Security requirement

Linux OS for Servers

Deutsche Telekom Group

Version	7.0
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
	3.65	Security requirement
Version	**State**	**Status**
7.0	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 Linux OS for servers inclusive requirments for iptables, mandatory access control solutions etc.

Table of Contents

1. Introduction

Linux is a popular operating system for server systems. It is used from single bare metal servers to high scaling cloud environments. This document includes security requirements for the base operating system and also for applications which are part of the operating system like, iptables, mandatory access control solutions etc. The security requirements are valid for all Linux distributions.

The security requirements have been prepared based on the general security policies of the Group in correlation with security best practices and vendor recommendations.

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.

A tool for hardening a Linux system according to requirements in this docuement can be found here.

1.1. Scope

In scope of this is document are server systems with Linux OS. This includes the following types of servers:

- bare metal servers
- virtualized servers (VMs)
- virtualization hosts

1.2. Not in Scope

This document is not intended to be used for workstations with Linux OS and container images based on Linux. Because of different use cases, associated threats and needed software, workstations need another set of requirements. Container images have to be treated differently as they normally do not include a full Linux OS.

2. System Hardening

Req 1	Unnecessary services must be disabled.

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

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

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

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

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

For this requirement the following warranty objectives are relevant:

ID: 3.01-5/7.0

Req 2	The accessibility of activated services must be restricted.

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

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

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

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

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

For this requirement the following warranty objectives are relevant:

ID: 3.01-6/7.0

Req 3	Unused software must not be installed or must be uninstalled.

Software could be installed during setup of Linux operating system which is not needed for the functionality of the server. Such software should not be installed or must be uninstalled after installation. Examples for software that is typically not needed on a server system are:

- inetd
- xinetd
- X Window System
- Avahi Server
- CUPS
- rsync service
- NIS server/client
- talk server/client
- telnet server/client
- tftp server/client
- ftp server/client
- rsh server/client

Note: It is not allowed to install software on a server that is not needed for operation, maintenance or general functionality!

Motivation: Vulnerabilities in software offer an attack window for attackers to infiltrate the system. By uninstalling not needed software the attack surface and the risk of a successful compromise can 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
- Disruption of availability

For this requirement the following warranty objectives are relevant:

ID: 3.65-3/7.0

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

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

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

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

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

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

[Example 2]

Disabling unused network interfaces of a server.

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

For this requirement the following warranty objectives are relevant:

ID: 3.01-4/7.0

Req 5	Dedicated partitions must be used for growing content that can influence the availability of the system.

An own partition must be created for directories that are used to store dynamic content. It is recommended to use a dedicated partition for the directories:

- /tmp
- /var

In specific cases it could be necessary to use the following partitions:

- /var/log und /var/tmp (anstelle von /var)
- /home (für File-Server mit einer großen Benutzerzahl)

Motivation: A filled filesystem can stop operation of a server. This can be triggered by an attacker to effect availability of a server.

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

For this requirement the following warranty objectives are relevant:

ID: 3.65-5/7.0

Req 6	Parameters nodev, nosuid and noexec must be set for partitions where this is applicable.

The named mount options must be set for the following partitions if they exist:

- /tmp (nodev, nosuid)
- /var/tmp (nodev, nosuid, noexec)
- /home (nodev)

Note: For installation reasons it could be necessary to remove 'noexec' from partition '/tmp' as this is used sometimes for script execution during software installation.

If separate partition exists also for:

- /dev/shm (RHEL, SLES) (nodev, nosuid, noexec)
- /run/shm (Ubuntu) (nodev, nosuid, noexec)

Motivation: It must be avoided for such partitions that an attacker can execute files with suid, to store device files and to save and execute files from this partition.

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

For this requirement the following warranty objectives are relevant:

ID: 3.65-6/7.0

Req 7	Automounting of filesystems using "`autofs`" must be disabled.

Automounting of file systems must be disabled to avoid the automated mounting and use of external file systems like USB sticks and CD-ROMs.

Motivation: With automounting enabled any external file system will be mounted to the server and can possibly misused.

Implementation example: Debian-based Linux:
```
$ sudo apt purge autofs
```
RedHat-based Linux:
```
$ sudo yum remove autofs
```

OR

```
$ sudo systemctl disable --now autofs
```

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

For this requirement the following warranty objectives are relevant:

ID: 3.65-7/7.0

Req 8	USB-Storage must be disabled.

Kernel module "usb-storage" serves as driver for USB mass storage device and must be prevented from being loaded by default. To accomplish this "modprobe" must be configured to not load "usb-storage" at system boot.
If need the module can be loaded at runtime, be used and be removed afterwards.

Motivation: USB storage devices are usually not used in productive server operation. Since they can be used to introduce malware or exfiltrate data they must be disabled.

Implementation example: `/etc/modprobe.conf` contains:
```
install usb-storage /bin/true
```

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

For this requirement the following warranty objectives are relevant:

ID: 3.65-8/7.0

Req 9	The use of "`at`" and "`cron`" must be restricted to authorized users.

The use of the tools 'cron' and 'at', that can be used to schedule automated execution of jobs on a Linux system, must be restricted to authorized users.

Motivation: Users can misuse these tools to execute jobs on a system.

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

For this requirement the following warranty objectives are relevant:

ID: 3.65-9/7.0

Req 10	Sticky bit must be set on all world-writable directories.

This feature prevents the ability to delete or rename files in world writable directories (such as /tmp) that are owned by another user.

Motivation: Setting the sticky bit on world writable directories prevents users from deleting or renaming files in that directory that are not owned by them.

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: 3.65-10/7.0

Req 11	No regular files that are world writable must exist.

World writable files are files that have write permission set for other. These files are writable by any user of the server. Such files must be detected and if existing the rights of these files must be changed to an adequate level.

Motivation: Data in world-writable files can be read, modified, and potentially compromised by any user on the system.

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: 3.65-11/7.0

Req 12	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

For this requirement the following warranty objectives are relevant:

ID: 3.01-18/7.0

Req 13	The default user "umask" must be 027 or more restrictive.

The setting of the umask defines which mode files or directories get when they are created by a user. The default umask on most Linux systems is less strict. This is the reason why a stricter umask must be configured.

A umask of 027 is required. This defines the permissions 'read, write, execute' (0) for the user, 'read, execute' (2) for group and no permissions (7) for others.

Motivation: With a strict umask the manipulation of files by unauthorized users can be prevented.

Implementation example: For systems with `bash` and `sh` only `/etc/profile.d/set_umask.sh` contains:

```
umask 027
```

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: 3.65-13/7.0

Req 14	Not needed SUID and SGID bits must be removed from executables.

Executables with SUID or SGID bits set run with extensive rights. Such executables pose a security risk. Therefore, executables with SUID and SGID bit set must be limited to the absolutly needed ones. From all others, the SUID and SGID bits must be removed.

An alternative is to grant more granular permission for such commands with Posix capabilities. This solution allows to enable only needed system functions for a binary file and not full root privileges as with SUID/SGID. It is highly recommended where even possible to use Posix capabilities instead of SUID/SGID!

The following executables are allowed to run with SUID and SGID if not Posix capabilities can be used:

- /bin/ping
- /sbin/pam_timestamp_check
- /sbin/unix_chkpwd
- /usr/bin/at
- /usr/bin/gpasswd
- /usr/bin/locate
- /usr/bin/newgrp
- /usr/bin/passwd
- /usr/bin/ssh-agent

- /usr/libexec/utempter/utempter
- /usr/sbin/lockdev
- /usr/sbin/sendmail.sendmail
- /usr/bin/expiry
- /bin/ping6
- /usr/bin/traceroute6.iputils
- /sbin/mount.nfs
- /sbin/umount.nfs
- /sbin/mount.nfs4
- /sbin/umount.nfs4
- /usr/bin/crontab
- /usr/bin/wall
- /usr/bin/write
- /usr/bin/screen
- /usr/bin/mlocate
- /usr/bin/chage
- /usr/bin/chfn
- /usr/bin/chsh
- /bin/fusermount
- /usr/bin/pkexec
- /usr/bin/sudo
- /usr/bin/sudoedit
- /usr/sbin/postdrop
- /usr/sbin/postqueue
- /usr/sbin/suexec
- /usr/sbin/ccreds_validate
- /usr/lib/dbus-1.0/dbus-daemon-launch-helper
- /usr/lib/policykit-1/polkit-agent-helper-1

Motivation: Executables with SUID and SGID are a high risk for a system. If such an executable has a vulnerability it could possibly lead to compromise of the system.

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

For this requirement the following warranty objectives are relevant:

ID: 3.65-14/7.0

Req 15	Core dumps must be disabled and crash reporting softwre must be removed.

A core dump includes complete memory content of a crashed or killed executable program. It can contain data with need of protection like passwords or cryptographic keys. Crash reporting software in some cases overrules core dump configuration (e.g. "Apport", "abrt") and it can automatically send reports to its vendor, thus as the case may be dis-

close data with need of protection.
Core dumps must be disabled during normal operation and may only be enabled in case of debugging. Errer reporting software must be removed, in case it cannot be removed it must be disabled.
On systems where core dumps are needed it is necessary to disable core dumps for setuid (`fs.suid_dumpable=0`) processes.

Motivation: Core dumps and error reports can include data with need of protection. To avoid information leakage core dumps and crash reporting must be disabled.

Implementation example: 1) configure kernel option & systemd-coredump
`/etc/sysctl.d/50-coredump.conf` contains:
```
kernel.core_pattern=/dev/null
```

2) uninstall crash reporting software (because could otherwise overridde kernel.core_pattern) :
Debian example:
```
$ sudo apt remove reportbug
```
Ubuntu example:
```
$ sudo apt remove apport
```
Redhat example:
```
$ yum remove abrt
```

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

For this requirement the following warranty objectives are relevant:

ID: 3.65-15/7.0

Req 16	Protection against buffer overflows must be enabled.

A protection function against buffer overflow attacks must be used on Linux servers. The following Kernel option must be configured:

- kernel.randomize_va_space=2 (seekernel.org, default setting on most Linux distributions)

Precondition is Processor feature

- NoExecute (NX) respectively
- eXecute Disable (XD)

which must be activated in system BIOS. If existing Prelink must be disabled if the ASLR feature PIE (Position-independent executable) is used. This is a tool to modify ELF shared libraries and ELF dynamically linked binaries. In modern Linux distributions like RHEL, SLES Prelink is obsolete and no longer included.
Self-compiled software needs to be compiled with support for these security features. GCC default settings fulfill that.

Motivation: Buffer overflow attacks can be used to unauthorizedly execute code on a system to influence availability or to get full access to a system.

Implementation example: `/etc/sysctl.conf` contains:

```
kernel.randomize_va_space=2
```

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

For this requirement the following warranty objectives are relevant:

ID: 3.65-16/7.0

2.1. Network Hardening

Req 17	IPv4 protocol stack must be securely configured.

If IPv4 is not used it must be completely disabled. Otherwise, the IPv4 stack on Linux servers must be hardened. For this the following configuration must be implemented:

- IPv4 forwarding must be disabled.
- IPv4 redirects must not be accepted.
- Secure IPv4 redirects must not be accepted.
- IPv4 packet redirect sending must be disabled.
- IPv4 source routed packets must not be accepted.
- Suspicious packets must be logged
- Broadcast ICMP requests must be ignored.
- Bogus ICMP responses must be ignored.
- Reverse Path Filtering must be enabled.
- TCP SYN Cookies must be enabled.
- TCP Timestamps must use random time offset (since Kernel 4.12) or be disabled (prior Kernel 4.12).
- An ICMP ratelimit must be configured.
- ARP must be restricted.

Motivation: An unhardened IPv4 protocol stack is vulnerable against several attacks like denial of service, traffic high jacking.

Implementation example: Example for a Debian-based Distribution:

```
vim /etc/sysctl.d/80-ipv4-hardening.conf

net.ipv4.ip_forward=0
net.ipv4.conf.all.accept_redirects=0
net.ipv4.conf.default.accept_redirects=0
net.ipv4.conf.all.secure_redirects=0
net.ipv4.conf.default.secure_redirects=0
net.ipv4.conf.all.shared_media=0
net.ipv4.conf.default.shared_media=0
net.ipv4.conf.all.send_redirects=0
net.ipv4.conf.default.send_redirects=0
net.ipv4.conf.all.accept_source_route=0
net.ipv4.conf.default.accept_source_route=0
net.ipv4.conf.all.log_martians=1
net.ipv4.conf.default.log_martians=1
net.ipv4.icmp_echo_ignore_broadcasts=1
net.ipv4.icmp_ignore_bogus_error_responses=1
net.ipv4.conf.all.rp_filter=1
net.ipv4.conf.default.rp_filter=1
net.ipv4.tcp_syncookies=1
net.ipv4.tcp_timestamps=1
net.ipv4.icmp_ratelimit=100
net.ipv4.icmp_ratemask=88089
net.ipv4.conf.all.arp_ignore=2
net.ipv4.conf.default.arp_ignore=2
net.ipv4.conf.all.arp_announce=2
net.ipv4.conf.default.arp_announce=2
net.ipv4.conf.all.arp_notify=0
net.ipv4.conf.default.arp_notify=0
net.ipv4.conf.all.arp_accept=0
```

```
net.ipv4.conf.default.arp_accept=0
```

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

For this requirement the following warranty objectives are relevant:

ID: 3.65-17/7.0

Req 18	IPv6 protocol stack must be securely configured.

If IPv6 is not used it must be completely disabled. Otherwise, the IPv6 stack on Linux servers must be hardened. For this the following configuration must be implemented:

- IPv6 forwarding must be disabled.
- IPv6 redirects must not be accepted.
- IPv6 source routed packets must not be accepted.
- IPv6 router advertisements must not be accepted.
- IPv6 router solicitations messages must not be sent.
- IPv6 autoconfiguration must be disabled

Motivation: An un-hardened IPv6 protocol stack is vulnerable against several attacks like denial of service, traffic high jacking.

Implementation example: Example for a Debian-based Linux-Distribution:

```
/etc/sysctl.d/80-ipv6-hardening.conf

net.ipv6.conf.all.forwarding=0
net.ipv6.conf.all.accept_redirects=0
net.ipv6.conf.all.accept_source_route=0
net.ipv6.conf.all.accept_ra=0
net.ipv6.conf.all.router_solicitations=0
net.ipv6.conf.all.autoconf=0
net.ipv6.conf.default.forwarding=0
net.ipv6.conf.default.accept_redirects=0
net.ipv6.conf.default.accept_source_route=0
net.ipv6.conf.default.accept_ra=0
net.ipv6.conf.default.router_solicitations=0
net.ipv6.conf.default.autoconf=0
```

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

For this requirement the following warranty objectives are relevant:

ID: 3.65-18/7.0

2.2. System Update

Req 19	The software used must be obtained from trusted sources and checked for integrity.

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

This requirement applies to all types of software:

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

as well as other software that may be used.

Trusted Sources

Trusted sources are generally considered to be:

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

Integrity Check

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

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

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

Extended integrity checking when pulling software from public registries

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

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

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

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

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

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

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

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

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

For this requirement the following warranty objectives are relevant:

ID: 3.01-2/7.0

Req 20	Linux distributions which are supported by their vendor must be used. Absolutely necessary is security vulnerability support.

Releases of Linux distributions with active support by their creator (distributor, producer or developer) must be used. Releases which are marked as end-of-support (EOS), end-of-life (EOS) or something the like by their vendors must **not** be used. If available long term support (LTS) versions are to prefer.

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 given 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 restrictions. Some suppliers generally define their support in levels that can contain restrictions already in the last phase immediately before the absolute end date of regular sup-

port.
If a product is used within support phases that are subject to restrictions, 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 it) operates a comprehensive security vulnerability management for the respective product that meets the above criteria and is considered to be reliably established.

Motivation: Safe production can only be achieved with software that obtains timely security patches. Known vulnerabilities and Proof-of-Concepts to exploit them make systems without vendor support an easy target for attackers.

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

For this requirement the following warranty objectives are relevant:

ID: 3.65-20/7.0

Req 21	If needed, active software licenses must be installed to ensure security updates.

Motivation: Some operating system vendors license their products and require the purchase of licenses. These allow access to (security) updates. Without update options, a system may not be operated.

Implementation example: Red Hat Enterprise Linux for example can be registered with: `subscription-manager register`

ID: 3.65-22/7.0

Req 22	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 chan-

nels are sensibly integrated into continuous processes of security patch management for a system, so that newly discovered vulnerabilities can be registered promptly and led into operational remedial measures.
As a complementary measure to the detection of potentially still contained types of vulnerabilities that have in principle already become known, targeted vulnerability investigations of the system can be carried out. Particularly specialized tools such as automated vulnerability scanners are suitable for this purpose. Examples include: Tenable Nessus, Qualys Scanner Appliance.

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

For this requirement the following warranty objectives are relevant:

ID: 3.01-10/7.0

Req 23	GPG check for repository server must be activated and corresponding keys for trustable repositories must be configured.

GPG check must be enabled and keys must be configured properly to verify integrity during installation of software from a repository server. On RedHat Linux it is necessary to activate the gpgcheck globally.

Motivation: The GPG check is necessary to guarantee the authenticity of used source an integrity of software. Without this check an attacker could possibly manipulate software packets before installation.

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:
- Integrity

ID: 3.65-24/7.0

Req 24	The operating system must have an Endpoint Detection and Response (EDR) solution.

An up-to-date Endpoint Detection and Response (EDR) solution must be used on the operating system. An EDR solution collects security-relevant activity data from processes and evaluates them centrally. Alarms can be generated from malicious behavior of processes or by a specific signature (like classic virus scanners). In addition, it is possible to react directly to suspicious program behavior via a central console. For example, the operating system or rather the client can be isolated or the malicious process terminated. Furthermore, an overview of the vulnerabilities of all operatings systems is forwarded to a central location.

Motivation: Normal virus scanners rely purely on signatures, whereas an EDR solution also searches for anomalies in the process behavior. Signatures have the disadvantage that new signatures have to be written for new virus variants before they can be detected.

Implementation example: Have a look at PSA Wiki - Sicherheitsanforderungen : Endpoint Detection and Response (EDR) for guidance.

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

For this requirement the following warranty objectives are relevant:

ID: 3.65-25/7.0

2.3. User Authentication

Req 25	User accounts must ensure the unique identification of the user.

Users must be identified unambiguously by the system.

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

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

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

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

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

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

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

For this requirement the following warranty objectives are relevant:

ID: 3.01-22/7.0

Req 26	System accounts must be non-login.

On Linux servers, several users are available that are needed for functionality of applications. These users are not intended to provide a shell. To avoid that such accounts are used to login the shell parameter in file '/etc/passwd' must be set to '/usr/sbin/nologin' or '/bin/false'.

Note: The system accounts root, sync, shutdown and halt are excluded from this requirement!

Motivation: This prevents system accounts from potentially being misused by attackers to run commands.

Implementation example: To prevent such users from logging in shell in "`/etc/passwd`" can be set to "`/usr/sbin/nologin`" or "`/bin/false`".

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

For this requirement the following warranty objectives are relevant:

ID: 3.65-27/7.0

Req 27	User accounts must be protected with at least one authentication attribute.

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

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

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

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

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

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

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

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: 3.01-20/7.0

Req 28	Privileged user accounts must be protected with at least two authentication attributes from different factors.

A privileged user account is a user account with extended authorizations within a system. Extended authorizations en-

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

Examples of privileged user accounts:

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

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

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

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

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

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

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

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: 3.01-21/7.0

Req 29	Authentication must be used for single user mode.

For system recovery, the so called single user mode is used. This mode can also be manually selected from the bootloader during system boot. Authentication must be enabled for single user mode to protect this access. This is especially relevant, if the console of the machine can be reached remotely (e.g. via lights-out-management or via virtual console in case of virtual machines).

Motivation: Without authentication, an unauthorized user can gain root privileges by forcing a reboot.

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

For this requirement the following warranty objectives are relevant:

ID: 3.65-30/7.0

2.4. Administrative Access

Req 30	The administration of the operating system must be done via a network interface which is independent from the production network.

Administrative access to a server must not be done via an interface which provides productive services. Access must be limited to legitimate systems. The administration of applications can also be done using this network interface.

The restriction can be done with, e.g., filter mechanisms, local access lists or a packet filter. This limitation has to be done as restrictive as possible, i.e., limit to single IP addresses or at least small IP ranges.

Motivation: In the event of a successful attack, an attacker may gain access to confidential information or even to the entire system. By restricting the accessibility to legitimate systems, the group of potential attackers can be reduced, and thus also the likeliness of a successful attack. Furthermore, systems must be manageable even in the case the customer network is down.

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

For this requirement the following warranty objectives are relevant:

ID: 3.37-10/7.0

Req 31	Administrative services and accesses must be bound to only those interfaces that have been set up to administer.

The administrative services (interactive by persons or machine-to-machine, e.g., SSH, HTTPS, RDP) must be bound to the appropriate interface(s). Due to the separation of management traffic from user traffic, this is the IP address in the management network. If the system - or parts of it - is managed by more than one interface, the management services have to be bound to the lowest possible number.

Depending on the respective service, the access to these services must be restricted to a few, trustworthy, necessary target or source addresses.

Motivation: This ensures that it can be clearly foreseen under which address these management services are reachable. In addition, a unique address is important for implementing filters and firewall rules that restrict access to these management services to legitimate addresses.

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

For this requirement the following warranty objectives are relevant:

ID: 3.37-11/7.0

Req 32	Network based access used for operating system administration must have integrity protection, be encrypted and securely authenticated.

Access is only permitted by using secure protocols (e.g., SSHv2, HTTPS, SNMPv3). The administrator must ensure that any network connection between his workstation or a management system and the operating system to be administrated is securely authenticated, encrypted and protected against tampering.

Motivation: If the administrator transmits changes to the configuration settings via unencrypted or unsecure connections, there is a risk that unauthorized parties gather system information (configuration settings, access IDs, etc.) to exploit security vulnerabilities.

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

For this requirement the following warranty objectives are relevant:

ID: 3.37-12/7.0

2.5. Location

Req 33	If the system is not located in a room with at least protection class "high" (PC3), the BIOS and, if available, other options for local management must be secured against unauthorized access.

The Protection Classes (PC) are defined in the Annex 1, "Physical Security of Buildings", of the Group Policy on "Physical Security". Typically, Datacenters are compliant to the requirements of PC3.

Servers operated in public or customer areas must be especially protected against unauthorized access and changes: The BIOS settings must be protected against export and tampering. Are further access options to the system configuration possible, e.g. by Intel AMT, iLO, LOM, and others, these must be protected as well. In case passwords are used, these must be exclusive to the individual server and must not allow conclusions to be drawn about a distinguishing feature of the server.

The BIOS must be configured in such a way that only the designated operating system can be started with it from the designated partition.

Motivation: Motivation: Changing BIOS settings can facilitate attacks. Since, for example, local rooms with technical installations seldom offer access protection to the servers, attackers could change the startup sequence of data storage media when the server is started in the BIOS without the password protection described. This would make it possible to start an alternative operating system which circumvents the security mechanisms of the implemented operating system.

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

For this requirement the following warranty objectives are relevant:

ID: 3.37-20/7.0

Req 34	If the system is not located in a room with at least protection class "high" (PC3), used data storages must be fully encrypted.

The Protection Classes (PC) are defined in the Annex 1, "Physical Security of Buildings", of the Group Policy

on "Physical Security". Typically, Datacenters are compliant to the requirements of PC3.

Data storages are all disks and flash memory in the systems.

Motivation: Access to devices which are operated outside of data centers with protected access is relatively easy. Physical data storage media can be easily stolen as a result.

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

For this requirement the following warranty objectives are relevant:

ID: 3.65-35/7.0

3. Logging

Req 35	Auditing must be enabled at boot by setting a kernel parameter.

Each process on the system carries an "auditable" flag which indicates whether its activities can be audited. Although "auditd" takes care of enabling this for all processes which launch after it does, adding the kernel argument ensures it is set for every process during boot.

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

Implementation example: For Debian based distributions e.g. you can edit `/etc/default/grub` . Go for variable `GRUB_CMDLINE_LINUX_DEFAULT` and add "`audit=1`" for the kernel parameter to activate. After doing this change it is need to issue `update-grub` of course for the changes to take effect.

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: 3.65-36/7.0

Req 36	Log rotation for logfiles must be configured.

Log rotation for logfiles must be enabled. Files must be rotated based on file size and max lifetime of file.

Motivation: An attacker can trigger log events to fill up the disk space of the server. This could lead to a denial of service of the server.

Implementation example: logrotate configuration contains:# restrict maximum size with maxsize and rotate

```
# restrict maximum size with maxsize and rotate
maxsize 10M
rotate 100

# control maximum age with daily rotation and maxage
daily
maxage 90

# permissions
create 640 syslog adm

# secure deletion
shred
```

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

For this requirement the following warranty objectives are relevant:

ID: 3.65-37/7.0

Req 37	System time must be synchronized against a reference time source.

A service like NTP (client programs: ntp, crony) must be used for time synchronization. Example for Deutsche Telekom AG NTP server pools are:

- ntp-pool-info_ntpp10.telekom.de
- ntp-pool-info_ntpp21.telekom.de

Motivation: Time synchronization must be done on a server to support time sensitive security mechanisms like the usage of certificates or solutions like Kerberos. A precise time is also mandatory to ensures consistent time records for logging events.

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: 3.65-38/7.0

Req 38	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: Using logrotate old logs can be deleted after 90 days. Option "shred" makes sure data is deleted safely (at host level).

```
$ cat /etc/logrotate.d/any-log
/var/log/* {
  daily
  maxage 90
  shred
}
```

For this requirement the following threats are relevant:

- Unauthorized access or tapping of data

For this requirement the following warranty objectives are relevant:

ID: 3.65-39/7.0

Req 39	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

3.1. Auditd

The auditd subsystem is an access monitoring and accounting for Linux. It can be used to define granular log events to be monitored under Linux OS. From security point of view log events can be used to detect malicious activities and to analyze possible breaches.

Req 40	"auditd" service must be used to log security relevant events.

On Linux server, "auditd" must be installed and configured to log security relevant events. Every event must be logged with a precise timestamp and a unique system reference.
Logdata of "auditd" counts as security reelvant event that has to be forwarded to a central log server, as described in chapter "External Logging".

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: 3.65-41/7.0

Req 41	Syscalls "execve" (execute program) must be logged.

On Linux servers, the following System events must be logged:

Event Type	Description	Category
execution of programs	All program executions must be logged.	Mandatory

Motivation: Forensics & mitigation *: In case of compromise of a Linux host it needs to be comprehensible which actions have been taken.*

Implementation example: As example an Ubuntu 20.04 host could be configured like this:

```
$ cat /etc/audit/rules.d/execve.rules
-a exit,always -F arch=b64 -S execve
-a exit,always -F arch=b32 -S execve
```

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: 3.65-42/7.0

Req 42	System events must be logged.

On Linux servers, the following System events must be logged:

Event Type	Description	Category
System Startup and Shutdown	All restarts or shutdowns of the OS must be logged.	Mandatory
(Un)Installation of software	After the commissioning of the server, every uninstallation and installation of software must be logged.	Mandatory
Change of system time	Modification of the local system time and change of ntp settings must be logged.	Mandatory
Connection of external device (storage)	The connection of external devices like USB-Flash drives, which can mount on the running server, must be logged.	Mandatory

Privileged commands execution	The use of privileged commands with SUID/SGID must be logged.	Mandatory
Loading/unloading of kernel modules	The loading and unloading of kernel module must be logged.	Mandatory
Change of scheduled jobs	Jobs which are executed periodically, must be monitored, if they are changed or deleted.	Optional

Motivation: It is unusual to make system changes when the OS is put into operation. An attacker who has access to the server could change the system for its malicious purpose. The logging of the system events is necessary to detect and backtrack these attacks.

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: 3.65-43/7.0

Req 43	Access and Authentication events must be logged.

On Linux servers, the following Access and Authentication events must be logged:

Event Type	Description	Category
Logon and Logoff	The Logon and Logoff of a User via external or local access must be logged.	Mandatory
Password Change	User Password changes or resets must be logged.	Mandatory
Escalation of privileges (sudo/sudoers)	It must be logged, if a user executes sudo or changes corresponding configuration files (sudoers).	Mandatory
Modification of DAC permissions	Modification of discretionary access control permissions must be logged.	Mandatory

Motivation: The logging of authentication and access events can be useful to backtrack who has access to a certain time. With these logs it is for example possible to detect a captured account which is used by an attacker.

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: 3.65-44/7.0

Req 44	Account and Group Management events must be logged.

On Linux servers, the following Account and Group Management events must be logged:

Event Type	Description	Category
Creation, modification and deletion of users	It must be logged, if users are created, modified or deleted.	Mandatory
Creation, modification and deletion of groups	It must be logged, if a groups are created, modified or deleted.	Mandatory

Motivation: The logging of account and group management events can be useful to backtrack user and group management. With these logs it is possible to detect malicious modification, creation and deletion of users and groups.

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: 3.65-45/7.0

Req 45	Configuration Change events must be logged.

On Linux servers, the following Configuration Change events must be logged:

Event Type	Description	Category
Disable logging	It must be logged, if the logging service is disabled.	Mandatory
Deletion and unauthorized modification of logs	The deletion of events must be logged. The unauthorized modification of logs must be logged.	Mandatory
Change of logging configuration	It must be logged, if there is a change of the logging configuration	Mandatory
Change of network configuration	Change of network and interface configuration must be logged.	Mandatory
Authentication Subsystem changes	Changes of Authentication Subsystems (e.g. LDAP- or Kerberos-Policy) must be logged.	Optional
Critical File changes	Depending on the use case, critical file changes should be logged.	Optional
Change of SELinux configuration	If used SELinux events and changes of configuration must be logged.	Optional
Change of AppArmor configuration	If used AppArmorevents and changes of configuration must be logged.	Optional

Motivation: Configuration changes could have a massive impact of the OS and consequently could be a security risk. It is necessary to identify all important configurations on the OS and log the changes.

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: 3.65-46/7.0

Req 46	Auditd configuration must be immutable.

Immutable mode must be set for auditd to avoid that audit rules can be modified with 'auditctl' command.

Motivation: If auditd is not in immutable mode, unauthorized users can initiate changes to hide malicious activity.

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: 3.65-47/7.0

3.2. External Logging

Req 47	Security relevant logging data must be sent to a remote system directly after their creation.

Security relevant logging data must be forwarded to an external system in appropriate logging files as well as being stored locally. Standard protocols like Syslog, SNMPv3 must be preferred. The transfer should be secured, i.e. encrypted and authenticated, when data with need of protection is transmitted. To enable testing for consistency and completeness, sequence numbers are to be used and, if feasible, TCP instead of UDP.

Hint: The receiver of the data must be informed, how the data provided should be assessed.

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

Implementation example: rsyslog configuration contains:

```
*.* @@host1.central-log.internal
*.* @@host2.central-log.internal
```

ID: 3.65-48/7.0

Req 48	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

3.2.1. RSyslog

Req 49	If RSyslog is used, the default permission of 640 or more restrictive for logfiles must be configured.

For RSyslog the default permissions of 640 or more restrictive used for log files must be configured in file 'rsyslog.conf'.

Motivation: Unauthorized access to log files is possible if they have wrong file permissions. This is a risk of unwanted information leakage, as such files can contain data with need of protection.

Implementation example: rsyslog configuration contains:

```
FileOwner syslog
FileGroup adm
FileCreateMode 0640
DirCreateMode 0750
Umask 0027
PrivDropToUser syslog
PrivDropToGroup adm
```

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: 3.65-50/7.0

Req 50	If RSyslog is used, at least one central logging server must be configured.

One or several external log servers must be configured in RSyslog configuration.

Motivation: If logging data is only stored locally, it can be manipulated by an attacker to conceal the attack and any manipulation done on the system. This is the reason why the events must be forwarded immediately after occurrence.

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: 3.65-51/7.0

3.2.2. Syslog-NG

Req 51	If Syslog-NG is used, the default permission of 640 or more restrictive for logfiles must be configured.

For Syslog-NG the default permissions of 640 or more restrictive used for new generated logging files must be configured in file 'syslog-ng.conf'.

Motivation: Unauthorized access to logging files is possible if they have wrong file permissions. This is a risk of unwanted information leakage, as such files can contain data with need of protection.

Implementation example: syslog-ng configuration contains:

```
create-dirs yes
dir-owner syslog
dir-group adm
dir-perm 0750
owner syslog
group adm
perm 0640
```

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: 3.65-52/7.0

Req 52	If Syslog-NG is used, at least one central logging server must be configured.

One or several external log servers must be configured in RSyslog configuration.

Motivation: If logging data is only stored locally, it can be manipulated by an attacker to conceal the attack and any manipulation done on the system. This is the reason why the events must be forwarded immediately after occurrence.

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: 3.65-53/7.0

4. Authentication attribute "password" & Pluggable Authentication Modules

PAM (Pluggable Authentication Modules) is a solution to provide independent modules for authentication schemes to programs. PAM is used by default in most common Linux distributions.

Req 53	If passwords are used as an authentication attribute, "Shadow Password Suite" must be configured to protect passwords by using a secure hashing function. Passwords must be kept at least 1 day and must be changed latest after 365 days.

Passwords must always be stored as hashes. Sha512-crypt with 640.000 rounds and Salt (96 Bit) must be used as a hashing algorithm to protect passwords. Passwords must be used at least 1 day and up to 365 days. Background of this requirement is described in Security requirement 3.01 Technical Baseline Security for IT/NT Systems Chapter 8. Authentication parameter password.

Motivation: It an unauthorized person gets access to a password file, the password can be misused if not stored in a secure way. Consecutively changing password can be a misused to circumvent protection by password history, therefore a minimum password age is necessary.

Implementation example: `/etc/login.defs` contains:

```
ENCRYPT_METHOD SHA512
SHA_CRYPT_MIN_ROUNDS 640000
PASS_MAX_DAYS 365
PASS_MIN_DAYS 1
```

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: 3.65-54/7.0

Req 54	If passwords are used as an authentication attribute, "PAM" must be configured to use strong salted password hash functions while doing sufficient calculation rounds to protect passwords.

PAM can be configured to take advantage of the secure cryptographic hash function with salt for passwords. To further increase security, several rounds of the hash function are executed. The default setting for SHA-512 is 5000 rounds. This is not sufficient due to cheap and readily available compute ressources, which make brute force attacks conceivable. Therefore, the number of rounds must be increased to at least 640,000.

Newer crpytographic libraries offer hash functions that are considered superior to SHA-512. "yescrypt" or "scrypt" are offered now for several Distributions and the former is often default for newer Distro and Manpages should be used when considering them. For example Ubuntu 20.04 "focal" LTS list more information here.

Motivation: If an unauthorized person gets access to a password file, the password can be misused if not stored in a secure way.

Implementation example: **Debian-based Distro example:**

- `/etc/pam.d/common-password` needs to contain `sha512` and `rounds=640000` options configured :

```
password [success=1 default=ignore] pam_unix.so obscure sha512
rounds=640000
```

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

For this requirement the following warranty objectives are relevant:

ID: 3.65-55/7.0

Req 55	If passwords are used as an authentication attribute, "PAM" must be configured to apply password complexity rules to force the use of passwords with a minimum length of 12 characters and a combination of three out of the following categories: upper cases, lower case, numbers and special characters.

PAM must be configuration that only passwords that comply with the following complexity can be used on the system:

- Minimum length of 12 characters
- Comprising at least three of the following categories: upper/lower case letters, numbers and special characters

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

Implementation example: Debian-based Distro example:

- `/etc/pam.d/common-password`contains:

```
password requisite pam_pwquality.so retry=3 minlen=12 minclass=3
enforce_for_root difok=2 maxsequence=3 maxrepeat=3
```

Note: **Suse Enterprise Linux** uses module "`pam_cracklib`", see here.

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

For this requirement the following warranty objectives are relevant:

ID: 3.65-56/7.0

Req 56	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 57	If a password is used as an authentication attribute, the reuse of previous passwords must be prevented.

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

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

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

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

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

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

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

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

Implementation example: Debian-based Distro example:

- `/etc/login.defs`contains :

```
PASS_MIN_DAYS 1
```

- `/etc/pam.d/common-password`contains (line order is important) :

```
password required pam_pwhistory.so use_authtok enforce-for-root
retry=3 remember=60
password [success=1 default=ignore] pam_unix.so obscure sha512
use_authtok
```

ID: 3.65-58/7.0

Req 58	If passwords are used as an authentication attribute, "PAM" must be configured to have protection online attacks like brute force and dictionary attacks that hinder password guessing.

Brute force and dictionary attacks aim to use automated guessing to passwords for user and machine accounts. To prevent these kind of attacks a limitation for invalid authentication reties must be configured. It is recommended to lock an account after 5 retries for 10 minutes.

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

Implementation example: Debian-based Distro example:

- `/etc/pam.d/common-auth`contains:

```
auth required pam_tally2.so onerr=fail audit silent deny=5 un-
lock_time=600 even_deny_root
```

For implementation details look here: www.linux.org/docs/man8/pam_tally2.html
For newer distributions like Ubuntu 22.04 LTS module "faillock" is also a solution: man8/pam_faillock.8.html

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

For this requirement the following warranty objectives are relevant:

ID: 3.65-59/7.0

Req 59	PAM must be configured so that Message of the day (motd) outputs do not contain any data with need of protection.

Message of the day (motd) outputs must not contain any data with need of protection like number of missing patches, used software or kernel version. The following files must be checked for such information:

- /etc/pam.d/login
- /etc/pam.d/sshd

Motivation: Data with need of protection in motd can give attackers information that are helpful for further attacks.

Implementation example: Debian-based Distro example:

```
user@server:~# sudo rm /etc/update-motd.d/*
```

For this requirement the following threats are relevant:

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

For this requirement the following warranty objectives are relevant:

ID: 3.65-60/7.0

5. iptables / ufw

To restrict the reachability of listening of services on a system (as required with Req. 2) it is recommended to use IPTables. TCP Wrapper, as another solution for traffic control, has some major drawbacks and cannot be recommended.

The following requirements are a minimal setup for IPTables. If needed additional rules must be configured.

Req 60	If iptables or ufw is used, policies for loopback traffic must be configured.

Loopback traffic is used between server processes. A policy for traffic to the loopback network (127.0.0.0/8) must be configured for all other network interfaces.

Motivation: To prevent spoofing attacks, the loopback network must be protected from such malicious traffic.

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

For this requirement the following warranty objectives are relevant:

ID: 3.65-61/7.0

Req 61	If iptables or ufw is used, rules for inbound and (if used) forwarding connections must be configured.

For inbound and (if used) forwarding connections rules must be configured.

- iptables: tables`INPUT&FORWARD`(https://ipset.netfilter.org/iptables.man.html)
- ufw: DIRECTION`incoming&routed` (http://manpages.ubuntu.com/manpages/focal/man8/ufw.8.html)

Motivation: Without rules all packets will be dropped by the default policy (default drop) which will prevent network usage.

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

For this requirement the following warranty objectives are relevant:

ID: 3.65-62/7.0

Req 62	If iptables or ufw is used, policies must exist for all ports in listening state.

It is necessary to configure rules for all available services (ports) on a server. If a complete reachability is not needed (for example for management services like SSH) a restriction to source IP addresses or IP networks must be implemented.

Motivation: A restriction of reachability of a network service minimizes the possible attack vector.

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

For this requirement the following warranty objectives are relevant:

ID: 3.65-63/7.0

Req 63	If iptables or ufw is used, the default policy for incoming or forwarding traffic must be to drop it.

To reject connections to unconfigured network services, the default policy for inbound and forwarded traffic must be configured to drop all packets not caught by other policies.

- iptables: tables`INPUT`&`FORWARD`(https://ipset.netfilter.org/iptables.man.html)
- ufw: DIRECTION`incoming`&`routed`(http://manpages.ubuntu.com/manpages/focal/man8/ufw.8.html)

Motivation: Uncontrolled access to network services is possible without proper default policy.

Implementation example:

```
$ sudo iptables -P INPUT DROP
$ sudo iptables -P FORWARD DROP

$ sudo ufw default deny incoming
$ sudo ufw default deny routed
```

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

For this requirement the following warranty objectives are relevant:

ID: 3.65-64/7.0

6. Mandatory Access Control

Req 64	If a system has Internet facing services, is a virtualization or container host, a MAC solution must be used to restrict these services respectively guest VMs.

Mandatory access control (MAC) enforces additional security policies for access. A MAC solution like SELinuix or AppArmor must be used on systems with a higher security demand like:

- Servers with Internet facing services.
- Host systems for hypervisor virtualization
- Host systems for container solutions

Note: On Debian based Linux (e.g. Ubuntu) and Suse Enterprise Linux AppArmor is used by default. On RedHat based Linux (e.g. RHEL, Oracle Linux, CentOS) SELinux is used. It is rec-ommended to use the pre-installed MAC solution of the used Linux OS.

Motivation: The stricter access model of MAC protects services better as Linux did it by default. In case of a successful compromise of a service the attacker is limited in accessing the system.

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

For this requirement the following warranty objectives are relevant:

ID: 3.65-65/7.0

6.1. SELinux

Req 65	If SELinux is used, it must not be disabled in bootloader configuration.

Most distributions have kernel default values to load SELinux infrastructure. Only setting kernel parameter `selinux=0` would lead to deactivation. SELinux must not be deactivated in bootloader configuration of Grub to be executed during system boot.

Motivation: The stricter access model of MAC protects services better as Linux did it by default. In case of a successful compromise of a service the attacker is limited in accessing the system.

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

For this requirement the following warranty objectives are relevant:

ID: 3.65-66/7.0

Req 66	If SELinux is used, it must be run in "enforcing" mode to actually enforce policy.

Processes with high security risk should be restricted with mandatory access control (MAC) like SELinux to reduce exploitability. Software like hypervisor, container software or network services reachable from the internet are typical examples for restriction. Using profiles SELinux restricts processes to their expected behavior. Either pre- or self-configured profiles must be used for software with high security risk.
For some Linux distributions SELinux runs per default in "permissive" mode, which leads to SELinux only logging warnings but not enforcing policy. Therefore it needs also to checked whether it is set to "enforcing" mode.

Motivation: The stricter access model of MAC protects services better as Linux does by default. In case of a successful compromise of a service the attacker is limited in accessing the system.

Implementation example: See `/etc/selinux/config` for configuration. For Debian-based distros using kernel parameter (`selinux=1 security=selinux enforcing=1`) on boot is recommend too. Red Hat on the other hand only uses the config file. **In any case it is best to consult distro documentation for proper understanding and configuration!**

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:

ID: 3.65-67/7.0

Req 67	If SELinux is used, the policy must be configured.

The SELinux policy must be configured at least for targeted network daemons or stricter if needed.

Motivation: The stricter access model of MAC protects services better as Linux did it by default. In case of a successful compromise of a service the attacker is limited in accessing the system.

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

For this requirement the following warranty objectives are relevant:

ID: 3.65-68/7.0

Req 68	If SELinux is used, SETroubleshoot and MCS Translation Service must not be installed.

The SETroubleshoot is an unnecessary daemon to have running on a server, especially if X Windows is disabled. The MCS Translation Service service is also not needed for operating SELinux. Both services must not be installed or deleted.

Motivation: Not needed software and services must not be installed on a system to reduce the amount of potentially vulnerable code.

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

- Unauthorized use of services or resources
- Disruption of availability

For this requirement the following warranty objectives are relevant:

ID: 3.65-69/7.0

6.2. AppArmor

Req 69	If AppArmor is used, it must not be disabled in bootloader configuration.

AppArmor must be activated in bootloader configuration of Grub to be executed during system boot.

Motivation: The stricter access model of MAC protects services better as Linux did it by default. In case of a successful compromise of a service the attacker is limited in accessing the system

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

For this requirement the following warranty objectives are relevant:

ID: 3.65-70/7.0

Req 70	If AppArmor is used, its state must be enforced.

Profiles for AppArmor must be enforced for daemons with high security risk like hypervisor, container software or Internet reachable network service on the server.

Motivation: The stricter access model of MAC protects services better as Linux did it by default. In case of a successful compromise of a service the attacker is limited in accessing the system.

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

For this requirement the following warranty objectives are relevant:

ID: 3.65-71/7.0

7. Regular Compliance Checks

This chapter includes requirements that show the default behavior of a Linux OS. These requirements must normally not be configured after installation. Nevertheless, these requirements are also important to mention as it is necessary to include them in regular compliance checks in order to identify misconfiguration and to guarantee an adequate security level during the complete lifecycle of a system.

Req 71	No legacy + entries must exist in files passwd, shadows and group.

The character "+" is used as a marker in configuration files to insert data from NIS (Network Information Service) maps. Such markers must not exist in the files '/etc/passwd', '/etc/shadows' and '/etc/group'.

Motivation: Attackers could possibly inject own code to gain privileged access if the character "+" exists in files '/etc/passwd', '/etc/shadows' and '/etc/group'.

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

For this requirement the following warranty objectives are relevant:

ID: 3.65-72/7.0

Req 72	A user's home directory must be owned by the user and have mode 750 or more restrictive.

A user can change the permissions of its home directory. It must be validated regularly that the mode of all user home directories is 750 and the corresponding user is owner of its home directory. This is only valid for accounts of real users and excludes system accounts. An exception of this requirement are home directories for users used for SFTP in a chroot environment. In this case the home directory must be owned by root.

Motivation: If accidently a wrong user becomes owner of another user's home directory or if permissions are wrong another user can access foreign data.

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: 3.65-73/7.0

Req 73	Default group for the root account must be GID 0.

The usage of GID 0 as default group for the root account is the default on most Linux systems. This must not be changed and should be checked in '/etc/passwd' file on a regular basis.

Motivation: Changing the GID for root may result in root-owned files that become accessible by non-privileged users.

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: 3.65-74/7.0

Req 74	Root must be the only UID 0 account.

Any account with UID 0 has root privileges on the server. It must be validated on a regular basis that only the root account has UID 0.

Motivation: A user with restricted rights can accidently become root if UID 0 is set for its account.

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: 3.65-75/7.0

Req 75	All groups in /etc/passwd must exist in /etc/group.

It must be checked on a regular basis that all groups definied `/etc/passwd` are also defined in `/etc/group` .

Sample entry from `/etc/passwd` :

```
thomas:x:1000:1000:Tho Mas,,,:/home/thomas:/bin/bash
```

The red marked number represents the users primary group and this one must exist in `/etc/group` too.

Motivation: Errors in group management can result in unspecific authentication and authorization behavior.

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

For this requirement the following warranty objectives are relevant:

ID: 3.65-76/7.0

Req 76	No duplicate UIDs and GIDs must exist.

It must be checked on a regular basis that no duplicate UIDs and GIDs in the files '/etc/passwd' and '/etc/group' exist on the server.

Motivation: Users and groups must be assigned to unique UIDs and GIDs for accountability and to ensure appropriate access protections.

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: 3.65-77/7.0

Req 77	No duplicate user or group names must exist.

The tools "useradd" and " groupadd" do not allow to set duplicate user or group names. Unfortunately, an administrator can do so by manually editing the corresponding configuration files. The file "`/etc/passwd`" and "`/etc/group`" must be checked for duplicate entries.

Motivation: Duplicate user or group name can result in unauthorized access to data.

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: 3.65-78/7.0

Req 78	The shadow group must be empty (only Debian-based Linux distributions).

On Debian-based Linux distributions the shadow group allows system programs which require access the ability to read the '/etc/shadow' file. No users must be assigned to the shadow group. The shadow group must be checked in file '/etc/group' on a regular basis that no users are assigned to it.

Motivation: With unauthorized access to the '/etc/shadow' file it is possible to run password cracking attacks against the stored password hashes.

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: 3.65-79/7.0

Req 79	No files and directories without assigned user or group must exist.

If users or groups are deleted from a system, their files and directories must also be deleted, or the ownership must be transferred to another user or group. Otherwise, files and directories without a user or group are left on the system. The system must be checked for files and directories without assigned user or group on a regular basis.

Motivation: If files and directories without an assigned user or group exist on system it could happen that a newly generated user can access this data if same UID or GID is assigned as used by the user or group deleted before.

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: 3.65-80/7.0

Req 80	Permissions of security relevant configuration files must have the distribution default values or more restrictive.

The permissions of configuration files must be correct and set to according user and group. See the following list of security relevant files:

- /etc/passwd; /etc/passwd-

- /etc/shadow; /etc/shadow-
- /etc/group; /etc/group-
- /etc/gshadow; /etc/gshadow-
- /boot/grub2/grub.cfg; /boot/grub2/user.cfg (RedHat based Linux)
- /boot/grub/grub.cfg (Ubuntu Linux)
- /var/log/*
- /etc/crontab; /etc/cron.*; /etc/cron.d
- /etc/sshd_config

Motivation: In configuration files data with need of protection is stored. With wrong privileges, an unauthorized user can possibly access these files and misuse data or even modify configuration.

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: 3.65-81/7.0