

Security requirement

Machine Learning

Deutsche Telekom Group

Version	4.0
Date	Jul 1, 2025
Status	Released

Publication Details

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

Reuterstrasse 65, 53315 Bonn
Germany

File name	Document number 3.81	Document type Security requirement
Version 4.0	State Jul 1, 2025	Status Released
Contact Telekom Security psa.telekom.de	Validity Jul 1, 2025 - Jun 30, 2030	Released by Stefan Pütz, Leiter SEC-T-TST

Summary
This document describes the security requirements for machine learning algorithms.

Copyright © 2025 by Deutsche Telekom AG.
All rights reserved.

Table of Contents

1.	Introduction	4
2.	AI-Componsens	5
2.1.	Machine Learning Algorithm	5
2.2.	Training of the machine learning algorithm	7
2.3.	Operation of the machine learning algorithm	9
2.4.	Access to the machine learning algorithm	14
2.5.	Access of the machine learning algorithm to other systems	25
3.	Systems using AI-Components	26
4.	Technical Baseline Security for IT/NT Systems	32
4.1.	Technical and organizational prerequisites for the use of components in the system	32
4.2.	Basic System Hardening	32
4.3.	Protecting availability and integrity	35
4.4.	Protecting data and information	36
4.5.	Session Protection	37
4.6.	Authentication and authorization	38
4.7.	Logging	39

1. Introduction

This document was prepared on the basis of the requirements set out in the security guidelines applicable within the Group. It should be noted that in addition to the security guidelines, there may also be AI ethics guidelines or other requirements in the Group that must also be observed.

The security requirement serves, among other things, as the basis for release in the PSA process. It also serves as an implementation recommendation for units that do not participate in the PSA process. These requirements must already be taken into account during the planning and decision-making processes. When implementing the safety requirement, the respective overriding national, international and supranational law must be observed.

In order to create a uniform basis for understanding, some central terms are clarified below:

Term	Description
Artificial Intelligence (AI)	Generic term for technologies that enable computers to simulate human-like intelligence and perform tasks that normally require human thought.
AI-Component	A system that implements AI itself. For example, an image recognition algorithm that can be accessed via an API.
Machine Learning	Machine learning is a generic term for the "artificial" generation of knowledge from experience. An artificial system learns from examples and can generalize them after the learning phase is complete.
Machine Learning Algorithm	One of many specific algorithms used in machine learning, for example, to detect patterns in data and make predictions.
Hyperparameter	Parameters that influence the learning process of a machine learning algorithm and are set before training.
Dimensionality reduction	A process of reducing the number of features or variables in a data set to reduce complexity and improve the efficiency of machine learning algorithms.

It is also important to note that not every product must meet every single requirement. This document is divided into requirements for AI-Components and requirements for systems that use AI-Components.

If compliance with the described requirements is not feasible or only feasible to a limited extent in individual cases, a risk assessment must be carried out together with a security and/or data protection expert (according to the requirement concerned) and possible alternative protective measures must be agreed.

2. AI-Componsens

In this document an AI-Component is a system that implements AI itself. For example, an image recognition algorithm that can be accessed via an API.

2.1. Machine Learning Algorithm

Req 1	The task of the algorithm, the context in which it is to be used, and the conditions under which the algorithm is allowed to make decisions must be clearly defined.
-------	--

For an algorithm to be used reliably, it is crucial that the task it is to perform is clearly defined, that the context in which it is to be used is fully understood, and that the framework conditions under which the algorithm is allowed to make decisions are precisely regulated. This is the only way to ensure that the algorithm delivers the desired results and does not lead to unexpected or undesired outcomes.

Required information for the definition of the task:

- Clear and unambiguous description of the specific task
- Example input and output data

Information needed to define the context:

- Type of data to be processed
- Source of training and test data
- List of involved stakeholders

Motivation: Clearly defining the task, the context, and the constraints under which the algorithm is allowed to make decisions not only minimizes the risk of unexpected results, but also ensures better verifiability of the algorithm.

For this requirement the following threats are relevant:

- Disruption of availability
- Denial of executed activities

For this requirement the following warranty objectives are relevant:

- Transparency

ID: 3.81-1/4.0

Req 2	The process of data collection and/or data generation must be clearly documented. This includes training, validation and test data.
-------	---

It is of great importance that the process of data collection and/or data generation is clearly documented to ensure transparency, traceability as well as protection of personal data. This includes not only the documentation of the training, validation and test data, but also the description of the methods and tools used as well as the selection criteria for the data. Clear documentation is essential to ensure the quality of the data and to identify possible sources of error.

Relevant information:

- Timestamp of collection/generation
- Description of data sources/types
- Methods and tools used for data collection and/or data generation
- Selection criteria for the data
- Processes for data preparation and cleaning
- Processes for anonymization or pseudonymization of personal data
- Processes for validation and verification of data quality

- Processes for splitting data into training, validation and test data
- Processes for updating and maintaining data

Motivation: Clear documentation of the data collection and/or data generation process is critical to the quality and trustworthiness of data used for training and testing the algorithm. Insufficient documentation can lead to errors and inaccuracies in the data, which in turn can lead to incorrect decisions and unreliable results.

For this requirement the following threats are relevant:

- Unauthorized modification of data
- Unnoticeable feasible attacks

For this requirement the following warranty objectives are relevant:

- Integrity
- Transparency

ID: 3.81-2/4.0

Req 3	The AI-Component and connected systems must be protected against misuse of the AI-Component.
-------	--

Hardening must take place so that the AI component, through quasi-legitimate use, cannot be abused.

Examples of such attacks are:

- Task Escape Attacks - Using the Large Language Model (LLM) for (potentially malicious) out-of-scope tasks
- Data Exfiltration Attacks - output of internal information provided to the LLM by, for example, a database

Motivation: Machine learning algorithms often have more capabilities than are needed by the developers. These can be potential vulnerabilities as they are used to abuse the AI component.

For example, if a publicly available Large Language Model is used as the AI-Component, it is important to ensure that the LLM's responses match the context of the project.

For this requirement the following threats are relevant:

- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources
- Unnoticeable feasible attacks
- Attacks motivated and facilitated by information disclosure or visible security weaknesses

For this requirement the following warranty objectives are relevant:

- Availability
- Confidentiality
- Integrity

ID: 3.81-3/4.0

Req 4	Deployed base algorithms used for transfer learning must be obtained from trusted sources and checked against existing AI-Component requirements (PSA requirements in this document).
-------	---

Transfer learning refers to all methods that enable the knowledge gained from solving a particular problem to be transferred to the processing of another problem.

To ensure that transfer learning is performed effectively and safely, it is critical that the baseline algorithms used for this process come from trusted sources and comply with existing AI-Component requirements (PSA requirements in

this document). It is important that these baseline algorithms are regularly reviewed and updated to ensure that they meet the latest standards and do not pose security risks. In addition, factors such as performance, scalability, and compatibility with other systems should also be considered when selecting baseline algorithms for transfer learning.

Motivation: If the base algorithms are obtained from unreliable or insecure sources, this can lead to errors, data leaks, and even security vulnerabilities and new attack scenarios. In addition, unreliable algorithms can affect the performance of the AI-Component and make it more difficult to achieve the desired results. Therefore, it is important that the base algorithms come from trusted sources to ensure effective and secure transfer learning.

For this requirement the following threats are relevant:

- Unnoticeable feasible attacks

For this requirement the following warranty objectives are relevant:

- Integrity
- Transparency

ID: 3.81-4/4.0

Req 5	The hyperparameters of the algorithm must be carefully selected and documented to ensure reproducibility.
-------	---

Hyperparameters are external configuration variables used for training machine learning algorithms and are set manually before training an algorithm. Examples of hyperparameters are the number of nodes and layers in a neural network and the number of branches in a decision tree. Hyperparameters determine important features such as model architecture, learning rate, and model complexity.

To ensure that the results of an algorithm are reproducible, it is important to carefully select and document the hyperparameters. Accurate documentation of hyperparameters also enables further improvement of the algorithm through hyperparameter optimization. If automated hyperparameter optimization is used, this process must also be traceable.

Motivation: If the hyperparameters of a machine learning algorithm are not carefully documented, this can lead to problems with the reproducibility and optimization of the algorithm. Without accurate documentation, it is difficult to obtain the same results when the algorithm is run again, which can affect the reliability of the algorithm. In addition, the lack of documentation also makes it difficult to optimize the hyperparameters for best possible results.

For this requirement the following threats are relevant:

- Disruption of availability

For this requirement the following warranty objectives are relevant:

- Availability
- Integrity
- Transparency

ID: 3.81-5/4.0

2.2. Training of the machine learning algorithm

Req 6	Collected/generated data must be pre-processed and cleaned prior to use for training, validation and testing purposes.
-------	--

To ensure the accuracy and reliability of training, validation and testing results, it is necessary to carefully review, clean and normalize the collected/generated data before use. In this process, erroneous, incomplete, or inconsistent data should be removed or corrected to avoid bias in the results. In addition, appropriate pre-processing of the data, such as feature extraction or dimensionality reduction, can improve the effectiveness of machine learning algorithms.

Possible cleaning procedures and preprocessing:

- Removing duplicates

- Removing outliers
- Filling missing values
- Normalization can be used to unify the scaling of the data and thus make it comparable
- Feature engineering, the preparation of data for processing in machine learning algorithms, can be used to extract relevant features from raw data and reduce the dimensionality of the data
- Removal of irrelevant features
- Balancing classes
- Each class of data (e.g. dog and cat) must be represented the same number of times in the data set

Motivation: It is important to carefully clean and prepare data before using it for training, validation, and testing purposes to achieve accurate and reliable results. Uncleaned or incomplete data can introduce bias and errors in the results and reduce the effectiveness of the machine learning algorithm. Performing cleanup and pre-processing can ensure that the data is accurate and consistent, leading to higher accuracy and reliability of results.

For this requirement the following threats are relevant:

- Unnoticeable feasible attacks

For this requirement the following warranty objectives are relevant:

- Integrity
- Transparency

ID: 3.81-6/4.0

Req 7	The data used for training, validation and testing purposes must be representative with regard to the assigned task. In addition, the data must be free of possible bias and inconsistencies.
-------	---

To ensure that the machine learning algorithm produces accurate and reliable results, the data used for training, validation, and testing purposes must be representative of the assigned task. This means that the data should have sufficient diversity and coverage to cover the different aspects of the task. In addition, it is mandatory that the data be free of possible biases and inconsistencies (bias) that may lead to inaccurate or unfair results. Examples of possible biases include unequal distribution of data points in certain categories or unequal representation of certain groups in the data.

Motivation: It is especially necessary that the data used for a machine learning algorithm be free of bias and inconsistency in order to produce accurate and fair results. If the data is biased, the results may also be skewed and lead to inaccurate or unfair decisions.

For example, an uneven distribution of data points in certain categories may cause the algorithm to inappropriately consider those categories or to draw incorrect conclusions. In addition, unequal representation of certain groups in the data may lead to unfair decisions based on prejudice or discrimination.

For this requirement the following threats are relevant:

- Disruption of availability

For this requirement the following warranty objectives are relevant:

- Integrity
- Transparency

ID: 3.81-7/4.0

Req 8	Security-relevant events, such as the training of the machine learning algorithm, must be logged with an exact time stamp and a unique system designation.
-------	--

It is necessary to accurately document activities, such as training the machine learning algorithm, to ensure effective

troubleshooting and traceability in the event of errors or security breaches. The following information must be recorded:

- Name of the person conducting the training
- Training timestamp
- Data set used
- Changes to training, validation and test data
- Name of the person making the changes
- Timestamp of the changes

Motivation: Accurately logging security-related events, such as training the machine learning algorithm, is essential to ensure effective troubleshooting and traceability in the event of errors or security breaches. Accurate logging allows potential security risks to be identified and fixed early, before they lead to major problems. In addition, clear identification of events enables fast and effective response to security incidents.

For this requirement the following threats are relevant:

- Denial of executed activities

For this requirement the following warranty objectives are relevant:

- Transparency

ID: 3.81-8/4.0

Req 9	Data requiring protection, such as training, validation and test data, as well as hyperparameters and implementation of the algorithm, must be protected from unauthorized viewing, modification and deletion during storage.
-------	---

To ensure the integrity and confidentiality of data used for algorithm training, validation, and testing, as well as the hyperparameters and implementation of the algorithm itself, during storage, it is essential to implement appropriate security measures. These include the use of encryption technologies, access controls, and periodic reviews of security measures. It is also necessary to ensure that all individuals who have access to this data are aware of the importance of data security and adhere to the appropriate security protocols.

Motivation: Unauthorized access to training, validation and test data can lead to unnoticed manipulation of these. If the algorithm is then trained with the manipulated data set, unforeseen behavior can occur and the integrity of the algorithm is no longer guaranteed.

If unauthorized access to the hyperparameters or to the implementation of the algorithm occurs, changes can be made that can have serious consequences. In addition, this information provides the opportunity for cloning the algorithm.

For this requirement the following threats are relevant:

- Unauthorized access or tapping of data
- Unauthorized modification of data
- Denial of executed activities
- Unnoticeable feasible attacks
- Attacks motivated and facilitated by information disclosure or visible security weaknesses

For this requirement the following warranty objectives are relevant:

- Confidentiality
- Integrity
- Transparency

ID: 3.81-9/4.0

2.3. Operation of the machine learning algorithm

Req 10	The results of the algorithm must be continuously checked and validated to ensure that they are correct and meet expectations. If the results of the algorithm deviate from the expected behavior, this must be corrected immediately.
--------	--

It is essential that the results of the algorithm are regularly checked and validated to ensure that they are not only correct, but also meet expectations. Inaccurate or erroneous output can have serious consequences. Therefore, it is necessary that the algorithm be continuously monitored to detect and correct potential problems early. This can be done through regular functional/ and penetration testing.

It is also necessary that the algorithm always shows the expected behavior. However, should deviations occur, these must be identified and remedied immediately to ensure reliable functioning of the algorithm. This requires that the affected machine learning algorithm is taken out of operation as long as the misbehavior has not been corrected.

Motivation: By continuously monitoring the behavior of the algorithm, potential problems can be detected and responded to at an early stage. In the absence of such monitoring, behavioral changes are not detected and cannot be remedied.

Implementation example: Depending on the area of application, for example in image recognition, random checks may be sufficient. However, if an AI-Component is used to configure productive systems, such as network elements, servers, applications, etc., the decision made by the AI-Component should be checked beforehand on a test system.

For this requirement the following threats are relevant:

- Disruption of availability
- Unnoticeable feasible attacks

For this requirement the following warranty objectives are relevant:

- Integrity

ID: 3.81-10/4.0

Req 11	Security-related events must be logged with an exact timestamp and a unique system reference.
--------	---

Security-relevant are any events that may have an impact on the information security (confidentiality, integrity, availability) of a system or indicate its misuse. Systems must log the occurrence of security-related events.

In order to enable the assignment and evaluation of these events, they must be logged together with a unique system name (e.g. host name, IP or MAC address) and the exact time of occurrence ("time stamp"). At a minimum, the timestamp of a logged event must include the following information:

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

The security relevance of specific events depends on the type and function of the system as well as the data processed in it. Therefore, it is not possible to specify a uniform list of events that are relevant to security. Relevant events must be identified according to the system. The following list contains typical security-related events and is accordingly to be understood as a basic guide for the identification of security-relevant events for a particular system.

Security-related Event	Information to log to the event
Failed login attempts	• affected user account • number of failed attempts • source of access (IP address, client-ID/client-name, local console, ...)

System access via user accounts (including system accounts or technical user accounts)	• affected user account • Access start/end time (login/logout) • source of access (IP address, client-ID/client-name, local console, ...)
Modifications of user accounts (including system accounts or technical user accounts)	• user account used to make the modification • user account to which the modification was made • type of modification made (create, activate, deactivate, delete, password reset, ...)
Changes to role assignments/permissions of user accounts (including system accounts or technical user accounts)	• user account used to make the modification • user account to which the modification was made • name of the affected role, type of the affected permission • type of modification made (role/permission added, role/permission removed, ...)
Use/initiation of critical system functions	• user account used to make the use • name/type of system function used
Access to sensitive data processed in the system	• user account used to access the data • Name of the affected data field (the contents of the data field should not be logged to prevent data leakage via log files) • type of access (read, modify, delete, ...)
Abnormal increase in the utilization of system resources, such as memory, CPU, network, or disk access, over an extended period of time	• threshold of the affected system resource • duration of exceeding the threshold • threshold exceedance Depending on the individual system, sensible threshold values must be defined.
Other events that may provide evidence of attempted or successful attacks against the system *)	• individually depending on the type of event

*) This requires identifying the most important forms of attack that the system can typically expect and ensuring that events relevant to their detection are actually logged by the system. As a rule, this can only be guaranteed by the developer/manufacturer of the system.

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.

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

For this requirement the following threats are relevant:

- Denial of executed activities
- Unnoticeable feasible attacks

ID: 3.01-33/9.0

Req 12	Applicable retention and deletion periods must be observed for security-relevant logging data that is recorded locally.
--------	---

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

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

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

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

Deviances

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

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

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

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

For this requirement the following threats are relevant:

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

ID: 3.01-34/9.0

Req 13	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 proto-

cols such as Syslog, HTTPS should be preferred.

Motivation: If logging data is only stored locally, it can be manipulated by an attacker who succeeds in compromising the system in order to conceal his attack and any manipulation he has performed on the system. This is the reason why the forwarding must be done immediately after the event occurred.

For this requirement the following threats are relevant:

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

For this requirement the following warranty objectives are relevant:

- Integrity

ID: 3.81-13/4.0

Req 14	For security-relevant logging data that is forwarded to the 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 log server.
- after 90 days, stored logging data must be deleted immediately on the 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 log server is not within the same operational responsibility as the source system of the login data, it must be ensured that the responsible operator of the log server is aware of the valid parameters for the logging data to be received and that they are adhered to in accordance with the regulations mentioned here.

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

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

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

For this requirement the following threats are relevant:

- Unauthorized access or tapping of data
- Denial of executed activities

- Unnoticeable feasible attacks

ID: 3.01-36/9.0

2.4. Access to the machine learning algorithm

Req 15	Access for the use of the AI-Component must not be possible without successful authentication and authorization. This applies both to API interfaces and to all other ways of interacting with the AI-Component.
--------	--

Successful authentication and authorization is required before the machine learning algorithm can be accessed or used. This applies to all types of interactions with the algorithm, including API interfaces and other access options.

This requirement applies to all systems for which Deutsche Telekom is responsible. I.e. if it is an external service (Bing Search, Chat-GPT, ...), this requirement does not apply.

Motivation: In the absence of authentication and authorization, there is a risk of unauthorized use of the algorithm and must therefore be prevented.

For this requirement the following threats are relevant:

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

For this requirement the following warranty objectives are relevant:

- Confidentiality

ID: 3.81-15/4.0

Req 16	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 sys-

tem information must always be deactivated.

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

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

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

Implementation example: [Example 1]

Deactivation of the display of the product name and the installed version of a Web server in its delivered error web pages.

[Example 2]

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

For this requirement the following threats are relevant:

- Attacks motivated and facilitated by information disclosure or visible security weaknesses

ID: 3.01-9/9.0

Req 17 User accounts must be protected with at least two authentication features from different factors.

All user accounts of the system must be protected from unauthorized use.

This is to be achieved by using authentication features that allow the legitimate user to be uniquely identified.

Authentication features are differentiated into superordinate groups, so-called "factors":

- Factor KNOWLEDGE: "something that only the legitimate user knows"
- Factor POSSESSION: "something that only the legitimate user possesses"
- Factor INHERENCE: "something that only the legitimate user is"

At least two authentication features must be combined.

These must come from different factors ("Multi-Factor Authentication"; short: "MFA").

A combination of authentication features of the same factor (example: two different PINs) is not allowed.

Examples of acceptable authentication features

- from the KNOWLEDGE factor: PINs, passphrases (passwords*)
- from the POSSESSION factor: smart cards, FIDO 2 Level 2 security keys, device-bound passkeys, cryptographic keys stored in a TPM/HSM, OTPs (One-Time Passwords) generated in an authenticator app that has been approved for use in Deutsche Telekom
- from the INHERENCE factor: biometric features such as fingerprints, iris or retina scans, face recognition

*) Important note on the use of passwords

Passwords must be strictly avoided as an authentication feature.

They are only acceptable if more secure alternatives cannot be implemented.

Authentication features that can be easily forged or manipulated are generally not permitted.

These include:

- Phone numbers and caller IDs
- IP addresses
- Geographical locations
- VPN affiliations and network information
- Status of clients, such as workstation systems or mobile devices

Note on technical user accounts

Technical user accounts used for machine access (also known as "machine-to-machine" or "M2M" accounts) are exempt from the obligation to use multi-factor authentication, as this is usually not feasible there.

Motivation: User accounts that are not protected by appropriate authentication features can be abused by an attacker to gain unauthorized access to a system and the data and applications stored on it. Due to a variety of common attack methods, the protection of user accounts by a single authentication feature is now considered weak, especially if the classic password is used here. A combination of authentication features from different factors significantly increases security for user accounts.

Implementation example: Examples of MFA

MFA Combination	Description	Application Note
Telekom "myCard" plus PIN	1st Factor: Certificate implemented in hardware ("Possession") 2nd Factor: PIN ("Knowledge")	(none)
FIDO2 Level 2 Security Key plus fingerprint	1st Factor: Crypto key stored in hardware ("Possession") 2nd Factor: Biometric feature ("Inherence")	Biometric feature must be pre-configured, otherwise the biometric sensor will only work as a "button" and does not provide the 2nd factor ("Inherence").
Device-bound Passkey in WH4B* on Nucleus PC plus PIN	1st Factor: Crypto key stored in hardware ("Possession") 2nd Factor: PIN ("Knowledge")	(none)
Device-bound Passkey in Secure Enclave on iOS Device plus FaceID	1st Factor: Crypto key stored in hardware ("Possession") 2nd Factor: Biometric feature ("Inherence")	(none)
Password plus OTP App approved for use (e.g. MS Authenticator)	1st Factor: Password ("Knowledge") 2nd Factor: specific generated One Time Password ("Possession")	Passwords are only eligible if more secure alternatives cannot be implemented. The synchronization of the authenticator app via a cloud service as well as sharing (e.g. via a password manager) the OTP is not permitted.

*) WH4B: "Windows Hello for Business"

For this requirement the following threats are relevant:

- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data

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

ID: 3.01-20/9.0

Req 18 Privileged user accounts must be protected with at least two authentication features that come from different factors and provide increased resilience against phishing attacks.

A privileged user account is a user account that has advanced permissions within a system. Advanced permissions allow access to configuration settings, features, or data assets that are not available to regular users of the system. In direct dependence on the specific tasks performed by means of a privileged user account within a system, the advanced permissions assigned may be specifically limited or may include fully 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; Assign permissions; password resets)
- Accounts that are authorized to legitimize, trigger, or stop business-critical operations
- Accounts with access to data classified as SCD (Sensitive Customer Data) in the interest of the 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 entire database)

To protect privileged user accounts, at least two authentication features must be combined. These must come from different factors ("KNOWLEDGE", "POSSESSION" or "INHERENCE"). A combination of authentication features of the same factor is not permitted. At least one of the authentication features must not be vulnerable to phishing attacks.

In a phishing attack, the legitimate user is trickily manipulated to reveal his login data to the attacker, for example by means of fake login pages. A phishing attack can also be carried out against a combination of multiple authentication features.

Phishing-resilient authentication features prevent attacks based on their technical implementation. These authentication features use cryptographic methods and create a strong bond between the user, the device and the service. They also require manual user interaction during authentication, which cannot be automated; for example, to submit a fingerprint scan to a hardware token.

Considered not vulnerable to phishing attacks are

- FIDO2 Level 2 Security Keys
- Certificates implemented in hardware

Important note on the use of passwords

Passwords must be strictly avoided as an authentication feature.

They are only acceptable if more secure alternatives cannot be implemented.

Note on technical user accounts

Technical user accounts used for machine access (also known as "machine-to-machine" or "M2M" accounts) are exempt from the obligation to use multi-factor authentication, as this is usually not feasible there.

Motivation: Privileged user accounts pose an increased risk to the security of a system. If an attacker successfully compromises such a user account, they will be granted broad privileges that allow them to take control of the system or system parts, disrupt system functions, view/manipulate processed data, or influence business-critical operations. The combination of multiple authentication features of different types already significantly minimizes the risk of a user account being compromised, but for privileged user accounts, special attention must be paid to resilience against phishing attacks. The phishing attacks that can be observed are becoming increasingly tricky and difficult for users to detect, so that multifactor authentication without increased resilience against phishing can no longer be considered sufficient for privileged user accounts.

Implementation example: Examples of MFA with increased resilience to phishing attacks

MFA Combination	Description	Application Note
Telekom "myCard" plus PIN	1st Factor: Certificate implemented in hardware ("Possession") 2nd Factor: PIN ("Knowledge")	(none)
FIDO2 Level 2 Security Key plus fingerprint	1st Factor: Crypto key stored in hardware ("Possession") 2nd Factor: Biometric feature ("Inherence")	Biometric feature must be pre-configured, otherwise the biometric sensor will only work as a "button" and does not provide the 2nd factor ("Inherence").
FIDO2 Level 2 Security Key plus PIN	1st Factor: Crypto key stored in hardware ("Possession") 2nd Factor: PIN ("Knowledge")	(none)

For this requirement the following threats are relevant:

- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources
- Denial of executed activities

ID: 3.01-21/9.0

Req 19 It must be ensured that machine users do not use hard-coded secrets such as passwords or tokens to authenticate or authorize themselves to the AI algorithm.

To ensure the security and integrity of the system, it is important to ensure that machine users do not use hard-coded secrets such as passwords or tokens to authenticate or authorize themselves to the AI algorithm. Instead, alternative methods of authentication and authorization should be implemented to avoid potential disclosures.

Motivation: Secrets like passwords, certificates, secret keys are a high risk if they are disclosed to unauthorized persons. Especially used in code or images stored in repositories can lead to an unintended disclosure.

Implementation example: For implementation, the following Secret Management Systems can be used:

- HashiCorp. Vault
- CyberArk Conjur
- AWS KMS
- Azure Key Vault
- Google Cloud KMS
- GitLab Protected Variables

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:

- Integrity
- Transparency

ID: 3.81-19/4.0

Req 20 User accounts must ensure the unique identification of the user.

Users must be identified unambiguously by the system.

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

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

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

Ways to prevent misuse of such user accounts by individuals include:

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

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

Motivation: Unambiguous user identification is mandatory to assign a user permissions that are necessary to perform the required tasks on the system. This is the only way to adequately control access to system data and services and to prevent misuse. Furthermore, it makes it possible to log activities and actions on a system and to assign them to individual users.

For this requirement the following threats are relevant:

- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources
- Denial of executed activities

ID: 3.01-22/9.0

Req 21 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 (known as "Password Hashing") that is recognized as effective for this purpose and is classified as secure at the current state of the art.

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 encoding 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.

For this requirement the following threats are relevant:

- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources
- Denial of executed activities
- Attacks motivated and facilitated by information disclosure or visible security weaknesses

ID: 3.01-24/9.0

Req 22	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 ("Completely Automated Public Turing test to tell Computers and Humans Apart") 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.

For this requirement the following threats are relevant:

- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources
- Denial of executed activities

ID: 3.01-25/9.0

Req 23	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 than 25 characters. This will provide more freedom to End Users when composing individual memorable 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 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 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.

For this requirement the following threats are relevant:

- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources
- Denial of executed activities

ID: 3.01-26/9.0

Req 24	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).

For this requirement the following threats are relevant:

- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources
- Denial of executed activities

ID: 3.01-27/9.0

Req 25	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.

For this requirement the following threats are relevant:

- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources
- Denial of executed activities
- Attacks motivated and facilitated by information disclosure or visible security weaknesses

ID: 3.01-28/9.0

Req 26	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.

For this requirement the following threats are relevant:

- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources
- Denial of executed activities
- Attacks motivated and facilitated by information disclosure or visible security weaknesses

ID: 3.01-29/9.0

Req 27	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, which 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.

For this requirement the following threats are relevant:

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

ID: 3.01-30/9.0

Req 28	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.

For this requirement the following threats are relevant:

- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources
- Denial of executed activities
- Attacks motivated and facilitated by information disclosure or visible security weaknesses

ID: 3.01-31/9.0

2.5. Access of the machine learning algorithm to other systems

Req 29	The results and decisions of the machine learning algorithm must not be transferred to productive systems without prior testing.
--------	--

It is crucial that the results and decisions generated by a machine learning algorithm are carefully checked and validated before they are transferred to productive systems. Although machine learning algorithms are capable of identifying complex patterns and relationships in large amounts of data, they are still not perfect and can be flawed. If these errors are not detected and corrected, they can have a serious impact on the affected system. Therefore, it is important that algorithm results and decisions are reviewed and validated to ensure that they are correct and reliable. This verification does not necessarily have to be performed by a human.

Motivation: Without prior verification of the results of the machine learning algorithm, decisions made can lead to system failures. This must be avoided as a matter of principle.

For this requirement the following threats are relevant:

- Disruption of availability

For this requirement the following warranty objectives are relevant:

- Availability
- Transparency

ID: 3.81-29/4.0

3. Systems using AI-Components

Req 30	The AI component must be able to be completely stopped and deactivated by a system administrator at any time if necessary.
--------	--

If a system has the functionality of an AI component, a system administrator must be able to stop and disable the AI component completely at any time.

Motivation: Motivation: In order to be able to guarantee complete control and supervision of an AI component, there must be an uncomplicated and fast solution for the immediate termination of the AI component. This ensures that further damage can be averted in the event of a compromise or unforeseeable or erroneous behavior of the AI component.

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

ID: 3.81-30/4.0

Req 31	The AI component must be able to be deactivated by the user at any time. This also includes individual sessions or user interactions.
--------	---

If a system has the functionality of an AI component, it must be able to be deactivated by the user. It is also crucial that the user can end individual interactions with the AI component and that the user is informed about the termination. In addition, there must be a documentation available on how the user can activate, deactivate and, if necessary, restore individual sessions.

Motivation: An AI component that cannot be disabled carries the risk of having an unintended impact on the system. Sensitive data could be transmitted without the user being aware of it. In case of a malfunction, the user must also have the option to terminate the AI component even if that would affect the original function.

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

ID: 3.81-31/4.0

Req 32	The AI component must not be able to turn itself back on after it has been turned off or disabled.
--------	--

If the AI component has been switched off or deactivated by the system or the user, the AI component must not be able to reactivate itself on its own. If the AI component has been deactivated, it must also be ensured that it is not automatically activated when the entire system is completely restarted if this function was previously deactivated.

Motivation: A user who has deliberately deactivated the AI component must be able to rely on it staying that way. If a compromised AI component is able to reactivate itself, there is a risk that an attacker will access information or data without the user noticing. In addition, actions and changes to the system could be made by the AI component that are not intended by the user.

For this requirement the following threats are relevant:

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

Req 33	An AI component may only be used as an assistance system and must not be able to make independent decisions.
--------	--

If a system has the functionality of an AI component, it must be ensured that it does not make independent decisions but may only be used as an assistance system. If an AI component is used, process decisions must always be made by a real person who has the necessary authorizations. An AI component must not be modified to make independent decisions.

Motivation: An AI component that makes autonomous decisions poses a threat to the entire system. If an AI component has the same authorizations, e.g. a system administrator or users with elevated privileges, an attacker could misuse the AI component for malicious purposes.

For this requirement the following threats are relevant:

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

Req 34	It must be ensured that an AI component does not have access to safety-critical information.
--------	--

If a system has the functionality of an AI component, it must be ensured that it does not have access to safety-critical information and actions. Safety-critical information, such as Passwords or certificates may only be accessed by authorized users or administrators.

Motivation: An AI component that has access to security-critical information or actions, such as user data, passwords or certificates, poses a significant security risk. If an AI component has been taken over by an attacker, he would also automatically have access to sensitive information and data of users and administrators.

For this requirement the following threats are relevant:

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

Req 35	Interactions with the AI component must be forwarded to a central log server.
--------	---

If a system has the functionality of an AI component, it must ensure that all logging data with the AI component is forwarded to a central log server. This ensures that system administrators can understand exactly how the AI component works.

Motivation: An AI component can be targeted by attackers or suffer system crashes. In order to be able to trace exactly when certain events took place, a central log server must be used in order to be able to forensically process these events and, if necessary, to initiate suitable measures to prevent them in the future.

The following basic regulation must be taken into account:

- Security-related logging data must be retained for a period of 90 days.
- After 90 days, stored logging data must be deleted immediately.

For this requirement the following threats are relevant:

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

ID: 3.81-35/4.0

Req 36 The AI component must have abuse detection, e.g. unethical and malicious actions or expenses.
This function of an abuse detection must be sufficiently described.

If a system has the functionality of an AI component, it must have an abuse detection. This ensures that the AI component is not misused for malicious intentions. Documentations must be available about the functioning and access of this abuse detection. It must not be possible for the user to deactivate the abuse detection.

Motivation: AI components such as a Large language models can be able to generate malicious and unethical content that could be misused by the user. These include, for example, Code generation for ransomware or instructions on how to carry out denial of service attacks. As a rule, such expenses can be prevented by using filters.

For this requirement the following threats are relevant:

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

ID: 3.81-36/4.0

Req 37 Interactions with the AI component must not allow any conclusions to be drawn about the training data that was used to train the AI component.

If a system has the functionality of an AI component, a user may not receive information from the AI component about which training data was used to train the AI model. The data used to train the model must not be publicly available. Extraction of the training data must be prevented by defined rules of behavior of the AI component.

Motivation: Each AI component undergoes a training process before commissioning in order to be able to prepare the component for its intended use. Training datasets that have been determined to be suitable by the system administrators are used for this purpose. It is possible that training data contains sensitive information that has data protection or security-critical relevance, which is why the user may not be given and information about the training data.

For this requirement the following threats are relevant:

- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data

ID: 3.81-37/4.0

Req 38 AI-generated content must always be clearly labeled as such.

If a system has the functionality of a generative AI component, the content generated by it must be clearly labeled as AI-generated. Especially in the case of embedded AI assistants, it must be possible to clearly distinguish between AI and non-AI-generated content. In the case of texts and words, this must be an addition to the content created. A watermark must be used for image and video material.

Motivation: Generative AI is able to generate text, images or video material that can then be used by the user. If emails are generated automatically or texts are supplemented by generative AI, the recipient must be informed that parts of the email were not written by the actual sender himself.

For this requirement the following warranty objectives are relevant:

- Integrity
- Transparency

Req 39	When interacting with an AI component, the user must be informed that the interaction is not with a natural person, but with an artificial intelligence.
--------	--

If a system has the functionality of an AI component in the form of a large language model (e.g. a chat bot) and it interacts with a user, the user must be informed that it is not a natural person, but an artificial intelligence and that the answers may be imprecise.

Motivation: A user of a large language model must be aware at all times that he is interacting with an artificial intelligence and not with a real person. This can lead to the model's answers being imprecise or illogical. So-called "hallucinations" can occur at any time and influence the conversation with the user. That's why it's important to point out any inaccuracies to the user at the beginning of the interaction.

For this requirement the following warranty objectives are relevant:

- Integrity
- Transparency

Req 40	Data transmitted to an AI-Component must be preprocessed before transmission. This includes denoising algorithms for images, videos and speech as well as similar preprocessing for inputs to other AI-Components, such as text-based algorithms.
--------	---

This preprocessing includes various steps that must be adapted to the type of input data (images, video, speech, text, etc.). This preprocessing ensures that the data is available in a form that can be analyzed optimally by the AI component, thus making manipulation-based attacks more difficult.

Examples for pre-processing:

1. Images and Video

- Denoising algorithms
- Scaling to uniform size
- Contrast adjustments
- Color correction
- Cropping of relevant areas

2. Speech

- Removal of background noise
- Normalization of speech volume
- Restriction of frequency ranges

3. Text

- Removal of potentially malicious words, phrases, or sentences

Motivation: By pre-processing the data, potential malicious user input can be filtered out before it reaches the AI component.

For this requirement the following threats are relevant:

- Unnoticeable feasible attacks

For this requirement the following warranty objectives are relevant:

- Availability
- Confidentiality
- Integrity

Req 41	It must be ensured that no confidential data is transferred by the system or the user to the AI components unless this is explicitly provided for. This includes both personal data and other information requiring protection.
--------	---

It must be ensured that no personal or other data requiring protection is transferred to AI-Components by the system or the user unless this is explicitly provided for. This requires careful review and control of data flows within the system to ensure that the confidentiality and integrity of the data is maintained at all times. In any case, confidential data should be anonymized before it is transferred to an AI-Component.

Regulation for the classification of information: <https://informationsdrehscheibe.telekom.de>

Motivation: All information transferred to AI-Components could be viewed by the operator or third parties and also used outside the agreed processing contract.

The best protection against unwanted use of confidential data is to hand over such data only to systems that are explicitly designed to process it.

For this requirement the following threats are relevant:

- Unauthorized access or tapping of data

For this requirement the following warranty objectives are relevant:

- Confidentiality

Req 42	The system must ensure that only data that meets the requirements of the AI-Components is transferred. Thus, for example, no .exe programs may be accepted as user input for an image recognition algorithm.
--------	--

In order for the AI-Component to operate effectively and securely, only data that meets the requirements for secure operation of this component may be transferred to it. It is important to avoid certain file formats or types of data to prevent incorrect or inaccurate processing. For example, .exe programs should not be accepted as input to an image recognition algorithm, as this type of data is incompatible with the algorithm's operation and can lead to undesirable results.

Motivation: Inadequate input data can have far-reaching consequences when processed. Input validation is intended to prevent undefined behavior from occurring in the AI-Component.

For this requirement the following threats are relevant:

- Disruption of availability

For this requirement the following warranty objectives are relevant:

- Integrity

Req 43	A responsible person must be named who is responsible for the decisions of the machine learning algorithm so that a clear assignment is possible in the event of errors or damage.
--------	--

AI-Components can cause great damage to a company in the event of misuse or in the event of an error. Therefore, it is important to ensure a clear assignment of responsibility.

Instructions and processes for all conceivable error scenarios must be defined in advance. Ethical and legal issues must also be clarified by a responsible person or department before commissioning and then updated on an ongoing

basis.

The designation of a responsible person, or an area, with clear indication of the contact details, is therefore necessary.

Motivation: Clear accountability for the use of AI-Components is important to ensure that in the event of errors or damage, a clear assignment is possible and thus quick action can reduce the damage that occurs.

For this requirement the following threats are relevant:

- Denial of executed activities

For this requirement the following warranty objectives are relevant:

- Transparency

ID: 3.81-43/4.0

Req 44	Predefined authentication attributes must be changed.
--------	---

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

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

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

For this requirement the following threats are relevant:

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

ID: 3.01-8/9.0

4. Technical Baseline Security for IT/NT Systems

4.1. Technical and organizational prerequisites for the use of components in the system

Req 45	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

ID: 3.01-1/9.0

4.2. Basic System Hardening

Req 46	Only required software may be used on the system.
--------	---

In the installation routines for software provided by the supplier, individual components of the software are often preselected as standard installations, which are not necessary for the operation and function of a specific system. This

also includes parts of software that are installed as application examples (e.g. default web pages, sample databases, test data), but are typically not used afterwards.

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

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

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

For this requirement the following threats are relevant:

- Unauthorized use of services or resources
- Attacks motivated and facilitated by information disclosure or visible security weaknesses

ID: 3.01-3/9.0

Req 47	Features that are not required in the software and hardware used must be deactivated.
--------	---

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

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

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

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

Implementation example: [Example 1]

Deactivation of debugging functions in the software that are used in the event of fault analysis, but do not have to be active during normal operation.

[Example 2]

Disabling unused network interfaces of a server.

For this requirement the following threats are relevant:

- Unauthorized use of services or resources
- Attacks motivated and facilitated by information disclosure or visible security weaknesses

ID: 3.01-4/9.0

Req 48	Unnecessary services must be disabled.
--------	--

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

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

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

Motivation: Active services that are not required unnecessarily increase the attack surface of a system and, as a direct

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

For this requirement the following threats are relevant:

- Unauthorized use of services or resources
- Attacks motivated and facilitated by information disclosure or visible security weaknesses

ID: 3.01-5/9.0

Req 49 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 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.

For this requirement the following threats are relevant:

- Unauthorized use of services or resources
- Attacks motivated and facilitated by information disclosure or visible security weaknesses

ID: 3.01-6/9.0

Req 50 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 immediately 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.

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

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

For this requirement the following threats are relevant:

- Unauthorized access to the system

- Unauthorized use of services or resources
- Attacks motivated and facilitated by information disclosure or visible security weaknesses

ID: 3.01-7/9.0

Req 51 Known vulnerabilities in the software or hardware of the system must be fixed or protected against misuse.

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

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

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

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

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

Additional information:

The primary sources of known vulnerabilities in software/hardware are lists in the release notes as well as the security advisories from the official reporting channels of the supplier or independent CERTs. In particular, the reporting channels are sensibly integrated into continuous processes of security patch management for a system, so that newly discovered vulnerabilities can be registered promptly and led into operational remedial measures.

As a complementary measure to the detection of potentially still contained types of vulnerabilities that have in principle already become known, targeted vulnerability investigations of the system can be carried out. Particularly specialized tools such as automated vulnerability scanners are suitable for this purpose. Examples include: Tenable Nessus, Qualys Scanner Appliance.

For this requirement the following threats are relevant:

- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources
- Disruption of availability
- Denial of executed activities
- Unnoticeable feasible attacks
- Attacks motivated and facilitated by information disclosure or visible security weaknesses

ID: 3.01-10/9.0

4.3. Protecting availability and integrity

Req 52 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.

For this requirement the following threats are relevant:

- Unauthorized use of services or resources
- Disruption of availability

ID: 3.01-12/9.0

Req 53 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

ID: 3.01-13/9.0

4.4. Protecting data and information

Req 54 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 for the transmission of data must be recognized and effective 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.

For this requirement the following threats are relevant:

- Unauthorized access or tapping of data
- Unauthorized modification of data
- Disruption of availability
- Unnoticeable feasible attacks
- Attacks motivated and facilitated by information disclosure or visible security weaknesses

ID: 3.01-15/9.0

4.5. Session Protection

Req 55	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.

For this requirement the following threats are relevant:

- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources
- Denial of executed activities

ID: 3.01-16/9.0

Req 56 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.

For this requirement the following threats are relevant:

- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources
- Denial of executed activities

ID: 3.01-17/9.0

Req 57 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.

For this requirement the following threats are relevant:

- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources
- Denial of executed activities

ID: 3.01-18/9.0

4.6. Authentication and authorization

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 data and system areas in need of protection.

For this requirement the following threats are relevant:

- Unauthorized access to the system
- Unauthorized access or tapping of data
- Unauthorized modification of data
- Unauthorized use of services or resources
- Denial of executed activities

ID: 3.01-23/9.0

4.7. Logging

Req 59	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" = "Universal Time Coordinated").

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 ("NetworkTimeProtocol") Server on the IP network
- DCF77 radio signal received via a physically connected receiver
- GPS radio signal received via a physically connected receiver

For this requirement the following threats are relevant:

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

ID: 3.01-32/9.0

Req 60	Security-relevant logging data must be provided to a security incident detection and response solution in near-real-time.
--------	---

An acceptable detection and response solution for the purposes of this requirement must meet the following criteria:

- There is a continuous analysis and correlation of the logging data provided by connected IT/NT Systems.
- The identification of threatening activities and ongoing attacks takes place on the basis of a continuously updated detection pattern database
- Integrated alerting processes trigger the initiation of countermeasures in the event of a find.
- Security experts are actively involved in the operation of the solution as well as the evaluation of alarms

Such a solution is not the same as a log server in terms of its function. Accordingly, the provision to a detection and response solution must always be in addition to an existing forwarding to a log server. An exception exists if the log server used provides the logging data to a detection and response solution immediately after receipt. In this case, it is sufficient for the system to forward the logging data only to the log server.

The logging data must be provided to the detection and response solution within 5 minutes ("near-real-time") from the time it is generated in the system.

The specific technical interfaces and protocols to be supported for deployment are determined by the detection and response solution available in the system's operating environment as well as the technical basis of the system itself.

Motivation: IT/NT systems are constantly exposed to security threats and attacks. By means of a central analysis and the correlation of recorded system events of entire IT landscapes in almost real time, current threats and ongoing attacks can be detected even beyond system boundaries and appropriate countermeasures can be initiated promptly. The prompt delivery of logging data to the detection and response solution is essential for its effectiveness.

Implementation example: **Example 1:** A system based on a classic server collects security-relevant logging data in the operating system's local syslog or Windows Event Log. Logging data is continuously forwarded to a Security Information and Event Management (SIEM) for further analysis, intrusion detection, and alarm handling.

Example 2: An Endpoint Detection and Response (EDR) agent is running on a system. The EDR agent continuously forwards the security-related logging data it collects on the system to the central backend of the EDR solution, where further analysis, intrusion detection, and alarm handling takes place

Example 3: Network components provide events to a SIEM that make it possible to detect suspicious communication behavior such as network scans, unauthorized data flows between network segments, or active connections or connection attempts to suspicious destinations on the Internet. In particular, gateway routers from the CNDTAG provide "Netflow" logs, while forward proxies report access to domains and/or IP addresses and firewalls log blocked connection attempts.

Example 4: Systems designed on the basis of cloud-native functions generate logging data directly from these functions using the cloud's own logging and monitoring infrastructure. In such a system, the generation of security-relevant logging data is specifically activated in the corresponding configuration options of the cloud-native functions used. The resulting logging data is continuously transferred to a SIEM (Security Information and Event Management) for further analysis, intrusion detection and alarm handling. Some cloud providers already provide some attack pattern detection in the cloud's own monitoring infrastructure and generate specific cloud alerts from it. These are also transferred to the SIEM.

Example 5: A system is based on a SaaS ("Software as a Service") application from a cloud provider. Typically, in the SaaS model, the cloud provider is responsible for all operational aspects of the application, including security management. The cloud customer itself has no influence on the underlying technical infrastructure or the operation of the application. Based on contractual agreements, recognized certificates, and documented self-disclosures from the cloud provider, it is formally verified that the cloud provider has implemented a security incident detection and response solution. Some cloud providers also generate dedicated logging data about security-relevant activities within the separate customer tenants of their SaaS application (e.g. changes to configuration settings, permission assignments or user accounts within the tenants) and make them available to cloud customers. These so-called audit logs are continuously transferred to a SIEM (Security Information and Event Management) for further analysis, attack detection and alarm handling.

For this requirement the following threats are relevant:

- Unnoticeable feasible attacks

ID: 3.01-37/9.0