Security requirement

MySQL /MariaDB database systems

Deutsche Telekom Group

Version	6.0
Date	Dec 1, 2023
Status	Released

Publication Details

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

Reuterstrasse 65, 53315 Bonn
Germany

File name	Document number	Document type
	3.24	Security requirement

Version	State	Status
6.0	Dec 1, 2023	Released

Contact	Validity	Released by
Telekom Security psa.telekom.de	Dec 1, 2023 - Nov 30, 2028	Stefan Pütz, Leiter SEC-T-TST

Summary
Security requirements for MySQL database systems and offical MariaDB fork

Table of Contents

1. Introduction

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

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

2. General requirements for MySQL / Maria database systems

Req 1	The MySQL-DB / MariaDB version must be in the active or extended lifecycle period as per the MySQL Lifecycle Policy or in case of Maria DB offical supported by commercial supporting companies.

An overview of the current MySQL Lifecycle Policy can be found at http://www.mysql.com/about/legal/lifecycle/. In addition, the platforms supported in the MySQL Enterprise Edition must be within the MySQL lifecycle (active support or extended support). A directory of the platforms supported can be found at: http://www.mysql.com/support/supportedplatforms/enterprise.html. As it stands (May 2011), version 5.0 or higher applies.

Motivation: Oracle offers special support agreements as part of lifecycle support. They are combined with services for fault clearance and the clearance of security vulnerabilities.

Implementation example: The MySQL-DB version installed and its type (Community or Enterprise) can be queried using the following commands:

```
show variables like “%version%”;
status;
select version();
```

For this requirement the following threats are relevant:

- Unauthorized access or tapping of data
- Unauthorized modification of data
- Disruption of availability
- Unnoticeable feasible attacks

For this requirement the following warranty objectives are relevant:

ID: 3.24-1/6.0

3. Database system hardening

This section defines the requirements for hardening the database system. The measures are comparable with those for operating system hardening. They reduce the likelihood of successful attacks on database systems and/or the consequences thereof. On UNIX/Linux operating systems, the “mysql_secure_installation” script is provided for hardening MySQL-DB. This bundles hardening measures such as:

- Setting a password for the super user account of the MySQL database system.
- Setting the login rights to local host only for the MySQL ROOT user.
- Deleting anonymous user accounts.
- Deleting the “test” database and its privileges.

It is recommended that the MySQL-DB is run in a chroot-enviroment. This can be accomplished by either starting with the option --chroot or adding the making the relevant entry in the my.conf file.

Req 2	The “test” database must be deleted.

Motivation: Any unnecessary information on the system presents a security risk. Attackers could gain additional information about the system from this.

Implementation example: Checking for the existence of and deleting the “test” database:

```
show databases like “test”;
drop database “test”;
```

On Unix/Linux operating systems, running the “mysql_secure_installation” script when the default database is installed deletes the “test” database. Running the http://dev.mysql.com/doc/refman/5.5/en/mysql-secure-installation.html script on Linux generally hardens the database and implements the following measures:

- Setting a password for the super user account of the MySQL database system.
- Setting the login rights to local host only for the MySQL ROOT user.
- Deleting anonymous user accounts.
- Deleting the “test” database and its privileges.

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

For this requirement the following warranty objectives are relevant:

ID: 3.24-2/6.0

Req 3	It must be ensured that no user accounts without a user name exist (anonymous accounts).

“Anonymous accounts” are user accounts with a blank (“”) user name.

Motivation: By default, logins to these user accounts are possible and the rights to these can be used by other users under certain circumstances.

Implementation example: Running the SQL command DROP USER ""; deletes relevant user accounts. On Unix/Linux operating systems, running the “mysql_secure_installation” script during the installation deletes the anonymous accounts.
Check: The use of anonymous accounts can be checked using the following SQL command:

```
select user from mysql.user where user = “”;
```

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

For this requirement the following warranty objectives are relevant:

ID: 3.24-3/6.0

Req 4	Only one instance of the MySQL / Maria database system must be installed on one operating-system instance (hardware platform or virtualization guest).

Motivation: If multiple database instances for different tasks (e.g., Internet and intranet) are running on one operating-system instance, the instances are not separated from one another. There is a risk that attackers could also corrupt the second database system. To separate the individual database instances deploy virtualization solutions.

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

For this requirement the following warranty objectives are relevant:

ID: 3.24-4/6.0

Req 5	(Default) passwords on (SUPER) user accounts must be changed.

With a standard installation on a Unix/Linux or Windows system, the system creates a super user account without a password or “root”.

Motivation: Default passwords for databases present a high security risk. The administrator therefore needs to change these.

Implementation example: On Unix/Linux operating systems, running the “mysql_secure_installation” script sets the SUPER user account (root) password.
The following SQL command can be used to check whether user accounts without passwords exist:

```
select user, password from mysql.user where length (password) =
0 or password = “”;
```

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

For this requirement the following warranty objectives are relevant:

ID: 3.24-5/6.0

Req 6	MySQL / Maria DB must be startet with the Option --safe-user-create.

It is important to prevent the GRANT process for a new user with a blank password. The administrator must therefore start the MySQL daemon using the following option:

```
--safe-user-create
```

Motivation: Blank passwords for databases present a high security risk. The administrator therefore needs to change these.

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

For this requirement the following warranty objectives are relevant:

ID: 3.24-6/6.0

Req 7 The start option “old-passwords” must not be used.

Motivation: The administrator must use the highest available hash strength in order to make brute force attacks difficult. Short password hashes (such as before version 4.1), on the other hand, have minimal security.

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

For this requirement the following warranty objectives are relevant:

ID: 3.24-7/6.0

Req 8 The start option secure-auth must be used.

Motivation: The administrator must use the strongest hashing algorithm available to block brute force attacks. Short hashes (as existing before version 4.1) do not offer sufficient security.

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

For this requirement the following warranty objectives are relevant:

ID: 3.24-8/6.0

Req 9 The variable MYSQL_PWD must not be used when saving passwords and/or in scripts.

Motivation: The use of MYSQL_PWD is highly insecure since the password can be read out with the command “ps”.

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

For this requirement the following warranty objectives are relevant:

ID: 3.24-9/6.0

Req 10	If used wit Unix/Linux the database daemon must run with a dedicated, non-administrative Unix/Linux account which is used on the system only for the database.

Motivation: The DB daemon must not run with root rights since otherwise every DB user can create files with FILE privileges.

Implementation example: The read and write access to the database directory must be restricted to database users. If the daemon starts with the dedicated mysql user account, the following entry should be made in the my.cnf file:

```
[mysqld]
user=mysql
```

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

For this requirement the following warranty objectives are relevant:

ID: 3.24-10/6.0

Req 11	If used with Windows, the MySQL-DB / MariaDB must run with restricted privileges under a network service account.

Motivation: The MySQL user, as the network service account, only has restricted rights to the operating system. This reduces the impact of security gaps in the database.

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

For this requirement the following warranty objectives are relevant:

ID: 3.24-11/6.0

Req 12	Unless it is needed, the symlink function must be deactivated.

The symlink function enables a user to access the server’s data directory.

Motivation: Every user with write access to the server’s data directory can delete any files on the system.

Implementation example: To deactivate the symlink function, the administrator must start the MySQL-DB using the following option,

```
--skip-symbolic-links
```

The status of the symlink function can be queried using the SQL command
show variables like “have_symlink”;
This is deactivated if the system returns “Disabled” as the result.

For this requirement the following threats are relevant:

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

For this requirement the following warranty objectives are relevant:

ID: 3.24-12/6.0

Req 13 The FILE privilege must only be restricted to the super user account.

Motivation: Users with FILE privileges can read all files on the host which can also be read by the MySQL server. Thus a user can create new files in every directory to which the MySQL server has write access.

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

For this requirement the following warranty objectives are relevant:

ID: 3.24-13/6.0

Req 14 The read and write access for the FILE privilege must be limited to a defined directory.

Motivation: Users with FILE privileges can read all files on the host which can also be read by the MySQL server. Thus a user can create new files in every directory to which the MySQL server has write access.

Implementation example: The start option,

```
--secure-file-priv
```

limits the read and write access of LOAD FILE and LOAD DATA, as well as select...-into outfile commands to a predefined directory.

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

For this requirement the following warranty objectives are relevant:

ID: 3.24-14/6.0

Req 15 The database must be started with the option --local-infile=0.

Motivation: With this option, no user can transfer files from the client system to the server via LOAD DATA

Implementation example: The administrator must start the MySQL-DB with the following option (or transfer to the configuration file my.cnf):

```
--local-infile=0
```

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

For this requirement the following warranty objectives are relevant:

ID: 3.24-15/6.0

Req 16	It must be ensured that only the MySQL/MariaDB user account is granted read and write access to all MySQL/MariaDB data and its subdirectories, as well as to log files and database files.

Depending on the installation type, these directories are available for standard Windows or Unix binary installations: $MYSQL_HOME/data; Linux RPM distribution: /var/lib/mysql; source distribution: /usr/local/var.

Motivation: Restricting the file permissions means that the information gained after successfully taking over the system is significantly reduced.

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

For this requirement the following warranty objectives are relevant:

ID: 3.24-16/6.0

Req 17	It must be ensured that only the MySQL / MariaDB users and authorized administrators have read and write access to query and binary log files.

The system saves executed commands of the MySQL-DB (system, DDL, DML, DCL and SELECT) in the log files without encryption.

Motivation: Restricting the file permissions means that the information gained after successfully taking over the system is significantly reduced.

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

For this requirement the following warranty objectives are relevant:

ID: 3.24-17/6.0

Req 18	If used with Linux, it must be ensured that only the MySQL / MariaDB users have read and write access to the file “mysql_history”.

Motivation: The file “~mysql/.mysql_history” contains unencrypted passwords and DDL commands. If the user accounts are managed via SQL commands such as CREATE USER, GRANT and SET PASSWORD, the passwords appear in plain text.

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

For this requirement the following warranty objectives are relevant:

ID: 3.24-18/6.0

Req 19	If a password is used as an authentication attribute, it must have at least 12 characters and contain three of the following categories: lower-case letters, upper-case letters, digits and special characters.

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

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

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

When a password is assigned, the system must ensure that the password meets these policies. This must be preferably enforced by technical measures; if such cannot be implemented, organizational measures must be established.
If a central system is used for user authentication [see also Root Security Requirements Document[i] "3.69 IAM (Identity Access Management) - Framework"], it is valid to forward or delegate this task to that central system.

Permissible deviation in the password minimum length

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

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

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

For this requirement the following threats are relevant:

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

For this requirement the following warranty objectives are relevant:

ID: 3.01-26/7.0

Req 20	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

For this requirement the following warranty objectives are relevant:

ID: 3.01-27/7.0

Req 21	If a password is used as an authentication attribute, it must be changed after 12 months at the latest.

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

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

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

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

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

For this requirement the following warranty objectives are relevant:

ID: 3.01-30/7.0

4. MySQL / Maria -specific requirements

Req 22	It must be ensured that only administrative accounts have access to the MySQL-DB “user table”.

Motivation: The user table contains all user information. It is therefore extremely important that it is specially protected. Only then can further attacks on the system be reduced.

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

For this requirement the following warranty objectives are relevant:

ID: 3.24-22/6.0

Req 23	It must be ensured that only authorized accounts have access to the MySQL / Maria database tables.

Motivation: The database table contains all database information. It is therefore extremely important that it is specially protected. Only then can further attacks on the system be reduced.

Implementation example: The administrator must start MySQL-DB with the following option:

```
--skip-show-database
```

This option can also be added to the file my.cnf. With this option, only those users with the “show databases” authorization may use the instruction “show databases”. This privilege should therefore only be granted to specific users.

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

For this requirement the following warranty objectives are relevant:

ID: 3.24-23/6.0

Req 24	The MySQL / Maria database authorization system must not be deactivated.

Motivation: Only if a rights system is implemented is it possible to protect the data and the system as a whole in the long term.

Implementation example: It must be ensured that the database system is NOT started using the following option:

```
--skip-grant-tables
```

This start option deactivates the entire authorization system so that every database user is granted unrestricted access to all databases. If the procedure for restoring the root password is required, the start options must only be changed by the system administrator. It is therefore important to monitor changes in the configuration files as well as daemon restarts using the modified settings.

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

For this requirement the following warranty objectives are relevant:

ID: 3.24-24/6.0

Req 25 The GRANT option “WITH GRANT” must not be used.

Using the option “WITH GRANT” means that the user with the relevant privilege can transfer his privileges to other users or roles.

Motivation: Only if a rights system is implemented is it possible to protect the data and the system as a whole in the long term. Forwarding your own rights increases the risk of the unauthorized use of data.

Implementation example: The users with this privilege can be listed using the following command:

```
select user, host from mysql.user where grant_priv =“y”;
```

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

For this requirement the following warranty objectives are relevant:

ID: 3.24-25/6.0

Req 26 Predominantly administrative privileges must only be assigned for DB administrator accounts (principle of least privilege).

The granting of the following privileges must remain restricted to the DB administrator account:

- SUPER
- PROCESS
- SHUTDOWN
- CREATE USER
- RELOAD
- SHOW DATABASES
- GLOBAL GRANT OPTION
- CREATE TEMPORARY TABLES
- LOCK TABLES

Motivation: Non-administrative users who have such privileges may jeopardize the confidentiality, integrity and availability of the MySQL-DB.

Implementation example: With the command show grants an administrator can check the privileges granted. A user with these privileges could procure a list of executable queries which contain sensitive data such as passwords. Furthermore, he can get read and write access to all files in the mysqld process or configure the DB differently.

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

For this requirement the following warranty objectives are relevant:
- Confidentiality
- Integrity
- Availability

ID: 3.24-26/6.0

Req 27	The “privilege tables” must be checked after every upgrade.

Motivation: Sometimes privileges are added to new MySQL versions, which did not exist in the previous versions. For this reason, the administrator must check the privilege tables for correct assignment of privileges after every upgrade. This also means that incompatibilities with the current MySQL version can be identified.

Implementation example: MySQL has a script for checking the tables and privileges after an upgrade. An executable version of “mysql_upgrade” will be installed on all platforms in all MySQL versions higher than 5.1.10. The script checks/repairs the tables (mysqcheck) and checks the privilege tables (mysq_fix_privilege_tables). With older versions the administrator must issue these commands manually. A description of all commands can be found at http://dev.mysql.com/doc/refman/5.5/en/mysql-upgrade.html.

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

For this requirement the following warranty objectives are relevant:

ID: 3.24-27/6.0

Req 28	The start option --allow-suspicious-udfs must be deactivated.

This option determines whether user-defined functions (UDFs) can be loaded at the start. This function is deactivated by default.

Motivation: User defined functions can run malicious code on the system or be carrier functions for malicious code. They must therefore be deactivated.

Implementation example: Check: It is important to check whether the following option is used in the my.cnf configuration file:

```
    --allow-suspicious-udfs
```

If this option is selected, the administrator must deselect it.

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

For this requirement the following warranty objectives are relevant:

ID: 3.24-28/6.0

Req 29	It must be ensured that there are no wildcards (“%”) in the host name during the user authentication.

This ensures that only connections from trustworthy sources are possible.

Motivation: To reduce the risk of the accounts of compromised systems jeopardizing the database system, permissions must be stringently assigned.

Implementation example: Check: The use of wildcards can be checked using the SQL command:

```
select user from mysql.user where host = "%";
```

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

For this requirement the following warranty objectives are relevant:

ID: 3.24-29/6.0

Req 30 It must be ensured that the super user can only log in to the local host.

Motivation: To reduce the risk of the accounts of compromised systems jeopardizing the database system, permissions must be stringently assigned. This includes the administration of the database itself.

Implementation example: On Unix/Linux operating systems, running the "mysql_secure_installation" script upon installation restricts the access permissions of the super user account to the local host.

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

For this requirement the following warranty objectives are relevant:

ID: 3.24-30/6.0

Req 31 The resources available per user account must be limited.

Motivation: To reduce the risk of compromised accounts jeopardizing the database system, permissions should be stringently assigned.

Implementation example: The available server resources can be limited using the variables below. This is possible both globally and for each individual user.

- MAX_QUESTIONS
- MAX_USER_CONNECTIONS
- MAX_QUERIES_PER_HOUR
- MAX_UPDATES_PER_HOUR
- MAX_CONNECTIONS_PER_HOUR

Further information can be found at http://dev.mysql.com/doc/refman/5.5/en/user-resources.html.

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

For this requirement the following warranty objectives are relevant:

5. Logging

Req 32	Accesses to database systems, as well as critical database procedures and database content must be logged.

Secure, traceable database operation requires important operating information to be logged. This includes, for instance, the logging of failed login attempts to uncover possible intrusion attempts.
Logging of security-relevant user actions shall comply with national legislation currently in force.
When implementing measures resulting from this Requirement, the applicable participation rights of the responsible employee representatives/trade unions as well as the works and collective agreements shall be observed.

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

For this requirement the following warranty objectives are relevant:

ID: 3.16-26/6.0

Req 33	Security relevant events must be logged with a precise timestamp and a unique system reference.

Systems must log the occurrence of security-relevant incidents. So that these events can be evaluated and classified, they must be logged together with a unique system reference (e.g., host name, IP or MAC address) and the exact time the incident occurred ("Timestamp").

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

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

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

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

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

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

Event	**Event data to be logged**
Incorrect login attempts	• User account, • Number of failed attempts, • Source (IP address, client ID / client name) of remote access

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

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

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

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

For this requirement the following warranty objectives are relevant:

ID: 3.01-33/7.0

Req 34	Applicable retention and deletion periods must be observed for security-relevant logging data that is recorded locally.

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

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

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

Deviances

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

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

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

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

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data
- Denial of executed activities
- Unnoticeable feasible attacks

For this requirement the following warranty objectives are relevant:

ID: 3.01-34/7.0

Req 35	Security-relevant logging data must be forwarded to a separate log server immediately after it has been generated.

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

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

For this requirement the following threats are relevant:
- Unauthorized modification of data
- Disruption of availability
- Denial of executed activities
- Unnoticeable feasible attacks

For this requirement the following warranty objectives are relevant:

ID: 3.01-35/7.0

Req 36	For security-relevant logging data that is forwarded to the separate log server, compliance with the applicable retention and deletion periods must be ensured.

The following basic rules must be taken into account:

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

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

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

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

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

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

For this requirement the following threats are relevant:
- Unauthorized access or tapping of data
- Denial of executed activities
- Unnoticeable feasible attacks

For this requirement the following warranty objectives are relevant:

ID: 3.01-36/7.0

Req 37	The system must provide logging data that is required to detect the system-specific relevant forms of attack in a SIEM.

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

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

The logging data must be sent to a SIEM immediately after the system event occurs.
SIEM (Security Information & Event Management) solutions collect event log data from various source systems, correlate it and evaluate it automatically in real time in order to detect anomalous activities such as ongoing attacks on IT/

NT systems and to be able to initiate alarms or countermeasures.
The immediate receipt of system events is therefore absolutely crucial for the SIEM to fulfill its protective functions.

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

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

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

ID: 3.01-37/7.0

Req 38	Important database services and instances must be monitored continually for misuse scenarios.

The monitoring of user actions for misuse shall comply with national legislation currently in force (for details see the “Security Requirement on Misuse Detection”).

Motivation: There are many conceivable ways to misuse database systems. Users generate an unusually high data usage rate or operate at unusual times of day. Attackers utilize unusual and critical commands for database queries, as well as tools and malware to extend their rights. To detect misuse, database systems should be continually monitored for misuse scenarios, e.g. by means of database triggers and log monitoring.

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

For this requirement the following warranty objectives are relevant:

ID: 3.16-32/6.0