up additional attack surface (e.g., the possibility of interactive login for technical users).

Implementation example: The user type must match the user's role. If the user is SAP application user, he needs the user type "dialog". When it comes to communication within the system or between systems, the user type "communication" is selected for the user. If jobs are to be executed on the system, the user receives the user type "system". The user type "service" should only be used in justified exceptional cases.

Property	Dialog (A)	Communication (C)	System (B)	Service (S)
GUI login	yes	no	no	yes
RFC login	yes	yes	yes	yes
Enforce of password change	yes	yes	no	no
Password expiration	yes	yes	no	no
Login ticket can be created	yes	yes	no	no

Check via report RSUSR200.

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: 3.22-85/i373

Req 86	An emergency user with full rights must be created for system administration.

A emergency user with SAP_ALL authorization must exist. The usage of this user must be steered by the Central Administrator Access tool (CAA). The auditing must be activated for this user.

Motivation: An emergency user with full rights (SAP_ALL) must exist on the system. Access via the SAP Auditing Log must be recognized in the log files as an emergency user. In addition, access via emergency user must be authenticated and documented in a traceable format.

Implementation example: Using the CAA-Tool (Central Administrator Access) by T-Systems.

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

For this requirement the following warranty objectives are relevant:

ID: 3.22-86/i373

2.8.4. ABAP-Stack: Authorizations

Req 87	The user administration must be divided into user, profile and role administration.

Motivation: It must be prevented that security policies (need-to-know, need-to-do, segregation of duties) can be bypassed by users being able to create users and assign profiles and roles and, consequently, being able to assign themselves full access to the data and the system.

Implementation example: SAP delivers template roles for this which still need to be adjusted to the circumstances of the user and authorization concept:

Template	Administrator
SAP_ADM_PR	Profile management
SAP_ADM_AU	Role management
SAP_ADM_US	User administration User administration is executed via the transaction SU01 and role and profile maintenance via transaction PFCG

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: 3.22-87/i373

Req 88	Operating system commands defined on SAP level must be protected for unauthorized access. This is done by restriction of transactions SM49 and SM69 within the authorization concept.

The access on operating system level must be restricted like the access on operating system level from SAP level. It must be avoided that arbitrary SAP user can start operating system commands from SAP level. The access from SAP level to operating system commands is performed with transactions (TAs) SM69 and SM49.

With TA SM69 logical operating system commands can be defined on SAP level. Beside the real function of the operating system command can be changed extra or without awareness of its effect by maintaining wrong inputs ("ls" on SAP level can get to "rm" on operating system level).

With TA SM49 logical operating system commands can be executed out of the SAP level.

The access on operating system commands by using the TA SM49 and its definition with TA SM69 must described in the authorization concept and must be granted very restricted only.

Motivation: The with the TAs SM69 and SM49 defined and executed operating system commands are started with the SAP administration user (<sid>adm) on operation system level. This operating system user has wide authorization on all SAP directories and SAP files on operating system. Furthermore these administrative accesses will not be logged on level of operating system. And it enables users on SAP level to bypass the authorization concept on operating system level. The uncontrolled and arbitrary use of operating system commands out of SAP level can lead to a compromising of the system due to the attacker gets several possibilities for executing operating system commands and SAP operating system commands with authorities of the SAP administration user on operating system level. Using the commands (security) relevant information can be read, changed and/or deleted. For example using the SAP tool "tp" a SAP transport on operating system level can be performed or the whole SAP directory tree can be deleted and a system outage will happen than.

Implementation example: The validation of the configuration must be done in several steps:

1. With the user information system TA SUIM it must be checked in which role/profile the TAs SM49 and SM69 (authorization object S_TCODE) exists and to which user these are granted.
2. Additionally for these user the authorization object S_LOG_COM must be checked. In the field "COMMAND" the logical command is entered. In that file a generic value can be used for example for a group out of the finance department the generic value can be "ZFI*" (the can contain several logical commands which are connected to different operating system commands.
3. With the help of the authorization concept and the job description it must be clarified if the need-to-do principle was really set up correct.

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: 3.22-88/i373

Req 89	Execution of operating system commands by using the report RSBDCOS0 must be protected for unauthorised access.

RSBDCOS0 is an ABAP report/program that mimics a console/shell and allows one to execute commands at the operating system level from SAPGUI.

In order to use RSBDCOS0, several authorization objects are required. See SAP Note 2297349.

Motivation: Executing operating system commands by using the report RSBDCOS0 extensive changes on the system can be made. The executed operating system commands are started with the SAP administration user (<sid>adm) on operation system level. This operating system user has wide authorization
on all SAP directories and SAP files on operating system. Furthermore these administrative accesses will not be logged on level of operating system. And it enables users on SAP level to bypass the authorization concept on operatingsystem level. The uncontrolled and arbitrary use of operating system commands out of SAP level can lead to a compromising of the system due to the attacker gets several possibilities for executing operating system commands and SAP operating system commands with authorities of the SAP administration user on operating system level. Using the commands (security) relevant information can be read, changed and/or deleted. For example using the SAP tool "tp" a SAP transport on operating system level can be performed or the whole SAP directory tree can be deleted and a system outage will happen than.

Implementation example: The validation of the configuration must be done in several steps (please see SAP note 2297349 also):

With the user information system TA SUIM it must be checked in which role/profile the following objects with the listed values contains and which user this role/profile is granted:

Objekt	Felder	Kommentar
S_TCODE	TCD=SE38 and/or SA38 and SM69 and SM49	
S_RZL_ADM	ACTVT=01	
S_LOG_COM	COMMAND=RSBDCOS0 OPSYSTEM=<name OS> HOST=<hostname>	OPSYSTEM and HOST values can be a wildcards *, for example: OPSYSTEM=* HOST=*
S_C_FUNCT	PROGRAM=RSBDCOS0 ACTVT=16 (Execute) CFUNCNAME=SYSTEM	Without this authorization, the user will encounter a CALL_C_FUNCTION_NO_AUTHORITY short dump

RSBDCOS0 also requires transactions SM69 and SM49 to be unlocked in transaction SM01 (now SM01_DEV and SM01_CUS see SAP Note 2234192).

With the help of the authorization concept and the job description it must be clarified if the need-to-do principle was really set up correct.

ID: 3.22-89/i373

Req 90	Download authorization for users must be restricted.

This requirement is the responsibility of the SAP specialist department (owner of the authorization concept).

Downloading and exporting SAP reports is enabled by default for all users (System -> List -> Back up). The download authorization must be restricted via an authorization concept in relation to specific tasks so that user groups for whose

efficient workflows the download authorization is important still have it but everyone else has their authorization withdrawn.

Motivation: Unauthorized communication results in the loss of confidentiality for vulnerable data or even in an infringement of the Data Protection Act as a result of inappropriate use. Confidential information, e.g., personal or finance department data, can be downloaded to a user's PC, modified there, linked to other data, processed and forwarded to unauthorized third parties.

Implementation example: Withdrawal of the authorization S_GUI activity 61

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

For this requirement the following warranty objectives are relevant:

ID: 3.22-90/i373

Req 91	ICF communication must be protected on the end-user layer.

Users are assigned to each enabled service in the SICF transaction. The SICF transaction may only be used to enable those services that are actually required in the application. Access by users to enabled services is to be implemented via the authorization concept.

Motivation: ICF services are not protected by default. This means that any user can access an ICF service when it is active.

Implementation example: The system administrator can assign a user the following authorizations:

Object: S_ICFREC
Field: ACTVT (activity):
- 02 Change
- 03 Display
- 06 Delete
- 16 Execute
- 63 Enable
- 70 Manage, administrate
- D1 Copy
- DL Download
- UL Upload

Field: USER (user name)
Field: CLIENT (client)
Field: SYSTEM (system ID)

Authorizations must be granted restrictive only (need-to-do-principle, least privilege principle).

See also: http://help.sap.com/saphelp_nw04/helpdata/de/80/b2dd3a6dac703be10000000a11405a/content.htm (in german)
More information: http://help.sap.com/saphelp_nw73/helpdata/de/48/cf963002b9230ee10000000a42189b/content.htm

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: 3.22-91/i373

Req 92	Administrative transactions that are used to operate and maintain the system (e.g., RZ10) must be inaccessible for normal users.

Administrative transactions used for the operation and maintenance of the system must be inaccessible to ordinary users. Administrative transactions include, for example PFCG, RZ10, RZ11, SCC4, SE16, SE38, SE80, SM21, SM31, STMS, SUIM, SU01, SU02 ...

Motivation: Restrictive authorizations are necessary in order to prevent unauthorized access by '"normal" 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

For this requirement the following warranty objectives are relevant:

ID: 3.22-92/i373

Req 93	Transactions and modules that are not required must not available in the system.

To ensure security in a SAP system, transactions and modules that are not used must be deactivated or deleted. If this is not possible, you must ensure that they are not used using roles and authorizations

Motivation: After an installation, all modules and transactions of an SAP system are available to all mandanten. By withdrawing authorizations and deactivierung from modules, the system is adapted for the client (tenant) and the "need to konw" is implemented.The functional scope of the license is adapted by deactivating or activiatet modules and transactions

Implementation example: Following transactions must be blocked in all clients via transaction SM01:
KA10,KA12,KA16,KA18,SARA;OBR1; OICP,OOSB; OM-DL,OMEH,OMWF,OOUS,OPF0,OTZ1,OY27,OY28,OY29,OY30; OMEI,OMWG,OOPR,OP15,OPE9,OTZ2,OY21; OMG7,OMWK,OPF1,OTZ3,OY20; OOSP; OVZ6;SCC1;SCC5;SE01;SE09,SE10;SE16,SM30,SM31;SM69;SU12.

ID: 3.22-93/i373

Req 94	If using transaction the RSUDO, the execution must be controlled by authentications.

The transaction RSUDO must be secured by authentication, because the transaction serves to expand your own authorizations. Similar to Windows "run as".

Motivation: Often Z-program's must be check, if they use RSUDO to expand his privileges.

ID: 3.22-94/i373

Req 95	The use and configuration of trusted/trusting relationships between SAP systems must be restrictive.

The trust relationship must be used carefully. Access between trusted processes is possible without explicit authentication.

Motivation: In the event of incorrect configuration, any user from the one system can access any user with the partner system and carry out actions within the framework of its permissions.

Implementation example: To show trusted System by transaction SMT2 or with in report RFCTRUST. It is important to ensure, that trusted relationship don't work in false direction over operation models.

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: 3.22-95/i373

Req 96	When SAP own single-sign-on (SSO) is used, the user permissions must be adapted according to the system environment.

SAP Netweaver Systems can use the local or shared SSO's. When the system use shared accounts between productive SAP systems, the permissions must be configure restrictly and costemized for each system.

Motivation: When an SSO trust relationship is established with a system that is configured with a lower protection level, this reduces the protection level of the trusting system, since attackers could attack the less protected system and from there gain access to the trusting system.

Implementation example: This is implemented for each client in the ABAP system via transaction SSO2 or STRUSTSSO2 of the ACL access control list. In the JAVA system: Call NetWeaver Administrator / System Administration / Configuration / Trusted Systems You can find more information at:
- http://help.sap.com/saphelp_nwpi71/helpdata/de/8d/903d41b77ba52fe10000000a155106/content.htm
- http://help.sap.com/saphelp_nwmobile71/helpdata/de/78/f1a8490e7011d6999500508b6b8a93/content.htm

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

For this requirement the following warranty objectives are relevant:

ID: 3.22-96/i373

Req 97	It must be ensured for using the Direct Imput Method that a 4-eyes-principle is established, the use of it is documented and a procedure for that is existing.

It must be ensured for using the Direct Imput Method that a 4-eyes-principle is established, the use of it is documented and a procedure for that is existing.

Motivation: By using the Direct Input Method the data are written with a reduced validation directly into the database. Additionally it is possible, to update the database entries without a protocol of the performed change. In the event of incorrect or careless use, the data integrity of the system is endangered.

Implementation example: Necessary for implementation:
1. There must be a process instruction for the Direct Input procedure.
2. In this process instruction it is described that a 4-eyes principle is implemented.
3. This process instruction describes how to document the use of the Direct Input method.

To check which users are allowed to use the procedure, the following must be done:

- The users with the appropriate authorizations can be determined via report RSUSR002. For this, 3 evaluations must be carried out, for the potion actions BMV0, SXDA and LSMW one each. To get the exact query data, please go to the following link: ht-tps://www.dsag.de/sites/default/files/150505_leitfaden_best-practice-sap-erp_rz.pdf

ID: 3.22-97/i373

Req 98	A authorization check for starting of Web-Dynpro-ABAP-applications must be performed (condition: use of S/4HANA).

With the use of S/4HANA, the authorization object S_START is introduced. The start authorization check against the authorization object S_START (Web Dynpro ABAP applications) was already delivered with ABAP Basis Releases 7.03 and 7.30.

Motivation: Unauthorized Access can lead to read, change and or deletion of data.

Implementation example: Check first if authorization check for WebDynpro is set to active. Start transaktion SU25. Under "Adjust the Authorization Checks (Optional)", start "Activate Web Dynpro Start Authorization Check (S_START)". The subsequent ALV list displays two lines:

Web Dynpro Application Configuration	R3TR	WDCA
Web Dynpro Application	R3TR	WDYA

If these two lines are not selected in the "Inactive" column, the start authorization check is active for the corresponding objects.You have the following options:

- You can add suitable authorizations for the authorization object S_START to your authorization roles. The exact procedure is described in SAP Note 1413011. In addition, in transaction SUIM under "Roles", you can start the "Search for Startable Applications in Roles" report that you can use to adjust roles in a simplified way. To do this, in the "Application Type" input field, select the "Web Dynpro Application" or "Web Dynpro Application Configuration" value.
- You can assign the role SAP_BC_WEBDYNPRO_ALL (or a copy of this role) to all users, which contains full authorization to start all Web Dynpro applications and Web Dynpro application configurations.
- Not recommended/allowed: You can deactivate the start authorization check for Web Dynpro applications and Web Dynpro application configurations.

ID: 3.22-98/i373

2.8.5. ABAP-Stack: Logging

Req 99	Changes on critical tables must be logged with the "table change logging" and the "logging of the change documents".

In the standard system, table logging is not active and must be activated in order to comply with legal requirements.

Motivation: Without table logging there is a risk of late or no response to potential unauthorized table data modifications, e.g., an adversary may change the bank account value in the relevant table and commit fraud actions by money transfer to another account. No active logging of change documents and table logging can lead to a prosecution.With SAP you have the possibility to log changes on tables. Using the table logging mainly customizing data are logged. Therefore it is traceable to see which user has when changed, added or deleted a data record. By default within the

SAP system's incl. SAP S/4 HANA are ca. 40.000 tables marked for logging. Beneath this are billing relevant tables, for which the logging of changes is statutory. Therefore customizing tables are under the duty to preserve for 10 years conformable to law §257 HGB. The table logging is not activated by default and is therefore a MUST. Specially the relevant enterprise own tables must covered. Own tables with billing relevant content must be logged as well. Changes on tables can be done in TWO ways, which must be activated separate. Either the changes are performed direct within the SAP system or they are perform by transports over the transport system. The direct changes within the system are logged by setting the profile parameters rec/client, the changes by transports are logged by setting the transport parameter RECCLIENT. All changes are saved in table DBTABLOG. The log data can be viewed by TA SCU3. The log enty shows the time stamp, the user and the application server on which the change was done.

Changes on economic data (master data or invoice document data like invoice document, credit card data or vendor data), which are to be subject to often changes, are logged by change documents. The logging is steered by change document objects. In such objects the tables are named, which are logged. The objects are administrated over TAs SCDO or SCDO_NEW. For each change document object tables are defined, for which change documents are written. Changes on such tables must be logged § 257 HGB and are as well under the duty to preserve for 10 years conformable to law §257 HGB. All change document objects are saved in table TCDOB. The data of logging are saved in tables CDHDR and CDPOS. Within the logs the username, the data of the change, the used TA and the kind of the change can be seen.

Implementation example: 1.1) The table logging is controlled by an appropriate value in the technical table configuration (TA SE13 or TA RDDPRCHK_AUDIT). To review all the tables logged, the DD09L table is called with the SE16N transaction. Mark all critical system tables (s. SAP note 112388) and especially the own relevant enterprise tables to be logged! These setting is generally performed in the development system and via transport set in the produktion system!

1.2) Activate the table logging for direct changes on tables with setting profle parameter:

- rec/client = ALL (or at least the operational client. Value "ALL" is possible due to in for example 000 and 066 no changes on tables occurs)

1.3) To log the changes on tables done by transports, set the following transport parameter on the transport domain controller in TA STMS to the same value as in the SAP profile parameter rec/client (s. SAP note 163694):

- RECCLIENT = <same value as SAP profile parameter rec/client>

2) SAP system activate for business objects not automatically the log for change documents. Log activation must be done by yourself! Mark especially the own relevant enterprise tables to be logged! Steps for activation for change documents log for an object:

1. TA SCD0: Create a change document
2. TA SE11: Activate this change document of the object in the maitain data
3. TA SCD0: Generate a update program for the object
4. Insert according requests in the relevant program's

Change documents for an object can be reviewed with TA SCD0 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
- Unauthorized use of services or resources

For this requirement the following warranty objectives are relevant:

ID: 3.22-99/i373

Req 100	The SAP Security Audit Log (SAL) must be activated and security relevant events must be logged (till SAP NW 7.4).

Systems must be able to log security relevant events and it must be possible to activate that logging functionality.To get log information on critical TAs and program's, the SAL must be activated.

Motivation: The detection of (targeted) attacks and successful or attempted compromises in the networks and systems of DTAG requires a comprehensive logging of security relevant events on targeted systems and systems which are involved in such attacks.Systems must be able to log security relevant events and it must be possible to activate that logging functionality. 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. It must be possible to activate the following security events:

- *Authentication of users or systems (successful and failed)*
- *Administrative access (incl. executed commands/changes)*
- *Unauthorized/failed and erroneous access attempts (e.g. access on not allowed functions by an account)*
- *Administration of accounts*
- *Changing of group membership of accounts.*
- *Further security relevant events specific to the system*

Implementation example:

1. Execute of TA RZ10 and setting of following parameters:
    - rsau/enable = 1 (activation of SAL)
    - rsau/selection_slots = 10 (max. amount of audit filters)
    - rsau/user_selection = 1 (activates generic enter of the user name, which helps to reduce the amount of filters)
    - System restart necessary
2. Execute of TA SM19 for configuring of filters within the static configuration for filtering security relevant events like described above. Further actions can be added like ...
    1. Debugging with replace, system change option changed, change documents deleted without archiving, ...
    2. All actions of the user DDIC, SAP*, emergency user, further unpersonalized user. E. g. a filter for DDIC is created, which is valid over all clients due to enter "*" within the field client.
    3. User with the profiles SAP_ALL, SAP_NEW, S_A.SYSTEM, S_A.ADMIN, S_A.DEVELOP, S_ADMI_ALL, R3_BASIC, S_A.CUSTOMIZ, S_ABAP_ALL, S_ADMI_SAP, S_ENTW, S_RZL_ADM
3. Activate the filter with setting the check for "active".
4. Restart the system for activation the configuration.

ID: 3.22-100/i373

Req 101	Security relevant events must be logged (from SAP NW 7.5).

Systems must be able to log security relevant events and it must be possible to activate that logging functionality. 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.It must be possible to activate the following security events:

- Authentication of users or systems (successful and failed)
- Administrative access (incl. executed commands/changes)
- Unauthorized/failed and erroneous access attempts (e.g. access on not allowed functions by an account)
- Administration of accounts
- Changing of group membership of accounts.
- Further security relevant events specific to the system

Motivation: The detection of (targeted) attacks and successful or attempted compromises in the networks and systems of DTAG requires a comprehensive logging of security relevant events on targeted systems and systems which are involved in such attacks.

Implementation example: To activate the SAL (Security Audit Log), perform the following steps (see SAP Note 2191612):

1. Run TA RSAU_CONFIG: And set up filters in a profile in the static configuration. Only the static configuration is still active after a system startup.
2. It is not recommended to record the logs in the database because you will need to store the logs on another secure system (Recording Destination and Recording Type fields).
3. Define enough filters.
4. You should select the generic user selection. This can save users who have the same strings from having to define filters.
5. Set up filters for security-related events, as described in the long text above. Depending on the SAP application, further actions or users can be logged, such as ...
    1. Debugging with Replace, system modifiability changed, change pads were deleted without archiving
    2. All actions of users DDIC, SAP*, emergency users, other unpersonalized users, e.g. a filter can be created with DDIC that is valid for all clients by inserting the "*" in the Tenant field.
    3. Users with profiles SAP_ALL, SAP_NEW, S_A.SYSTEM, S_A.ADMIN, S_A.DEVELOP, S_ADMI_ALL, R3_BASIC, S_A.CUSTOMIZ, S_ABAP_ALL, S_ADMI_SAP, S_ENTW, S_RZL_ADM
6. Activate the filter.
7. Perform a restart to activate the settings that have been performed.

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: 3.22-101/i373

2.8.6. ABAP-Stack: Communication Security

Req 102	External access to the SAP gateway must be restricted via the “secinfo” configuration file. External service registration at the SAP gateway can be restricted via the “reginfo” configuration file.

Anyone who reaches the SAP gateway via IP is able to start programs on the SAP system if the “secinfo” file has not been created manually. This is why “secinfo” must be maintained.

IMPORTANT: If there is a “secinfo” but no “reginfo”, the registration at the SAP gateway is also blocked via “secinfo”. This is why the “reginfo” must also be maintained, because otherwise the SAP system would no longer be available for registrations as usual and SAP developers are not familiar with this error in practice.

Motivation: Attackers are able to start programs in the SAP system via the SAP gateway if “secinfo” is not maintained. Responsible maintenance of “secinfo” is therefore required. In addition, external programs are able to register at the gateway and then offer their services via this gateway. The external service has the security responsibility. Existing, already registered services cannot be overwritten (ID already available) – and calling up the registered external services from within SAP would additionally require a corresponding RFC destination to be maintained. This means, there is an option for restricting registration at the gateway via “reginfo”, but it is not required for the security of the SAP system.

Implementation example: secinfo and reginfo should be moved to the global directory via gw/sec_info = $(DIR_GLOBAL)/secinfo gw/reg_info = $(DIR_GLOBAL)/reginfo in order to reduce the maintenance required for each instance. Example secinfo – multiple lines based on the following template (#VERSION=2 is not a comment!)

```
#VERSION=2
P USER=* TP=* HOST=* USER-HOST=<host portal>
P USER=* TP=* HOST=* USER-HOST=<central host instance>
P USER=* TP=* HOST=* USER-HOST=<host application server>
P USER=* TP=* HOST=* USER-HOST=<host SolutionManager>
P USER=* TP=* HOST=* USER-HOST=<host TREX>
P USER=* TP=* HOST=* USER-HOST=<host portal>
```

Example reginfo: (#VERSION=2 is not a comment!)

```
#VERSION=2
P TP=*
```

Further informations: http://help.sap.com/saphelp_nw70ehp3/helpdata/en/e2/16d0427a2440fc8bfc25e786b8e11c/content.htm

Requirement-ID: 5fd17246

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: 3.22-102/i373

Req 103	Registrations of services at the SAP gateway must be restricted using the configuration file "reginfo".

Registrations of services at the SAP gateway must be restricted using the configuration file "reginfo". If the file "RegInfo" is not maintained, external programs can register their own services and modify the processing of the data unintentionally.

Motivation: The configuration of the "reginfo" file prevents external programs and possibly the type of processing from influencing each other unintentionally. Existing, already registered services cannot be overwritten.

Implementation example: secinfo and reginfo should be moved to the global directory via gw/sec_info = $(DIR_GLOBAL)/secinfo gw/reg_info = $(DIR_GLOBAL)/reginfo in order to reduce the maintenance required for each instance. Example secinfo – multiple lines based on the following template (#VERSION=2 is not a comment!)

Example reginfo: (#VERSION=2 is not a comment!)

```
#VERSION=2
P TP=*
Requirement-ID: 3f7813be
```

ID: 3.22-103/i373

Req 104	The access from external SAP RFC must be controlled.

External access on the SAP system via RFC, must be controlled by the SAP gateway. This must be done by setting two parameters and maintaining the ACL (Access Control List) file.

The parameter gw/acl_file defines the name of the ACL file. With that ACL (Access Control List) you can configure, who can connect to the gateway. Is that parameter not set, no access control exists!

The parameter gw/acl_mode defines the behavior of the gateway if the ACL file (gw/sec_info or gw/reg_info) doesn't exist. With the value 1 external and registered server within the system (applications server of the same system) are allowed. All other server will be denied or they must be entered in the file under gw/sec_info or gw/reg_info and gw/acl_file.

With the ACL will be steered, which connection the gateway accept and which not. Basement for this are the IP-adresses of the clients. The last rule is a general deny (deny 0.0.0.0/0). One row within the ACL musr have following syntax:

```
<permit | deny> <ip-address> [tracelevel] [# comment]
```

Motivation: Not to loose the control over the data, it must be avoided, that unauthorized access is possible.

Implementation example: Führen Sie alle Punkte aus, um eine sichere Konfiguration umzusetzen:

1) Execute TA RZ10 and set folowing parameters within the DEFAULT.PFL:

- gw/acl_file = <directory>/<file nameDateiname>
- gw/acl_mode = 1

```
2) Example for entries in the ACL file:

permit 10.1.2.0/24        # permit client network
deny   0.0.0.0/0           # deny the rest

Requirement-ID: ccb4c58b
```

ID: 3.22-104/i373

Req 105	Access to the SAP Message Server must be restricted to avoid DOS scenarios.

It must be guaranteed that the SAP Message Server does not process any relevant user load. Normal users should therefore only be able to work on the SAP system via load distribution and the SAP Message Server should only be used for limited access, e.g., by authorized administrators. If this is not guaranteed, there would be a risk of system instability. The following parameters are to be set:

- Maximum number of HTTP ports the Message Server can additionally open.
- Maximum number of HTTP clients that can log in to the Message Server in parallel.
- Timeout for network operations. If no further actions take place during this period, the connection to the HTTP client is interrupted. Timeout is stated in seconds.
- Maximum length of an HTTP header which can be imported.

Motivation: Setting these parameters minimizes the risk of compromise or DoS (Denial of Service) attacks.

Implementation example: The following default values are set and can be modified to suit individual project contexts:

- ms/http_max_ports:20
- ms/http_max_clients:1000
- ms/http_timeout:20
- ms/http_bufferln:65636

Requirement-ID: dfca5a9f

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

For this requirement the following warranty objectives are relevant:

ID: 3.22-105/i373

Req 106	System, which using RFC-Callback function, must work with withelist's.

As a rule, the RFC-callback function should be omitted.

The danger with this methodology is that the rights of the calling user from the source system are used in the process, and thus extensive rights can often be abused.
Using the function call "CALL FUNCTION <function> DESTINATION "BACK">, it is possible to execute malicious RFC functions on an SAP system when the calling system contacts a compromised system via RFC. This is because the compromised system can use DESTINATION "BACK" to establish a back channel.

To minimize the risk of an attack, SAP offers the creation of whitelists.

Motivation: Manipulated RFC-callback functions can allow unauthorized access to the compromised system.

Implementation example: The profile parameter rfc/callback_security_method must be set to 3.

If it is necessary to use the RFC callback function or not to use, the following must be implemented:

1. The profile parameter rfc/callback_security_method must be set to "3". This allows only the callback calls that are allowed in the whitelist (if the callback function is not necessary to use, the withelist is kept empty)
2. The whitelist for the RFC callback function, must be created with transaction SM59 (Determination of the call-backs and generation of the whitelist via simulation mode possible).
3. The Security Audit Log (SAL) must be activated and the following 3 options must be checked in the detail selection under the audit class "RFC-function call":
    - RFC callback executed (Destination &A, Called &B, Callback &C)
    - RFC callback rejected (Destination &A, Called &B, Callback &C)
    - RFC callback in simulation mode (Destination &A, Called &B, Callback &C)

It is recommended to log the actual calls before blocking all non-whitelist entries (parameter value 3). If you have set the rfc/callback_security_method parameter to 2 and thus selected the simulation mode, you can trace all callback calls in the SM20 (prerequisite for this is point 3.). And it is possible to generate and activate a whitelist in transaction SM59 and sort out unwanted callback calls.

To view the current entries of the whitelist, access the table RFCCBWHITELIST in SE16. All entries are listed there. To see which connections are also active, it is sufficient to look at table RFCCBWHITELIST_A.

Requirement-ID: 7033e1c0

ID: 3.22-106/i373

2.8.7. ABAP-Stack: Parameter

Req 107	Backward compatibility for the password must be prevented.

If the parameter is not equal to 0, the password is stored in addition to the current secure password, the password is also stored in the insecure R3 method. The stored BCODE hash can be quickly decrypted with simple PC hardware.

Motivation: To be downward compatible with SAP systems < 6.0, you can set the parameter to 0.

Implementation example: Excecute the transaction RZ10. Setting the parameter as follows:

- login/password_downwards_compatibility = 0

ID: 3.22-107/i373

Req 108	The system must not allow multiple registrations.

Multiple dialog logins by the same user in the same client must be deactivated by the SAP system, but it's allow to open in one session more Tabs,

Motivation: It's allow to open in one session more Tabs, but it's not allow, that one user as more the 1 registrations at the same time. This is steered by the parameter login/disable_multi_gui_login, it the user can login only once or multiple on the same client. Setting the value 1, the multiple login is deactivated.

The parameter login/multi_login_users steers, which user is allowed to login multiple on the same client. This parameter must be set to empty.

Implementation example: Execute TA RZ10 and set following parameters:

- login/disable_multi_gui_login = 1
- login/multi_login_users = <empty>

Requirement-ID: 352984ee

ID: 3.22-108/i373

Req 109	The authorization check for authorization objects must be activated.

The profile parameter auth/object_disabling_active is set in default on value Y and with that, it is possible to deactivate globally authorization checks on authorization objects with TAs SU25 respectively AUTH_SWITCH_OBJECTS. So it is possible that for example the check on the authorization object S_TCODE can be deactivated for the entire system. This must be avoided!

Motivation: Is the deactivation of authorization checks set, the SAP system doesn't perform authorization checks on selected authorization objects. A unauthorized access on data is possible.

Implementation example: 1) Execute TA RZ10 and set following parameter:

- auth/object_disabling_active = N

2) The right for execution of TAs respectively AUTH_SWITCH_OBJECTS must be set restrictive (authorization object S_TCODE).

Requirement-ID: eeee0592

ID: 3.22-109/i373

Req 110	When calling function modules from a remote system or program, an authorization check must be performed.

When calling RFC function modules from an RFC client program or another system, an authorization check is performed in the called system on the authorization object S_RFC. This checks for the name of the function group to which the function module belongs.

The parameter auth/rfc_authority_check controls whether a logon and an authorization check are performed for system function modules (see SAP Note 176064).

Motivation: Unauthorized logon to the system and unauthorized access on functions and thus to data must be prevented via automatic system checks.

Implementation example: The auth/rfc_authority_check parameter must be set to a value of 1 or higher. By executing

TA RZ10 you can set the parameter.

For higher security the parameter should be configured either to 6 or better to 9:

- 6=logon and authorization check required, but not for the RFC_PING function module. The RFC_PING function module is executed without logon and without authorization check.
- 9 =Logon and authorization check required for all function modules

The value 9 prevents an attacker from anonymously accessing RFC functions and thus obtaining additional system information for attack preparation. Potential functions for this would be:

RFC_PING: Simple accessibility check
RFC_SYSTEM_INFO: Information about the system (OS, database, version, identifier)
RFC_GET_LOCAL_DESTINATIONS: Returns all currently active RFC destinations at the same database
RFC_GET_LOCAL_SERVERS: Information about the local servers
SYSTEM_INVISIBLE_GUI: Sets the visibility of the current SAP GUI to invisible.

Requirement-ID: 033e7aef

ID: 3.22-110/i373

Req 111	SAP login tickets must be transferred securely.

If login tickets are used, they must be transferred securely. This can be controlled via a parameter.

Motivation: Unauthorized system access via a tapped SAP login ticket must be prevented.

Implementation example: Set following parameter with transaction RZ10:

- login/ticket_only_by_https = 1
  so the browser sends the login ticket only by an existing HTTPS connection.

ID: 3.22-111/i373

2.8.8. ABAP-Stack: ICM (Internet Communication Manager)

Req 112	Access to the SAP system via the ICM must be restricted by using an ACL filter, an authentication handler, and/or the HTTP rewrite handler.

Access on a SAP system must be secured with different procedures. Depending on the access ACL filter (ACL = Access Control List), authentication handler or HTTP rewrite handler can be used. The ICM offers different filter mechanisms. We recommend to use the easiest mechanism which can meet your security requirments. That means e. g. if an ACL is sufficient, use this, if it is the authentication handler than use this and if it is the rewrite handler use this. Please avoid to complex configurations, which can lead to a lack of security.

URLs should be filtered because otherwise information about the infrastructure and the configuration can be read (see also SAP Note 870127).

Filter Mechanism	ACL File	Authentication Handler	HTTP Rewrite Handler

Usage	Use ACL files to restrict access to specific client IP addresses or client IP address ranges when the restriction does not depend on the content of the HTTP request (i.e., not even the URL), and no HTTP error page is desired.	Use the Authentication Handler to set up URL filters (as a "white list" or "black list"). Rules in the Authentication Handler can also refer to specific client IP addresses, or to IP addresses of the server..	Use the HTTP rewrite handler for filters that cannot be mapped by ACL files or the Authentication Handler. The Rewrite Handler is a powerful tool for different filter mechanisms. It makes it possible to check numerous data of an HTTP request on the basis of a set of rules and to link them with each other.
Dynamic reloading of the configuration file	possible	possible	possible
White - respectively black list	• yes, both • also mixt possible	• yes, both • also mixt possible	• yes, both • also mixt possible
Filtering on URLs, handling of upper and lower case letters	no	• yes • standard configuration is case insensitive • can be configured	• yes • upper - and lower case sensitive can be configured per filter rule
Security Logging	yes	yes	no
Filtering on client IP addresses, including netmasks	yes	yes	yes
Parameter	**icm/server_port_<xx> option ACLFILE**	**icm/HTTP/auth_<xx> option PERMFILE**	**icm/HTTP/mod_<xx> option FILE**

Motivation: Prevent unauthorized web access on your backend system to avoid data and confidentiality loss.

Implementation example: Whatever filtering technique (ACL file, Authentication Handler and / or HTTP Rewrite Handler) you choose, please note the following:

- Evaluate the connection log content (see the written LOGFILE via the parameter`icm/HTTP/logging_xx`) , to determine the relevant URLs and describe them with exact rules in your URL permission table (White List).
- Be sure to filter / block the following paths in your URL permission table:
  - `/sap/public/icman/*`
    `/sap/wdisp/info`
- If you specify black lists (deny entries) in URL filters, use case-insensitive filters because AS ABAP treats URLs as case-insensitive.
- Instead of filtering the URLs, URLs can be deactivated if possible.

1) Set ACL files / filters for the system.
The option ACLFILE specifies the file that is used as access control list (ACL). With this an access control list can be

set up, which connections the ICM accepts and which not. For each port of the ICM a separate ACL file can be created. The string obeys the following syntax:

```
icm/server_port_<xx> = PROT=<protocol name>, PORT=<port or ser-
vice name>[, TIMEOUT=<timeout>, PROCTIMEOUT=<proctimeout>, EXT-
BIND=1, HOST=<host name>, VCLIENT=<SSL client verification>,
SSLCONFIG=ssl_config_<xx>, ACLFILE=<path for ACL file>]

# Description of the access points
icm/server_port_0 = PROT=HTTP , PORT=0 , TIMEOUT=30 , PROC-
TIMEOUT=60, ACLFILE=<Pfad zur ACL-Datei>
icm/server_port_1 = PROT=HTTPS , PORT=1443, TIMEOUT=30 , PROC-
TIMEOUT=60, ACLFILE=<Pfad zur ACL-Datei>

A row within the ACL file must follow the syntax:

<permit | deny> <ip-address> [tracelevel] [# comment]
```

The rules are checked sequential from the beginning („top down“). The first fitting rule will be taken for the connection („first match“). When no rule is fitting the connection is refused. A deny rule as a last rule must be entered (deny 0.0.0.0/0).

Example content of an ACL file:
permit 10.1.2.0/24 # permit client network
...
deny 0.0.0.0/0 # deny the rest

2) Set Authentication Handler for the system.

To protect the backend system (AS ABAP or AS Java), there is an HTTP subhandler (filter) that can block requests based on various criteria. If the filter is activated, it is passed through for each HTTP(S) request to the ICM before the request is sent to another HTTP handler (file access, cache, administration, redirect) or to the backend system (AS ABAP or AS Java). You can filter requests by the following criteria: URL, client IP address, server IP address, user name/user group/user password, pattern search in the URL. To set up limitations using URL filter, use follwoing parameter with its syntax:

```
icm/HTTP/auth_<xx> = PREFIX=<URL-prefix>
[,PERMFILE=<permission file>, AUTHFILE=<authentication file>,
FILTER=<name>]
```

Example for setting the parameter:

```
icm/HTTP/auth_0 = PREFIX=/, PERM-
FILE=D:\usr\sap\WDP\SYS\profile\perm_filter.txt
```

With the sub option PERMFILE the kind of the access to your permission file is configured. Create a permission file and enter rules. The rules are applied in the order they appear in the file, from top to bottom, and the first rule that matches the request is applied. There is an implicit "deny all" rule that is added to the end of the file, at runtime. Therefore, it is recommended to use "positive rules" (create only P or S rules, as everything else will be denied by default).

The permission file has following syntax:

- Rows for comment begin with a # and will be ignored
- Other rows are built up like:P/D/S <URL-pattern> <USER> <GROUP> <CLIENT-IP> <SERVER-IP>
- The letter infront of the row has follwing meaning:
    - P (Permit)
    - D (Deny)

- S (Secure) only secure connection are allowed (HTTPS)
- <URI-pattern> ist der Abschnitt der URL, der im Abschnitt Cache-Key als translated path bezeichnet ist

- For the URL-Pattern you can use the wildcard *, but only at the beginning or the end of the <URL-pattern>.
- For the client - and server IP adess you can use an exact compare, the wildcard * or a net mask syntax.

Example of a white list with forcing HTTPS by setting the "S" at the beginning of the line:

```
# This is the "permission file" used by the ICM/SAP Web Dispatcher Au-
thentication handler
# webadmin and ping only from Admin-workstations <IP-Admin workstation> allowed
S /sap/bc/ping * * <IP-Admin workstation/net class> *
S /sap/wdisp/admin/* * * <IP-Admin workstation/net class> *
# Access from Webapps and WebGUI (https only) from internal IPs
S /sap/bc/webdynpro/* * * <internal IP/net class> *
S /sap/bc/gui/sap/its/webgui * * <internal IP/net class> *
# Access from Webapps and WebGUI (https only) from Internet
S /sap/bc/srt/rfc/meinwebservice * * * * *
# Everything not listed above will be denied because of the final implicit rule "D * * * * *"
```

3) Set HTTP rewrite handler for the system.

The HTTP rewrite handler is a very powerful tool. You can use it for various actions:

- Delete, add, and enhance HTTP header fields:
  You can delete or add HTTP header fields, or enhance them with additional values.
- Rewrite URLs:
  You can rewrite both the URL path and the query string of an incoming HTTP request into another previously-defined URL path or query string. You can link the execution of URL modifications to one or more conditions.
- Redirect or filter URLs:
  You can redirect the URL of an incoming HTTP request to a different application server or URL. You can also filter incoming HTTP requests by defined patterns.
- No modification action / nop action (no operation):
  Modifiers (options), such as break, restart, and noescape can occur after a modification action (rule). So that you can use modifiers without definitions too, the nop action is provided.

To set up access restrictions via HTTP rewrite handlers, use the parameter with the following syntax:

```
icm/HTTP/mod_<xx> = PREFIX=<URL-Prefix>[, FILE=<action file>]
```

SAP recommendation: To have the same filter rules on all instances, place the filter rules file in a global directory and set the parameter in the default profile DEFAULT.PFL. The filter rules are located in the icm_filter_rules.txt file in the $(DIR_GLOBAL)/security/data directory:

```
i c m / H T T P / m o d _ 0 = P R E -
FIX=/,FILE=$(DIR_GLOBAL)/security/data/icm_filter_rules.txt
```

The activation of the rewrite handler is configured by an action file (use option FILE). The action file is a text file with one action (rule) per line. Comment lines start with a gate (#). The sequence within the action file describes the sequence of performing the single action. You can perform several URL changes in series. You can connect the execution of the HTTP headerfield changes with one or more conditions. No implicit "deny all" rule is added at the end, unlike when using the Authentication Handler. Notice also that a "negative list" has to be created, instead of a "positive list" like in the Authentication Handler.

Use the following syntax to define a rule in the action file (s. SAP Help):
<operation> <pattern> <dest> [<option>]

Enforce HTTPS with the rewrite handler:
To prevent the use of unencrypted HTTP, you have to change the protocol of the rewrite handler from HTTP to HTTPS. This will prevent error messages in the browser if users inadvertently access the system with an HTTP URL. Note that not all HTTP clients follow this redirect. While the redirect configuration ensures that no HTTP access to the system is possible, it is possible that individual users of the system, e.g. web service end points, must be switched from HTTP to

HTTPS.
Example of strict use of HTTPS:

```
# We allow access to the WEB GUI, but only through HTTPS
# The indentation is optional
if %{SERVER_PROTOCOL} !stricmp "HTTPS" [and]
if %{PATH} regimatch ^/sap/bc/gui/sap/its/webgui(.*)
RegForbiddenURL ^/sap/bc/gui/sap/its/webgui(.*) - [break]
```

Example avoid execute a ping:

```
# We deny access to the "ping" service
RegForbiddenURL ^/sap/public/ping(.*) - [break]
```

ID: 3.22-112/i373

Req 113	The ICM server (port) must be configured securely.

The ICM must be configured so that only secure protocols have access. This is done by setting the parameter icm/server_port_<xx> to use the HTTPS protocol. With the PROT and PORT options you specify the protocol and the corresponding port by the port number or service name. Exactly only one service can be bound to a port.

Motivation: Unencrypted communication can be read which can lead to a threat for the system. User-IDs with the related password can be tapped and can be misused for an unauthorized login, which can lead to an unauthorized access on data.

Implementation example: By setting the parameter icm/server_port_<xx> the use of HTTPS is enforced and accesses can be controlled. This profile parameter of the ICM configures the server port. The string obeys the following syntax:

```
icm/server_port_<xx> = PROT=<protocol name>, PORT=<port or service>[,
TIMEOUT=<timeout>, PROCTIMEOUT=<proctimeout>, EXTBIND=1, HOST=<server
name>, VCLIENT=<SSL client verification>, SSLCONFIG=ssl_config_<xx>,
ACLFILE=<path to ACL-file>]
```

Description of access points (_0 and _1 are predefined values in AS ABAP for outgoing connections, _2 is an example for an incoming HTTPS connection, _3 is an example for an outgoing HTTPS connection):

```
icm/server_port_0 = PROT=HTTP,PORT=0,TIMEOUT=30,PROCTIMEOUT=60
icm/server_port_1 = PROT=SMTP,PORT=0,TIMEOUT=120,PROCTIMEOUT=120
icm/server_port_2 = PROT=HTTPS , PORT=443, TIMEOUT=30 , PROC-
TIMEOUT=60, ACLFILE=<path to ACL-file>
icm/server_port_3 = PROT=HTTPS,PORT=0,TIMEOUT=30,PROCTIMEOUT=60,
ACLFILE=<path to ACL-file>
```

Please note that the value 0 for PORTS means that no port is opened for incoming connections for the specified protocol. Ports for incoming connections must be configured explicitly for security reasons.

Attention: SAP delivers predefined values with the ICM. These must be checked for validity and, if necessary, deactivated with " e.g.:

```
icm/server_port_0 =“
icm/server_port_1 =“
```

ID: 3.22-113/i373

Req 114	The administration of the ICM via the web-based administration interface must be secured.

You have to set the parameter icm/HTTP/admin_<xx> when you want to monitor and administrate the ICM via the web-based administration interface (browser).

Motivation: The web interface for the administration of the ICM must not be accessible for all users. This must be protected to prevent unauthorized access.

The display of system information must also be prevented, as this information could be misused to attack the system.

Implementation example: The parameter icm/HTTP/admin_<xx> has following syntax:

```
icm/HTTP/admin_<xx> = PREFIX=<URL-prefix>, DOCROOT=<root directory of
the administration files> [, AUTHFILE=<file name of the authorisation
file>, PORT=<TCP/IP-port on which admin requests are accepted>,
HOST=<server name or IP adress on which admin requests are accepted>,
CLIENTHOST=<client name or IP adress from which admin requests are ac-
cepted>, ALLOWPUB=<value> ]
```

Example for setting the parameters within the profile:

```
icm/HTTP/admin_0 = PRE-
FIX=/sap/admin,DOCROOT=$(DIR_ICMAN_ROOT/
admin,PORT=1443,HOST=localhost;<IP host
ICM>,CLIENTHOST=localhost;<hostname from Administrator clients>,
[ALLOWPUB=FALSE]
```

Setting the value for PREFIX: URL prefix for which this HTTP subhandler should be called. By default, the URL prefix for administration is set to /sap/admin.

Setting the value for DOCROOT: root directory of the administration files.

Setting the value for AUTHFILE: filename of the authentication file (default value: icmauth.txt) containing the hash values of the passwords for the admin users. If no user authentication is required (e.g. because it is already done in the Authorization Handler), this can be achieved by specifying AUTHFILE=none.

Setting the value for PORT (MUST): By default, admin requests are accepted on all local TCP/IP ports. This is not allowed. A restriction to a TCP/IP port that is only locally accessible and on which admin requests are accepted must be implemented. This way the use of HTTPS can be easily implemented.
! Note: This port must have been specified via the icm/server_port_<xx> parameter. An HTTPS port must be set up for this, otherwise the administrator passwords will be transmitted in clear text when logging in. Insecure protocols are not allowed!
Example:

```
icm/server_port_<xx>=.... PROT=HTTPS, PORT=1443, ....
```

Setting the values for HOST (MUST): If nothing is specified here, admin requests are accepted from all host names. This is not allowed. The restriction of access to the local host must be implemented, so that only the local users can use the web-based interface. For this, specify the value localhost or 127.0.0.1 and the <IP of the ICM host>.

Setting the values for CLIENTHOST (optinal): If nothing is specified here, admin requests are accepted from any client machine. Administration must be restricted to specific client machines (machines of ICM administrators). To do this, enter the value localhost and the <machine names or IP addresses of the administrators>.

Setting the value for ALLOWPUB: If ALLOWPUB=TRUE is set, public and read accesses to certain administration pages are allowed under the path "public/index.html" without login (e.g. "Monitor", "Active Services", "Core Thread Status", "Hostname Buffer", "Release Information" and "MPI Status"). Access to these pages without logging in must be prevented. Even if access is restricted with the HOST and CLIENTHOST subparameters (see above), system information that could be used to attack the system must only be viewable by logging on to the system. ALLOWPUB=FALSE must be set.

For more information s. SAP note 870127 and the SAP Online help.

ID: 3.22-114/i373

Req 115	The authorization file "icmauth.txt" must be secured for unauthorized access.

Within the authorization file (Default: icmauth.txt) all users with their group membership are listed in plain text. The

passwords are encrypted within that file. The path is following the conventions of the operating system. For security reasons this file must be accessible by the administrations user of the SAP systems (<SID>adm) only.

Motivation: The file "icmauth.txt" must be protected against unauthorized access, to decrease the threat for an system attack. The encrypted password can be easily cracked and be used than for an unauthorized access.

Implementation example: Set restricted access rights on the authorization file icmauth.txt with following command (UNIX):

```
chmod 700 <authorization file>
```

The location of the file and its name are specified in the `icm/authfile` parameter.

ID: 3.22-115/i373

Req 116	The access to the ICM administration directory must be set restrictively.

The administration directory of the ICM must have restrictive permissions. Normaly the administration directory is defined by the same variable and directory is $DIR_ICMAN_ROOT/admin and contains the files for the administration by web interface.

Motivation: The administration directory contains data with a high protection requirement (e. g. file icmauth.txt).

Implementation example: Changing the access rights for the administration directory with the following commands:

```
chown <sid>adm:sapsys <directory name>
chmod 750 <file name> or if more restrictive is possible => 700
```

ID: 3.22-116/i373

Req 117	The web administration is personalized and not done by the user "icmadm".

To monitor and administer the ICM from a browser, you need a user.

The icmadm user is the default user for the web administration interface and is stored in the icmauth.txt file when the administration user is set up.

Personalized administrators are created for traceable administration.

Motivation: Unauthorized administration access can lead to the possibility to read and change the data and or of a system outage.

Implementation example: The following steps must be performed when setting up the administration users:

1. As <sid>adm, create the icmadm administration user with icmon. This generates the icmauth.txt authorization file.
2. Create personified administration users with icmon.
3. Try access with the personalized users.
4. Change the password of the default user icmadm according to the password guidelines.

ID: 3.22-117/i373

Req 118	The access on the command "icmbnd" for binding of ports must be set restrictive.

Usually the ICM binds the ports by itself (recommended). That the ICM can also bind the "well known ports", the ports from 0 to 1023, it must use the external command icmbnd. Due to setting the option EXTBIND=1 by the definition ot the parameter icm/server_port_<xx> the command icmbnd can be executed by the ICM for configuration the ports < 1023.
The command icmbnd can be executed directly (without the configuration using parameter icm/server_port_<xx>)

(not recommended).

The path of the file icmbnd is defined in following parameters of the profile:

- exe/icmbnd
- DIR_EXECUTABL

Motivation: Unauthorized execution of the command icmbnd can lead to a misconfiguration. Thereby attacks over the ICM can be performed on downstream systems. This can lead to unauthorized access on data or to system outage.

Implementation example: Set access rights restrictive on the file icmbnd:

```
chown root:sapsys icmbnd

chmod 4750 icmbnd
```

ID: 3.22-118/i373

Req 119	Don't display error messages, genarted by default service.

When the service use default error messages, normally the service display Versions and Name.

Motivation: No information about the system may be output in "error pages" that could be exploited to attack the system.

Implementation example: The parameter /HTTP/show_detailed_errors must be set to "false"

ID: 3.22-119/i373

Req 120	The information about the server in the HTTP response to the client must be limited.

The parameter controls whether or not the server header field should be inserted in HTTP responses from the server to the client. The server header field comprises the values of parameters is/server_name and is/server_version. For security reasons, SAP recommends not to pass this information to the client.

Motivation: No information may be passed to the client that could be exploited to launch an attack.

Implementation example: To do this, set the is/HTTP/show_server_header parameter as follows:

```
is/HTTP/show_server_header = FALSE
```

ID: 3.22-120/i373

Req 121	The security log for the ICM must be enabled.

You can use this parameter icm/security_log to control the output of the security log from the ICM and SAP Web Dispatcher.

Administrators can use the security log (icm/security_log) to help identify any potential unauthorized access to the system. The following irregularities are logged:

- Data with invalid syntax
- Attempted access to objects that do not exist (NOT found)
- Access to objects that is not permitted due to filter rules (permission denied)
- Logon errors to Web administration (in ICM and Web Dispatcher)

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.

Implementation example: Set the parameter icm/security_log to enable the security log.

Syntax of the parameter:

```
icm/security_log = LOGFILE=<file name>, LEVEL=<security level>,
MAXSIZEKB=<max size in KB>, SWITCHTF=<options>, FILEWRAP=on
```

LOGFILE: Name of the file. Time specifications make the name unique.
LEVEL: Specifies the level (1 -3) of logging. Level 1 only records the reason for the entry. Level 2 records additional information about the status of the connection and the start of the data that gave rise to the entry (this is the default value if option LEVEL is not specified). Level 3 logs all the data that gave rise to the entry.
MAXSIZEKB: Specifies the size of the log file. If this size is exceeded, the current file is closed and a new one (with a new name) is opened. By defining LOGFILE the name becomes unique and thus overwriting is prevented.
SWITCHTF: A new log file can not only be created if the file has reached a certain size, but also when the time data changes (every new hour, day or month possible).
FILEWRAP: If FILEWRAP=on is active, every time a new file is opened, the existing log file is reset and overwritten (!). Therefore, there is always only one log file with the current log data. If you omit this option, once the size has been exceeded a new file is written (see MAXSIZEKB and SWITCHTF).

Example setting the parameter (option FILEWRAP is not set!):

```
icm/security_log = LOGFILE=dev_icm_seclog-%d-%m-%y_%h:%t:%s, LEVEL=2,
MAXSIZEKB=10000, SWITCHTF=day
```

ID: 3.22-121/i373

Req 122	HTTP logging must be enabled.

You can use these parameters icm/HTTP/logging_<xx> (for incoming requests) and icm/HTTP/logging_client_<xx> (for outgoing requests) to control the output of the connection log from the ICM and SAP Web Dispatcher and log activities.

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.

Implementation example: Set the following two parameters:

1) Logging of incoming connections

Syntax of the parameter:

```
icm/HTTP/logging_<xx> = PREFIX=<URL prefix>, LOGFILE=<log file name>
[, LOGFORMAT=<format>, FILTER=<filter>, MAXSIZEKB=<size in KBytes>,
SWITCHTF=<options>, FILEWRAP=on]
```

PREFIX: URL prefix that is called for this HTTP subhandler (for example, /).
LOGFILE: Name of the output file in the file system. With each restart the ICM checks whether the file specified as the log file exists. If it does it continues to write to this file, if it does not, it creates a new file. You can attach another timestamp to the actual file name, which makes the file unique.
LOGFORMAT: There are the different formats for log files. You can reproduce the “NCSA Combined Log Format” known to other Web servers by using the following format string: %h %l %u %t "%r" %s %b "%{referer}i" "%{user-agent}i"
FILTER: The filter property specifies that an HTTP request is logged only when a certain header field (for example, an HTTP header field) is in the request or response.
MAXSIZEKB: Specifies the size of the log file. If this size is exceeded, the current file is closed and a new one (with a new name) is opened. By defining LOGFILE the name becomes unique and thus overwriting is prevented.

Example setting the parameter (option FILEWRAP is not set!):

```
icm/HTTP/logging_0 = PREFIX=/, LOG-
FILE=incoming_icm_log-%d-%m-%y_%h:%t:%s, LOGFORMAT=%h %l %u %t "%r" %s
%b "%{referer}i" "%{user-agent}i", MAXSIZEKB=10000, SWITCHTF=day
```

2) Logging of outgoing connections
Syntax of the parameter:
```
icm/HTTP/logging_client_<xx> =  PREFIX=<URL prefix>, LOGFILE=<log file name> [, LOGFORMAT=<format>, FILTER=<filter>, MAXSIZEKB=<size in KBytes>, SWITCHTF=<options>, FILEWRAP=on]
```

The options are the same as for icm/HTTP/logging_<xx>.

Example setting the parameter (option FILEWRAP is not set!):
```
icm/HTTP/logging_client_<xx>  =  PREFIX=/,  LOG-FILE=outgoing_icm_log-%d-%m-%y_%h:%t:%s, LOGFORMAT=%h %l %u %t "%r" %s %b "%{referer}i" "%{user-agent}i", MAXSIZEKB=10000, SWITCHTF=day
```

Attention for icm/HTTP/logging_<xx> and icm/HTTP/logging_client_<xx>: The log files for incoming and outgoing requests must have different names, that is, you cannot not write both directions to one file.

ID: 3.22-122/i373

2.9. Securing the Java-Stack

2.9.1. Java-Stack: Operating System Directories

Req 123	The directories and files in which configuration files from the Java installation are stored on the server must be configured with restrictive access authorizations.

All directories and files below JC<no> must belong to SIDadm and may not be overwritten or modified by other operating system users. In the case of double stack systems (ABAP+JAVA) it should be noted that the JAVA stack is an add-in installation that runs under the same SID as the ABAP stack. The JAVA stack is installed in the ABAP stack installation directory.

Motivation: If access to the Java stack resources is not restricted and protected, attackers might be able to access functions and data without being authorized to do so. Unauthorized access to resources in conjunction with vulnerabilities in the implementation of authorization on the application layer (e.g., no explicit authorization checks, etc.) can cause the entire Java stack to be compromised.

Implementation example:
```
cd JC
find . \! –user sidadm | xargs ls -lad
```
all result entries must be changed to sidadm:
```
chown sidadm <file/directory list>
find . \( -type f -o -type d \) -perm +022 | xargs ls –lad
```
Example output with incorrect write access rights:

- -rwxrwxr-x 1 sidadm sapsys 625826 2010-09-11 00:38 ./exe/libsapcsa.so.old
- -rwxrwxr-x 1 sidadm sapsys 7062188 2010-09-14 21:34 ./exe/sapstartsrv.old
- drwxrwxr-x 6 sidadm sapsys 4096 2009-02-24 17:01 ./igs
- drwxr-xrwx 2 sidadm sapsys 4096 2010-10-09 20:35 ./log

The following commands then have to be removed for deletion in the example:
```
chmod o-w log
chmod g-w igs exe/libsapcsa.so.old exe/sapstartsrv.old
```
If the “find” command only delivers output such as the following, this merely means that no entries with incorrect write access rights have been found and that there is no need for action:
```
drwxr-xr-x 7 sidadm sapsys 4096 2010-10-09 20:44 .
```

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: 3.22-123/i373

Req 124	Access to the SAP installation logs must be severely restricted.

For the ABAP and Java stack, the installation files (logs) are stored on the standard SAP operating directory /usr/sap/<SAPSID>/SYS. This is the default value of the DIR_INSTALL variable, which specifies the location for the SAP installation directory.

Motivation: Installation logs contain data with a high protection requirement. After the installation is complete, only authorized users may access it.

Implementation example: Changing the access rights for the installation logfiles with the following commands:

```
chown root <file name>
chmod 700 <file name>
```

If possible, the installation logs can also be deleted.

ID: 3.22-124/i373

Req 125	The directory "SIDADM-Home" and its content must be protected from non-authorized access.

On operation system, the home directory from "SIDADM-Home" must be protected from non-authorized access.

Motivation: The content under the HOME-directory must be protect, there are useful information's about the application.

Implementation example:

OS-Verzeichnisname	OS-Zugriffsberechtigung	OS-User/OS-Gruppe
<home Verzeichnis von <sapsid>adm>	700	<sapsid>adm/sapsys
<home Verzeichnis von <sapsid>adm>/*	700	<sapsid>adm/sapsys

OS-Commands for setting user and group permissions:

```
chown <OS-User>:<OS-Gruppe> <Verzeichnisname>
chmod <OS-Berechtigung> <Verzeichnisname>
```

Beispiel:

```
chown <sapsid>adm:sapsys <home Verzeichnis von <sapsid>adm>
chmod 700 <home Verzeichnis von <sapsid>adm>
```

ID: 3.22-125/i373

Req 126	On operation system the directory "/spamnt/<SAPSID>" and their sub-directory's with its contents must be protected from non-authorized access.

On operation system, the directory "/spamnt/<SAPSID>" and there sub-directory's must be protected from non-authorized access, Under the directory "/sapmnt/<SAPSID>" are very importent subdircotrys and content for the applikation.

Motivation: Usually the files are on a global host and are distributed via NFS, SMB. If is possible to access to directory and files, a manipulation for this data-file can be change the working of the application.

Implementation example:

OS directory name	OS access	OS-user/OS-group
/sapmnt/<SAPSID>/exe	775	<sapsid>adm/sapsys
/sapmnt/<SAPSID>/global	700	<sapsid>adm/sapsys
/sapmnt/<SAPSID>/profile	755	<sapsid>adm/sapsys

OS command for setting access and the OS-users with the relevant OS-group:

```
chown <OS-user>:<OS-group> <OS directory>
chmod <OS access> <OS directory>
```

Example:

```
chown <sapsid>adm:sapsys /sapmnt/<SAPSID>
chmod 751 /sapmnt/<SAPSID>
```

ID: 3.22-126/i373

Req 127	The directory "/usr/sap/>SAPSID> und its content must be protected for non authorized access of the operation system.

The content in the directory "usr/sap/<SAPSID>" is for the application essential. There must be protected for access from operations system.

Motivation: Unauthorized access to directory's and files can be change Information, at worst case the application is stop working.

Implementation example:

OS directory name	OS access	OS user/OS group
/usr/sap/<SAPSID>	751	<sapsid>adm/sapsys
/usr/sap/<SAPSID>/<Instance ID>	755	<sapsid>adm/sapsys
/usr/sap/<SAPSID>/<Instance ID>/sec	700	<sapsid>adm/sapsys
/usr/sap/<SAPSID>/SYS	755	<sapsid>adm/sapsys
/usr/sap/<SAPSID>/SYS/*	755	<sapsid>adm/sapsys

OS command for setting access and the OS-users with the relevant OS-group:

```
chown <OS-user>:<OS-group> <OS directory>
chmod <OS access> <OS directory>

Example:
chown <sapsid>adm:sapsys /usr/sap/<SAPSID>
chmod 751 /usr/sap/<SAPSID>
```

ID: 3.22-127/i373

Req 128	The SAP transport directory and its content must be protected for non-authorized access.

The content of the transport directory must be protected for non-authorized access. Usually the directory is deployed on a host and distributed via SMB or NFS protocol.

Motivation: The content of the transport directory must be protected for non authorized access. The directorys are importend for proper work.

Implementation example:

OS-Verzeichnisname/OS-Dateien	OS-Zugriffsberechtigung	OS-User/OS-Gruppe
/usr/sap/trans	775	<sapsid>adm/sapsys
/usr/sap/trans/*	770	<sapsid>adm/sapsys
/usr/sap/trans/.sapconf	775	<sapsid>adm/sapsys

OS-Kommandos zum Setzen der OS-Zugriffsberechtigungen und des OS-Users und der OS-Gruppe:

```
chown <OS-User>:<OS-Gruppe> <Verzeichnisname>
chmod <OS-Berechtigung> <Verzeichnisname>
```

Beispiel:

```
chown <sapsid>adm:sapsys /usr/sap/trans
chmod 775 /usr/sap/trans
```

ID: 3.22-128/i373

Req 129	On operating system level SAP tools must be secured from unauthorized access.

Ensure that the SAP tools have his own users, groups and directory's on operation system.

Motivation: Having unauthorized access to the files the file content could be changed. Changes on the file content could lead to system failures and/or to an outage.

Implementation example: For minimum following tools access must be restricted:

Tool name	Directory	Access	User/Group	Function
sapevt	/usr/sap/<SAPSID>/SYS/exe/run	750	<sapsid>adm/sapsys	Netweaver events starting on operating system level
saposcol	/usr/sap/<SAPSID>/SYS/exe/run	4710	root/sapsys	Collects information-en from the operating system

ID: 3.22-129/i373

2.9.2. Java-Stack: Communication

Req 130	Tools for administration of the AS Java must be bind to the AdminLAN interface.

To reduce the attack vector, it is necessary that the administration tools and the associated ports must use the Admin-LAN (a separate network interface that is separated from Office-LAN). Tools that use plain text protocols are also no longer permitted in the protected AdminLAN (e.g. the Shell-Administrator use telnet). The Sell Administrator can be called locally.

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: Administrative web access via the Netweaver Administrator (NWA) is realized via a jump server to the AdminLAN.
For security reasons, access to the AS Java via Telnet is restricted to the host 127.0.0.1 (localhost) for the Shell Administrator tool.

ID: 3.22-130/i373

2.9.3. Java-Stack: User and authentication

Req 131	The Account administrator, J2EE_ADMIN_<SID> or J2EE_ADMIN must be protected from unauthorized access.

For the AS Java the administrator (data source data base), J2EE_ADMIN_<SID> (data source ABAP) or J2EE_ADMIN user must exist due to it is used by certain applications on the AS Java to perform administrative and installation tasks, for example software deployment and undeployment. The default administration user must be locked. For the daily administration tasks personalized accounts with equivalent administrative privileges for the AS Java administrators must be created.
For the certain applications on the AS Java to perform administrative and installation tasks, for example software deployment and undeployment the default administrator user must be unlocked from the personalized administrators and after the neccessary tasks it must be locked again.

Motivation: The AS Java default administrator user is mandatory and is used for performing administrative and installation tasks. But unauthorized access for this technical user must be prevented.

Implementation example: Perform following steps (see SAP note 2526747):

- Create personalized accounts for the AS Java administrators
- Set the password for the AS Java default administration user according to the DTAG security policies
- Lock the standard user

For the SAP AS Java user management the Netweaver Administrator (NWA) is used.

If user administration for AS Java is performed in AS ABAP, transactions SU01 or PFCG are used for user and role administration.

ID: 3.22-131/i373

Req 132	The Account guest, J2EE_GUEST_<SID> or J2EE_GUEST must be protected from unauthorized access.

The guest (data source data base), J2EE_GUEST_<SID> (data source ABAP) or J2EE_GUEST user must exist within the SAP Java Stack due to it is used for all areas without authorization checks e. g. for the login screen and for starting the system. The user must receive a password that corresponds to the DTAG security policies and it must be locked (s. SAP Note 2526747).

Attention: The user name for J2EE_GUEST can be freely defined. Apply the requirements to the user you have defined as the J2EE_GUEST user.

Motivation: Guest access is mandatory and is used for all areas without an authorization check (for the login screen and for starting the system). But unauthorized access for this technical user must be prevented.

Implementation example: Perform following steps (see SAP note 2526747):

- Set the password for the AS Java default administration user according to the DTAG security policies

- Lock the standard user

For the SAP AS Java user management the Netweaver Administrator (NWA) is used.

If user administration for AS Java is performed in AS ABAP, transactions SU01 or PFCG are used for user and role administration.

ID: 3.22-132/i373

Req 133	When using Adobe Document Services (ADS), the adsuser communications user must be set up securely.

Used for communication between the AS Java and the Adobe Document Services (ADS). ADS is only available in connection with an AS Java. The data source for the user can be Java or ABAP, depending on how the user management was defined in the UME. In order to set up the adsuser user securely, the password must be set according to the DTAG security policies as well as the correct user type and the valid role must be set.

Motivation: Unauthorized access by this technical user must be prevented.

Implementation example: The adsuser is securely configured if, ...

- ... you set the password according to the DTAG security policies. Note: The set password must be made known to the destionation service.
- ... the user is assigned the user type "service user" for Java or "system" for ABAP.
- ... the user is only given the ADSCaller role

For the SAP Web AS Java user management the Netweaver Administrator (NWA) is used. For the SAP Web AS ABAP user management and role maintenance transactions SU01 or PFCG can be used.

ID: 3.22-133/i373

Req 134	The service user ADS_AGENT must be set up securely.

The service user ADS_AGENT in the SAP Web AS ABAP is required for the communication between the SAP Web AS ABAP and the Adobe document services when you use PDF-based printforms in the ABAP environment. In order to set up the ADS_AGENT user securely, the password must be set according to the DTAG security policies as well as the correct user type and the valid role must be set.

Motivation: Unauthorized access by this technical user must be prevented.

Implementation example: The ADS_AGENT is securely configured if, ...

- ... you set the password according to the DTAG security policies. Note: The set password must be made known to the destionation service.
- ... the user is assigned the user type "service".
- ... the user is only given the SAP_BC_FP_ICF (Double Stack) or SAP_BC_FPADS_ICF (Single Stack) role.

The user is defined in ABAP. Use transaction SU01 for this.

ID: 3.22-134/i373

Req 135	The system user sapjsf or sapjsf_XYZ must be set up securely.

sapjsf or sapjsf_XYZ is a system user and handles the communication between the User Management Engine (UME) and the user administration of the SAP NetWeaver Application Server (AS) ABAP. It is an user which is created in

ABAP at the time the Java stack is installed or later (AS ABAP as data source) and it is necessary that Java is able to login to the ABAP via RFC with or without SNC.

Recommendation: Since it is possible for multiple AS Java to be connected to one AS ABAP, the user should be renamed “sapjsf_XYZ”, whereby XYZ refers to the name of the relevant AS Java.

Motivation: Unauthorized access by this technical user must be prevented.

Implementation example: The sapjsf or sapjsf_XYZ user is securely configured if, ...

- ... you set the password according to the DTAG security policies.
- ... the user is assigned the user type "system" for ABAP.
- ... the user is only given the following roles as of release 6.20:
    - SAP_BC_JSF_COMMUNICATION_RO provides all authorizations for read access to user data, for authenticating remote users, and several low-level RFC authorizations. For example, users can still change their own password. This role provides sufficient authorization if you do not want to perform administrative changes from the UME: for example, add a new user or change a last name.
    - SAP_BC_JSF_COMMUNICATION is the same as the above role, but additionally provides authorization to modify and delete all user-related data.

If you change the password of this user in transaction SU01, you must also change the password stored for this user in the AS Java database accordingly. You have to restart the AS Java for this procedure. You should therefore plan this required downtime for the AS Java. You also have to enter the new password for the service user for UME-ABAP communication in the RFC connection UMEBackendConnection in the AS Java (destination service in the SAP NetWeaver Administrator). Test the connection and, if it fails, enter the connection information again until the connection is successful. Restart the AS Java. The AS Java now supports the new password of the system user for UME-ABAP communication. If the AS Java cannot start, activate the emergency user and maintain the password of the system user in the RFC destination. If necessary, please read further details in the SAP help.

ID: 3.22-135/i373

Req 136	When using an AS Java, the user administration must be coupled to an "Active Directory" or LDAP service.

When using an AS Java, the user administration must be coupled to an "Active Directory" or LDAP service due to the user administration of an AS Java cannot reach the same security level comparable to that of an AS ABAP in user administration.

Motivation: The default Useradministration from AS JAVA has not the same protections as the group AD or LDAP.

Implementation example: Connection to a central IAM.

ID: 3.22-136/i373

Req 137	System cookies must be protected via the HttpOnly attribute.

Java EE applications can use system cookies to track user data (such as sessions tracking, logon data, and so on). These cookies contain sensitive information about the user. Therefore, to prevent potential misuse of session information, cookies should not be exposed to client side scripts. To increase the security protection of system cookies, you can enable the use of the additional system cookie attribute HttpOnly.

Declaring a cookie as HttpOnly increases the security of your system because it eliminates access to this cookie in the Web browser from client-side scripts, applets, plugins, and so on. This can have side effects because some applications use such technologies and also rely on this information. These applications may no longer function correctly be-

cause they cannot access this information. Please check if applications need this informartion.

Motivation: Cookies that have the HttpOnly attribute cannot be accessed by unauthorized JavaScript. This is a protection against cross-site scripting.

Implementation example: Secure the session cookies with following configuration:

- Set the HTTP service property "SystemCookiesDataProtection" to value "true". With that the HttpOnly attribute for system cookies is active.
- Set the ICMproperty "disable_url_session_tracking" to value "false". This enables URL session tracking.

Note: If both parameter are set to value "true", AS Java cannot handle HTTP sessions correctly.

ID: 3.22-137/i373

Req 138	For secure transmission of system cookies you must set the "secure" attribute.

You use the HTTP service property SystemCookiesHTTPSProtection to set the Secure attribute to the Web session tracking and the load balancing (sapdb_) cookies it is related to. If the property is set to true, the Secure attribute is set to the system cookie and cookies marked as secure are only transmitted in case the communication channel is a secure one. In this case, it is obligatory that you use SSL since these cookies are not transferred via plain HTTP and session tracking on HTTP will not work.

You use the SecuritySessionIDHTTPSProtection property of the HTTP Service to protect the security session identifier cookie from being sent over unsecured HTTP connection. When the property is set to true, the Secure attribute for the cookie is set and the browser will send it only over HTTPS.

Motivation: Prevent unauthorized access to the session cookie by ensuring that the cookie is only transported over a secure connection.

Implementation example: To protect the system cookie from insecure transmission, set the HTTP service property "SystemCookiesHTTPSProtection" to value "true".

ID: 3.22-138/i373

Req 139	Secure the logon ticket cookies by setting the HttpOnly attribut.

Logon tickets are cookies that are used for user authentication and Single Sign-On on AS Java.

Motivation: Prevent unauthorized access to the Logon Ticket cookie.

Implementation example: To set this attribute for logon tickets, set the user management engine (UME) property "ume.logon.httponlycookie" to the value "true".

ID: 3.22-139/i373

2.9.4. Java-Stack: Authorization management

Req 140	A distinction between user and role administration must be implemented in both the user administration and the system administration.

A separation between role and user administration as well as user administration and system administration is necessary.

Motivation: In the standard configuration, there is no separation of user and role administration for J2EE administrators. In order to prevent misuse, these areas are to be organizationally separated.

Implementation example:

Responsibility	Java
Role administration	Content administration
Profile administration	Content administration
User administration	User administration

Recommendation: Do not use local user administration for application users in UME, but link them to the ABAP user administration (in the case of dual stack), to the Active Directory or LDAP.

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: 3.22-140/i373

Req 141	It must be ensured that administration authorizations are only granted to the administrators responsible.

In the default configuration of AS Java, no distinction is made between users and administrators in the user administration. It must be ensured that distinctions are made via the authorization concept.

Motivation: Introduce Segragation of Duties to prevent unauthorized administrative actions on the AS Java.

Implementation example: Comparison with the customer-specific authorization concept to verify which user owns a certain UME action:

Export all roles via J2EE UME IdentityManagement (<log>://<host>:<port>/useradmin => Roles (Search and mark all) => Export).

After determining the roles that contain critical authorizations, they can be checked against user assignments.

http://help.sap.com/saphelp_nwmobile71/helpdata/de/4a/e06f429c789041e10000000a1550b0/frameset.htm

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: 3.22-141/i373

Req 142	The self-registration function for users must be disabled.

Self-registration means that new users can register themselves when logging in. In the default configuration, the User Management Engine (UME) is set up to allow a basic form of self-registration. On the login screen, users can register themselves by filling out a form. After that, they are registered users and can access all content assigned to the default "All" group. They do not need to be approved by an administrator. If you want to set up self-registration with an approval workflow, you must configure User Management for enterprise use.

The self-registration function for users should be disabled if it is not required for the application. (Example: eBest self-

registration)

Motivation: The UME self-registration function is enabled by default. In this scenario, attackers can create an account in the Java stack and then exploit any existing vulnerabilities in the authorization concept. This can then lead to unauthorized access to the data and functions in AS Java.

Implementation example: This is achieved by setting the ume.logon.selfreg parameter to FALSE.

Or you can **uncheck** the following checkboxes in the Netweaver Administrator (NWA) configuration under the "User Management UI" tab:

- Enable self-registration of guest users.
- Enable self-registration of companies

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: 3.22-142/i373

Req 143	Security relevant events must be logged.

Systems must be able to log security relevant events and it must be possible to activate that logging functionality. 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. It must be possible to activate the following security events:

- Authentication of users or systems (successful and failed)
- Administrative access (incl. executed commands/changes)
- Unauthorized/failed and erroneous access attempts (e.g. access on not allowed functions by an account)
- Administration of accounts
- Changing of group membership of accounts.
- Further security relevant events specific to the system

Motivation: The detection of (targeted) attacks and successful or attempted compromises in the networks and systems of DTAG requires a comprehensive logging of security relevant events on targeted systems and systems which are involved in such attacks.

Implementation example: Use SAP NetWeaver Administrator to configure log and trace files. Open SAP NetWeaver Administrator and choose: Troubleshooting -> Logs and Traces -> Log Configuration

Navigate to security audit log and set severity to 'Info'.

ID: 3.22-143/i373

2.9.5. Java-Stack: Transport Management System (TMS)

Req 144	When using AS Java, the secure Enhanced Change and Transport System (CTS+) must be used.

To be able to transport Non-ABAP objects in a controlled manner, the Enhanced Change and Transport System (CTS+) must be used.

Motivation: By using the Extended Change and Transport System (CTS+), cross-system synchronization of changes (e.g. portal changes) is possible, workflow scenarios can be flexibly configured, functional and technical approval steps can be planned and, among other things, audit-proof tracking and reporting of the CTS (Change and Transport System) can be used.

Implementation example: Enable CTS+:

Please see the SAPHelp:
http://help.sap.com/saphelp_nw70ehp2/helpdata/de/bb/6fab6036a146baa58e42fac032ab7b/frameset.htm

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: 3.22-144/i373

2.9.6. Java-Stack: ICM (Internet Communication Manager)

Req 145	Access to the SAP system via the ICM must be restricted by using an ACL filter, an authentication handler, and/or the HTTP rewrite handler.

Access on a SAP system must be secured with different procedures. Depending on the access ACL filter (ACL = Access Control List), authentication handler or HTTP rewrite handler can be used. The ICM offers different filter mechanisms. We recommend to use the easiest mechanism which can meet your security requirments. That means e. g. if an ACL is sufficient, use this, if it is the authentication handler than use this and if it is the rewrite handler use this. Please avoid to complex configurations, which can lead to a lack of security.

URLs should be filtered because otherwise information about the infrastructure and the configuration can be read (see also SAP Note 870127).

Filter Mechanism	ACL File	Authentication Handler	HTTP Rewrite Handler
Usage	Use ACL files to restrict access to specific client IP addresses or client IP address ranges when the restriction does not depend on the content of the HTTP request (i.e., not even the URL), and no HTTP error page is desired.	Use the Authentication Handler to set up URL filters (as a "white list" or "black list"). Rules in the Authentication Handler can also refer to specific client IP addresses, or to IP addresses of the server..	Use the HTTP rewrite handler for filters that cannot be mapped by ACL files or the Authentication Handler. The Rewrite Handler is a powerful tool for different filter mechanisms. It makes it possible to check numerous data of an HTTP request on the basis of a set of rules and to link them with each other.
Dynamic reloading of the configuration file	possible	possible	possible
White - respectively black list	• yes, both • also mixt possible	• yes, both • also mixt possible	• yes, both • also mixt possible

Filtering on URLs, handling of upper and lower case letters	no	• yes • standard configuration is case insensitive • can be configured	• yes • upper - and lower case sensitive can be configured per filter rule
Security Logging	yes	yes	no
Filtering on client IP addresses, including netmasks	yes	yes	yes
Parameter	**icm/server_port_<xx> option ACLFILE**	**icm/HTTP/auth_<xx> option PERMFILE**	**icm/HTTP/mod_<xx> option FILE**

Motivation: Prevent unauthorized web access on your backend system to avoid data and confidentiality loss.

Implementation example: Whatever filtering technique (ACL file, Authentication Handler and / or HTTP Rewrite Handler) you choose, please note the following:

- Evaluate the connection log content (see the written LOGFILE via the parameter`icm/HTTP/logging_xx`) , to determine the relevant URLs and describe them with exact rules in your URL permission table (White List).
- Be sure to filter / block the following paths in your URL permission table:
  - `/sap/public/icman/*`
    `/sap/public/icf_info/*`
    `/sap/wdisp/info`
- If you specify black lists (deny entries) in URL filters, use case-insensitive filters because AS ABAP treats URLs as case-insensitive.
- Instead of filtering the URLs, URLs can be deactivated if possible.

1) Set ACL files / filters for the system.
The option ACLFILE specifies the file that is used as access control list (ACL). With this an access control list can be set up, which connections the ICM accepts and which not. For each port of the ICM a separate ACL file can be created. The string obeys the following syntax:

```
icm/server_port_<xx> = PROT=<protocol name>, PORT=<port or service name>[, TIMEOUT=<timeout>, PROCTIMEOUT=<proctimeout>, EXTBIND=1, HOST=<host name>, VCLIENT=<SSL client verification>, SSLCONFIG=ssl_config_<xx>, ACLFILE=<path for ACL file>]

# Description of the access points
icm/server_port_0 = PROT=HTTP , PORT=0 , TIMEOUT=30 , PROCTIMEOUT=60, ACLFILE=<Pfad zur ACL-Datei>
icm/server_port_1 = PROT=HTTPS , PORT=1443, TIMEOUT=30 , PROCTIMEOUT=60, ACLFILE=<Pfad zur ACL-Datei>

A row within the ACL file must follow the syntax:

<permit | deny> <ip-address> [tracelevel] [# comment]
```

The rules are checked sequential from the beginning („top down“). The first fitting rule will be taken for the connection („first match“). When no rule is fitting the connection is refused. A deny rule as a last rule must be entered (deny 0.0.0.0/0).

Example content of an ACL file:
permit 10.1.2.0/24 # permit client network
...
deny 0.0.0.0/0 # deny the rest

2) Set Authentication Handler for the system.

To protect the backend system (AS ABAP or AS Java), there is an HTTP subhandler (filter) that can block requests based on various criteria. If the filter is activated, it is passed through for each HTTP(S) request to the ICM before the request is sent to another HTTP handler (file access, cache, administration, redirect) or to the backend system (AS ABAP or AS Java). You can filter requests by the following criteria: URL, client IP address, server IP address, user name/user group/user password, pattern search in the URL. To set up limitations using URL filter, use follwoing parameter with its syntax:

```
icm/HTTP/auth_<xx> = PREFIX=<URL-prefix>
[,PERMFILE=<permission file>, AUTHFILE=<authentication file>,
FILTER=<name>]
```

Example for setting the parameter:

```
i c m / H T T P / a u t h _ 0     =     P R E F I X = / ,     P E R M -
FILE=D:\usr\sap\WDP\SYS\profile\perm_filter.txt
```

With the sub option PERMFILE the kind of the access to your permission file is configured. Create a permission file and enter rules. The rules are applied in the order they appear in the file, from top to bottom, and the first rule that matches the request is applied. There is an implicit "deny all" rule that is added to the end of the file, at runtime. Therefore, it is recommended to use "positive rules" (create only P or S rules, as everything else will be denied by default).

The permission file has following syntax:

- Rows for comment begin with a # and will be ignored
- Other rows are built up like:P/D/S <URL-pattern> <USER> <GROUP> <CLIENT-IP> <SERVER-IP>
- The letter infront of the row has follwing meaning:
  - P (Permit)
  - D (Deny)
  - S (Secure) only secure connection are allowed (HTTPS)
  - <URI-pattern> ist der Abschnitt der URL, der im Abschnitt Cache-Key als translated path bezeichnet ist
- For the URL-Pattern you can use the wildcard *, but only at the beginning or the end of the <URL-pattern>.
- For the client - and server IP adess you can use an exact compare, the wildcard * or a net mask syntax.

Example of a white list with forcing HTTPS by setting the "S" at the beginning of the line:

```
# This is the "permission file" used by the ICM/SAP Web Dispatcher Au-
thentication handler
# webadmin and ping only from Admin-workstations <IP-Admin workstation> allowed
S /sap/bc/ping * * <IP-Admin workstation/net class> *
S /sap/wdisp/admin/* * * <IP-Admin workstation/net class> *
# Access from Webapps and WebGUI (https only) from internal IPs
S /sap/bc/webdynpro/* * * <internal IP/net class> *
S /sap/bc/gui/sap/its/webgui * * <internal IP/net class> *
# Access from Webapps and WebGUI (https only) from Internet
S /sap/bc/srt/rfc/meinwebservice * * * * *
# Everything not listed above will be denied because of the final implicit rule "D * * * * *"
```

3) Set HTTP rewrite handler for the system.

The HTTP rewrite handler is a very powerful tool. You can use it for various actions:

- Delete, add, and enhance HTTP header fields:

  You can delete or add HTTP header fields, or enhance them with additional values.
- Rewrite URLs:

  You can rewrite both the URL path and the query string of an incoming HTTP request into another previ-

ously-defined URL path or query string. You can link the execution of URL modifications to one or more conditions.

- Redirect or filter URLs:
  You can redirect the URL of an incoming HTTP request to a different application server or URL. You can also filter incoming HTTP requests by defined patterns.
- No modification action / nop action (no operation):
  Modifiers (options), such as break, restart, and noescape can occur after a modification action (rule). So that you can use modifiers without definitions too, the nop action is provided.

To set up access restrictions via HTTP rewrite handlers, use the parameter with the following syntax:
```
icm/HTTP/mod_<xx> = PREFIX=<URL-Prefix>[, FILE=<action file>]
```

SAP recommendation: To have the same filter rules on all instances, place the filter rules file in a global directory and set the parameter in the default profile DEFAULT.PFL. The filter rules are located in the icm_filter_rules.txt file in the $(DIR_GLOBAL)/security/data directory:
```
icm/HTTP/mod_0 = PREFIX=/,FILE=$(DIR_GLOBAL)/security/data/icm_filter_rules.txt
```

The activation of the rewrite handler is configured by an action file (use option FILE). The action file is a text file with one action (rule) per line. Comment lines start with a gate (#). The sequence within the action file describes the sequence of performing the single action. You can perform several URL changes in series. You can connect the execution of the HTTP headerfield changes with one or more conditions. No implicit "deny all" rule is added at the end, unlike when using the Authentication Handler. Notice also that a "negative list" has to be created, instead of a "positive list" like in the Authentication Handler.

Use the following syntax to define a rule in the action file (s. SAP Help):
<operation> <pattern> <dest> [<option>]

Enforce HTTPS with the rewrite handler:
To prevent the use of unencrypted HTTP, you have to change the protocol of the rewrite handler from HTTP to HTTPS. This will prevent error messages in the browser if users inadvertently access the system with an HTTP URL. Note that not all HTTP clients follow this redirect. While the redirect configuration ensures that no HTTP access to the system is possible, it is possible that individual users of the system, e.g. web service end points, must be switched from HTTP to HTTPS.
Example of strict use of HTTPS:
```
# We allow access to the WEB GUI, but only through HTTPS
# The indentation is optional
if %{SERVER_PROTOCOL} !stricmp "HTTPS" [and]
if %{PATH} regimatch ^/sap/bc/gui/sap/its/webgui(.*)
RegForbiddenURL ^/sap/bc/gui/sap/its/webgui(.*) - [break]
```

Example avoid execute a ping:
```
# We deny access to the "ping" service
RegForbiddenURL ^/sap/public/ping(.*) - [break]
```

ID: 3.22-145/i373

Req 146	The ICM server (port) must be configured securely.

The ICM must be configured so that only secure protocols have access. This is done by setting the parameter icm/server_port_<xx> to use the HTTPS protocol. With the PROT and PORT options you specify the protocol and the corresponding port by the port number or service name. Exactly only one service can be bound to a port.

Motivation: Unencrypted communication can be read which can lead to a threat for the system. User-IDs with the related password can be tapped and can be misused for an unauthorized login, which can lead to an unauthorized access on data.

Implementation example: By setting the parameter icm/server_port_<xx> the use of HTTPS is enforced and accesses can be controlled. This profile parameter of the ICM configures the server port. The string obeys the following syntax:

```
icm/server_port_<xx> = PROT=<protocol name>, PORT=<port or service>[,
TIMEOUT=<timeout>, PROCTIMEOUT=<proctimeout>, EXTBIND=1, HOST=<server
name>, VCLIENT=<SSL client verification>, SSLCONFIG=ssl_config_<xx>,
ACLFILE=<path to ACL-file>]
```

Description of access points (_0 till _3 are predefined values in AS Java for outgoing connections, _4 is an example for an incoming HTTPS connection, _5 is an example for an outgoing HTTPS connection):

```
icm/server_port_0 = PROT=HTTP , PORT=5$ (SAPSYSTEM) 00,
TIMEOUT=60 , PROCTIMEOUT=600
icm/server_port_1 = PROT=P4 , PORT=5$ (SAPSYSTEM) 04
icm/server_port_2 = PROT=IIOP , PORT=5$ (SAPSYSTEM) 07
icm/server_port_3 = PROT=TELNET , PORT=5$ (SAPSYSTEM) 08,
HOST=localhost
icm/server_port_2 = PROT=HTTPS , PORT=443, TIMEOUT=30 , PROC-
TIMEOUT=60, ACLFILE=<path to ACL-file>
icm/server_port_3 = PROT=HTTPS,PORT=0,TIMEOUT=30,PROCTIMEOUT=60,
ACLFILE=<path to ACL-file>
```

Please note that the value 0 for PORTS means that no port is opened for incoming connections for the specified protocol. Ports for incoming connections must be configured explicitly for security reasons.

Attention: SAP delivers predefined values with the ICM. These must be checked for validity and, if necessary, deactivated with " e.g.:

```
icm/server_port_0 =“
icm/server_port_1 =“
...
```

ID: 3.22-146/i373

Req 147	The administration of the ICM via the web-based administration interface must be secured.

You have to set the parameter icm/HTTP/admin_<xx> when you want to monitor and administrate the ICM via the web-based administration interface (browser).

Motivation: The web interface for the administration of the ICM must not be accessible for all users. This must be protected to prevent unauthorized access.

The display of system information must also be prevented, as this information could be misused to attack the system.

Implementation example: The parameter icm/HTTP/admin_<xx> has following syntax:

```
icm/HTTP/admin_<xx> = PREFIX=<URL-prefix>, DOCROOT=<root directory of
the administration files> [, AUTHFILE=<file name of the authorisation
file>, PORT=<TCP/IP-port on which admin requests are accepted>,
HOST=<server name or IP adress on which admin requests are accepted>,
CLIENTHOST=<client name or IP adress from which admin requests are ac-
cepted>, ALLOWPUB=<value> ]
```

Example for setting the parameters within the profile:

```
icm/HTTP/admin_0 = PRE-
FIX=/sap/admin,DOCROOT=$(DIR_ICMAN_ROOT/
admin,PORT=1443,HOST=localhost;<IP host
ICM>,CLIENTHOST=localhost;<hostname from Administrator clients>,
```

[ALLOWPUB=FALSE]

Setting the value for PREFIX: URL prefix for which this HTTP subhandler should be called. By default, the URL prefix for administration is set to /sap/admin.

Setting the value for DOCROOT: root directory of the administration files.

Setting the value for AUTHFILE: filename of the authentication file (default value: icmauth.txt) containing the hash values of the passwords for the admin users. If no user authentication is required (e.g. because it is already done in the Authorization Handler), this can be achieved by specifying AUTHFILE=none.

Setting the value for PORT: By default, admin requests are accepted on all local TCP/IP ports. This is not allowed. A restriction to a TCP/IP port that is only locally accessible and on which admin requests are accepted must be implemented. This way the use of HTTPS can be easily implemented.
! Note: This port must have been specified via the icm/server_port_<xx> parameter. An HTTPS port must be set up for this, otherwise the administrator passwords will be transmitted in clear text when logging in. Insecure protocols are not allowed!
Example:
`icm/server_port_<xx>=.... PROT=HTTPS, PORT=1443, ....`

Setting the values for HOST: If nothing is specified here, admin requests are accepted from all host names. This is not allowed. The restriction of access to the local host must be implemented, so that only the local users can use the web-based interface. For this, specify the value localhost or 127.0.0.1 and the <IP of the ICM host>.

Setting the values for CLIENTHOST: If nothing is specified here, admin requests are accepted from any client machine. This is not allowed. Administration must be restricted to specific client machines (machines of ICM administrators). To do this, enter the value localhost and the <machine names or IP addresses of the administrators>.

Setting the value for ALLOWPUB: If ALLOWPUB=TRUE is set, public and read accesses to certain administration pages are allowed under the path "public/index.html" without login (e.g. "Monitor", "Active Services", "Core Thread Status", "Hostname Buffer", "Release Information" and "MPI Status"). Access to these pages without logging in must be prevented. Even if access is restricted with the HOST and CLIENTHOST subparameters (see above), system information that could be used to attack the system must only be viewable by logging on to the system. ALLOWPUB=FALSE must be set.

ID: 3.22-147/i373

Req 148	The authorization file "icmauth.txt" must be secured for unauthorized access.

Within the authorization file (Default: icmauth.txt) all users with their group membership are listed in plain text. The passwords are encrypted within that file. The path is following the conventions of the operating system. For security reasons this file must be accessible by the administrations user of the SAP systems (<SID>adm) only.

Motivation: The file "icmauth.txt" must be protected against unauthorized access, to decrease the threat for an system attack. The encrypted password can be easily cracked and be used than for an unauthorized access.

Implementation example: Set restricted access rights on the authorization file icmauth.txt with following command (UNIX):

```
chmod 700 <authorization file>
```

The location of the file and its name are specified in the `icm/authfile` parameter.

ID: 3.22-148/i373

Req 149	The access to the ICM administration directory must be set restrictively.

The administration directory of the ICM must have restrictive permissions. Normaly the administration directory is defined by the same variable and directory is $DIR_ICMAN_ROOT/admin and contains the files for the administration by web interface.

Motivation: The administration directory contains data with a high protection requirement (e. g. file icmauth.txt).

Implementation example: Changing the access rights for the administration directory with the following commands:

```
chown <sid>adm:sapsys <directory name>
chmod 750 <file name> or if more restrictive is possible => 700
```

ID: 3.22-149/i373

Req 150	The web administration is personalized and not done by the user "icmadm".

To monitor and administer the ICM from a browser, you need a user.

The icmadm user is the default user for the web administration interface and is stored in the icmauth.txt file when the administration user is set up.

Personalized administrators are created for traceable administration.

Motivation: Unauthorized administration access can lead to the possibility to read and change the data and or of a system outage.

Implementation example: The following steps must be performed when setting up the administration users:

1. As <sid>adm, create the icmadm administration user with icmon. This generates the icmauth.txt authorization file.
2. Create personified administration users with icmon.
3. Try access with the personalized users.
4. Change the password of the default user icmadm according to the password guidelines.

ID: 3.22-150/i373

Req 151	The access on the command "icmbnd" for binding of ports must be set restrictive.

Usually the ICM binds the ports by itself (recommended). That the ICM can also bind the "well known ports", the ports from 0 to 1023, it must use the external command icmbnd. Due to setting the option EXTBIND=1 by the definition ot the parameter icm/server_port_<xx> the command icmbnd can be executed by the ICM for configuration the ports < 1023.
The command icmbnd can be executed directly (without the configuration using parameter icm/server_port_<xx>) (not recommended).

The path of the file icmbnd is defined in following parameters of the profile:

- exe/icmbnd
- DIR_EXECUTABL

Motivation: Unauthorized execution of the command icmbnd can lead to a misconfiguration. Thereby attacks over the ICM can be performed on downstream systems. This can lead to unauthorized access on data or to system outage.

Implementation example: Set access rights restrictive on the file icmbnd:

```
chown root:sapsys icmbnd

chmod 4750 icmbnd
```

ID: 3.22-151/i373

Req 152	Don't display error messages, genarted by default service.

When the service use default error messages, normally the service display Versions and Name.

Motivation: No information about the system may be output in "error pages" that could be exploited to attack the system.

Implementation example: The parameter /HTTP/show_detailed_errors must be set to "false":

```
is/HTTP/show_detailed_errors = FALSE
```

ID: 3.22-152/i373

Req 153	The information about the server in the HTTP response to the client must be limited.

The parameter controls whether or not the server header field should be inserted in HTTP responses from the server to the client. The server header field comprises the values of parameters is/server_name and is/server_version. For security reasons, SAP recommends not to pass this information to the client.

Motivation: No information may be passed to the client that could be exploited to launch an attack.

Implementation example: To do this, set the is/HTTP/show_server_header parameter as follows:

```
is/HTTP/show_server_header = FALSE
```

ID: 3.22-153/i373

Req 154	The security log for the ICM must be enabled.

You can use this parameter icm/security_log to control the output of the security log from the ICM and SAP Web Dispatcher.

Administrators can use the security log (icm/security_log) to help identify any potential unauthorized access to the system. The following irregularities are logged:

- Data with invalid syntax
- Attempted access to objects that do not exist (NOT found)
- Access to objects that is not permitted due to filter rules (permission denied)
- Logon errors to Web administration (in ICM and Web Dispatcher)

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.

Implementation example: Set the parameter icm/security_log to enable the security log.

Syntax of the parameter:

```
icm/security_log = LOGFILE=<file name>, LEVEL=<security level>,
MAXSIZEKB=<max size in KB>, SWITCHTF=<options>, FILEWRAP=on
```

LOGFILE: Name of the file. Time specifications make the name unique.
LEVEL: Specifies the level (1 -3) of logging. Level 1 only records the reason for the entry. Level 2 records additional information about the status of the connection and the start of the data that gave rise to the entry (this is the default value if option LEVEL is not specified). Level 3 logs all the data that gave rise to the entry.
MAXSIZEKB: Specifies the size of the log file. If this size is exceeded, the current file is closed and a new one (with a new name) is opened. By defining LOGFILE the name becomes unique and thus overwriting is prevented.
SWITCHTF: A new log file can not only be created if the file has reached a certain size, but also when the time data changes (every new hour, day or month possible).
FILEWRAP: If FILEWRAP=on is active, every time a new file is opened, the existing log file is reset and overwritten (!). Therefore, there is always only one log file with the current log data. If you omit this option, once the size has been exceeded a new file is written (see MAXSIZEKB and SWITCHTF).

Example setting the parameter (option FILEWRAP is not set!):

```
icm/security_log = LOGFILE=dev_icm_seclog-%d-%m-%y_%h:%t:%s, LEVEL=2,
MAXSIZEKB=10000, SWITCHTF=day
```

ID: 3.22-154/i373

Req 155	HTTP logging must be enabled.

You can use these parameters icm/HTTP/logging_<xx> (for incoming requests) and icm/HTTP/logging_client_<xx> (for outgoing requests) to control the output of the connection log from the ICM and SAP Web Dispatcher and log activities.

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.

Implementation example: Set the following two parameters:

1) Logging of incoming connections

Syntax of the parameter:

```
icm/HTTP/logging_<xx> = PREFIX=<URL prefix>, LOGFILE=<log file name>
[, LOGFORMAT=<format>, FILTER=<filter>, MAXSIZEKB=<size in KBytes>,
SWITCHTF=<options>, FILEWRAP=on]
```

PREFIX: URL prefix that is called for this HTTP subhandler (for example, /).
LOGFILE: Name of the output file in the file system. With each restart the ICM checks whether the file specified as the log file exists. If it does it continues to write to this file, if it does not, it creates a new file. You can attach another timestamp to the actual file name, which makes the file unique.
LOGFORMAT: There are the different formats for log files. You can reproduce the “NCSA Combined Log Format” known to other Web servers by using the following format string: %h %l %u %t "%r" %s %b "%{referer}i" "%{user-agent}i"
FILTER: The filter property specifies that an HTTP request is logged only when a certain header field (for example, an HTTP header field) is in the request or response.
MAXSIZEKB: Specifies the size of the log file. If this size is exceeded, the current file is closed and a new one (with a new name) is opened. By defining LOGFILE the name becomes unique and thus overwriting is prevented.

Example setting the parameter (option FILEWRAP is not set!):

```
icm/HTTP/logging_0 = PREFIX=/, LOG-
FILE=incoming_icm_log-%d-%m-%y_%h:%t:%s, LOGFORMAT=%h %l %u %t "%r" %s
%b "%{referer}i" "%{user-agent}i", MAXSIZEKB=10000, SWITCHTF=day
```

2) Logging of outgoing connections

Syntax of the parameter:

```
icm/HTTP/logging_client_<xx> = PREFIX=<URL prefix>, LOGFILE=<log file
name> [, LOGFORMAT=<format>, FILTER=<filter>, MAXSIZEKB=<size in
KBytes>, SWITCHTF=<options>, FILEWRAP=on]
```

The options are the same as for icm/HTTP/logging_<xx>.

Example setting the parameter (option FILEWRAP is not set!):

```
icm/HTTP/logging_client_<xx> = PREFIX=/, LOG-
FILE=outgoing_icm_log-%d-%m-%y_%h:%t:%s, LOGFORMAT=%h %l %u %t "%r" %s
%b "%{referer}i" "%{user-agent}i", MAXSIZEKB=10000, SWITCHTF=day
```

Attention for icm/HTTP/logging_<xx> and icm/HTTP/logging_client_<xx>: The log files for incoming and outgoing requests must have different names, that is, you cannot not write both directions to one file.

ID: 3.22-155/i373

2.10. SAProuter

Req 156	The SAP-Router must be install on harden operations system image.

The SAProuter is a application with the task to protect the SAP system. Usually SAProuter build up in "DMZ".

Motivation: The SAProuter is comparable with a Proxy, but it's only allowed to use it for the SAP protocol family (DIAG, RFC, SNC). It protects SAP systems like a proxyserver the webserver.

ID: 3.22-156/i373

Req 157	The SAProuter software must be the only service running on the host or VM.

The SAProuter is similar as a firewall or proxy server. Usually the SAProuter is build up in network areas with internet traffic. When the SAProuter system is corrupt it is easy to handle a single service.

Motivation: With single services in the DMZ, the risk for other applications is decreased.

ID: 3.22-157/i373

Req 158	It is not allowed, that the SAProuter application is in the same subnet as the protected system.

The application must be an IP-Address-Range different from the SAP-Router, because the SAProuter can only protect the SAP system, if there is not workaround to bypass the SAProuter.

Motivation: The SAProuter can only protect the SAP system, if there not workaround to bypass the SAProuter.

Implementation example: SAP application = 192.168.100.1
SAProuter = 10.10.10.10

ID: 3.22-158/i373

Req 159	In the operation system from SAProuter application ,the function "IP-Forward" must be disabled.

In the operation system from SAP-Router-Application ,the function "IP-Forward" must be disabled

Motivation: By activation IP-Forward function, the SAP-Router is bypassing.

ID: 3.22-159/i373

Req 160	Access to the folder, subfolders and files of the SAProuter must be set up restricted.

The SAProuters must be installed and executed with its own user and group and SAProuter administration on OS level is traceable personalized by mechanisms, authorizations and logs

Motivation: Protect rules to configure the SAProuter.

Implementation example: For syntax use the german description.

ID: 3.22-160/i373

Req 161	Neither the SAProuter user nor the group are allowed having root privileges.

Neither the SAProuter user nor the group are allowed having root privileges.

Motivation: Protect the operation system and other applications, that after vulnerability the SAP-Router-Service runs not as administrator.

ID: 3.22-161/i373

Req 162	If SNC is used, the SNC entries in the saprouttab file must be in the first lines.

The entries in the saprouttab file are read and evaluated from top to bottom (First Match). The first line that fits is used as the rule for the connection.

Motivation: Dragging up the SNC rules ensures that SNC rules are used before unencrypted connections.

Implementation example: Content saprouttab:

```
# Initiating SNC for all connections to host2 :
 KT = "p:CN=saprout2, OU=TEST01, O=myCompany, C=US"   host2   *
# Accepting all connections

 D      *      *      *
```

https://help.sap.com/doc/saphelp_nw73ehp1/7.31.19/en-us/65/8d09ab5c7e46028f633bb01a09b380/content.htm?no_cache=true
https://help.sap.com/doc/saphelp_nw75/7.5.5/en-US/48/6c7a3fc1504e6ce10000000a421937/content.htm?no_cache=true

ID: 3.22-162/i373

Req 163	The last row of saprouttab-file must be deny all connections.

To ensure that no unauthorized connection is true, the last row must be a "deny" all role.

Motivation: To ensure that no unauthorized connection is true, the last row must be a "deny all" roleSAP note 1895350

Implementation example: Last rows within the saprouttab:

```
KD    *    *    *    *


D     *    *    *    *
```

"KD" prohibit all SNC connections and "D" all unencrypted connections (for SAP protocols RFC and DIAG).

ID: 3.22-163/i373

Req 164	Maintain only necessary connections in the saprouttab table.

In the configuration file for SAProuter must be only connections,they are necessary.

Motivation: Only connections that are required for using the application, should be permitted in order to keep the protection for the application as high as possible.

Implementation example: Check if only the necessary connections are maintained within the saprouttab (s. SAP note 1895350).

ID: 3.22-164/i373

Req 165	In the route permission table (saprouttab) it isn't allowed to use wildcards (*).

In the configuration file for SAP-Router (saprouttab, rule set for the access of the SAProuter), it is not allowed to use wildcards in the "P" / "KP" and "S" / "KS" rows for the destination host (<dest-host>) and the destination port (<dest-serv>). The use of IP address ranges for the destination host and or port ranges for the destination service not allowed either.

Motivation: By narrowing down to the destination hosts and / or services unauthorized access to other hosts and other services is prevented.

Implementation example: Example for an entry wihin the saprouttab:

P/S <IP-adress/hostname> <complete IP-adress/hostname> 3299 <password>

ID: 3.22-165/i373

Req 166	SAProuter must only be used by protocol DIAG, RFC and SNC.

Only the protocols DIAG, RFC and SNC are allowed to pass the SAProuter. An exception is in case of the SAP support, which uses different services like https.

Motivation: Different protocols like https must be qualified e. b. by the SAP Web Dispatcher. It is not allowed to bypass these protocols using the SAProuter. The SAProuter only pass DIAG and RFC protocols.

Implementation example: By Usage of the des "S" (in case of SNC "KS") instead of "P" (in case of SNC "KP") for all positive entries only the SAP services are allowed for a connection:

KS <host 1> <host2> <Port> <password>

S <host 3> <host4> <Port> <password>

ID: 3.22-166/i373

Req 167	Every SAProuter must have a separate route permission table (file name: saprouttab) assigned.

The default path looks like ./saprouttab by starting in the SAProuter folder, where the "saprouter" file is located.

Motivation: The SAP-Router is not working fine, with a configuration-file

Implementation example: Start SAProuter with option -R <path>.

```
<path to SAProuter file>SAProuter -r -R <path to SAProuter file>/
saprouttab
```

ID: 3.22-167/i373

Req 168	It must be prevented that the SAProuter does establish any connections to itself (loopback).

In the default configuration, the SAProuter does not allow a route to itself. The loopback from the SAProuter to itself can be explicitly allowed with the -X option. This option can compromise the security of your system, as potential attackers could administer the SAProuter from an external client (see SAP Note 1853140).

Motivation: If loopback connections are allowed, changes to the SAProuter configuration can be made through unauthorized access.

Implementation example: Run SAP-Router binary WITHOUT the option -X

```
<path to SAProuter files>/SAProuter -r -R <Path to SAProuter file>/
saprouttab
```

ID: 3.22-168/i373

Req 169	The Logging must be turned on for the SAProuter (SAP note 1895350).

The logging content for option -G is showing the connections. Option -T only provides information about connection failures.

Set option -J to restrict the size of the log file to avoid big files. If you do not use option -J, the log file can become as large as is necessary. If you use option -J, once the log file reaches the defined size, it is renamed to <log file name>_a_<start date>_<start time>-<end date>_<end time>.

Additionally Option -E should be set, when options -G und -T are used. Option -E avoids an overwrite of the old log files when the SAProuter must be restarted.

Motivation: About to set the function Log on, it is possible to view the traffic about the SAP-Router.

Implementation example: For logging the SAProuter connections option -G <path> and -T <path> must be set. Additionally option -J must be set to avoid to big files. For this a suitable value must be set.

```
<path>SAProuter -r -G <path> -T <path> -J <size in bytes> -E
```

ID: 3.22-169/i373

Req 170	SAProuter must be start as service.

To ensure that the SAProuter is running after a reboot, it must be started by a script. Only with that, it is enshured that the environment is always working continous.

Motivation: To ensure that the SAProuter is running after a reboot, it must be started by a script. Only with that, it is enshured that the environment is always working continous.

Implementation example: SAProuter must be started automatically when OS is starting:

```
<path>SAProuter -r
```

When it is started without option -S, the default port 3299 is used.

ID: 3.22-170/i373

2.11. Specifications SAP Web Dispatcher

Req 171	Access to the SAP system via the SAP Web Dispatcher must be restricted by using an ACL filter, an authentication handler, and/or the HTTP rewrite handler.

Access on a SAP system must be secured with different procedures. Depending on the access ACL filter (ACL = Access Control List), authentication handler or HTTP rewrite can be used. The SAP Web Dispatcher offers differen filter mechanisms. We recommend to use the easiest mechanism which can meet your security requirments. That means e. g. if an ACL is sufficient, use this, if it is the authentication handler than use this and if it is the rewrite handler use this. Please avoid to complex configurations, which can lead to a lack of security.

URLs should be filtered because otherwise information about the infrastructure and the configuration can be read (see also SAP Note 870127).

Filter Mechanism	ACL File	Authentication Handler	HTTP Rewrite Handler

Usage	Use ACL files to restrict access to specific client IP addresses or client IP address ranges when the restriction does not depend on the content of the HTTP request (i.e., not even the URL), and no HTTP error page is desired.	Use the Authentication Handler to set up URL filters (as a "white list" or "black list"). Rules in the Authentication Handler can also refer to specific client IP addresses, or to IP addresses of the server..	Use the HTTP rewrite handler for filters that cannot be mapped by ACL files or the Authentication Handler. The Rewrite Handler is a powerful tool for different filter mechanisms. It makes it possible to check numerous data of an HTTP request on the basis of a set of rules and to link them with each other.
Dynamic reloading of the configuration file	possible	possible	possible
White - respectively black list	• yes, both • also mixt possible	• yes, both • also mixt possible	• yes, both • also mixt possible
Filtering on URLs, handling of upper and lower case letters	no	• yes • standard configuration is case insensitive • can be configured	• yes • upper - and lower case sensitive can be configured per filter rule
Security Logging	yes	yes	no
Filtering on client IP addresses, including netmasks	yes	yes	yes
Parameter	icm/server_port_<xx> option ACLFILE	icm/HTTP/auth_<xx> option PERMFILE	icm/HTTP/mod_<xx> option FILE

Motivation: Prevent unauthorized web access on your backend system to avoid data and confidentiality loss.

Implementation example: Whatever filtering technique (ACL file, Authentication Handler and / or HTTP Rewrite Handler) you choose, please note the following:

- Evaluate the connection log content (see the written LOGFILE via the parameter`icm/HTTP/logging_xx`) , to determine the relevant URLs and describe them with exact rules in your URL permission table (White List).
- Be sure to filter / block the following paths in your URL permission table:
  - /sap/public/icman/*
  - /sap/public/icf_info/*
  - /sap/wdisp/info
- If you specify black lists (deny entries) in URL filters, use case-insensitive filters because AS ABAP treats URLs as case-insensitive.
- Instead of filtering the URLs, URLs can be deactivated if possible.

1) Set ACL files / filters for the system.

The option ACLFILE specifies the file that is used as access control list (ACL). With this an access control list can be set up, which connections the SAP Web Dispatcher accepts and which not. For each port of the SAP Web Dispatcher a separate ACL file can be created. The string obeys the following syntax:

```
icm/server_port_<xx> = PROT=<protocol name>, PORT=<port or service
name>[, TIMEOUT=<timeout>, PROCTIMEOUT=<proctimeout>, EXTBIND=1,
HOST=<host name>, VCLIENT=<SSL client verification>, SSLCON-
FIG=ssl_config_<xx>, ACLFILE=<path for ACL file>]

# Description of the access points
icm/server_port_0 = PROT=HTTP , PORT=0 , TIMEOUT=30 , PROCTIMEOUT=60,
ACLFILE=<Pfad zur ACL-Datei>
icm/server_port_1 = PROT=HTTPS , PORT=1443, TIMEOUT=30 , PROC-
TIMEOUT=60, ACLFILE=<Pfad zur ACL-Datei>
```

A row within the ACL file must follow the syntax:

```
<permit | deny> <ip-address> [tracelevel] [# comment]
```

The rules are checked sequential from the beginning („top down“). The first fitting rule will be taken for the connection („first match“). When no rule is fitting the connection is refused. A deny rule as a last rule must be entered (deny 0.0.0.0/0).

Example content of an ACL file:
permit 10.1.2.0/24 # permit client network
...
deny 0.0.0.0/0 # deny the rest

2) Set Authentication Handler for the system.

To protect the backend system (AS ABAP or AS Java), there is an HTTP subhandler (filter) that can block requests based on various criteria. If the filter is activated, it is passed through for each HTTP(S) request to the SAP Web Dispatcher before the request is sent to another HTTP handler (file access, cache, administration, redirect) or to the backend system (AS ABAP or AS Java). You can filter requests by the following criteria: URL, client IP address, server IP address, user name/user group/user password, pattern search in the URL. To set up limitations using URL filter, use follwoing parameter with its syntax:

```
icm/HTTP/auth_<xx> = PREFIX=<URL-prefix>
[,PERMFILE=<permission file>, AUTHFILE=<authentication file>,
FILTER=<name>]
```

Example for setting the parameter:

```
icm/HTTP/auth_0 = PREFIX=/, PERM-
FILE=D:\usr\sap\WDP\SYS\profile\perm_filter.txt
```

With the sub option PERMFILE the kind of the access to your permission file is configured. Create a permission file and enter rules. The rules are applied in the order they appear in the file, from top to bottom, and the first rule that matches the request is applied. There is an implicit "deny all" rule that is added to the end of the file, at runtime. Therefore, it is recommended to use "positive rules" (create only P or S rules, as everything else will be denied by default).

The permission file has following syntax:

- Rows for comment begin with a # and will be ignored
- Other rows are built up like:P/D/S <URL-pattern> <USER> <GROUP> <CLIENT-IP> <SERVER-IP>
- The letter infront of the row has follwing meaning:
  - P (Permit)
  - D (Deny)
  - S (Secure) only secure connection are allowed (HTTPS)
  - <URI-pattern> ist der Abschnitt der URL, der im Abschnitt Cache-Key als translated path bezeichnet ist
- For the URL-Pattern you can use the wildcard *, but only at the beginning or the end of the <URL-pattern>.

- For the client - and server IP adess you can use an exact compare, the wildcard * or a net mask syntax.

Example of a white list with forcing HTTPS by setting the "S" at the beginning of the line:

```
# This is the "permission file" used by the ICM/SAP Web Dispatcher Au-
thentication handler
# webadmin and ping only from Admin-workstations <IP-Admin worksta-
tion> allowed
S /sap/bc/ping * * <IP-Admin workstation/net class> *
S /sap/wdisp/admin/* * * <IP-Admin workstation/net class> *
# Access from Webapps and WebGUI (https only) from internal IPs
S /sap/bc/webdynpro/* * * <internal IP/net class> *
S /sap/bc/gui/sap/its/webgui * * <internal IP/net class> *
# Access from Webapps and WebGUI (https only) from Internet
S /sap/bc/srt/rfc/meinwebservice * * * * *
# Everything not listed above will be denied because of the final im-
plicit rule "D * * * * *"
```

3) Set HTTP rewrite handler for the system.

The HTTP rewrite handler is a very powerful tool. You can use it for various actions:

- Delete, add, and enhance HTTP header fields:
  You can delete or add HTTP header fields, or enhance them with additional values.
- Rewrite URLs:
  You can rewrite both the URL path and the query string of an incoming HTTP request into another previously-defined URL path or query string. You can link the execution of URL modifications to one or more conditions.
- Redirect or filter URLs:
  You can redirect the URL of an incoming HTTP request to a different application server or URL. You can also filter incoming HTTP requests by defined patterns.
- No modification action / nop action (no operation):
  Modifiers (options), such as break, restart, and noescape can occur after a modification action (rule). So that you can use modifiers without definitions too, the nop action is provided.

To set up access restrictions via HTTP rewrite handlers, use the parameter with the following syntax:

```
icm/HTTP/mod_<xx> = PREFIX=<URL-Prefix>[, FILE=<action file>]
```

SAP recommendation: To have the same filter rules on all instances, place the filter rules file in a global directory and set the parameter in the default profile DEFAULT.PFL. The filter rules are located in the icm_filter_rules.txt file in the $(DIR_GLOBAL)/security/data directory:

```
i c m / H T T P / m o d _ 0 = P R E -
FIX=/,FILE=$(DIR_GLOBAL)/security/data/icm_filter_rules.txt
```

The activation of the rewrite handler is configured by an action file (use option FILE). The action file is a text file with one action (rule) per line. Comment lines start with a gate (#). The sequence within the action file describes the sequence of performing the single action. You can perform several URL changes in series. You can connect the execution of the HTTP headerfield changes with one or more conditions. No implicit "deny all" rule is added at the end, unlike when using the Authentication Handler. Notice also that a "negative list" has to be created, instead of a "positive list" like in the Authentication Handler.

Use the following syntax to define a rule in the action file (s. SAP Help):

```
<operation> <pattern> <dest> [<option>]
```

Enforce HTTPS with the rewrite handler:
To prevent the use of unencrypted HTTP, you have to change the protocol of the rewrite handler from HTTP to HTTPS. This will prevent error messages in the browser if users inadvertently access the system with an HTTP URL. Note that not all HTTP clients follow this redirect. While the redirect configuration ensures that no HTTP access to the system is possible, it is possible that individual users of the system, e.g. web service end points, must be switched from HTTP to HTTPS.

Example of strict use of HTTPS:

```
# We allow access to the WEB GUI, but only through HTTPS
# The indentation is optional
if %{SERVER_PROTOCOL} !stricmp "HTTPS" [and]
if %{PATH} regimatch ^/sap/bc/gui/sap/its/webgui(.*)
RegForbiddenURL ^/sap/bc/gui/sap/its/webgui(.*) - [break]
```

Example avoid execute a ping:

```
# We deny access to the "ping" service
RegForbiddenURL ^/sap/public/ping(.*) - [break]
```

ID: 3.22-171/i373

Req 172	Between the SAP Web dispatcher and the backend/application must be set up a secure protocol (HTTPS)

The parameter wdisp/ssl_encrypt determines, how the SAP Web Dispatcher handles inbound HTTP requests. The following values are permitted:

0: Forward the request unencrypted.

1: Encrypt the request again with SSL, in case the request arrived via HTTPS protocol.

2: Always forward the request encrypted with SSL. You can also configure the SAP Web Dispatcher for End-to-End SSL, by specifying the protocol ROUTER when you define the icm/server_port_<xx> parameter.

Motivation: the Webdispacher control the connections between client and application in relation to the Internet-Protocol HTTP(S).

Implementation example: Set profilparemeter wdisp/ssl_encrypt = 2

ID: 3.22-172/i373

Req 173	The SAP Web Dispatcher server (port) must be configured securely.

The SAP Web Dispatcher must be configured so that only secure protocols have access. This is done by setting the parameter icm/server_port_<xx> to use the HTTPS protocol. With the PROT and PORT options you specify the protocol and the corresponding port by the port number or service name. Exactly only one service can be bound to a port.

Motivation: Unencrypted communication can be read which can lead to a threat for the system. User-IDs with the related password can be tapped and can be misused for an unauthorized login, which can lead to an unauthorized access on data.

Implementation example: By setting the parameter icm/server_port_<xx> the use of HTTPS is enforced and accesses can be controlled. This profile parameter of the SAP Web Dispatcher configures the server port. The string obeys the following syntax:

```
icm/server_port_<xx> = PROT=<protocol name>, PORT=<port or service>[,
TIMEOUT=<timeout>, PROCTIMEOUT=<proctimeout>, EXTBIND=1, HOST=<server
name>, VCLIENT=<SSL client verification>, SSLCONFIG=ssl_config_<xx>,
ACLFILE=<path to ACL-file>]
```

Description of access points (_0 and _1 are predefined values in AS ABAP for outgoing connections, _2 is an example for an incoming HTTPS connection, _3 is an example for an outgoing HTTPS connection):

```
icm/server_port_0 = PROT=HTTP,PORT=0,TIMEOUT=30,PROCTIMEOUT=60
icm/server_port_1 = PROT=SMTP,PORT=0,TIMEOUT=120,PROCTIMEOUT=120
icm/server_port_2 = PROT=HTTPS , PORT=443, TIMEOUT=30 , PROC-
TIMEOUT=60, ACLFILE=<path to ACL-file>
```

```
icm/server_port_3 = PROT=HTTPS,PORT=0,TIMEOUT=30,PROCTIMEOUT=60,
ACLFILE=<path to ACL-file>
```

Please note that the value 0 for PORTS means that no port is opened for incoming connections for the specified protocol. Ports for incoming connections must be configured explicitly for security reasons.
Attention: SAP delivers predefined values with the SAP Web Dispatcher. These must be checked for validity and, if necessary, deactivated with " e.g.:

```
icm/server_port_0 =“
icm/server_port_1 =“
```

ID: 3.22-173/i373

Req 174	The administration of the SAP Web Dispatcher via the web-based administration interface must be secured.

You have to set the parameter icm/HTTP/admin_<xx> when you want to monitor and administrate the SAP Web Dispatcher via the web-based administration interface (browser).

Motivation: The web interface for the administration of the SAP Web Dispatcher must not be accessible for all users. This must be protected to prevent unauthorized access.

The display of system information must also be prevented, as this information could be misused to attack the system.

Implementation example: The parameter icm/HTTP/admin_<xx> has following syntax:
`icm/HTTP/admin_<xx> = PREFIX=<URL-prefix>, DOCROOT=<root directory of the administration files> [, AUTHFILE=<file name of the authorisation file>, PORT=<TCP/IP-port on which admin requests are accepted>, HOST=<server name or IP adress on which admin requests are accepted>, CLIENTHOST=<client name or IP adress from which admin requests are accepted>, ALLOWPUB=<value>]`

Example for setting the parameters within the profile:

`icm/HTTP/admin_0 = PREFIX=/sap/admin,DOCROOT=$(DIR_ICMAN_ROOT)/admin,PORT=1443,HOST=localhost;<IP SAP Web Dispatcher>,CLIENTHOST=localhost;<hostname from Administrator clients>, [ALLOWPUB=FALSE]`

Setting the value for PREFIX: URL prefix for which this HTTP subhandler should be called. By default, the URL prefix for administration is set to /sap/admin.

Setting the value for DOCROOT: root directory of the administration files.

Setting the value for AUTHFILE: filename of the authentication file (default value: icmauth.txt) containing the hash values of the passwords for the admin users. If no user authentication is required (e.g. because it is already done in the Authorization Handler), this can be achieved by specifying AUTHFILE=none.

Setting the value for PORT (MUST): By default, admin requests are accepted on all local TCP/IP ports. This is not allowed. A restriction to a TCP/IP port that is only locally accessible and on which admin requests are accepted must be implemented. This way the use of HTTPS can be easily implemented.
! Note: This port must have been specified via the icm/server_port_<xx> parameter. An HTTPS port must be set up for this, otherwise the administrator passwords will be transmitted in clear text when logging in. Insecure protocols are not allowed!
Example:
`icm/server_port_<xx>=.... PROT=HTTPS, PORT=1443, ....`

Setting the values for HOST (MUST): If nothing is specified here, admin requests are accepted from all host names. This is not allowed. The restriction of access to the local host must be implemented, so that only the local users can use the web-based interface. For this, specify the value localhost or 127.0.0.1 and the <IP of the SAP Web Dispatch-

er>.

Setting the values for CLIENTHOST (optional): If nothing is specified here, admin requests are accepted from any client machine. Administration must be restricted to specific client machines (machines of SAP Web Dispatcher administrators). To do this, enter the value localhost and the <machine names or IP addresses of the administrators>.

Setting the value for ALLOWPUB: If ALLOWPUB=TRUE is set, public and read accesses to certain administration pages are allowed under the path "public/index.html" without login (e.g. "Monitor", "Active Services", "Core Thread Status", "Hostname Buffer", "Release Information" and "MPI Status"). Access to these pages without logging in must be prevented. Even if access is restricted with the HOST and CLIENTHOST subparameters (see above), system information that could be used to attack the system must only be viewable by logging on to the system. ALLOWPUB=FALSE must be set.

ID: 3.22-174/i373

Req 175	The authorization file "icmauth.txt" must be secured for unauthorized access.

Within the authorization file (Default: icmauth.txt) all users with their group membership are listed in plain text. The passwords are encrypted within that file. The path is following the conventions of the operating system. For security reasons this file must be access able by the administrations user of the SAP systems (<sid>adm) only.

Motivation: The file "icmauth.txt" must be protected against unauthorized access, to decrease the threat for an system attack. The encrypted password can be easily cracked and be used than for an unauthorized access.

Implementation example: Set restricted access righs on the authorization file icmauth.txt with following command (UNIX):

```
chmod 700 <authorization file>
```

The location of the file and its name are specified in the `icm/authfile` parameter.

ID: 3.22-175/i373

Req 176	The access to the WebDispatcher administration directory must be set restrictively.

The administration directory of the ICM must have restrictive permissions. Normaly the administration directory is defined by the same variable and directory is $DIR_ICMAN_ROOT/admin and contains the files for the administration by web interface.

Motivation: The administration directory contains data with a high protection requirement (e. g. file icmauth.txt).

Implementation example: Changing the access rights for the administration directory with the following commands:

```
chown <sid>adm:sapsys <directory name>
chmod 750 <file name> or if more restrictive is possible => 700
```

ID: 3.22-176/i373

Req 177	The web administration is personalized and not done by the user "icmadm".

To monitor and administer the ICM from a browser, you need a user.

The icmadm user is the default user for the web administration interface and is stored in the icmauth.txt file when the administration user is set up.

Personalized administrators are created for traceable administration.

Motivation: Unauthorized administration access can lead to the possibility to read and change the data and or of a system outage.

Implementation example: The following steps must be performed when setting up the administration users:

1. As <sid>adm, create the icmadm administration user with wdispmon oder sapwebdisp -bootstrap. This generates the icmauth.txt authorization file.
2. Create personified administration users with icmon.
3. Try access with the personalized users.
4. Change the password of the default user icmadm according to the password guidelines.

ID: 3.22-177/i373

Req 178	The access on the command icmbnd for binding of ports must be set restrictive.

The access on the command icmbnd for binding of ports must be set restrictive.Usually the SAP Web Dispatcher binds the ports by itself (recommended). That the SAP Web Dispatcher can also bind the "well known ports", the ports from 0 to 1023, it must use the external command icmbnd. Due to setting the option EXTBIND=1 by the definition ot the parameter icm/server_port_<xx> the command icmbnd can be executed by the SAP Web Dispatcher for configuration the ports < 1023.

The command icmbnd can be executed directly (without the configuration using parameter icm/server_port_<xx>) (not recommended).

The path of the file icmbnd is defined in following parameters of the profile:

- exe/icmbnd
- DIR_EXECUTABL

Motivation: Unauthorized execution of the command icmbnd can lead to a misconfiguration. Thereby attacks over the SAP Web Dispatcher can be performed on downstream systems. This can lead to unauthorized access on data or to system outage.

Implementation example: Set access rights restrictive on the file icmbnd:

```
chown root:sapsys icmbnd

chmod 4750 icmbnd
```

ID: 3.22-178/i373

Req 179	HTTP-Standard error messages generated by the server must be not visible by the client.

For security reasons the details should not passed to the client. In HTTP-Standard error messages normely the service give Infomation about the error (serverversion, softwareversion, time,...).

Motivation: If the Service display error, version, date, patch, it is esay to find vulnerabilitys or exploids

Implementation example: Set is/HTTP/show_detailed_errors parameter to false. Check the SAP Web Dispatcher profiles:

```
is/HTTP/show_detailed_errors = false
```

This parameter determines the standard form of HTTP error messages that the server (ICM or SAP Web dispatcher) creates and sends to the client.

ID: 3.22-179/i373

Req 180	The information about the server in the HTTP response to the client must be limited.

The parameter controls whether or not the server header field should be inserted in HTTP responses from the server to the client. The server header field comprises the values of parameters is/server_name and is/server_version. For security reasons, SAP recommends not to pass this information to the client.

Motivation: No information may be passed to the client that could be exploited to launch an attack.

Implementation example: To do this, set the is/HTTP/show_server_header parameter as follows:
`is/HTTP/show_server_header = FALSE`

ID: 3.22-180/i373

Req 181	The security log for the SAP Web Dispatcher must be enabled.

You can use this parameter icm/security_log to control the output of the security log from the ICM and SAP Web Dispatcher.

Administrators can use the security log (icm/security_log) to help identify any potential unauthorized access to the system. The following irregularities are logged:

- Data with invalid syntax
- Attempted access to objects that do not exist (NOT found)
- Access to objects that is not permitted due to filter rules (permission denied)
- Logon errors to Web administration (in ICM and Web Dispatcher)

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.

Implementation example: Set the parameter icm/security_log to enable the security log.

Syntax of the parameter:
`icm/security_log = LOGFILE=<file name>, LEVEL=<security level>, MAXSIZEKB=<max size in KB>,` SWITCHTF=<options>, FILEWRAP=on

LOGFILE: Name of the file. Time specifications make the name unique.
LEVEL: Specifies the level (1 -3) of logging. Level 1 only records the reason for the entry. Level 2 records additional information about the status of the connection and the start of the data that gave rise to the entry (this is the default value if option LEVEL is not specified). Level 3 logs all the data that gave rise to the entry.
MAXSIZEKB: Specifies the size of the log file. If this size is exceeded, the current file is closed and a new one (with a new name) is opened. By defining LOGFILE the name becomes unique and thus overwriting is prevented.
SWITCHTF: A new log file can not only be created if the file has reached a certain size, but also when the time data changes (every new hour, day or month possible).
FILEWRAP: If FILEWRAP=on is active, every time a new file is opened, the existing log file is reset and overwritten (!). Therefore, there is always only one log file with the current log data. If you omit this option, once the size has been exceeded a new file is written (see MAXSIZEKB and SWITCHTF).

Example setting the parameter (option FILEWRAP is not set!):
`icm/security_log = LOGFILE=dev_icm_seclog-%d-%m-%y_%h:%t:%s, LEVEL=2, MAXSIZEKB=10000, SWITCHTF=day`

ID: 3.22-181/i373

Req 182	HTTP logging must be enabled.

You can use these parameters icm/HTTP/logging_<xx> (for incoming requests) and icm/HTTP/logging_client_<xx>

(for outgoing requests) to control the output of the connection log from the ICM and SAP Web Dispatcher and log activities.

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.

Implementation example: Set the following two parameters:

1) Logging of incoming connections
Syntax of the parameter:
```
icm/HTTP/logging_<xx> = PREFIX=<URL prefix>, LOGFILE=<log file name>
[, LOGFORMAT=<format>, FILTER=<filter>, MAXSIZEKB=<size in KBytes>,
SWITCHTF=<options>, FILEWRAP=on]
```

PREFIX: URL prefix that is called for this HTTP subhandler (for example, /).
LOGFILE: Name of the output file in the file system. With each restart the ICM checks whether the file specified as the log file exists. If it does it continues to write to this file, if it does not, it creates a new file. You can attach another timestamp to the actual file name, which makes the file unique.
LOGFORMAT: There are the different formats for log files. You can reproduce the “NCSA Combined Log Format” known to other Web servers by using the following format string: %h %l %u %t "%r" %s %b "%{referer}i" "%{user-agent}i"
FILTER: The filter property specifies that an HTTP request is logged only when a certain header field (for example, an HTTP header field) is in the request or response.
MAXSIZEKB: Specifies the size of the log file. If this size is exceeded, the current file is closed and a new one (with a new name) is opened. By defining LOGFILE the name becomes unique and thus overwriting is prevented.

Example setting the parameter (option FILEWRAP is **not** set!):
```
icm/HTTP/logging_0 = PREFIX=/, LOG-
FILE=incoming_icm_log-%d-%m-%y_%h:%t:%s, LOGFORMAT=%h %l %u %t "%r" %s
%b "%{referer}i" "%{user-agent}i", MAXSIZEKB=10000, SWITCHTF=day
```

2) Logging of outgoing connections
Syntax of the parameter:
```
icm/HTTP/logging_client_<xx> = PREFIX=<URL prefix>, LOGFILE=<log file
name> [, LOGFORMAT=<format>, FILTER=<filter>, MAXSIZEKB=<size in
KBytes>, SWITCHTF=<options>, FILEWRAP=on]
```

The options are the same as for icm/HTTP/logging_<xx>.

Example setting the parameter (option FILEWRAP is **not** set!):
```
icm/HTTP/logging_client_<xx> = PREFIX=/, LOG-
FILE=outgoing_icm_log-%d-%m-%y_%h:%t:%s, LOGFORMAT=%h %l %u %t "%r" %s
%b "%{referer}i" "%{user-agent}i", MAXSIZEKB=10000, SWITCHTF=day
```

Attention for icm/HTTP/logging_<xx> and icm/HTTP/logging_client_<xx>: The log files for incoming and outgoing requests must have different names, that is, you cannot not write both directions to one file.

ID: 3.22-182/i373

2.12. Databases

Req 183	The access for SAP BR*Tools must be restricted (Oracle Databases).

SAP provides the tools BRBACKUP, BRARCHIVE, BRRESTORE, BRRECOVER, BRSPACE, BRCONNECT, and BRTOOLS to manage and protect the data in your Oracle database. For more information, see SAP Notes 8523, 113747 and SAP Help documentation.

Motivation: Having unauthorized access to the files the file content could be changed. Changes on the file content could lead to system failures and/or to an outage.

Implementation example: The access to the br*tools must be set restricted:

Tool name	Directory	Authorization	User/Group	Function
BRBACKUP	/sapmnt/<SID>/exe	4775	ora<dbsid>/sapsys	SAP tool for the Oracle database enables to back up database files
BRARCHIVE	/sapmnt/<SID>/exe	4775	ora<dbsid>/sapsys	Archive offline redo log files
BRCONNECT	/sapmnt/<SID>/exe	4775	ora<dbsid>/sapsys	Database administration tool (clean up logs, update statistics, adapt next extent, change password of dbusers, ...)
BRRESTORE	/sapmnt/<SID>/exe	755	<sid>adm/sapsys	Restore an entire database backup or parts of it
BRRECOVER	/sapmnt/<SID>/exe	755	<sid>adm/sapsys	Recover your database
BRSPACE	/sapmnt/<SID>/exe	755	<sid>adm/sapsys	Manage the space in your database
BRTOOLS	/sapmnt/<SID>/exe	755	<sid>adm/sapsys	User Interface: Collection of commands from DBA-Cockpit

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
- Unnoticeable feasible attacks

For this requirement the following warranty objectives are relevant:

ID: 3.22-183/i373

Req 184	The Oracle Database Role PUBLIC is not allowed to be changed.

'PUBLIC' is a special database role. It is created and configured at database creation. It does not show up in DBA_ROLES or SESSION_ROLES.

Because PUBLIC is accessible to every database users, all privileges and roles granted to PUBLIC are accessible to every database user. It is therefore not advisable to grant additional privileges and role to PUBLIC.

Content of SAP note 2553347 describes the issues, when role PUBLIC will be changed:

- You should not revoke any privileges/roles from database role PUBLIC (in 11g and higher).
- You should not grant any privileges/roles to database role PUBLIC.

- If the configuration of role PUBLIC in your SAP database was changed and privileges were revoked from PUBLIC, patching or upgrading your SAP database can fail. You should restore the original default configuration before you patch your database or upgrade to a newer release.

Motivation: Unauthorized database access on database content muss be prevented by granting no further authorities to role PUBLIC (following need-to-do- and need-to-know-principle).

Implementation example: By default, PUBLIC has no roles or system privileges granted:

- `SQL> select granted_role from dba_role_privs where grantee = 'PUBLIC';`

=> no rows selected

- `SQL> select PRIVILEGE from dba_sys_privs where grantee = 'PUBLIC';`

=> no rows selected

Only object privileges are granted to PUBLIC. Which object privileges are granted, depends on Oracle release and installed database components. In 12.2, more than 3000 privileges are granted to PUBLIC. You can check that with:

- `SQL> select count(*) from dba_tab_privs where grantee = 'PUBLIC';`

```
COUNT(*)
```

Further support you will get with SAP note 2553347.

ID: 3.22-184/i373

Req 185	(Default) databases that are not required must be deleted on the database system

(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.

ID: 3.22-185/i373

Req 186	Only instances with the same data protection class can be operated in the same database.

Database instances (with physical or virtualized hardware) that be operated on an operating system instance, must be -have the same protection requirements (data protection class),
-are under a single customer administrative authority and;
-are operated, in terms of administration, by the same group of people,may be operated on one operating system instance (with physical or virtualized hardware).

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 server hardware is used multiple times by multiple database systems, the risk increases of a largergroup of people obtaining unauthorized access to systems for which they are not responsible technically or interms of administration.

ID: 3.22-186/i373

Req 187	Not needed SQL functions that are not required.

Not needed SQL functions that are not required (e.g. T-SQL, PL/SQL, SQL PL, extended stored procedures) and/or packages from the database system muss be deleted.

Motivation: In many instances an attacker exploits such functions to transfer malware onto the system or execute commands
with privileged rights

ID: 3.22-187/i373

Req 188	Database functions which permit access to operating-system files must be deleted or deactivated.

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 the target system. These procedures are often provided with unnecessarily high, mainly privileged rights or allow anyone to execute programs without authentication.
For example, malware exploits the procedures MS-SQL: xp_cmdshell and Oracle: utl_tcp.

ID: 3.22-188/i373

Req 189	Database functions which allow access to other network services must be deleted or deactivated.

Database functions which allow access to other network services (e.g. ,SMTP, HTTP, SNMP, FTP etc.), must be deleted or deactivated.

Motivation: 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.

ID: 3.22-189/i373

Req 190	Database functions which allow access to the operating system or network services, must not be accessible

Database functions which allow access to the operating system 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.

ID: 3.22-190/i373

Req 191	Database accesses between various database systems must be comply with the least privilege principle.

Database accesses between various database systems must be comply with the least privilege principle.

Motivation: Limiting the access rights reduces the attack surface.

ID: 3.22-191/i373

Req 192	Database accesses between various database systems must be made via an individual user account.

Database accesses between various database systems must be made via an individual user account.

Motivation: Accesses shall be made via a special user. Only in this way can access rights be individually restricted to the absolute minimum level.

ID: 3.22-192/i373

Req 193	Accesses to database systems, as well as critical "stored procedure" and database content must be logged.

Accesses to database systems, as well as critical database procedures and database content must be logged.

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.
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

ID: 3.22-193/i373

Req 194	Database encryption must be enabled.

The sensitive database data must be protected against access from the operating system level. This can be achieved via database encryption.

Motivation: Unauthorized access to the database data via the operating system level must be prevented.

Implementation example: Example for SAP HANA DB:
To protect data saved to disk from unauthorized access at operating system level, the SAP HANA database supports data encryption in the persistence layer. Data volume encryption protects the data area on disk, while redo log encryption protects the log area on disk.

Data and Log Volume Encryption are not enabled per default. Ideally, encryption is enabled in the database immediately on creation.

Set up encryption within the global.ini configuration file in the system database:
section [database_initial_encryption]

- persistence_encryption = on
- log_encryption = on

Who can enable or disable data and log volume encryption as well as backup encryption in the tenant database initially depends on how the following parameter is set:
section [database_initial_encryption]

- encryption_config_control = local_database (default)
  the tenant database administrator can enable or disable encryption from the tenant database using the ALTER SYSTEM statement.
- encryption_config_control = system_database
  then only the system database administrator can enable or disable encryption from the system database using the ALTER DATABASE statement.

Attention: In case of a subsequent transfer of the responsibility for the encryption of the data and log volumes from the system database administrator to the tenant database administrator or vice versa, this must be done via an additional ALTER statement. A simple conversion of the parameters is not sufficient!

Requirement-ID: 3e1d6282

ID: 3.22-194/i373

2.13. Hana

Req 195	The connection between the application and the database must be encrypted.

The connection between the application and the SAP database must be encrypted.

Motivation: In the cloud, it cannot be ensured that an unencrypted connection is sufficiently secure.

Implementation example: please use the german Part

ID: 3.22-195/i373

Req 196	Use the PROD space to deploy your applications or create new spaces for the applications as required.

To ensure isolation, do not deploy your applications to the SAP space. In addition, do not assign the SpaceDeveloper role to platform users in the SAP space, unless it is absolutely necessary.

Motivation: To keep the attack vector on the SAP application low, it must be exist a separation between SAP Netweaver systems and other applications.

Implementation example: https://help.sap.com/viewer/742945a940f240f4a2a0e39f93d3e2d4/2.0.04/en-US/a752216352b14d1e90bc09f1f2107217.html

ID: 3.22-196/i373

Req 197	Sap-Hana Studio must be installed in a separate vlan.

SAP-Hana-Studio is the configuration tool for Hana. Since some application administrators also work with the SAP Hana Studio, the tool must be installed in a protected area.

Motivation: HANA-Studio is an administration tool that should only be accessible by administrators in order not to endanger the connected system.

ID: 3.22-197/i373

Req 198	Applications outside the multi-target application (MTA) must be strongly isolated from the MTA's resources.

Applications outside the multi-target application (MTA) must be strongly isolated from the MTA's resources.The instances of applications in the same space must be run with different operating system (OS) user.

Motivation: Compliance with the separation of data and roles

Implementation example: Each space can have a different OS user. Resources from different applications may be isolated by leveraging the concept of organizations and spaces in combination with separated operating system users. A microservice-driven architecture of a solution is typically characterized by the cooperation of several service instances fulfilling dedicated task. Only by combining microservices can the solution meet its overall requirements. In the XS ad-

vanced context, several applications work closely together, forming what is referred to as a multi-target application, or MTA. To illustrate, let's assume a simple MTA consists of two applications. A UI-based application needs to read database tables, which in turn are written by the other application. Consequently, both applications need exclusive access to the same database schema. The applications should also have a common authorization concept that applies to the same pool of end users. However, all other applications outside this MTA should be strongly isolated from the MTA's resources, in other words they should not be allowed to access the stored data nor the MTA's HTTP endpoints. The isolation of spaces depends on different OS users. Spaces can be mapped to dedicated OS users, and only spaces running with different OS users are isolated from each other. Applications running in the same space share all resources such as data storage, user authorizations, and passwords.

1. Don’t use <sid>adm or any other high privileged OS user as a space OS user.
2. Restrict the privileges of the space OS user as much as possible.

Organization	Space	Space OS User	Content
Initial organization	SAP	sap<sid>xsa	Contains pre-installed SAP applications, for example: • Deploy-service for MTAs • Product-Installer and component-registry for application installation and update This space is an appropriate location for other system-relevant applications from SAP such as the optional administration UI or the Application Role Builder tool.
Initial organization	PROD	<sid>xsa	Empty after installation The space PROD can be used to deploy your custom applications.

ID: 3.22-198/i373

Req 199	SAP HANA backups must be encrypted.

Considerations for Backup Encryption:

- It takes longer to create encrypted backups than unencrypted backups.
- It takes longer to recover a database using encrypted backups than from unencrypted backups.
- If backup encryption is enabled, both data backups and log backups are encrypted.

Motivation: Backup encryption safeguards the privacy of the SAP HANA business data by preventing unauthorized parties from reading the content of backups.

Implementation example: Encryption of backups is not enabled by default. Ideally, encryption of database backups is enabled in the database immediately upon creation.

Encryption of backups is set by the following parameter in the global.ini configuration file in the system database:

Sektion [database_initial_encryption]

- backup_encryption = on

Who can enable or disable encryption of backups in the Tenant database depends initially on how the following parameter is set:
Sektion [database_initial_encryption]

- encryption_config_control = local_database (default)
  the tenant administrator can enable or disable the encryption of the backups in the tenant database using the ALTER SYSTEM statement.
- encryption_config_control = system_database
  only the administrator of the system database can enable or disable the encryption of the backups of the tenant database(s) with the ALTER DATABASE statement.

Attention: In case of a subsequent transfer of the responsibility for the encryption of the backups from the system database administrator to the tenant database administrator or vice versa, this must be done via an additional ALTER statement. A simple conversion of the parameter is not sufficient!

ID: 3.22-199/i373

2.13.1. HANA: Databases user Roles and Privileges

Req 200	The database administrator SYSTEM should be deactivated after installation.

Use Databaseadministrrtor SYSTEM to create database users with the minimum privilege set required for their duties (for example, user administration, system administration). After that deactivate databaseadministrator SYSTEM.

Motivation: The database administrator SYSTEM is the most powerful database user with irrevocable system privileges. The database administrator SYSTEM is active after installation active and should be deactivated for regular operations

ID: 3.22-200/i373

Req 201	On production systems it's not allowed to use on a system privilege DEVELOPMENT authorizes some internal ALTER SYSTEM commands.

On production environment never exist the privilege DEVELOPMENT, because authorizes some internal ALTER SYSTEM commands. It's only reserved for the users SYSTEM and _SYS_REP.

Motivation: ALTER SYSTEM commands can be change system environments, they have to be treated very restrictive

ID: 3.22-201/i373

Req 202	The users SYSTEM and _SYS_REPO users have all system privileges by default, so they must be protected.

Critical combinations of system privileges should not be granted together.

ID: 3.22-202/i373

Req 203	Do not use the authentication rule "DATA ADMIN" in a production system.

The system privilege DATA ADMIN is a powerful privilege and may not be used in a productive environment of any role. The system privilege DATA ADMIN is a powerful privilege. It authorizes a user to read all data in system views, as well as to execute all data definition language (DDL) commands in the SAP HANA database. Only the users SYSTEM

and _SYS_REPO users have this privilege by default

ID: 3.22-203/i373

Req 204	Repository roles must be treated restrictive and regularly checked.

Repository permissions must be treated restrictively and verified regularly. The standard user _SYS_REPO automatically has all these roles. Depending on the installation, repository authorizations can also be assigned to standard user SYSTEM.

Motivation: Application-specific repository roles should only be granted to application users.

Implementation example: SAP HANA is delivered with a set of preinstalled software components implemented as SAP HANA Web applications, libraries, and configuration data. The privileges required to use these components are contained within repository roles delivered with the component itself.
The standard user _SYS_REPO automatically has all of these roles. Some may also be granted automatically to the standard user SYSTEM to enable tools such as the SAP HANA cockpit to be used immediately after installation.

As repository roles can change when a new version of the package is deployed, either do not use them directly but instead as a template for creating your own roles, or have a regular review process in place to verify that they still contain only privileges that are in line with your organization's security policy.
Furthermore, if repository package privileges are granted by a role, we recommend that these privileges be restricted to your organization’s packages rather than the complete repository. Therefore, for each package privilege (REPO.*) that occurs in a role template and is granted on .REPO_PACKAGE_ROOT, check whether the privilege can and should be granted to a single package or a small number of specific packages rather than the full repository.

ID: 3.22-204/i373

Req 205	It must not allow that user are granted to debug privileges.

The privileges DEBUG and ATTACH DEBUGGER should not be assigned to any user for any object in production systems.

Motivation: The privileges DEBUG and ATTACH DEBUGGER should not be assigned to any user for any object in production systems.

ID: 3.22-205/i373

Req 206	The default user "system" is not allowed to use for daily operation.

The role CONTENT_ADMIN contains all privileges required for working with information models in the repository of the SAP HANA database. The user SYSTEM has the role CONTENT_ADMIN by default.

Motivation: Only the database user used to perform system updates should have the role CONTENT_ADMIN. Otherwise do not grant this role to users, particularly in production systems. It should be used as a role template only

ID: 3.22-206/i373

Req 207	The role SAP_INTERNAL_HANA_SUPPORT can only be granted to member of the SAP support.

The role SAP_INTERNAL_HANA_SUPPORT contains system privileges and object privileges that allow access to certain low-level internal system views needed by SAP HANA development support in support situations. No user has the role SAP_INTERNAL_HANA_SUPPORT by default.

Motivation: This role should only be granted to SAP HANA development support users for the their support activities

Implementation example: The role is requested by SAP in support cases, for example.

ID: 3.22-207/i373

2.13.2. HANA: Network Configurations

Req 208	Only used ports such as SQL and HTTP are opened.

It must be ensured that the ports are only opened for the corresponding instance.During installation, ports such as SQL 3<instance_no>15 and HTTP 80<instance_no> are opened by default

Motivation: Only ports that are needed for running your SAP HANA scenario should be open.

ID: 3.22-208/i373

Req 209	System replications and backup must be communicate about an internal interface.

System replications and Backups must be configured via a separate internal network channel [system_replication_communication - list interface parameter .internal]. Another option is to set the list interface parameter global (.global). If the list parameter used, communication must be secured (by using TSL/SSL).

Motivation: The recommended setting depends on whether or not a separate network is defined for internal communication:

- *If a separate internal network channel is configured for system replication, the parameter[system_replication_communication] listeninterface parameter should be .internal. You also need to add key-value pairs for the IP addresses of the network adapters for the system replication in the[system_replication_communication] internal_hostname_resolution section.*
- *If a separate network is not configured for system replication, the parameter [system_replication_communication] listeninterface parameter should be set to .global. However, in this case, it is important to secure communication using TSL/SSL and/or to protect the SAP HANA landscape with a firewall. In addition, set the parameter[system_replication_communication] allowed_sender to restrict possible communication to specific hosts. The parameter value must contain a list of the foreign hosts that are part of the SAP HANA system replication landscape.*

ID: 3.22-209/i373

2.13.3. HANA: Operations System and File System

Req 210	The access permission of exported files to SAP HANA, can be configured using the "import_export" parameter in the indexserver.ini configuration file.

The access permission of files exported to the SAP HANA server can be configured using the [import_export] file_securityparameter in the indexserver.ini configuration file. The default permission set is 640 ([import_export] file_security=medium).

Motivation: Files are not displayed in the directory.

Implementation example: The access permission of files exported to the SAP HANA server can be configured using the [import_export] file_security parameter in the indexserver.ini configuration file.
file_security=<value>

ID: 3.22-210/i373

Req 211	The default log-output from the audit trail must be logged as the syslog,

SAP writes in different log directories. It must be ensured that at least the log output from the "Audit Trail" writes to the syslog of the system. The default global audit trail target is syslog (SYSLOGPROTOCO)

Motivation: If you are using syslog, ensure that it is installed and configured according to your requirements (for example, for writing the audit trail to a remote server).

ID: 3.22-211/i373

Req 212	The system generates core dumps (files) (for example, crash dump files) must be write to secure area, there only access to privileged user.

The system generates core dump files (for example, crash dump files) automatically. Runtime (RTE) dump files can be triggered explicitly, for example by using the SAP HANA database management console (hdbcons) or as part of a full system information dump (fullSystemInfoDump.py).

Motivation:

- *Generate runtime dump files to analyze specific error situations only, typically at the request of SAP support.*
- *Delete dump files that are no longer needed.*

Implementation example: RTE dump files must be generated by the <sid>adm user.To create RTE dump files in a running system as part of a full system information dump in the SAP HANA studio, a user requires the EXECUTE privilege on procedure SYS.FULL_SYSTEM_INFO_DUMP_CREATE.

Dump files are stored in the trace directory and have the same access permissions as other trace files (see above).

Runtime dump files created as part of a full system information dump can be retrieved by users with the EXECUTE privilege on the procedure SYS.FULL_SYSTEM_INFO_DUMP_RETRIEVE using the SAP HANA studio. At operating system level, any user in the SAPSYS group can access their storage location: /usr/sap/SID/SYS/global/sapcontrol/snapshots

ID: 3.22-212/i373

Req 213	It must be ensured that all data in a Hana database (including tenants) corresponds to the operating model and the associated protection requirements.

Since all tenants of a database use the same data store and therefore also the same SAML token for authentication, it must be ensured that all data meet the same protection requirements.The master token can also be used to read data in the tenants.

Motivation: To prevent users of one tenant database being able to log on to other databases in the system (including the system database).

ID: 3.22-213/i373

Req 214	It must be disable database features, that provide direct access to filesystem or network access.

To safe e your system, it is must be possible to disable certain database features. In particular that are actions with direct access to the file system, the network, or other resources. The right way is to use import and export functions.

Motivation: Review the list of features that can be disable if they not required in your implementation scenario.

ID: 3.22-214/i373

Req 215	No direct login with the Unix standard user <sid>adm and sapadm is allowed.

The installation procedure creates the “super” OS user <sid>adm for the entire SAP HANA system. As the owner of all operating-system processes, The <sid>adm user is very 'powerful' from a security perspective. It is not allow to use the account for operations task normally

Motivation: The user <sid> adm has very high authorizations and must be protected.

Implementation example: The installation procedure creates the “super” OS user <sid>adm for the entire SAP HANA system. As the owner of all operating-system processes, The <sid>adm user is very 'powerful' from a security perspective.sapadm user is used for Host Agent SAPHOstCOntrol, that is sapstartsrv executable running for host agent. The administrator user sapadm of the sap Host agent was created during the installation of host agent but it doesnt not get assigned a password. With start the host agent, sapstartsrv process runs under sapadm user. Example :
sapadm 8974 1 0 Oct18 ? 02:04:34 /usr/sap/hostctrl/exe/sapstartsrv pf=/usr/sap/hostctrl/exe/host_profile -D
Please check below help portal for more information.
https://help.sap.com/saphelp_nw70ehp2/helpdata/en/27/bbca94674848518f6bc4ea919e56ba/frameset.htmsapadm password can only be changed from DPS (after the installation).Login via an authorization tool such as SUDO.for more details please see ULC:
http://ts-fachapplikationen.telekom.de/cs/cs/open/152366482

https://help.sap.com/viewer/666e73444beb4f1582ec605e3ea95aaa/CURRENT_VERSION/en-US/f35e144d63224704831653b1eff76ddd.html

ID: 3.22-215/i373

Req 216	It must be to use exclusively person-related users, for tasks on the operating system level.

It must be creation different personalized user accounts for task on operationsystem and application.

Motivation: It must be sure, that accounts for administration operating system or application are different accounts. It must be possible to use different operation models

ID: 3.22-216/i373

Req 217	The content of the HANA-DB directories must be protected to unauthorized access at operating system level.

The content of the HANA-DB directories must be protected to unauthorized access at operating system level.

Motivation: Unauthorized access to the Hana-DB directory's may result to the Data in the DB-Base are corupt.

Implementation example:

OS-Verzeichnisname	OS-Zugriffsberechtigung	OS-User/OS-Gruppe
/hdb/<(DB)SID>/	755	<(db)sid>adm/sapsys
/hdb/<(DB)SID>/sapdata*	750	<(db)sid>adm/sapsys
/hdb/<(DB)SID>/sapdata*/ mnt0000<m>	755	<(db)sid>adm/sapsys
/hdb/<(DB)SID>/sapdata*/ mnt0000<m>/hdb0000<n>/data-valume_0000.dat	755	<(db)sid>adm/sapsys
/hdb/<(DB)SID>/saplog	755	<(db)sid>adm/sapsys
/hdb/<(DB)SID>/saplog/ mnt0000<m>	755	<(db)sid>adm/sapsys

/hdb/<(DB)SID>/saplog/ mnt0000<m>/hdb0000<n>	755	<(db)sid>adm/sapsys
/hdb/<(DB)SID>/saplog/ mnt0000<m>/hdb0000<n>/logsegment_000.dat	755	<(db)sid>adm/sapsys
/hdb/<(DB)SID>/hdblcm	755	root/root
/hdb/<(DB)SID>/HDB<instancexx>	750	<(db)sid>adm/sapsys
/hdb/<(DB)SID>/profile	755	<(db)sid>adm/sapsys
/hdb/<(DB)SID>/global	750	<(db)sid>adm/sapsys
/hdb/<(DB)SID>/exe	550	<(db)sid>adm/sapsys
/hdb/<(DB)SID>/hdbstudio	755	<(db)sid>adm/sapsys
/hdb/<(DB)SID>/hdbclient	755	root/root
/hdb/<(DB)SID>/lm_structure	755	<(db)sid>adm/sapsys
/hdb/<(DB)SID>/federation	750	<(db)sid>adm/sapsys
/usr/sap/<(DB)SID>	755	<(db)sid>adm/sapsys
/usr/sap/hostctrl	755	root/sapsys
/usr/sap/SMDAgent	775	<(smdagent)sid>adm/sapsys

OS-Kommandos zum Setzen der OS-Zugriffsberechtigungen und des OS-Users und der OS-Gruppe:

```
chown <OS-User>:<OS-Gruppe> <Verzeichnisname>
chmod <OS-Berechtigung> <Verzeichnisname>
```

Beispiel:

```
chown <(db)sid>adm:sapsys /usr/sap/<(DB)SID>
chmod 755 /usr/sap/<(DB)SID>
```

ID: 3.22-217/i373

2.13.4. HANA: Multitenant

Req 218	A tenant must be created and set up on the SAP HANA instance for each client (in the sense of a customer).

To maintain mandant (client) separation at the database level at HANA, each client must use its own tenant. Client separation using authorizations is not permitted at database level.

Motivation: By default, all database processes run under the default OS user <sid>adm which is low isolation mode in HANA MDC. In that mode cross-tenant attacks through operating system mechanisms are possible. To increase the security of a multiple container system, it is possible to configure the system for high isolation. With high isolation, the processes of individual tenant databases run under dedicated OS users belonging to dedicated OS groups and the processes of the system database run under the <sid>adm user. Internal database communication is authenticated.

implementation example: https://help.sap.com/viewer/78209c1d3a9b41cd8624338e42a12bf6/2.0.04/en-US/a92f66e5a6564c1bafd33244b94558dc.html

ID: 3.22-218/i373

Req 219	When more than one tenant is used the "high-isolation-mode" is to be set up to avoid cross-database attacks through OS mechanisms.

To increase security in a multi tenant database container, the "High Isolation" mode must be used. In this mode, the operating system processes of individual tenant database run under individual operating system user.

Motivation: Migration of further databases into SAP HANA multi tenant database container (MDC) can reduce costs for infrastructure and database management.
The SAP HANA multi tenant database concept allows to run and manage multiple tenant databases in one SAP HANA system. By default, all database processes run under the default OS user <sid>adm which is low isolation mode in HANA MDC. In that mode cross-tenant attacks through operating system mechanisms are possible. To increase the security of a multiple container system, it is possible to configure the system for high isolation. With high isolation, the processes of individual tenant databases run under dedicated OS users belonging to dedicated OS groups and the processes of the system database run under the <sid>adm user. Internal database communication is authenticated.

Implementation example: To enable the mode "high isolation" you must perform different steps. Beside the creation of tenant-dedicated operating system user and groups you must execute the Python script "convertMDC.py". Additionally you must setup HANA internal encryption. To get the exact steps for enabling the mode "high isolation" please goto the SAP Help to the "SAP HANA Tenant Databases".

https://help.sap.com/viewer/78209c1d3a9b41cd8624338e42a12bf6/2.0.04/en-US/a92f66e5a6564c1bafd33244b94558dc.html

ID: 3.22-219/i373

Req 220	Tenant-comprehensive access must be not allowed.

Access between two different tenants is not allowed.

Motivation: If the Database is using by tentants, it must be used.

Implementation example: On the Overview page of the system database in the SAP HANA cockpit, open Configuration of System Properties by clicking the corresponding administration link. Select the configuration file global.ini goto the section "cross_database_access" to parameter "enable" and set it from "TRUE" to "FALSE".

ID: 3.22-220/i373

Req 221	In the environment of the database tenant and the redo log, the authorization to change system-relevant keys must be in separate authorizations/roles.

The authorization to change the keys for the database tenant and redo log, are equipped with their own authorization and are worth protecting.

ID: 3.22-221/i373

Req 222	If using multi-tenant, port 17 must be protect.

When MDC (Multi Tenant Database Container) is using, on the host port 17 must be protect, because the complete database including all the tenants can be accessed via port 17.

Motivation: The database is addressed by port 17 via SQL

ID: 3.22-222/i373

2.13.5. HANA: Extended Services (XS) Engine

Req 223	The XS-Service port must be secure for direct using.

The XS server must be protected, he can address the database directly via SQL

Motivation: By opening the XS port, the system is vulnerable. It is possible to access the database via the http protocol.

ID: 3.22-223/i373

Req 224	The XS server are not allowd to runs with self-signed certificate.

By default, the XS server runs with self-signed certificate for all domains. It is not allowd to use the XS by default.

Motivation: Configure the XS server to accept a custom certificate for all your domains, especially the shared domain (used for XS CLI communication). Custom certificates can be upload by using the xs set-certificate command for each domain.

Implementation example: Use trusted certificates.

ID: 3.22-224/i373

Req 225	When using SAP HANA and the XS port, the authorization concept of the ABAP stack must not be bypassed.

Ensure that the authorizations defined in the Abap stack are continued in the SAP Hana database.

Motivation: It is possible to bypass an authorization concept based on an adapter using the XS Engine The XS Engine authorization concept is independent of the adapter authorization concept.

ID: 3.22-225/i373

2.13.5.1. HANA: User

Req 226	The standard passwords of the database standard users have to be changed according to the password policy!

Predefined XS Advanced system users (login possible):
XSA_ADMIN
HDI_BROKER_CONTROLLER
sap_sb

Predefined Technical SAP HANA Users (no login possible)
SYS_XS_RUNTIME
SYS_XS_UAA
SYS_XS_UAA_SEC
SYS_XS_HANA_BROKER
SYS_XS_SBSS
_SYS_DI
_SYS_DI_*_CATALOG
_SYS_DI_SU
_SYS_DI_TO

Predefined XS Advanced system users are generated automatically during the installation. Predefined technical users are generated during installation and configuration.

Motivation: Uncontrolled and unauthorized usage of standard user is not allowed. This can lead to read and change of the data and to a system outage.

Implementation example: Command for changing the password:
ALTER USER <user> {FORCE_FIRST_PASSWORD_CHANGE ::= PASSWORD <password>}

To check the user status use following statement, which list all existing user within HANA DB:

Select * from "SYS"."USERS";

An overview of existing users can be also seen using the SAP HANA Studio.

ID: 3.22-226/i373

Req 227	A minimum configuration must be implemented for a HANA system.

A minimum configuration must be implemented for a Hana system.

Install custom SSL certificates (XS trust-certificate and XS set certificate commands)
Appoint at least one XS advanced user to be OrgManager of each organization
Register all required service brokers
Create all required shared domains
Create all required custom buildpacks
Create all required runtimes
Configure logical databases
Set up global environment variables (xs set_running | staging_environment_variable_groups command)

Motivation: Prevention of unauthorized access by unneeded components.

Implementation example: Check the status on XS engine level on OS:
xs service-brokers
xs domains
xs domain-certificates
xs buildpacks
xs runtimes
xs tenant-databases

Check setting of parameter with following command:
`SELECT KEY, VALUE FROM M_INIFILE_CONTENTS WHERE KEY = '<parameter>'`

ID: 3.22-227/i373

Req 228	For the XS advanced engine it must be an authorization and user management concept

Split the following roles to different XS advanced personalized users:

- XS_AUTHORIZATION_ADMIN (managing roles, role-collections)
- XS_USER_ADMIN (assigning role-collections to XS advanced users)

Motivation: The policies (need-to-know, need-to-do, segregation of duties) must be prevented.

Implementation example: Check if privileges are granted to other users:
`SELECT DISTINCT USER_NAME FROM USER_PARAMETERS WHERE PARAMETER = 'XS_RC_XS_CONTROLLER_ADMIN'`

ID: 3.22-228/i373

Req 229	User XSA_ADMIN must be locked.

The XSA_ADMIN user can use the controller without any restriction. It is the only user who can carry out the initial set-

up of the models and must be locked due to this reasons.

Motivation: Avoid the usage of the XSA_ADMIN users.

Implementation example: Deactivate the XSA_ADMIN with the following SQL statement:
```
ALTER USER XSA_ADMIN DEACTIVATE USER NOW
```

In an emergency case you can reactivate this user with the SQL statement:
```
ALTER USER XSA_ADMIN ACTIVATE USER NOW
```

Hint: Only a user with system privilege USER ADMIN can perform these actions.

ID: 3.22-229/i373

Req 230	Logging on to XS advanced only with a personal OS user with a home directory that is not readable to other OS users is allowed.

The XS advanced session is stored in the file system of the current OS user. This must not be readable for other operating system users.

Motivation: Prevention of access via anonymous default users.

Implementation example: Change of the access to the HOME-directory for XS advanced user using following commands:
```
chmod 700 <directory name>
```

ID: 3.22-230/i373

2.13.5.2. HANA: Extended Services (XS) EngineUsers on operations layer

Req 231	No direct login with the OS level standard users is allowed.

The initial setup includes OS user <sid>xsa user for the PROD space and OS user sap<sid>xsa for the SAP spaceUser.

<sid>xsa: OS user for staging and running applications in the pre-configured PROD space

sap<sid>xsa: OS user for staging and running applications in the pre-configured SAP space

ID: 3.22-231/i373

2.14. Connectivity to external webservices

Req 232	Basic security functionality for externally accessible web services must be implemented in the DMZ and be protected with a web service gateway depending on the criticality.

A DMZ is a zone where all external traffic is terminated.
Databases and applications are located in the MZ, therefore in a zone where no direct external access is possible.

The following functions must be realized in the access and validation tier (DMZ):

- authentication and authorization for the access (i.e. is the consumer of the web service allowed for access)
- basic validation based on the given description such as WSDL, JSON schema or XSD
- pre- and postprocessing tasks (such as terminating an TLS encryption or security relevant logging)
- termination of tcp connections and preventing all direct access to web service end points in the MZ
- limit the maximum connections

To ensure these functionalities a web service gateway should always be used.

Motivation: All measures must be taken in the DMZ to protect the services in the MZ against attacks.

Implementation example:

- 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.

ID: 3.22-232/i373

Req 233	If data requiring special protection is processed in a web service, this data must be individually protected by using end-to-end mechanisms at the application level (end-to-end), such as XML encryption.

Since transport encryptions are already terminated on outer layers, as well as possible payload logging and "hop by hop" communication, a pure transport encryption in most cases does not offer sufficient protection for particularly confidential data.

Examples of data requiring special protection are Medical data, Criminal records, Bank details of a person, Quarterly figures before publication and Draft contracts with high financial volume.

Motivation: Due to the special need for protection, confidentiality protection is required for certain data even if they are encrypted for transport or transmitted via secure networks.

ID: 3.22-233/i373

Req 234	Only the minimal set web services must be made available to the public.

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 Web Services.

Motivation: By offering only the minimal required set of web services, the attack window can be reduced.

Implementation example: On internet web service gateways, only those web services may be made available that are required.
The same applies, for example, to the outbound connections of a container namespace.

ID: 3.22-234/i373

Req 235	Every provider and consuments of a web service must authenticated and authorise each other when transmitting data requiring protection.

The Provider of a web service must ensure that the consumer is auhorised to access this web service and to receive this data or use these functions.
The consumer must ensure that he is interacting with the correct supplier.

Authorization checks must be considered in every function. It is not allowed to rely on a special trust status due to the authentication of a device at network level or connections from e.g. the same network or namespace.

The authorization check must ensure that the consumer only receives the data he is allowed to receive

Motivation: In order to prevent unpermitted usage of resources or output of data, a proper authentication and authorization is necessary.
Access to a weather web service that does not require protection, does not require authentication or authorization.

Implementation example: To grant general authorizations for the access to a web service a certificate-based authentication is suitable.
To grant an authorization at data field level, JW tokens with individual "claims" are suitable, for example, in combination with filters in order to issue only those data fields that are authorized for the consumer.

The JW token or certificate must be verified by each function / container / microservice.

ID: 3.22-235/i373

Req 236	The mechanism to authenticate and authorize must rely on strong cryptographic algorithms / frameworks.

Strong cryptographic frameworks/algorithms in this context are e.g. XML signature or TLS client certificates with adequate algorithms like SHA3- / SHA2 hashes and ECDSA / RSA signatures.
Additional examples are JW-, oAuth- and STS-tokens if transferred over tls.

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

Implementation example: Certificates according to the certificate requirements for HTTPS:

JW Tokens:

- Hash procedure SHA 256 or higher
- no "none" algorithms ({"alg":"none"})
- ES, RS prefer algorithms over HS
- as short as possible lifetimes

ID: 3.22-236/i373

Req 237	All web service requests and answers must be validated by the web service provider and consumers against a detailed specification.

Each data record transmitted by a web service must match the expected data fields, the expected length and / or the expected range, as well as, if necessary, with a formal specification to describe the acceptable data.
If the expected data fields are not always definable, only expected data fields may be processed.

Motivation: The detailed specification (including formal specification) enables a much better description of the data expected in data fields of a web service.

Implementation example: A complete web service description can be found e.g. B. WSDL 2.0 or Swagger can be achieved.
Examples of data format descriptions are JSON schemas or XML schema definitions.
An example of a formal specification is regular information such as B. "([A-Za-z0-9] + [-._ + & #% / = ~]) * [A-Za-z0-9] + @ ([- A-Za-z0-9] + [.]) + [A-Zaz] {2,6} "for specifying a valid email address.

ID: 3.22-237/i373

Req 238	If the web service does not contain a formal specification of the input data, black / white lists must be used to prevent illegal characters from being accepted.

Data that is not expected from the following parts of the application may have unexpected and undesired effects.
If the input data in the DMZ is properly sanitized, such attacks are made considerably more difficult.

Motivation: Black / whitelisting can effectively protect against certain types of attacks, such as SQL-, LDAP-, XML-, XPath-, XQuery-, code-, command injection.
The use of whitelisting is preferable to blacklisting because blacklists tend to become obsolete over time

Implementation example: For web services that are run via a web server, the open source tool mod_security is a good solution.

ID: 3.22-238/i373

Req 239	Responses from REST Web Services must be based on HTTP status codes.

The HTTP protocol provides standardized codes for each status of a connection.

Further additional Information such as received data (payload) or technical details about the error may not be output but must be stored with an unique error reference in the log.
The error reference may be output.

Kategorie	Beschreibung
1xx: Informational	Communicates transfer protocol-level information
2xx: Success	Indicates that the client's request was accepted successfully
3xx: Redirection	Indicates that the client must take some additional action in order to complete their request
4xx: Client Error	Indicates that the client seems to have erred
5xx: Server Error	Indicates that the server seems to have erred

Motivation: The HTTP protocol already provides standardized codes for each status of a connection.
By using these status codes, informations requiring protection, such as error and system messages, are prevented from being included in the output.
The consumer of the Web service can also use these standardized codes to generically determine the cause of the error.
Reflecting the received data (payload) can lead to security gaps (especially in web applications).

Implementation example: An example of a few HTTP status codes and their meaning:

HTTP status code	Message	Description
200	OK	The request was processed successfully
400	Bad Request	The request is malformed
406	Not Acceptable	The content type requeststst by the client in the accept-header is not offered by the web service
413	Payload to large	The request is larger than the server is willing or able to process

All HTTP status codes and their usage are specified in the RFC7231.

ID: 3.22-239/i373

Req 240	Functionalities of REST web services must be based on HTTP methods.

The HTTP protocol provides standardized methods that are required for all functionalities.
By using these methods a uniform and transparent authorization control is enabled, which is supported by web servers, reverse proxies and web service gateways.

Motivation: By using standardized methods, upstream layers such as web servers, reverse proxies and web service gateways are able to perform a basic validation of the requests.
This also ensures a common understanding for consumers of how a web service should be used.

Implementation example: Examples of the most common HTTP methods:

HTTP method	Description
GET	The GET method requests transfer the current selected resource
HEAD	The HEAD method is identical to GET, except that the server does not send the HTTP BODY of the response
POST	The POST method requests the processing of the transmitted data
PUT	The PUT method requests that the state of the target resource be created or replaced
DELETE	The DELETE method requests the removal of the target resource
PATCH	The PATCH method requests a set of changes described in the request

ID: 3.22-240/i373

Req 241	Confidential data may not be transferred in the URL.

Confidential data may only be transferred outside the URL (e.g. HTTP-BODY, HTTP-HEADER).
When using HTTP headers, only fields that are not recorded in log files may be used.

Examples for confidential data are:

- API Keys
- Security Tokens (e.g. oAuth-, JW-Token)
- Passwords

Motivation: All data contained in the URL appears in log files of e.g. web server, reverse proxy and web service gateway.
Log files are often collected at a central location or transferred to SIEM systems for monitoring.
In order to make this possible and at the same time protect confidential data, it must not be transmitted in the URL.

Implementation example: Log files are collected

- api keys
- security tokens (z. B. oAuth-, JW-Token)
- password

ID: 3.22-241/i373

Req 242	Content types must be validated.

A REST request and response must match the intended content type in the header.
Otherwise this could cause misinterpretation at the consumer / provider side and lead to code-injection / code-execution.

Motivation: REST web services often allow multiple data formats for request and response.
The consumer must specify the data format of the request and the desired data format of the Web service response in the "Content-Type" and "Accept" headers.
The provider may only process the request if it supports the data format for the request and the response.

Implementation example: Reject requests with unexpected or missing content type headers with HTTP code 415 "Unsupported media type".

ID: 3.22-242/i373

Req 243	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.

ID: 3.22-243/i373

Req 244	Events must be logged with an exact time stamp and trigger alarms depending on their criticality. The logging of users/user actions and payloads must be coordinated with data protection and data security.

Events must be given an event identifier according to their event type.

Security relevant events must be specially marked in order to be able to trigger alarms as required.
Suitable event identifiers are, for example „info“, „error“, „warning“, „alert“, „emergency“.These event labels can be used to automatically trigger alarms and facilitate troubleshooting.
It is also possible to control deletion periods and log access authorizations based on these event labels.

Motivation: To ensure safe operation and error analysis, a logging, monitoring and alarming concept must be created. Safety-relevant events must be immediately forwarded to a suitable company, which can analyse the problem and take countermeasures.

Implementation example: The event identifier as well as the need for an alarm depends on the need for protection and the interface through which the web service is accessible.

A web service that can be accessed from the Internet is more likely to be subject to validation and authorization errors, so an alarm is not always appropriate. However, if this occurs with a web service that can only be accessed from trusted areas, e.g. from its own namespace, administrative special networks or M2M connections, this can be an indication of an intruder.

Likewise, the need for an alarm may depend on the frequency of the event.
An access attempt to access a resource without appropriate authorization or a validation error may be irrelevant if it occurs once. If this happens frequently, it can be an indication of a targeted attack on the application or the application's functions are not working properly.

The following table gives examples of events, event identifiers, and also the log context to be stored.

Category	Event	Event Identifier	Log Context
Access	Access to the web service	info	Applications-, Container-ID, Authenticity, IP
Validation error	Incorrect data format	error, debug	Applications-, Container-ID, Authenticity, IP, Payload
	Incorrect encoding	error, debug	Applications-, Container-ID, Authenticity, IP, Payload
	incorrect daten elements	error, debug	Applications-, Container-ID, Authenticity, IP, Payload
	validation against formal definition fails	error, debug	Applications-, Container-ID, Authenticity, IP, Payload
	Incorrect range of values	error, debug	Applications-, Container-ID, Authenticity, IP, Payload
Output error	Datenbase / supplying systems not accessible	error, debug	Applications-, Container-ID, Authenticity, Request-ID, IP
	Error in the database records	error, debug	Applications-, Container-ID, Authenticity, Request-ID, IP
	Output incomplete	error, debug	Applications-, Container-ID, Authenticity, Request-ID, IP
Authorisation error	Access to resources without proper authorization	warning	Applications-, Container-ID, Authenticity, IP, Payload
	Signature validation error for JW Token	alert	Applications-, Container-ID, Authenticity, IP, Payload
	Perform actions that do not match the role / ACL	alert	Applications-, Container-ID, Authenticity, IP, Payload
Authentication error	Failed	warning	Applications-, Container-ID, IP
	Success	info	Applications-, Container-ID, IP
Privileged actions	Add / Delete of accounts	trace	Applications-, Container-ID, Authenticity, IP
	Change of privileges	trace	Applications-, Container-ID, Authenticity, IP
	Exporting data requiring protection	trace	Applications-, Container-ID, Authenticity, IP
	Import of daten	trace	Applications-, Container-ID, Authenticity, IP
	Change of configuration	trace	Applications-, Container-ID, Authenticity, IP
Runtime error	Crash / restart of the application / container	emergency	Applications-, Container-ID, Stacktrace

connectivity problems	emergency	Applications-, Container-ID
performance problems	alert	Applications-, Container-ID
Virus detection e.g. file upload	alert	Applications-, Container-ID, Authenticity, IP

ID: 3.22-244/i373

Req 245	A SOAP request that cannot be processed immediately must ensure the confidentiality and integrity of the cached message.

In general, queuing can take place for various reasons, e.g. B. if the systems in the MZ do not respond quickly enough. In general, queuing should take place in the MZ if possible. If a SOAP request or response cannot be processed immediately and is cached in the DMZ, the confidentiality and integrity of the cached message must be ensured.

Motivation: An attacker can gain access to the content of the cached messages without additional encryption.

Implementation example: The messages can be stored in AES encrypted form in a database. In order to ensure integrity, a hash value must also be formed over the message.

ID: 3.22-245/i373

2.15. Netweaver as Webapplication

Req 246	The web server service must be bound to one interface.

The web server service may only be bound to the interfaces via which it must be accessible. In most cases, the web server service only needs to be bound to one interface.

Motivation: The more interfaces allow access to the web server, the higher the risk of attack

ID: 3.22-246/i373

Req 247	HTTP methods that are not required must be deactivated.

HTTP methods that are not required must be deactivated.

Motivation: HTTP methods that are not required must be deactivated. The TRACE / TRACK method must not be used on a productive web server. Standard requests to web servers use only GET and POST. If other methods are required, they must be processed safely.

Implementation example: For ABAP, Java and HANA internal SAP Web Dispatcher via suitable parameter to set.

Configure the allowed HTTP methods via parameter icm/HTTP/mod_<xx> with option FILE as follows:

```
if %{REQUEST_METHOD} !stricmp "GET" [AND]
if %{REQUEST_METHOD} !stricmp "POST"
RegForbiddenURL ^/(.*) –
```

In the example only GET and POST are allowed.

SAP Help: https://help.sap.com/doc/saphelp_snc70/7.0/de-DE/ca/27caf57d2645f7aff689875f23d5d8/content.htm?no_cache=true

ID: 3.22-247/i373

Req 248	Components of the web server that are not required must be uninstalled.

Components of the web server that are not required for operation or function must be uninstalled or deactivated.

Motivation: Every extension, component or function can have security vulnerabilities.

Implementation example: All optional components and extensions of the web server must be deactivated if they are not required.Examples:

- CGI
- Server Side Includes(SSI)
- WebDAV

ID: 3.22-248/i373

Req 249	CGI technology must not be used.

CGI technology must not be used.

Motivation: By an improper CGI configuration, a large number of attack can result in a web server. Since modern web servers have different, more secure and more powerful alternatives to CGI, the use of CGI is neither necessary nor recommended.

ID: 3.22-249/i373

Req 250	If server side includes (SSI) are active, the execution of system commands must be deactivated.

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 as an additional loadable module in most web server products, can potentially be used by attackers. In particular, SSI's "exec" function could be used to execute system commands, which is a risk.

ID: 3.22-250/i373

Req 251	If WebDAV is used for writing files, access must not be granted without successful authentication.

If WebDAV is used for writing files, access must not be granted without successful authentication.

Motivation: WebDav enables the online update of content that is made available by the web server. This function could therefore be misused to change website content.

Implementation example: All access rights to files that can be accessed via WebDAV, must be set as restrictively as possible. In addition, WebDAV access must be restricted to the required directories.

ABAP: ICM tree. Check if service "dav" active.
TA SICF => Service execute => tree Default host => sap => bc => ecm

ID: 3.22-251/i373

Req 252	If WebDAV ist used, the access to needed directories must be restricted regarding the authorized user.

Access rights to all files accessible by WebDAV must be configures 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.

ID: 3.22-252/i373

Req 253	Default content must be removed.

Default content (examples, help files, documentation, aliases) provided with the standard installation must be removed.

Motivation: By using examples, information about the installed software (version) could be obtained or executable examples could contain security vulnerabilities.

ID: 3.22-253/i373

Req 254	The creation of directory lists (indexing) must be deactivated.

The creation of directory lists (indexing) must be deactivated.

Motivation: Directory lists contain information about files and directory structures that could be misused.

ID: 3.22-254/i373

Req 255	The HTTP header must not contain any information about the version of the web server.

Information about the web server in HTTP headers, must be kept to a minimum. The HTTP header must not contain any information about the version of the web server and the modules / extensions used.

Motivation: All information about the web server could provide information about security vulnerabilities.

Implementation example: Use the following settings to reduce the server information to a minimum:

- Set the parameter is/HTTP/show_server_header to FALSE. When you change this, the "Server:" header field is no longer set in HTTP responses.
- Set the parameter is/HTTP/show_detailed_errors to FALSE. After you do this, the system does not return any details about the error to the client.

ID: 3.22-255/i373

Req 256	Information about the web server in error pages delivered by the web server must be removed.

Information about the web server in error pages delivered by the web server must be removed. The standard error pages must be replaced by user-specific error pages. User-defined error pages must not contain version information of the web server and the modules / extensions used. The error messages must not contain any internal information, such as B. contain internal server names, error codes etc.

Motivation: All information about the web server could provide information about security vulnerabilities.

ID: 3.22-256/i373

Req 257	The unauthorized overwriting of the web server configuration must be prevented.

The unauthorized overwriting of the web server configuration must be prevented. The web server configuration can be

overwritten, for example, by creating additional config files.

Motivation: Changes to security-relevant settings must be prevented.

ID: 3.22-257/i373

Req 258	The web server may only deliver files that are intended for delivery.

The web server may only deliver files that are intended for delivery. All files that are located directly or indirectly (for example via links or in virtual directories) in the document directory of the web server must be provided with restrictive access rights. In particular, the web server must only be able to access files that are intended for delivery.

Motivation: If additional files or directories are linked into the document directory of the web server via links or virtual directories, there is a possibility that a user can access files via the web server that he should not be able to see. This must be prevented by careful configuration.

ID: 3.22-258/i373

Req 259	ECC certificates must implement a key length of at least 3072 bits.

Certificates must have a key length of at least 3072 bits when using RSA or at least 256 bits when using ECC.

Motivation: In order to guarantee the security of the certificates over the validity period, the cryptographic keys must have a corresponding length. According to general assessment, a key length of 3072 bits offers sufficient protection for the next few years. With ECC algorithms, shorter key lengths already offer the same level of security.

ID: 3.22-259/i373

Req 260	Access to the web server must be logged.

Access to the web server must be logged and contain the following information:

- Access time
- Source (IP address)
- Account (if known)
- URL
- Status code of the response from the web server

Motivation: When analyzing security incidents, it is very important to have basic information about how an attack occurred. Since a web server represents an external interface, certain information is only available on the web server even when attacking a downstream system and must therefore be logged there.

ID: 3.22-260/i373

Req 261	The web application must use HTTP Strict Transport Security (HSTS) to force all future connections from the browser to be established in encrypted form only. For this purpose, the web application must set the HTTP response header “Strict-Transport-Security”.

The web application must use HTTP Strict Transport Security (HSTS) to force all future connections from the browser to be established in encrypted form only. For this purpose, the web application must set the HTTP response header “Strict-Transport-Security

Motivation: If a browser 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 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

Implementation example: ABAP, Java and SAP Web Dispatcher:

From Release 7.53 set parameter:
icm/HTTP/strict_transport_security = max-age=31536000; includeSubDomains
HSTS does not work with servers running on non-standard HTTP/HTTPS ports because the user agents simply change the protocol from HTTP to HTTPS without paying attention to the non-standard ports.

Alternative and without restrictions for standard HTTP/HTTPS ports (s. SAP-Note 2042819):

Set up parameter icm/HTTP/mod_<xx> option FILE with the following lines in the file:

First, create a modification rule for redirecting HTTP requests to HTTPS:

```
if %{SERVER_PROTOCOL} !stricmp "https"
RegIRedirectUrl /(.*) https://<host>:<https_port>/$1
```

Where is the <host> Web dispatcher or ICM host, and <https_port> is the HTTPS server port.

Then, for HTTPS requests, set a modification rule to set the response header for Strict Transport Security, e.g.:

```
if %{SERVER_PROTOCOL} stricmp "https"
SetResponseHeader Strict-Transport-Security "max-age=31536000; in-
cludeSubDomains"
```

You can customize the header value to suit your specific needs.

S. SAP Hinweis 2202116

ID: 3.22-261/i373

Req 262	The web application must not use URL parameters or other fields that are captured in log files to transmit data with need of protection.

The web application must not use URL parameters or other fields that are captured in log files to transmit data with need of protection.

Motivation: In the case of the GET method, parameters are transmitted in the URL and are usually 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 HTTP header "Referer" and then appear in log files there. This part of the problem may also be addressed (for most browsers) by using the HTTP response header "Referrer-Policy" (for example, with the value "no-referrer"). Moreover, it cannot be ruled out that a user copies a used URL including data with need of protection. In the case of the POST method, for example, parameters are transmitted in the request body.

ID: 3.22-262/i373

Req 263	The web application must not store any sensitive data on the client side, either temporarily or persistently.

This applies both for cookies and for the different 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 may transmit session identifiers or (session/access) tokens in non-persistent cookies or store this data in the session storage of the browser temporarily.
As a further exception, the web application may store data for identifying the user, such as the user name or an ID, in persistent cookies or other local storages, if personal data is stored in encrypted form and if the user has granted consent.

The web application must also prevent data with need of protection that the user enters in an HTML form from being stored 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. 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: Motivation: All data stored on the client can in principal be analyzed and manipulated, either locally by direct access or possibly by a successful attack from outside.
In case of cookies, a web application can set an expiry date via the "expires" or "max-age" attribute. But then this is a "persistent cookie". The browser stores persistent cookies on the device of the user until the expiry date. Every person with access to the device can read the persistent cookies. This can happen in an Internet café, in other cases of shared use, or through a successful attack. Cookies without an expiry date are not persistent, meaning they are not permanently stored. The browser deletes these cookies as soon as it is closed. However, 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, among other things, for HTML5 applications provide further mechanisms for saving data locally to the client. But these local storages can be analyzed or tampered with on the client, too. For example, the DOM objects "localStorage" and "sessionStorage" are such an alternative that can be used to store data in a browser. However, in local storage data is stored permanently, while data stored in session storage is deleted with the end of the session. But regarding session storage it must be noted that a new session is created, when a website is opened in a new tab or a new browser window. All scripts of a domain, from which the data was stored, are able to access the stored data (of the same local user profile in case of local storage or of the same tab/browser window in case of session storage). Unlike cookies via the "HttpOnly" flag, there is no inherent protection mechanism against crosssite scripting for these storages.

Deactivating autocomplete prevents data that was entered from being stored locally regardless of the browser configuration. In this context it should be noted that most current browsers ignore the "autocomplete" attribute for "password" fields.

ID: 3.22-263/i373

Req 264	The web application must not use any user input data for direct access to files and directories as well as to other server-side resources.

Instead, indirect object references of server-side resources must be implemented.

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 would not usually be able to access.
If, for example, a web application shows the Deutscht.txt help file, when a user enters “Deutsch”, and the web application does also not sufficiently validate the user input, the attacker can possibly cause the web application to show the user file 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. Furthermore, if there is a corresponding security gap, an attacker can even incorporate external files into a script and thus cause code to be executed (remote file inclusion).

ID: 3.22-264/i373

Req 265	The web application must not use any input data to create shell commands or commands for the active program (for example, via "eval").

The web application must not use any input data to create shell commands or commands for the active program (for example, via "eval").

Motivation: This prevents OS command injection and code injection.
OS command injection can occur when input data is integrated into a command that is then passed to the operating system for execution. This may allow an attacker to execute arbitrary OS commands on the server running the web application. In that case, he can usually compromise the system completely. Very often, an attacker can even exploit such a vulnerability to compromise other parts of the hosting infrastructure by exploiting trust relationships and extending his attack to other systems.
With code injection, on the other hand, an attacker plants executable code. This can be server-based (for example, via php) or client-based (for example, via JavaScript). If the attacker is successful, this code is executed and can cause major damage, too. However, an attacker can also use code injections to insert XSS or SQL injection attacks.

For example, this code could be part of a web application to administer a web server:
string dirName = "C:\\filestore\\" + Directory.text;
ProcessStartInfo psInfo = new ProcessStartInfo ("cmd", "/c dir " + dirName);
...
Process proc = Process.Start(psInfo);
This function displays the contents of a directory. The script inserts the value of the parameter "Directory" as provided by the user into a specified command. However, an attacker could use shell metacharacters to inject his own commands and have them executed: directoryname && rm -rf.

If the PHP function eval() is used and untrusted data is passed to it that an attacker can influence
$username = $_POST['username'];
eval("echo $username");
code injection is possible, for example by entering mustermann; phpinfo().

If on the other hand in case of JavaScript a JSON document is interpreted via "eval"
eval({"menu":{"address":{"line1":addressLine1,"line2":"","line3":""}}});
and then the "addressLine1" variable from the request is assigned the value
",arbitrary:alert('executed!'),continue:"
injected code will be executed here, too.

All user inputs represent a potential attack surface for this. Direct execution via "eval" (or comparable functions) is highly problematic as the data is interpreted directly without further checks regarding security or validity. This applies on both the server side and the client side.

Implementation example: For example, this code could be part of a web application to administer a web server:
string dirName = "C:\\filestore\\" + Directory.text;
ProcessStartInfo psInfo = new ProcessStartInfo ("cmd", "/c dir " + dirName);
...
Process proc = Process.Start(psInfo);
This function displays the contents of a directory. The script inserts the value of the parameter "Directory" as provided by the user into a specified command. However, an attacker could use shell metacharacters to inject his own commands and have them executed: directoryname && rm -rf.

If the PHP function eval() is used and untrusted data is passed to it that an attacker can influence
$username = $_POST['username'];
eval("echo $username");
code injection is possible, for example by entering mustermann; phpinfo().

If on the other hand in case of JavaScript a JSON document is interpreted via "eval"
eval({"menu":{"address":{"line1":addressLine1,"line2":"","line3":""}}});
and then the "addressLine1" variable from the request is assigned the value
",arbitrary:alert('executed!'),continue:"
injected code will be executed here, too.

All user inputs represent a potential attack surface for this. Direct execution via "eval" (or comparable functions) is highly problematic as the data is interpreted directly without further checks regarding security or validity. This applies

on both the server side and the client side.

ID: 3.22-265/i373

Req 266	The length of the web application's session identifier must be at least 120 bit. All the relevant characters must be generated at random.

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.

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.

Implementation example: Verification: Log on to the SAP application with the Chrome browser and open it in the developer tools (Ctrl + Shift + i). In the "Application" menu bar, select the URL of the website in the "Cookies" context menu. In the line with *session*, read the column "Size" [bytes]. Depending on the ASCII code used, the number in the "Size" column must be multiplied by 7 or 8. The result must be greater than 120.

ID: 3.22-266/i373

Req 267	If stateless session/access tokens (such as JSON Web Tokens, JWT) are used, the web application must prevent manipulation, replay attacks or other types of illegitimate use by taking appropriate measures.

In the case of JWT, for example, they must be integrity-protected via secure cryptographic methods and algorithms, preferably using cryptographic signatures. The recipient of the JWT must check the integrity according to its own configuration/application logic (hence, not based on the header information of the JWT) and in particular not accept any unsecured JWT with "{"alg": "none"}". The validity period must also be as short as possible. And the tokens must always be protected by TLS.

Motivation: It must be prevented that an attacker can misuse such tokens. It is therefore necessary that the tokens are transmitted securely and verified restrictively. However, it should be noted in this context that solutions based on JWT may be comparatively easy to implement, especially to realize "stateless" services, but are not suitable for all types of web applications, especially because they are associated with some problems, such as the question of storing the tokens in the browser or the realization of a restrictive server-side timeout and logout mechanism.

ID: 3.22-267/i373

Req 268	To transmit session identifiers or stateful/stateless (session) tokens, the web application must not use URL parameters.

Such identifiers or tokens may only be transmitted outside the URL, that is in a (session) cookie or, alternatively, in another HTTP header, for example.

Motivation: Session cookies offer the advantage that important security-relevant properties can be easily defined by the web application and are guaranteed accordingly by the browsers (no permanent storage, secure transmission, no script access, restricted validity range). If other mechanisms are used, these properties must be ensured by the respective implementation.

The use of URL parameters is not permissible: In this case, it cannot be ruled out that a user copies a used URL including a valid identifier and passes it on. Furthermore, there is a risk of the identifier being sent via the "Referer" header to another web application. And last but not least, using cookies or HTTP headers minimizes the risk, that a web application is attacked by means of session fixation: For an attack of this type, the attacker first establishes a session, and then foists the identifier in question on a victim. This is usually done by means of a link that contains the 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.

ID: 3.22-268/i373

Req 269	A session identifier or stateful/stateless (session) token must not be stored in the browser persistently.

A session identifier or stateful/stateless (session) token must not be stored in the browser persistently.

Motivation: In case of session cookies a web application can set an expiry date for a cookie via the "expires" or "max-age" attribute. But then this is a "persistent cookie". The browser stores persistent cookies on the device of the user until the expiry date. Every person with access to the device can read the persistent cookies. This can happen in an Internet café, in other cases of shared use, or through a successful attack. Cookies without an expiry date are not persistent, meaning they are not permanently stored. The browser deletes these cookies as soon as it is closed.

If session cookies are not used, the browser's session storage can be used, for example, as this is also not persistent, and the content is deleted when the browser is closed. But it must be noted that a new session is created, when a website is opened in a new tab or a new browser window. Using the local storage of the browser, however, the identifiers would be stored persistently.

ID: 3.22-269/i373

Req 270	The web application must set the attribute "secure" in the session cookie.

In addition, it is recommended to use cookie names with the prefix "__Host-" (or alternatively "__Secure-") to protect the integrity of a cookie.

If the web application consistently sets the HTTP response header "Strict-Transport-Security" and thus ensures that the browser sends encrypted requests only, it is no longer mandatory to set the attribute "secure". However, it is still recommended as an additional security measure.

Motivation: The attribute "secure" prevents the browser from sending cookies unencrypted. This happens, for example, with unencrypted contents of a web application. However, this can also happen through an active attack in which an attacker injects or presents unencrypted links or references. So even if a page is only accessible via HTTPS, a page controlled by the attacker could cause the victim's browser to make an unencrypted request. An attacker could intercept this request and would have access to the cookie. If the "secure" attribute is set, this attack is unsuccessful: the browser does not add appropriately secured cookies to unencrypted HTTP requests.

The cookie prefixes protect against cookie injection or cookie clobbering: Even an unencrypted HTTP response can set a "secure" cookie. For example, a man-in-the-middle attacker could therefore set or modify "secure" cookies. Or an attacker could exploit problems with subdomains, e.g. if he controls evil.example.de and sets a cookie with "domain=example.de" for requests to this subdomain or if he finds an XSS on insecure.example.de and sets a cookie with "domain=example.de". Additional rules for browser behavior can be determined via a prefix in the cookie name (if the browser supports this function): If the cookie name has a prefix "__Secure-", a compatible browser will only set this cookie if it is set by an HTTPS response and the "secure" flag is set (example: Set-Cookie: __Secure-ID=123; Secure; Domain=example.com). If the cookie name has a prefix "__Host-", a compatible browser will only set this cookie if it is set by an https response, the "secure" flag is set, the "path" flag is set to root ("/") and the "domain" flag is not set (example: Set-Cookie:__Host-ID=123; Secure; Path=/).

Implementation example: ABAP:
Set parameter:
login/ticket_only_by_https = 1

Verification: Log in to the SAP application using the Chrome browser and open the developer tools (Ctrl + Shift + i). In the "Application" menu bar, select the URL of the website in the "Cookies" context menu. In the line with *session*, check the "Secure" column to see if there is a check mark.

ID: 3.22-270/i373

Req 271	The web application must set the “HttpOnly” attribute in the session cookie.

The web application must set the “HttpOnly” attribute in the session cookie.

Motivation: The “HttpOnly“ attribute does not enforce (as the name suggests) that a cookie is transmitted via HTTP only. Indeed, it prevents the access for JavaScript via the "document.cookie"-API. Therefore, it prevents XSS attacks on the cookie.

Implementation example: ABAP:
Set parameter:
icf/set_HTTPonly_flag_on_cookies = 0

Verification: Log in to the SAP application using the Chrome browser and open the developer tools (Ctrl + Shift + i). In the "Application" menu bar, select the URL of the website in the "Cookies" context menu. In the line with *session*, check the "HttpOnly" column to see if there is a check mark.

ID: 3.22-271/i373

Req 272	The web application must not set the “domain” attribute in the session cookie.

The “domain” attribute specifies the domain name for which the cookie is valid. The browser sends the cookie with all requests that the browser sends to this domain and its subdomains. 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 subdomains, so the session cookie is prevented from being sent to third party web applications, too.

Additionally, 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.

If, for example, the web application xyz.telekom.net sets a cookie without a "domain" attribute, the cookie is then sent in all requests to xyz.telekom.net and its subdomains (such as abc.xyz.telekom.net). However, if the web application sets the "domain" attribute to “telekom.net”, every web application in a subdomain of telekom.net will receive the cookie.

Implementation example:

- For the session cookie, configure the following properties:
  - Global_app_config/cookie_config/session_cookie/path
  - Global_app_config/cookie_config/session_cookie/domain
  - Global_app_config/cookie_config/session_cookie/max-age
- For the application cookie, configure the following properties:
  - Global_app_config/cookie_config/application_cookie/path
  - Global_app_config/cookie_config/application_cookie/domain
  - Global_app_config/cookie_config/application_cookie/max-age

You can also configure cookies locally using the web-j2ee-engine.xml application deployment descriptor.

https://help.sap.com/saphelp_em92/helpdata/de/49/75556b9bea200ce10000000a42189c/content.htm?no_cache=true

ID: 3.22-272/i373

Req 273	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.

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.

Motivation: 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 applications. 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.

If, for example, a web application under telekom.net/myapplication/index.jsp sets a cookie with the path specification ";path=/", the cookie is then sent in all requests to the telekom.net domain and potentially also to other, less trustworthy applications that were placed in the root or any other directories.
However, if the path specification is set to ";path=/myapplication/", the cookie is sent only for requests to telekom.net/myapplication/ (and to subdirectories, but not to higher level directories). The concluding slash sign must not be missed out, as otherwise the cookie will also be sent to other directories with matching names, like telekom.net/myapplication-exploited.

If no path specification is given, the browser uses the path of the current HTTP request based on which the cookie was set as the default.

Implementation example: If, for example, a web application under telekom.net/myapplication/index.jsp sets a cookie with the path specification ";path=/", the cookie is then sent in all requests to the telekom.net domain and potentially also to other, less trustworthy applications that were placed in the root or any other directories.
However, if the path specification is set to ";path=/myapplication/", the cookie is sent only for requests to telekom.net/myapplication/ (and to subdirectories, but not to higher level directories). The concluding slash sign must not be missed out, as otherwise the cookie will also be sent to other directories with matching names, like telekom.net/myapplication-exploited.

ID: 3.22-273/i373

Req 274	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 reasonable 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 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 permit multiple logins for other reasons. But in such exceptional cases, it is recommended that the user then has the option of deliberately terminating the other parallel sessions via a corresponding function, for example, in the case of a password change.

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 a successful attack is taking place.

ID: 3.22-274/i373

Req 275	The web application must have a function that allows a signed in user to logout at any time.

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 data against unauthorized access. A

specific logout can be used to end a session in order to ensure that this session cannot be continued by an unauthorized person.

Implementation example: The web application must provide a way to log out.

ID: 3.22-275/i373

Req 276	If a user accessed the web application by means of a single sign-on procedure (SSO) and then logs out of the web application, both sessions must be terminated – the session with the web application and the session with the SSO portal.

If the session with the SSO portal cannot be ended automatically, as an alternative the 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: 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 could possibly take it over without being noticed.

ID: 3.22-276/i373

Req 277	If a user accessed the web application by means of a SSO 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.

Motivation: When the user logs out of the SSO portal, he may not realize that other applications are still in use. If the sessions with these applications are not terminated automatically, an attacker could possibly take them over without being noticed.

ID: 3.22-277/i373

Req 278	After a user is inactive for a specified amount of time, a timeout must occur in the web application and the user must be automatically logged out of the web application.

The precise amount of time after which a timeout occurs must be specified individually for each application. It depends on the application's sensitivity and purpose. For critical web applications accessible from the Internet and with access to data with need of protection (where, for example, a user is supposed to access personal data of different customers), it is recommended that the timeout occurs after 60 minutes at the latest, whereas for internal web applications without access to data with need of protection, this period could be chosen to be significantly longer. It is also recommended that this time period be a configurable system parameter.

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

ID: 3.22-278/i373

Req 279	When a user logs out or a timeout occurs, the web application must invalidate the corresponding session identifier or stateful/stateless (session) token on the server side.

When a user logs out or a timeout occurs, the web application must invalidate the corresponding session identifier or

stateful/stateless (session) token 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 gains access to this session. This is possible, for example, if the attacker uses the same computer as the victim or if he has determined the session ID using a different attack.

ID: 3.22-279/i373

Req 280	If the session management of the web application is based (exclusively) on session cookies, 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 requests on a website that he controls, which (in case the user is accessing that website) would trigger valid actions of the user for the web application to be protected.

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

Generally, “CSRF tokens” (also known as synchronizer tokens) are incorporated as a hidden field for this. A CSRF token must be generated on the server side, must be unpredictable and must be individual at least for every session. The web application must check on the server side for all relevant requests whether the token has been transferred and its value matches the expected value before the requested action is performed. Many frameworks provide corresponding protection mechanisms. In this case, it is usually advisable to use them instead of implementing a corresponding solution yourself.
If implementation as a hidden field is not possible or reasonable, the CSRF token can be transmitted in a specific HTTP header (for example, “X-CSRF token”).
However, information that the browser sends automatically, such as cookies, does not provide any protection.

As an additional protection mechanism, it is recommended to set the attribute "SameSite" (usually with the value "Lax") in the session cookie.

Motivation: Requests that are not protected in this way are susceptible to CSRF (also known as XSRF or session riding). With this, a victim is made to unknowingly send a prepared HTTP request. This occurs, for example, through a visit to a malicious website that contains a relevant link to another web application (for example, as an "img", "script" or "iframe" tag). However, the victim cannot recognize this link. But the browser follows this link in the background and sends the session cookie automatically, too, as long as the user has an active session for this other web application.

*If, for example, a user of a web application can transfer money # https://example.com/app/transferFunds ?amount=1500 &destinationAccount=4673243243
and the corresponding request does not include a secret token, an attacker can prepare a request that will transfer money from the victim’s account to the attacker's account. The attacker embeds this request in an image tag and stores it on a site under his control. [bad URL at img=https://example.com/app/transferFunds ?amount=1500 &destinationAccount=attackersAcct#]*

*
If the victim visits this prepared site, while already authenticated to example.com, the forged request will include the user’s cookie and will therefore be executed.*

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

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 attack. However, some frameworks do not support CSRF tokens that are individual for each request. This variant may also lead to problems using the web application, for example when working in several tabs or windows in parallel.

By setting the "SameSite" attribute in the session cookie, the browser is instructed not to send the session cookie with

a cross-site request. This basically prevents CSRF attacks (if the browser supports this function). If the value "Strict" is set, the cookie is not sent with any such request; however, this can have undesired effects on browser behavior, for example, if the user selects a link to another web application and the session cookie for this web application is not sent, so that a user who was originally logged in has to log in again. If the value "Lax" is set, in the case of cross-site requests the cookie is only sent for HTTP methods that are considered secure (GET) and for top-level navigation (hence, not for inline frames, image tags or similar). Only for a web application that is stringently implemented regarding the HTTP methods will the value "Lax" therefore also provide sufficient protection against CSRF attacks.

With another CSRF variant, login-CSRF, the victim does not have to have a currently valid session. In this case, the attacker forges a login request for the vulnerable web application, using the attacker's credentials in this forged request. If the attack is successful, the victim is logged in with the attacker's account. All subsequent requests are then made with the attacker's account. One scenario where such an attack can be interesting is an unintentional login to a search engine with the consequence that the attacker can track the victim's subsequent search requests.

A logout-CSRF attack, on the other hand, could lead to DoS scenarios, for example.

ID: 3.22-280/i373

Req 281	The web application must use a mechanism against clickjacking attacks which prevents the web application from being embedded by other unauthorized web applications.

This is usually achieved by outputting the HTTP header “X-Frame-Options” in the web application's responses. This specifies whether the contents of the web application may be displayed within a frame (that is a frame or iframe element) of another web application. The header is set to “DENY” or “SAMEORIGIN”.

If it is not possible to use this header, clickjacking can also be prevented via the Content Security Policy (CSP) and its directive "frame-ancestors". When using the CSP header, in addition to the values 'none' and 'self', it is also possible to specify specific domains (incl. wildcards) to allow dedicated domains to embed the web application.

In particular, forms und functions in the web application that can cause a data change or status change must be protected accordingly. However, it is recommended to also protect pages that only present information from being displayed within a frame, as this is associated with side effects that can lead to further attack scenarios.

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 the victim never intended to click. The attacker inserts parts of the legitimate application into his page, usually by means of an inline frame, but masked or hidden by other elements of his own page. The victim is then led to click on content that is supposedly on the attacker's page. However, in reality this executes a click and an action on the legitimate web application.

Further attack scenarios for web applications that can be displayed within a frame result, for example, from the fact that in the case of Internet Explorer, the "document mode" is inherited by the page in the frame. Such a forced "downgrade" of the rendering engine by the attacker can also reactivate vulnerabilities (above all, for XSS) that have only been fixed in newer versions of Internet Explorer.

The “X-Frame-Options” HTTP header prevents a page from 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 is located.

The HTTP header "Content-Security-Policy" can also be used to prevent the content from being displayed in a frame. If the directive "frame-ancestors 'none' " is specified, the content is generally prevented from being displayed in a frame. The directive "frame-ancestors 'self' " restricts the display in frames to pages within the same domain where the web application is located. However, since one or more domains (including wildcards) can also be specified in the directive, which can then display the web application within a frame, this variant is more flexible than the "X-Frame-Options" HTTP header. However, the directive is not yet supported by all popular browsers.

JavaScript code might also be used to check whether a page containing the code is actually the top page in the frame hierarchy, to then prevent the display within a frame. Unfortunately, various techniques are known which get past the common variants of these JavaScript mechanisms.

ID: 3.22-281/i373

Req 282	Authorizations for cross-domain access must be defined restrictively.

The web application may allow solely such domains 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. Intranet applications must not grant authorizations for cross-domain access to domains that do not belong to the intranet.

Cross-Origin Resource Sharing (CORS) is a mechanism to allow exceptions to the rules of the Same Origin Policy (SOP) and to implement communication between web applications via the user's browser. CORS enables the browser to initiate cross-origin requests and to allow the response to be read. These requests are made, for example, by means of XMLHttpRequests (XHR). The mechanism is based on the exchange of header information. Browsers send the "Origin" header, which indicates the domain of the application that sends the request. Servers in turn send back the "Access-Control-Allow-Origin" header to express which domain is allowed to access their resources. This is then ensured by the browsers accordingly.
Authorization for such cross-domain access must therefore be defined restrictively without the use of wildcards in the "Access-Control-Allow-Origin" header. If the value of the "Access-Control-Allow-Origin" header is generated dynamically (possibly depending on the respective "Origin" header of the request), the "Origin" header must be validated, permitted domains must be defined restrictively and additionally the "Vary: Origin" header must be set in order to prevent attacks based on cache poisoning. The headers may also only be sent by the server for selected URLs/resources for which cross-domain access is intended. They may not be sent across the board for the entire application.

The "window.PostMessage"-API expands every window and every frame with a new method that allows text messages to be sent from one window to another. In that case, 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 (for example, using "eval"). To avoid DOM-based XSS attacks, the DOM of a page must not be modified based on the exchanged data (for example, by means of “innerHTML”).

The web application must not use any RIA services that circumvent the SOP. This includes AJAX service bridges, HTML bridges, AJAX proxies and aggregate sites. Workarounds and deprecated functions must also not be used to enable cross-domain access. This applies, in particular, to fragment identifiers (the part of a URL after the hash symbol), to JSONP (JSON with padding) and, as well, to the property "document.domain", by which a web application could change its origin (to a parent domain). Generally, standardized technologies should be used instead of proprietary technologies or workarounds.

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

Motivation: The 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. A restrictive definition of authorizations which may be necessary can reduce the associated risk. If cross-domain access is granted too freely, it may even be possible to access data with need of protection that is otherwise protected by authentication. This is because when cross-domain requests are sent, browsers send these requests together with the cookies for the domain of the web application to which these requests go. Their responses can be evaluated on the client side and content with need of protection can possibly be accessed. And also 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.

ID: 3.22-282/i373

Req 283	If a password is used as an authentication attribute, the registration process for the web application must ensure that the user chooses his password on his own: Either the user sets his password during registration, or he receives an individual initial password 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 visu-

alize, which criteria of the password policy are met by the password.

Motivation: Predefined passwords are frequently not treated with care by the users, possibly even noted down, as they are usually not both secure and easy to remember. In addition, 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

ID: 3.22-283/i373

Req 284	If a password is used as an authentication attribute, the web application must have a function that is protected against misuse, by means of which a user can reset the password.

Misuse that leads to the existing secure authentication mechanisms being bypassed via this function must be prevented.
For this, we recommend that users must select a “security question" and enter the correct answer (for critical applications possibly several questions and answers). The web application must request this data as part of the registration process or after the user's initial login.
If the user has forgotten his password, there is an appropriate function to select when logging in. Once the user has answered the relevant security question correctly, the password is reset. The web application creates a new initial password or a link with an activation token. It can send the initial password or activation token to the user via push notification or SMS, alternatively via e-mail, to predefined contact information. This function must in no way allow the user to successfully log in to the web application or to directly change his password, just by answering the security question.
It must also be prevented that this function offers an attacker the possibility to automatically determine valid usernames. For this purpose, CAPTCHAs can be queried or threshold values for invalid entries/attempts can be provided, which lead to a (temporary) blocking of this functionality (for example, for an IP address). In the case that a user has also forgotten the answer to his security question, a fallback mechanism can also be provided, if required. For this, the user is offered a further process, but which can usually not be performed automatically, for example calling a hotline. Also, for such a fallback mechanism, it is important to ensure that it cannot be misused.

Motivation: If a user has forgotten his 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 accounts. It also should not be possible for a password reset process to be (automatically) misused, for example, by resetting passwords en mass and bothering other users.

ID: 3.22-284/i373

Req 285	If initial passwords or activation tokens are used, these must be protected against malicious use and brute force attacks.

In particular, they must lose their validity after a suitable period of time. The precise validity period must be set individually for each application. It depends on the application's sensitivity and purpose, as well as on the password assignment procedure. If, for example, in case of a web application accessible from the Internet, the user starts the registration on his own and initial passwords (or activation tokens) are sent electronically, they must lose their validity within a few hours after they are sent. In case of an internal application and transport via internal networks, this period can be significantly longer (up to several days).

Initial passwords must comply with the same requirements regarding complexity like passwords chosen by the users.

The length of activation tokens must be at least 120 bit (like the length of session identifiers). All the relevant characters must be generated at random.

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.
Without any protection mechanism an attacker can possibly determine an initial password or an activation token by

automated creation of character combinations.

ID: 3.22-285/i373

Req 286	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.

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 general error message makes it more difficult for attackers to find valid usernames.

ID: 3.22-286/i373

2.15.1. Content-Management-Systeme (CMS)

Req 287	Access to the editing environment of the content management system (CMS) 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, such as 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, for example, by using two-factor authentication.

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

ID: 3.22-287/i373

Req 288	The CMS 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).

ID: 3.22-288/i373

Req 289	The CMS 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.

ID: 3.22-289/i373

Req 290	If the CMS provides functions for creating content that is to be published later, 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 edited without being viewed in advance by web application users.

ID: 3.22-290/i373

Req 291	The CMS 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 web application 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 XSS attacks could be created.

ID: 3.22-291/i373

Req 292	Preview functionalities of the CMS 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.

ID: 3.22-292/i373

2.16. Proxy-Server

Req 293	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. Access may only take place from and via a network that is fully under Deutsche Telekom Group control.

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

ID: 3.22-293/i373

Req 294	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 important that all components involved in data exchange observe the specified caching behavior. For this requirement the following threats are relevant

ID: 3.22-294/i373

Req 295	A procedure must be configured on outbound proxy servers, to block undesirable targets or content.

A procedure must be configured on outbound proxy servers which can be used to block access to undesired target addresses or undesired content (filtering).

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.

ID: 3.22-295/i373

Req 296	In the case of M2M connections via outbound proxy servers, access to non-required URLs must be blocked.

In the case of M2M connections via outbound proxy servers, access to non-required URLs must be blocked.

Motivation: Limiting callable URLs reduces the risk of downloading malware via this connection and reduces the overall attack surface

Implementation example: 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

ID: 3.22-296/i373

Req 297	The HTTP method "CONNECT" is not allowed.

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

ID: 3.22-297/i373

Req 298	Outbound proxy servers must not disclose the client’s network address outside the intranet.

Outbound proxy servers must not disclose the client’s network address outside the intranet.

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.

ID: 3.22-298/i373

Req 299	The proxy server must record information on each connection in a log file.

The proxy server must record information on each connection in a log file.

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

Implementation example: 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

ID: 3.22-299/i373

Req 300	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

ID: 3.22-300/i373

Req 301	An inbound network address (IP address) and a port may only allow access to the relevant applications

An inbound network address (IP address) and a port may only allow access to the relevant applications.

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

ID: 3.22-301/i373

Req 302	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 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.

Implementation example: It is recommended, that the proxy server responds to requests containing non-required methods with HTTP status code 405 (Method Not Allowed).

ID: 3.22-302/i373

Req 303	All inbound network connections must be terminated on the reverse proxy server.

All incoming network connections must be terminated on the reverse proxy server. A reverse proxy server must reject incoming 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.

ID: 3.22-303/i373

Req 304	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.

ID: 3.22-304/i373

Req 305	A reverse proxy server must check the URL of incoming requests for conformity to a specified pattern and reject them if necessary.

A reverse proxy server must check the URL of incoming requests for conformity to a specified pattern. The pattern depends on the application addressed via the reverse proxy server and therefore usually has to be defined individually. Unsuitable patterns must be rejected.

Motivation: URLs that do not conform 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 give an attacker unauthorized access to the application.

Implementation example: Requests for which the URL does not match the pattern, especially URLs contains invalid characters, should be rejected by the proxy server.

ID: 3.22-305/i373

Req 306	Connections over outbound proxy servers must be authenticated and authorized.

The identity of a user or a machine must be clearly established. The authentication of a machine can take place via the network address (IP), a directory service or a certificate. It must be checked whether the user or the machine is authorized to access the target resource.

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.

ID: 3.22-306/i373

2.17. Sap-Fiori Application Server

Req 307	It must be ensured, that the network separation between the SAP Netweaver-Server and the UI5 / Fiori-App is performed.

It must be ensured that the network separation between the SAP Netweaver server and the UI5 /Fiori apps is performed.

Motivation: Protection of the SAP application against unauthorized access.

Implementation example: Fiori-Apps always needs a connection to a Netweaver-Server. Reglary it is built up within the DMZ. As an alternative to the Netweaver server for Fiori apps a gateway can be used.

ID: 3.22-307/i373

2.18. SAP Fiori Apps

Guidelines for the development of SAP Fiori/UI5 Applications and other client-apps.

Req 308	When using Fiori applications from the SAP Cloud Platform, the SAP Cloud connector must always be used.

By usage of Fiori apps form the SAP Cloud Platform it is always required to use the central SAP Cloud connector (comparable to the AWS landing zone).

Motivation: Without a separation gateway as the SAP Cloud connector or a proxy for the appropriate protocol the access is not allowed.

Implementation example: When accessing Fiori apps from the SAP cloud, the SAP Cloud connector must be used.

ID: 3.22-308/i373

Req 309	By using Fiori a personalized login must be performed.

By usage of Fiori one or even several rolls must exist on the backend system. To use the application, each user must authorize himself with his own ID.

Motivation: A distinct identification ensures the tracebility of changes on the system.

Implementation example: Login via user name/password or MyCard.

ID: 3.22-309/i373

Req 310	By developing Fiori apps it must be ensured that there is no mixing of different web technologies.

By developing Fiori apps it must be ensured that there is no mixing of different web technologies.

Motivation: Protection of the system against different technologies and their unique features.

Implementation example: SAP delivers two versions of version Development Toolkit: SAPUI5 and the open source variant OpenUI5.
SAP has developed the JavaScript framework OpenUI5 under an Apache 2.0 license to create operating system independent business applications. Since OpenUI5 is based on CSS3, HTML5 and the JavaScript API of ECMA-Script 5 (ES5), only browsers with HTML5 capabilities are supported. With iOS-, Android-, MacOS- and Windows, the relevant platforms are included. Apps developed on the client UI technology OpenUI5 run in the browser and thus on any terminal device. When the user accesses an app, a request is sent to the respective server to load the application into the browser.

ID: 3.22-310/i373

Req 311	By development of a Fiori-App a framework must be used.

By development of a Fiori-App a integrated development environment must be used.

Motivation: The usage of a development environment helps to get a minimum level of quality and security. Eclipse is used to develop various programs (especially for Java). For SAP Fori applications the native runtime container from the AppStores should be used.

Implementation example: SAP delivers two versions of version Development Toolkit: SAPUI5 and the open source variant OpenUI5.
SAP has developed the JavaScript framework OpenUI5 under an Apache 2.0 license to create operating system independent business applications. Since OpenUI5 is based on CSS3, HTML5 and the JavaScript API of ECMA-Script 5 (ES5), only browsers with HTML5 capabilities are supported. With iOS-, Android-, MacOS- and Windows, the relevant platforms are included. Apps developed on the client UI technology OpenUI5 run in the browser and thus on any ter-

minal device. When the user accesses an app, a request is sent to the respective server to load the application into the browser.

ID: 3.22-311/i373

Req 312	The exchange of data (e. g. OData, XML) must be performed encrypted.

The exchange of data between the application and the backend especially by usage from ODATA and XML must be performed via an encrypted connection.

Motivation: For example, by using the Open Data (OData) protocol the data requests and data are transmitted between the application and the backend. With a transport encryption the data are protected against the access of unauthorized people.

ID: 3.22-312/i373

Req 313	It must be avoided that data is retrieved from the server via JSON or XML.

It must be avoided to retrieve data via JSON or XML from the server. The protocols can only retrieve complete data sets from the server and therefore do not scale as good as OData.

Motivation: In order to build perfonmant apps it makes sense to load only the data which are needed. In contrast JSON or XML read the complete data set.

ID: 3.22-313/i373

Req 314	By using Fiori apps it must be ensured that known web attacks are intercepted by appropriate countermeasures.

By using Fiori apps it must be ensured that known web attacks are intercepted by appropriate countermeasures.

Motivation: SAP Fiori is a web application written in HTML and therefore it is just as vulnerable for common attacks as classic web applications. To avoid unauthorized access SAP Fiori must be hardened.

Implementation example: Cross-Site Scripting
Clickjacking framing protection

ID: 3.22-314/i373

Req 315	No "offline app's" may be used.

Offline apps are apps that save data on the mobile device. If no connection to the server exists, the app is able to represent the data offline. If the data has a protection class "internal" or higher, no "offline Apps" may be used.

Motivation: Special attention must be paid to data with a protection class "internally" or higher that they are not stored at the device. Stored data can be read by third parties if the device got lost.

ID: 3.22-315/i373

2.19. using Cloud

Req 316	Clear exit strategy must exist before onboarding a project or a system on a public cloud / SAP Cloud Service.

Despite good intentions and contracts, it is possible that the CSP will not be able to fulfill the needs of DTAG and respective systems in the future. In order to account for potential legal, financial or technical problems which may arise in the future, it is necessary to have a clear plan how to get services and the data out of a specific cloud provider and at any given time.

In the traditional world, this problem was solved by dual (or multi) vendor strategy for avoiding vendor lock-ins, but the transition from vendor to vendor was possible because equipment was bought and owned by DTAG. This is not the case when using public cloud providers. Having 2 or more CSPs at disposal is not sufficient for dual (or multi) vendor strategy, it is also necessary to move impacted systems and data as well, either from the CSP to another CSP or from the CSP to own premises.

If possible, cloud consumer should consider using more generic cloud services, so that transition to the public cloud from another CSP (or into own premises) is done with minimal modifications (or no modifications in best case scenario).

Contracts and court proceedings cannot be considered as a viable replacement for an exit strategy.

Motivation: With the use of the cloud, one enters into a dependency. The exit strategy is to try not to become completely dependent.

ID: 3.22-316/i373

Req 317	Encryption of all data in transit, at rest or during processing (during processing if available) must be done according to the need for data protection.

In a public cloud environment, all stored data (at rest) and all non-open data in transit must be encrypted to prevent unauthorized access to data, since a public cloud is operated by an external company and it is a multi-tenant environment. Mandatory encryption applies to consumption of cloud APIs and to all services running inside cloud (e.g. communication between virtual machines).

Upcoming technology, like homomorphic encryption or VM memory encryption, enables encryption of data during processing, but it is still not widely available. If such technology is available, it must be used. Availability and use of such technology may expand number of use cases for applications running in the cloud.

Data encryption must be used for all IaaS services and for PaaS services whenever possible. SaaS services typically do not offer direct encryption due to their nature but offer configuration options for underlying encryption which must be used in a reasonable way (i.e. encryption must be turned on).

It is not necessary to encrypt open data (e.g. while processing raw internet traffic). Although it is a good practice, for performance reasons it is permitted to omit encryption between different services inside one unit which is fully under DTAG's control, e.g.:

- between containers inside a virtual machine if they always act as a whole unit and there is no chance that one of them will land in another VM
- between processes inside a single VM or a single container

If available encryption methods and key management models allow it, encryption of operating system boot disk for virtual machines is also necessary.

Data encryption allows easier deletion of data, e.g. necessary for GDPR (General Data Protection Regulation) compliance, by simple deletion of the key used for encryption of data at rest.

In case a system, data storage or a communication path contains a mix of open and non-open (internal, confidential) data, it is recommended to encrypt everything.

Motivation: Protection against third parties viewing the data. This can be caused by software errors or faulty configurations.

ID: 3.22-317/i373

Req 318	Process for handling abuse and incident notifications from and to the cloud service provider must be established.

Cloud service providers typically notify customers on abuse and incidents which they detect, and sometimes even on identified irregularities (e.g. huge jump in resource consumption). The landing zone operators need to, in cooperation with responsible SOC, establish a process which enables handling of such notifications 24/7.

In the same way, it needs to be possible for a cloud user or landing zone operator to report incidents and abuse to the cloud service provider, which can be organized through the landing zone operators in agreement with responsible SOC.

ID: 3.22-318/i373

Req 319	IAM solution of DTAG must be used for user authentication if supported by CSP. If not, automation for managing identities in the cloud must be in place.

Double administration of users brings issues like untimely creation and deletion of accounts (which can lead to data leaks), so it is necessary that accounts in the cloud are mapped to accounts in own DTAG's IAM. In terms of technologies - Active Directory identity federation, AFDS Shared (AAD expected in near future), SAML 2.0, OpenID+OAuth (2.0) and LDAPS (inside VPN connection only) are viable options.

In an unlikely case of not being able to integrate DTAG's own IAM solution for managing identities in the cloud, it is necessary to automate account management. Such automation for management of identities has to fulfill the "IAM (Identity Access Management) - Framework" (3.69.) requirements 01-57 and 65.

Preferred solution would be session based since no employee personal data would be exchanged with the cloud provider.

It is worth mentioning that this requirement also applies to applications running in the cloud, however, that is already covered with other security requirements (for operating systems and applications).

ID: 3.22-319/i373

Req 320	Dedicated physical or dedicated virtual connection must be used for connectivity between application in the public cloud and on-premise resources of Deutsche Telekom, or between applications hosted in different public clouds. Internet access is allowed only for workloads which are both cloud and Internet native, but they must be explicitly approved by the security department.

Dedicated physical connections are typically fiber (or copper) cables connected between CSP and DTAG premises. Dedicated virtual connections are VPN connections between a VPN endpoint inside DTAG premises and VPN endpoint in the public cloud. In case of connectivity between two cloud providers, VPN is also needed.

An application running in cloud which requires often or permanent connectivity (one time data transfer only may utilize other means if suitable) to/from DTAG resources, whether running in DTAG own premises or in another public cloud, needs to utilize VPN set up between public cloud where the application runs and DTAG premises (or another public cloud). It is not meant that each application should utilize a separate VPN connection, but that applications should use same connection which is set up once between cloud provider and DTAG for all resources belonging to DTAG in that CSP.

Preferred methods are in the following order:

1. dedicated physical connection (typically more reliable if used exclusively)
2. dedicated virtual connection
3. VPN over Internet (less reliable)
4. (restricted) plain Internet connectivity; approval from PSM necessary

VPN connection which uses shared backbone network with other customers or Internet is typically less reliable in case of congestions (e.g. due to DDoS attack) and that can affect reachability of services.

Plain Internet access is allowed only for fully cloud and Internet native workloads and it is necessary to restrict address ranges allowed for communication to minimum which is possible. This method is mainly reserved for trials, tests or clouds where DTAG has extremely small footprint.

ID: 3.22-320/i373

Req 321	Connections between public clouds and resources and networks inside DTAG premises must be protected by an independent security device, located inside DTAG premises and under DTAG control. A service running in the cloud must not have direct access to Internet and internal DTAG networks at the same time (the functionality must be split over multiple VMs, containers or in general - multiple tiers).

Connections between public cloud and internal networks of DTAG (e.g. CN-DTAG, Intranet) must be protected with a security device which needs to, depending on specific use case, provide:

- stateful firewalling (as bare minimum)
- web application firewall
- application level gateway
- API gateway
- DoS/DDoS protection
- ...

The security device in this context may contain out of several physical and virtual devices acting as a single unit. Protection provided by the device must be in general bidirectional - mainly protecting DTAG's networks from threats from the public cloud, but also protecting resources inside public cloud from malware and malicious insiders inside DTAG's networks. Since it is expected that applications hosted in the public cloud will utilize security groups and other security features offered by the public cloud itself, controls necessary on the security device do not to be very detailed for the connectivity from DTAG internal network towards the network in the public cloud. For the other direction, protection of DTAG's internal networks from threats from the public cloud, implementation must be done fully on the security device.

For HTTP(S) based services, the provided device should protect against OWASP Top 10 web vulnerabilities at minimum.

The configuration of the device should be automated in a way so that it can cope with rapid provisioning cycles typically found in highly automated cloud environments (e.g. it should be handled so that it can cope with dynamic IP address assignments in cloud).

Internal networks of DTAG are not built in modern way with microsegmentation in mind, and applications deployed in the cloud need to follow guidelines for connectivity to existing DTAG internal networks if they need such connectivity. Applications in the cloud are not allowed to have direct Internet connections and connection to DTAG internal networks at the same time because there would be no (at least) 2 layers of security protection in that case. This would allow attackers much easier infiltration into DTAG network and resources through a backdoor placed, intentionally or unintentionally, in an application running in the cloud. For special use cases, e.g. management and monitoring of an Internet facing application running in the cloud, an appropriate solution has to be determined on case-by-case basis.

ID: 3.22-321/i373

Req 322	Resource separation must be performed by using virtual private clouds (VPC). Network communication of an application/service in the cloud must be restricted by security groups. Advanced network protection mechanisms offered by CSP must be use

Virtual private cloud is a concept for separation (primarily network separation) of resources used by a system (or a group of systems) in the public cloud from other resources in the public cloud. It is implemented through set of policies which define connectivity and reachability of resources, even throughout different data centers, availability zones or regions of the public cloud. To achieve easy and effective implementation of basic network perimeter, one must use VPC(s). Using VPC can reduce the need for having separate VPN connections to each data center of CSP and can be used to set appropriate routing tables for controlling network traffic. VPC concept also allows cloud consumers to bring their own IP address ranges for resources used in that cloud.

Security groups in cloud are, in addition to VPC, cornerstone of limiting possible communication paths in the cloud. As for any other firewall which be found in traditional deployments, security groups must be made as restrictive as possible to allow only necessary communication paths.

In addition to security groups, cloud providers typically offer DoS, DDoS protection, rate limiting and other network protection mechanisms. Such mechanisms must be used, especially for Internet facing services running in the cloud. Depending on the cloud provider and offered advanced protection mechanisms, additional detection and/or protection mechanisms might be necessary.

Certain cloud providers do not use the term VPC but offer setting it up through means of virtual networks.

Motivation: Compliance with the separation between the applications

ID: 3.22-322/i373

Req 323	Test, reference and production systems must be separated from each other in terms of network technology. Within the network unit, separation must take place via tenants.

Applications must be (logically) separated from each other. Separation is also required for different network technical units.

This approach enables a mandatory separation of duties between the systems and prevents unwanted access to data and systems.

In addition to separating accounts, different tenants / projects must be protected according to the same rules. It is understandable that production systems need the highest protection requirements, but test and development systems must be protected equally.

Motivation: The need-to-know principle must be followed.
Access between the network units must be prevented.
For example, the access data of these systems may not be saved in plain text on GIT repositories.

ID: 3.22-323/i373

Req 324	Applications must be, as always, built in multitier models.

This requirement is well known best security practice already, but one should be reminded that this approach is still valid for usage of public clouds (or clouds in general).

Traditional applications, most likely during transition period, still need to have usual network separation between tiers, which is safeguarded by firewalls. This requires building a network structure in public cloud which mimics traditional network separation and placing appropriate security controls between tiers. This is typically done by using virtual networks and corresponding security groups.

Cloud native applications need to follow microsegmentation approach for achieving separation between tiers, which typically results in a flatter network with each component safeguarded with security groups and other policies which are available in cloud. Security requirements from other documents still apply.

Example taken from current version of Container Security Requirements:

- Containers that are assigned to different security zones (e.g. DMZ-Zone, Application/Database-Zone) must be deployed on different hosts/host pools. A pool refers to a group of at least one or more VMs. In case of using Container as a Service (CaaS) platforms, these pools can be shared for several applications. For critical applications dedicated hosts/host pools must be used. If the only exposed service is an ingress controlled (e.g. Nginx) hardened regarding Security Guidelines you can implement this by using only container specific measures e.g. Pod/Network Security Policies.

ID: 3.22-324/i373

2.20. SAP Cloud Platform as PaaS

Req 325	If the sap cloud platform is used without connections being used in local Telekom data centers (e.g. beers), the cert interface must be integrated as if the security framework were being used.

If the cloud application has no connection to Hitnet via the landing zone, the application must provide the appropriate interfaces and connections.

Motivation: A cloud application must be operated in the same way as an on-premise installation. It must be ensured that information such as accesses (especially faulty accesses) are included in global monitoring.

ID: 3.22-325/i373

Req 326	When using the SAP cloud platform, the service used must take place in an SAP data center operated in Germany (e.g. St.Leon-Rot).

When using the Sap Cloud platform, it must be ensured that the processing instance is located in Europe or Germany. It must also be ensured that backup and other services are produced also in Europe. In addition, Telekom processes must be followed (BIA = Business Impact Analysis, TIA = Transfer Impact Assessment, and an existing SOC2 certificate from the provider for the service must be in place).

Motivation: Ensure compliance with legal requirements.

ID: 3.22-326/i373

Req 327	Every interface that runs over the security framework must be documented.

At least the following points must be documented:

- Source /Sink/
- Protocol /Type of encryption/
- Framework in the SCP/
- Data protection requirements/ if required ->
- Created users in the SCP/ Type of authorization.

Motivation: It must be traceable all times, which systems communicate with the cloud in which way and via which users.

ID: 3.22-327/i373

Req 328	When using the SAP cloud platform, at least the following must be ensured: database encryption, separation of the different environments, encrypted communication between applications and cloud applications themselves.

When using the SAP cloud platform, it must be ensured that database fields worthy of protection are encrypted, that different environments (test, acceptance, production, ...) are separated from each other (at least one client, a different environment would be better) and that communication to and from the cloud via secure protocols (e.g. https).

Motivation: Data in the cloud must be technically secure. Depending on the need for data protection, encryption is necessary to protect itself as much as possible from the provider and operating personnel. Even if database encryption is used, it is not possible to protect the data from a person who has physical access to the system.

ID: 3.22-328/i373

2.20.1. Specification Cloud Connector (only if a SaaS in the SAP Cloud is used)

Req 329	The consistency of roles and permissions for cloud, on-premise and mobile apps must be ensured.

When using different components (on-premise, cloud, app via cloud, ...) the user permissions on all systems must match the original permission. When accessing an app, it must be ensured that the user has the same permissions in the app as on the local system.

Motivation: It must be ensured that users are not assigned more rights than the highest rights they have for the system

ID: 3.22-329/i373

Req 330	The cloud connector must not be replaced by a proxy or reverse proxy.

The cloud connector must not be replaced by a reverse proxy. In addition to the HTTP protocol, the cloud connector also provides other functions that allow you to use other SAP-typical protocols between the worlds.

Motivation: The cloud connector can establish a VPN tunnel to the SCP. If it does not do so, it must be ensured that the system protects all connections (HTTP, RFC, TCP, UDP) due to avoid an unauthorized access.

ID: 3.22-330/i373

Req 331	The Cloud Connector must be operated in such a way that an encrypted connection is always established.

The Cloud Connector must be operated in such a way that a secure connection is always established. If a VPN tunnel is used, the way it is set up must prevent the systems from being accessed natively and unprotected from the SCP platform

Motivation: Data protection if used Cloud-Services

ID: 3.22-331/i373

Req 332	For applications in the cloud it is not allowed to use selfsigned certificates.

The SCP cloud accepts the Telekom CA as a trusted provider, and both partners must be able to verify at any time whether the certificate used is valid and valid for any type of communication. The use of self-signed certificates is not acceptable for any type of communication.

Motivation: For each client it must be verifiable whether it is a trustworthy system.

Implementation example: Use the Telekom CA

ID: 3.22-332/i373

Req 333	The SAP Cloud Connector must be used. He is responsible for logging and forwarding all relevant data to the Securtiy Devense Center.

The cloud connector is responsible for logging and forwarding all relevant log data from the cloud applications. The data must be made available to operations and the Security Defense Center.

Motivation: Regardless of the location of an application, the Log- data for operation and Cyber-Defense-Center must be made available.

ID: 3.22-333/i373

2.21. Microsoft Azure Cloud via DTIT Landing Zone and product "Enterprise"

The security requirements within this chapter are only valid for the product "Enterprise" of the DTIT Landing Zone. With the product "Enterprise" the DTIT Landing Zone is used as network separation and connection between CN-DTAG (Hitnet) and the Azure Cloud with specified protection mechanism.

Req 334	SAP systems, which are operated in the Azure Cloud, must only be accessible via "hitnet" (CN-DTAG).

Access to the SAP-systems, which are operated in the Azure Cloud (product Enterprise) is only possible over the DTIT Landing Zone with their security requirements (e.g. monitoring, protection, secure connections, authentication) from the hitnet, e. g. TPAM tool: https://yam-united.telekom.com/pages/t-pam/apps/content/home.

Access to the applicaiton within the Azure Cloud from clients via internet is not allowed.

Motivation: The SAP system must be protected for unauthorized access.

ID: 3.22-334/i373

Req 335	Applications from the different Azure products "Discovery", "Voyager" and "Enterprise" are not allowed to communicate to each other.

We differentiate in Azure products "Enterprise", "Voyager" and "Discovery" for the DTIT Landing Zone. A communication between the different Azure products is not allowed. In the different Azure products a different protection of the applications in terms of patching, monitoring, interface access, ... is set up.

Motivation: An unauthorized access to the application must be avoided.

ID: 3.22-335/i373

Req 336	All SAP systems which belonging to a transport domain are within one subscription. The environments DEV, QAS and PRD are separated by network security groups (NSGs).

In a project (SAP system landscape), all systems (development/acceptance/production) that belong to a transport domain are within one subscription. The separation of the environments is done by NSGs.

This is an exception rule for the Azure Cloud compared to the NON-SAP systems, as it is not possible to comply with SAP's mode of operation via the transport directory (no shares across the boundaries of subscriptions are technically feasible).

Motivation: Unauthorized Access over system borders must be prevented. Separation over NSGs the access is controlled.

ID: 3.22-336/i373

Req 337	For a SAP system landscape (SAP project) within the Azure Cloud only specified Azure services may be used.

Azure services may not be used within an SAP system landscape (SAP project) of Azure, as these cannot be restricted via the NSG in some cases.

Exception for usable Azure Services:
Virtual Machine
Storage Account
Loadbalalancer
Key Vault
Azure Automation Account
Azure Firewall

Motivation: In the Azure Services there are many services that require connections to the internet. In contrast to the subscription, these cannot be configured in the NSG so that the protection of the application is guaranteed.

Furthermore these services are using authentication data, which are central managed in Azure.

ID: 3.22-337/i373

Req 338	For SAP systems in the Azure Cloud, a SAProuter for the connection to the CNDTAG must be set up.

For SAP systems in the Azure Cloud, a SAProuter for the connection to the CNDTAG must be set up. As a counterpart, a central system is available in the CNDTAG network. The SAProuter is only allowed to route the SAP protocols DIAG, RFC, and SNC.

Motivation: The SAProuter is used to ensure SAP-specific communication to the Hitnet.

ID: 3.22-338/i373

Req 339	For SAP systems in the Azure Cloud, a SAP Web Dispatcher for the connection to the CNDTAG must be set up.

For SAP systems in the Azure Cloud, a SAP Web Dispatcher (or something equal) for the connection to the CNDTAG must be set up. As a counterpart, a central system is available in the CNDTAG network. The SAP Web Dispatcher is only allowed to route the SAP protocols http and https.

Motivation: The SAP Web Dispatcher is used to ensure web communication to the Hitnet.

ID: 3.22-339/i373

Req 340	Application access via DIAG, RFC and SNC protocol must be lead through the central proxy "SAPaaS Router Service".

Application accesses from the CN-DTAG (Hitnet) to SAP systems located in the Azure Cloud product "Enterprise" must conduct their communication via the central, upstream SAProuters of the "SAPaaS Router Service".

According to the protocol to be transferred (for SAP must be used a suitable system):
RFC and SNC => SAProuter

Motivation: The SAProuter in particular is an important component for the SAP-specific protocol RFC and SNC. In many cases it is also used as a net-separating element.

Via the central, upstream SAProuters of the "SAPaaS Router Service" a secure connection between OnPremise and Azure cloud world with the product "Enterprise" established.

ID: 3.22-340/i373

Req 341	Application access via the http-protocol must be lead through the central proxy "SAPaaS Proxy Service".

Application accesses from the CN-DTAG (Hitnet) to SAP systems located in the Azure Cloud product "Enterprise" must conduct their communication via the central, upstream SAP Web Dispatchers of the "SAPaaS Proxy Service".

According to the protocol to be transferred (for SAP must be used a suitable system):
SAP-http, HTTPS, API => SAP Web Dispatcher

Motivation: The SAP Web Dispatcher in particular is an important component for the http and https traffic. In many cases it is also used as a net-separating element.

Via the central, upstream SAP Web Dispatchers of the "SAPaaS Proxy Service" a secure connection between On-Premise and Azure cloud world with the product "Enterprise" established.

ID: 3.22-341/i373

Req 342	Application traffic between subscriptions must always be routed via the network separating central element ("SAPaaS Router Service" or "SAPaaS Proxy Service").

Application traffic between subscriptions must always be routed via the network-separating central element within the Cloud ("SAPaaS Router Service" or "SAPaaS Proxy Service"). Firewalls and Network Security Groups are considered network-separating elements in the Azure Cloud too.

Motivation: In order to create a network separation between the different environments, all connections must run via suitable proxies or routers. Furthermore, we ensure via the proxies and routers that the communication is encrypted at least on an application basis.

ID: 3.22-342/i373

Req 343	All application connections must be encrypted at the application level.

End-to-end encryption at application level must be implemented (at application level via SNC or HTTPS).

Motivation: Protection of credentials and data at application level.

ID: 3.22-343/i373

Req 344	All connections within the network security group (NSG) must be encrypted.

In order to secure the data against third parties, communication between the applications must also be encrypted within an NSG.

Motivation: Secure communication in terms of end-to-end visibility and thus the lowest possible risk to third parties (e.g. cloud operators).

ID: 3.22-344/i373

Req 345	A backup of the data outside the cloud must exist.

If the data is only stored in the cloud, it must be ensured that an acceptable data set (data volume, actuality) is also stored outside the cloud.

Motivation: If the cloud can no longer be used, it must be agreed that there is a possibility to obtain a backup with data independently of the cloud. Which time window is used, or how much data may be lost if the cloud can no longer be used for a long time, must be determined in each individual case.

ID: 3.22-345/i373

Req 346	The access from job steering tool Automic Automation (UC4) for applications within the cloud must be secured.

When using Automic Automation (UC4) a separate tenant/client must be built up for the cloud application, so that a separation between the OnPrem world and the cloud world is done.

Furthermore only encrypted connections can be used.

Motivation: When Automic Automation (UC4) is used, the communication must be based on encrypted protocols and authentication.

Unauthorized access by high privileged Automic Automation (UC4) users must be avoided by using encrypted protocols and secure athentication.

ID: 3.22-346/i373

2.21.1. Central Management Subscription "SAPaaS central Management Service"

Req 347	In Azure Cloud for the product "Enterprise" of the DTIT Landing Zone only one central management subscription for the administrative access on the SAP applications in the subscriptions exists, the "SAPaaS central Management Service".

All SAP Azure applications are administrated from these central management subscription "SAPaaS central Management Service".
A 2FA is required.

Motivation: Unauthorized privileged access must be avoided.

ID: 3.22-347/i373

Req 348	For administrator access Cyberark / T-PAM must be used.

For the administrator access (on customer projects/systems) via "SAPaaS central Management Service" Cyberark / T-PAM Services (central bereitgestellt) must be used and the key rotation must be activated.

Motivation: Due to Cyberark/T-PAM tool no personalized admin account in SAP is necessary.

ID: 3.22-348/i373

Req 349	No user application access may take place in the central management subscription "SAPaaS central Management Service".

This central management subscription "SAPaaS central Management Service" is only to be used for operational purposes.

Motivation: Enforce a secure and uniform application administration way for SAP system landscapes.

ID: 3.22-349/i373

Req 350	Only spezified Azure services may be used in the central management subscription "SAPaaS central Management Service".

Only spezified Azure services may be used in the central management subscription "SAPaaS central Management Service", as these cannot be restricted via the NSG in some cases.

Exception for usable Azure Services:
Virtual Machine
Storage Account
Loadbalalancer
Key Vault
Azure Automation Account
Azure Firewall

Motivation: From the Azure Services there are services that require connections to the internet.

Furthermore these services are using authentication data, which are central managed in Azure.

ID: 3.22-350/i373

2.22. Google Cloud via DTIT Landing Zone with product "SAP on GCP"

The security requirements within this chapter are only valid for the product "SAP on GCP" of the DTIT Landing Zone. With the product "SAP on GCP" the DTIT Landing Zone is used as network separation and connection between CN-DTAG (Hitnet) and the Google Cloud with specified protection mechanism.

Req 351	SAP systems, which are operated in the Google Cloud, must only be accessible via "hitnet" (CN-DTAG).

Access to the SAP-systems, which are operated in the Google Cloud (product SAP on GCP) is only possible over the DTIT Landing Zone with their security requirements (e.g. monitoring, protection, secure connections, authentication) from the hitnet, e. g. TPAM tool: https://yam-united.telekom.com/pages/t-pam/apps/content/home.

Access to the applicaiton within the Google Cloud from clients via internet is not allowed.

Motivation: The SAP system must be protected for unauthorized access.

ID: 3.22-351/i373

Req 352	Applications from the different Google Cloud folders (managed and unmanaged) are not allowed to communicate with each other.

In the Google Cloud, a distinction is made between managed and unmanaged folders. Communication between the different Google Cloud Folders (managed and unmanaged) is not allowed.

Motivation: Unauthorized access to the application in the managed folder must be prevented. Projects in the managed folder have a connection to the CNDTAG and associated compliance rules are implemented.

ID: 3.22-352/i373

Req 353	SAP applications must be deployed via COSMOS in the Google Cloud via the "SAP on GCP" product.

SAP applications must be deployed via COSMOS in the Google Cloud via the "SAP on GCP" product. COSMOS (Cloud Orchestration SAP Managed Operation Service) is the orchestration tool for SAP managed services for SAP application landscapes in the Google Cloud via product "SAP on GCP".

The transport directory is implemented via COSMOS.

Provision of SAP systems via Cloud Shepherd is not permitted and not possible.

Motivation: Automated provision and implementation of security requirements increase efficiency and security levels.

Implementation example: https://cosmos.sap.t-systems.com

ID: 3.22-353/i373

Req 354	No additional preapproved services may be used in SAP on GCP.

No additional preapproved services may be used in SAP on GCP.

Motivation: Safe operation of the application must be ensured.

ID: 3.22-354/i373

Req 355	For SAP systems in the Google cloud a central connection "SAPaaS Router Service" and "SAPaaS Proxy Service" between CN-TAG and the Google Cloud is provided in connection with the DTIT Landing Zone.

The central connection ("SAPaaS Router Service" and "SAPaaS Proxy Service") consists out of SAProuters and SAP Web Dispatchers. These are delivered together with the DTIT Landing Zone.

Motivation: The "SAPaaS Router Service" and "SAPaaS Proxy Service" consists of SAProuters and SAP Web Dispatchers for the coupling. This is provided together with the DTIT Landing Zone.This ensures SAP-specific communication.

Please use for requests our support entry gate:
ttps://jira.telekom.de/servicedesk/customer/portal/1286

ID: 3.22-355/i373

Req 356	Application access via DIAG, RFC and SNC protocol must be lead through the central proxy "SAPaaS Router Service".

Application accesses from the CN-DTAG (Hitnet) to SAP systems located in the Google Cloud in Folder „SAP on GCP“ must conduct their communication via the central, upstream SAProuters of the "SAPaaS Router Service".

According to the protocols to be transferred (for SAP must be used a suitable system):
RFC and SNC => SAProuter

Motivation: The SAProuter in particular is an important component for the SAP-specific protocol RFC and SNC. In many cases it is also used as a net-separating element.

Via the central, upstream SAProuters of the "SAPaaS Router Service" a secure connection between OnPremise and

Google cloud world with the product "SAP on GCP" established.

ID: 3.22-356/i373

Req 357	Application access in direction Google Cloud via the http-protocol must be lead through the proxy (SAP Web Dispatcher).

Application accesses from the CN-DTAG (Hitnet) to SAP systems located in the Google Cloud product "SAP on GCP" must conduct their communication via the upstream SAP Web Dispatchers.

According to the protocol to be transferred (for SAP must be used a suitable system):
SAP-http, HTTPS, API => SAP Web Dispatcher

Motivation: The SAP Web Dispatcher in particular is an important component for the http and https traffic. In many cases it is also used as a net-separating element.

Via the upstream SAP Web Dispatchers a secure connection between OnPremise and Google cloud world with the product "SAP on GCP" established.

ID: 3.22-357/i373

Req 358	Outbound connections from Google Cloud to the internet must be set up at the project level.

Outbound connections from Google Cloud to the internet must be set up at the project level. For this purpose, the managed service "Secure Web Proxy" must be used in GCP.

A uniform implementation is to be provided promptly via COSMOS.

Motivation: Ensuring secure communication on the Internet.

ID: 3.22-358/i373

Req 359	Application traffic between application landscapes within the Google Cloud must always be routed via the network separating element ("SAPaaS Router Service", SAPaaS Proxy Service").

Application traffic between application landscapes (in the context of the managed folders) within the Google Cloud must always be routed via the network separating element ("SAPaaS Router Service", SAPaaS Proxy Service"). Managed Firewalls and Security Groups are considered network-separating elements in the Google Cloud too.

Motivation: In order to create a network separation between the different environments, all connections must run via suitable proxies or routers. Furthermore, we ensure via the proxies and routers that the communication is encrypted at least on an application basis.

ID: 3.22-359/i373

Req 360	All application connections must be encrypted at the application level.

End-to-end encryption at application level must be implemented (at application level via SNC or HTTPS).

Motivation: Protection of credentials and data at application level.

ID: 3.22-360/i373

Req 361	A backup of the data must exist geo redundancy.

If the data is only stored in the cloud, it must be ensured that an acceptable data set (data volume, actuality) is also

stored geo redundancy.

Motivation: A geo-redundant back ensures high availability in the event of damage to a data center.

ID: 3.22-361/i373

Req 362	The access from job steering tool Automic Automation (UC4) for applications within the cloud must be secured.

When using Automic Automation (UC4) a separate tenant/client must be built up for the cloud application, so that a separation between the OnPrem world and the cloud world is done.

Furthermore only encrypted connections can be used.

Motivation: When Automic Automation (UC4) is used, the communication must be based on encrypted protocols and authentication.

Unauthorized access by high privileged Automic Automation (UC4) users must be avoided by using encrypted protocols and secure athentication.

ID: 3.22-362/i373

2.22.1. Central Management Project "SAPaaS central Management Service"

Req 363	Administrative access to the SAP application landscapes in the Google Cloud in the "SAP on GCP" product of the DTIT Landing Zone must be provided via the "SAPaaS central Management Service".

Administrative access to the SAP application landscapes in the Google Cloud in the "SAP on GCP" product of the DTIT Landing Zone must be provided via the "SAPaaS central Management Service". All SAP applications within the Google Cloud are administrated from these central management project "SAPaaS central Management Service".

Hint: A 2FA is required.

Motivation: Unauthorized privileged access must be avoided.

ID: 3.22-363/i373

Req 364	No user application access may take place in the central management subscription "SAPaaS central Management Service".

This central management subscription "SAPaaS central Management Service" is only to be used for operational purposes.

Motivation: Enforce a secure and uniform application administration way for SAP system landscapes.

ID: 3.22-364/i373

2.23. FCI Cloud via Landing Zone with produkt "SAP on FCI"

The security requirements only apply to the "SAP on FCI" product. With the product "SAP on FCI", the DTIT Landing Zone acts as a separating and coupling network between the CN-DTAG (Hitnet) and the FCI.

Req 365	SAP systems operating in the FCI cloud may only be accessible via the "Hitnet" (CN-DTAG).

Access to the SAP systems operated in the FCI Cloud (in the product "SAP on FCI") may only be regulated via DTIT Landing Zone with its protective measures (e.g. monitoring, protection, secure connections, authentication) via the Hitnet. Clients are not allowed to access the application in the FCI Cloud from the Internet.

Motivation: The SAP system must be protected against unauthorized access.

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

For this requirement the following warranty objectives are relevant:

ID: 3.22-365/i373

Req 366	SAP applications must be made available via COSMOS in the FCI cloud via the "SAP on FCI" product.

SAP applications must be made available via COSMOS in the FCI cloud via the "SAP on FCI" product.
(The transport directory is also implemented via COSMOS).

OSMOS is the construction kit for the creation of system and application landscapes.
https://cosmos.sap.t-systems.com

Motivation: Easy relaying, quick installation, harmonized operation.

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

For this requirement the following warranty objectives are relevant:

ID: 3.22-366/i373

Req 367	For SAP systems in the FCI Cloud, a central connection "SAPaaS Router Service" and "SAPaaS Proxy Service" is provided between CN-TAG and the FCI Cloud in conjunction with the DTIT Landing Zone.

For SAP systems in the FCI Cloud, a central connection "SAPaaS Router Service" and "SAPaaS Proxy Service" is provided between CN-TAG and the FCI Cloud in conjunction with the DTIT Landing Zone.

The central connections "SAPaaS Router Service" and "SAPaaS Proxy Service" consist of SAProuters, SAP Web Dispatchers (Web Proxy) and components that are necessary for the SAP system and its coupling to different networks. These are provided in addition to the DTIT Landing Zone.

Motivation: The "SAPaaS Router Service" and "SAPaaS Proxy Service" ensure SAP-specific communication. The "SAPaaS Router Service" and "SAPaaS Proxy Service" are provided by a team in DT-IT, similar to the landing zone.

For inquiries, please use our support gate:
https://jira.telekom.de/servicedesk/customer/portal/1286

ID: 3.22-367/i373

Req 368	Application traffic between application landscapes in the FCI cloud must always be routed via a central network-separating element ("SAPaaS Router Service", "SAPaaS Proxy Service").

Application traffic between application landscapes in the FCI cloud (CWD1 - CWDn) must always be routed via a central network-separating element ("SAPaaS Router Service", "SAPaaS Proxy Service").

Motivation: Implementation of different application protections in terms of patching, monitoring, interface access.

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

For this requirement the following warranty objectives are relevant:

ID: 3.22-368/i373

Req 369	Application access via the DIAG, RFC and SNC protocols must be routed via the central, upstream proxy "SAPaaS Router Service".

Application access from CN-DTAG (Hitnet) to SAP systems that are running in the "SAP on FCI" folder in the FCI Cloud must route their communication via the central, upstream SAProuter of the "SAPaaS Router Service".

According to the protocols to be transferred (a system suitable for SAP must be used):
RFC and SNC => SAProuter

Motivation: In particular, the SAProuter is an important component for the SAP-specific RFC and SNC protocols. In many cases, it is also used as a network separating element.

A secure connection from the on-premise to the FCI cloud with the product "SAP on FCI" is established via the central, upstream SAProuters of the "SAPaaS Router Service".

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

For this requirement the following warranty objectives are relevant:

ID: 3.22-369/i373

Req 370	Application access to the FCI Cloud that uses the http protocol must be carried out via an upstream proxy (SAP Web Dispatcher).

Application access from CN-DTAG (Hitnet) to SAP systems that are located in the FCI Cloud in the "SAP on FCI" area must route their communication via the upstream SAP Web Dispatchters.

According to the protocols to be transferred (a system suitable for SAP must be used):
SAP-http, HTTPS, API => SAP Web Dispatcher

Motivation: In particular, SAP Web Dispatcher is an important component for the http and https protocols. In many cases, it is also used as a network separating element.

The upstream SAP Web Dispatchers ensure a secure connection from the on-premise to the FCI Cloud with the "SAP on FCI" product.

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

For this requirement the following warranty objectives are relevant:

ID: 3.22-370/i373

Req 371	For outbound connections from the FCI Cloud to the Internet, a proxy must be used in the HitNet/ CNDTAG.

For outbound connections from the FCI Cloud to the Internet, a proxy must be used in the HitNet/CNDTAG.

Motivation: A central proxy server secures the platform, as only one central point is allowed to communicate with the Internet. In addition, this offers the possibility of easier implementation of virus scanners or cache servers, filter lists, etc.

ID: 3.22-371/i373

Req 372	All application connections must be encrypted at the application level.

All application connections must be encrypted at the application level.
End-to-end encryption at the application level must be implemented (at the application level via SNC or HTTPS).

Motivation: Protection of credentials and data at the application level.

ID: 3.22-372/i373

Req 373	A backup of the data must be redundant.

It must be ensured that an acceptable data set (amount of data, up-to-dateness) is stored redundantly in another availability zone.

Motivation: Ensuring a backup to prevent data loss.

Implementation example: For example: Data center Magdeburg / data center Biere

ID: 3.22-373/i373

Req 374	Job control via Automic Automation (UC4) must be implemented for applications in the FCI Cloud via secure access.

Job control via Automic Automation (UC4) must be implemented for applications in the FCI Cloud via secure access.
When using Automic Automation (UC4), a separate tenant/client must be used for the application in the FCI Cloud, so that a separation is created between application landscapes and Automic Automation (UC4).

Motivation: Secure access to systems and separation of different environments (OnPrem, FCI, Google, Azure, Non-SAP).

ID: 3.22-374/i373

Req 375	When using automation (UC4), communication must be based on secure protocols and secure authentications.

When using automation (UC4), communication must be based on secure protocols and secure authentications.

Motivation: UC4 users have high privileges and need to be protected.

Implementation example: Use of certificates by the agents.

ID: 3.22-375/i373

2.23.1. Central Management Services (administrative access)

Req 376	Administrative access to the SAP application landscapes in the FCI Cloud in the product "SAP on FCI" of the DTIT Landing Zone must be provided via a central management service.

Administrative access to the SAP application landscapes in the FCI Cloud in the product "SAP on FCI" of the DTIT Landing Zone must be provided via a central management service.

All SAP applications are administered from this central management service.
2FA is required.

Motivation: Uncontrolled privileged access must be prevented.

Implementation example: Examples of centralized management services:

- Centrales HANA Studio
- Centrales HANA Cockpit
- COSMOS Management Services

ID: 3.22-376/i373

Req 377	In the central management service, only employees from the operational area are allowed access.

In the central management service, only employees from the operational area (OS and SAP basis) are allowed access.

Motivation: Ensuring secure and uniform application administration access for SAP system landscapes.

ID: 3.22-377/i373

Req 378	Administrative access to the central management services must be made via the USP tool.

Administrative access to the central management services must be made via the USP tool.

Motivation: Ensure secure authentication for administrators.

ID: 3.22-378/i373