

Sicherheitsanforderung

Secure Shell (SSH)

Deutsche Telekom Gruppe

Version	5.0
Datum	01.12.2025
Status	Freigegeben

Impressum

Herausgeber
Deutsche Telekom AG
Vorstandsbereich Technology & Innovation
Chief Security Officer

Reuterstrasse 65, 53315 Bonn
Deutschland

Dateiname	Dokumentennummer 3.04	Dokumententyp Sicherheitsanforderung
Version 5.0	Stand 01.12.2025	Status Freigegeben
Fachlicher Ansprechpartner Telekom Security psa.telekom.de	Gültigkeitsdauer 01.12.2025 - 30.11.2030	Freigegeben von Stefan Pütz, Leiter SEC-T-TST

Zusammenfassung
Sicherheitsanforderungen für SSH Server, SFTP Server und das SSH Protokoll.

Copyright © 2025 by Deutsche Telekom AG.

Alle Rechte, auch die des auszugsweisen Nachdrucks, der fotomechanischen Wiedergabe (einschließlich Mikrokopie) sowie der Auswertung durch Datenbanken oder ähnliche Einrichtungen, vorbehalten.

Inhaltsverzeichnis

1.	Einleitung	4
2.	sshd_config	5
3.	Zugriffsrechte & Schlüsselmaterial	17

1. Einleitung

SSH (Secure Shell) ist eine Client-/Server-Anwendung inklusive zugehörigem Netzwerkprotokoll, die für den Zugriff auf Systeme auf der Terminal-Ebene und die Übertragung von Daten genutzt werden kann. Da SSH typischerweise für den Managementzugriff genutzt wird, besteht ein hoher Schutzbedarf. Das vorliegende Dokument beinhaltet Sicherheitsanforderungen für den SSH Server, SFTP Server und das SSH Protokoll.

Es wird die Nutzung von OpenSSH empfohlen, da diese Lösung weitverbreitet ist und über eine große Entwicklergemeinschaft verfügt. Entsprechend orientieren sich die Anforderungen an [OpenSSH](#).

Dieses Dokument wurde auf Basis der Vorgaben aus den im Konzern gültigen Sicherheitsrichtlinien erstellt und basiert auf Best-Practice-Sicherheitsvorgaben und Herstellerempfehlungen. Die Sicherheitsanforderung dient u.a. als Grundlage zur Freigabe im PSA-Prozess. Sie dient darüber hinaus als Umsetzungsempfehlung für Einheiten, die nicht am PSA-Verfahren teilnehmen. Diese Anforderungen müssen bereits während der Planungs- und Entscheidungsprozesse berücksichtigt werden. Bei der Umsetzung der Sicherheitsanforderung ist das jeweils vorrangige nationale, internationale und supranationale Recht zu beachten.

2. sshd_config

Req 1 Die Version 2 vom SSH-Protokoll muss verwendet werden. (Ab OpenSSH 7.4+ automatisch erfüllt).

Nutzerrollen: Betrieb, Entwicklung, Integration

SSHv1 muss permanent in der Konfiguration des SSH-Servers deaktiviert werden. Mit OpenSSH 7.4 wurde die Unterstützung von SSHv1 komplett entfernt, so dass keine extra Konfiguration mehr notwendig ist.

Quelle:

[TR-02102-4] BSI-Technische Richtlinie: Kryptographische Verfahren - Empfehlungen und Schlüssellängen - Teil 4 SSH, Version 2025-01

Motivation: Die Version 1 des SSH-Protokolls besitzt kryptographische Schwächen und ist obsolet. Bei einer Nutzung von SSHv1 ist die Vertraulichkeit und Integrität der übertragenen Daten nicht sichergestellt.

Umsetzungsbeispiel: OpenSSH Parameter: Protocol 2

Für diese Anforderung sind folgende Bedrohungen relevant:

- Unberechtigter Zugang zum System
- Unberechtigte Einsichtnahme oder Abgriff von Daten
- Unberechtigte Modifikation von Daten
- Unberechtigte Nutzung von Diensten oder Ressourcen
- Störung der Verfügbarkeit
- Unbemerkt durchführbare Angriffe

ID: 3.50-46/8.0

Req 2 SSH MaxStartups muss auf "10:30:100" oder weniger eingestellt sein. (OpenSSH Standardeinstellung)

Der MaxStartups Parameter legt die maximale Anzahl simultaner (noch) nicht authentifzierter Verbindungen zum SSH-Daemon fest. Dieser Wert muss auf "10 : 30 : 100" oder weniger eingestellt sein. Das entspricht der OpenSSH Standardeinstellung.

Motivation: Um ein System vor Überlast durch eine Übermenge von simultanen Verbindung zu schützen limitiert man die maximale Anzahl gleichzeitiger Verbindungsversuche.

Umsetzungsbeispiel: `sshd_config` enthält:

`MaxStartups 10:30:100`

Für diese Anforderung sind folgende Bedrohungen relevant:

- Störung der Verfügbarkeit

ID: 3.04-2/5.0

Req 3 Es dürfen nur die folgenden Algorithmen zum Schlüsselaustausch verwendet werden: curve448-sha512, sntrup761x25519-sha512@openssh.com, curve25519-sha256@libssh.org, curve25519-sha256, ecdh-sha2-nistp521, ecdh-sha2-nistp384, cdh-sha2-nistp256, diffie-hellman-group18-sha512, diffie-hellman-group16-sha512, diffie-hellman-group-exchange-sha256 oder mlkem768x25519-sha256@openssh.com.

Nutzerrollen: Betrieb, Entwicklung, Integration

Für den Austausch von Schlüsseln dürfen die folgenden Algorithmen verwendet werden:

Algorithmus	Referenzspezifikationen
curve448-sha512	[RFC-7748]
sntrup761x25519-sha512@openssh.com	
curve25519-sha256@libssh.org	[RFC-7748]
curve25519-sha256	[RFC-7748]
ecdh-sha2-nistp521	[RFC-5656]
ecdh-sha2-nistp384	[RFC-5656]
ecdh-sha2-nistp256	[RFC-5656]
diffie-hellman-group18-sha512	[RFC-8268]
diffie-hellman-group16-sha512	[RFC-8268]
diffie-hellman-group-exchange-sha256	[RFC-4419]
mlkem768x25519-sha256@openssh.com	

Hinweis zu curve448 und curve25519:

curve448 und curve25519 werden nicht (explizit) vom BSI empfohlen, aber bisher sind keine Schwachstellen bekannt, so dass sie als sicher einzustufen sind.

Quellen:

[RFC-4419] Diffie-Hellman Group Exchange for SSH

[RFC-5656] Elliptic Curve Algorithm Integration

[RFC-7748] Elliptic Curves for Security

[RFC-8268] More MODP Diffie-Hellman Groups

[RFC-8731] Secure Shell (SSH) Key Exchange Method Using Curve25519 and Curve448

[TR 02102-4] BSI-Technische Richtlinie: Kryptographische Verfahren - Empfehlungen und Schlüssellängen - Teil 4 SSH, Version 2025-01

Motivation: Bei einem Angriff kann gegebenenfalls die Verschlüsselung von übertragenen Daten kompromittiert werden, wenn schwache Chiffren und Algorithmen verwendet werden.

Umsetzungsbeispiel: sshd_config enthält:

```
KexAlgorithms curve448-sha512, curve25519-sha256@libssh.org, curve25519-sha256, ecdh-sha2-nistp521, ecdh-sha2-nistp384, ecdh-sha2-nistp256, diffie-hellman-group18-sha512, diffie-hellman-group16-sha512, diffie-hellman-group-exchange-sha256
```

Für diese Anforderung sind folgende Bedrohungen relevant:

- Unberechtigte Einsichtnahme oder Abgriff von Daten

ID: 3.50-48/8.0

Req 4 Es dürfen nur die folgenden Chiffren verwendet werden: aes256-gcm@openssh.com, aes128-gcm@openssh.com, chacha20-poly1305@openssh.com, aes256-ctr, aes192-ctr oder aes128-ctr.

Nutzerrollen: Betrieb, Entwicklung, Integration

Veraltete und unsichere Chiffren und Algorithmen dürfen nicht benutzt werden. Die folgenden Chiffren können für SSH genutzt werden:

Cipher	Referenzspezifikationen
aes256-gcm@openssh.com	[RFC-5647]

aes128-gcm@openssh.com	[RFC-5647]
chacha20-poly1305@openssh.com	[RFC-8439]
aes256-ctr	[RFC-4344]
aes192-ctr	[RFC-4344]
aes128-ctr	[RFC-4344]

Quellen:

[RFC-4344] The Secure Shell (SSH) Transport Layer Encryption Modes

[RFC-5647] AES Galois Counter Mode for the Secure Shell Transport Layer Protocol

[RFC-8439] ChaCha20 and Poly1305 for IETF Protocols

Motivation: Bei einem Angriff kann gegebenenfalls die Verschlüsselung von übertragenen Daten kompromittiert werden, wenn schwache Chiffren und Algorithmen verwendet werden.

Umsetzungsbeispiel: sshd_config enthält:

Ciphers aes256-gcm@openssh.com, aes128-gcm@openssh.com, chacha20-poly1305@openssh.com, aes256-ctr, aes192-ctr, aes128-ctr

Für diese Anforderung sind folgende Bedrohungen relevant:

- Unberechtigter Zugang zum System
- Unberechtigte Einsichtnahme oder Abgriff von Daten
- Unberechtigte Modifikation von Daten
- Unberechtigte Nutzung von Diensten oder Ressourcen
- Abstreiten von durchgeführten Aktionen

ID: 3.50-49/8.0

Req 5 Es dürfen nur die folgenden MAC-Algorithmen verwendet werden: hmac-sha2-512-etm@openssh.com, hmac-sha2-512, hmac-sha2-256-etm@openssh.com oder hmac-sha2-256.

Nutzerrollen: Betrieb, Entwicklung, Integration

Es ist wichtig die Nutzung von unsicheren MAC-Algorithmen zu vermeiden. Beispiele für solche veralteten Algorithmen sind MD5 und SHA1. Die folgenden MAC-Algorithmen sind zugelassen und müssen entsprechend für den SSH-Server konfiguriert werden:

Algorithmus	Referenzspezifikationen
hmac-sha2-512-etm@openssh.com	[RFC-6688]
hmac-sha2-512	[RFC-6688]
hmac-sha2-256-etm@openssh.com	[RFC-6688]
hmac-sha2-256	[RFC-6688]

References:

[RFC-6688] SHA-2 Data Integrity Verification for the Secure Shell (SSH) Transport Layer Protocol

Motivation: Bei einem Angriff kann gegebenenfalls die Verschlüsselung von übertragenen Daten kompromittiert werden, wenn schwache Chiffre und Algorithmen verwendet werden.

Umsetzungsbeispiel: sshd_config enthält:

MACs hmac-sha2-512-etm@openssh.com, hmac-sha2-512, hmac-sha2-256-etm@openssh.com, hmac-sha2-256

Für diese Anforderung sind folgende Bedrohungen relevant:

- Unberechtigte Modifikation von Daten

ID: 3.50-50/8.0

Req 6 Nur die folgenden Host Key Algorithmen (alias "public key signature algorithms" oder "server authentication algorithms") dürfen verwendet werden: ssh-ed448, ssh-ed25519, ssh-ed25519-cert-v01@openssh.com, sk-ssh-ed25519@openssh.com, sk-ssh-ed25519-cert-v01@openssh.com, ecdsa-sha2-nistp521, ecdsa-sha2-nistp521-cert-v01@openssh.com, ecdsa-sha2-nistp384, ecdsa-sha2-nistp384-cert-v01@openssh.com, ecdsa-sha2-nistp256, ecdsa-sha2-nistp256-cert-v01@openssh.com, sk-ecdsa-sha2-nistp256@openssh.com oder sk-ecdsa-sha2-nistp256-cert-v01@openssh.com. Ein Host Key muss einen Host eindeutig identifizieren. (1:1 Beziehung).

Nutzerrollen: Betrieb, Entwicklung, Integration

Veraltete und unsichere Algorithmen dürfen nicht benutzt werden. Die folgenden Algorithmen können für SSH genutzt werden. Die Bezeichnungen können von Software zu Software variieren:

Algorithmus	Referenzspezifikationen
ssh-ed448	[RFC-8709]
ssh-ed25519	[RFC-8709]
ssh-ed25519-cert-v01@openssh.com	[RFC-8709]
sk-ssh-ed25519@openssh.com	[RFC-8709]
sk-ssh-ed25519-cert-v01@openssh.com	[RFC-8709]
ecdsa-sha2-nistp521	[RFC-5656]
ecdsa-sha2-nistp521-cert-v01@openssh.com	[RFC-5656]
ecdsa-sha2-nistp384	[RFC-5656]
ecdsa-sha2-nistp384-cert-v01@openssh.com	[RFC-5656]
ecdsa-sha2-nistp256	[RFC-5656]
ecdsa-sha2-nistp256-cert-v01@openssh.com	[RFC-5656]
sk-ecdsa-sha2-nistp256@openssh.com	[RFC-5656]
sk-ecdsa-sha2-nistp256-cert-v01@openssh.com	[RFC-5656]
rsa-sha2-512	[RFC-8332]
rsa-sha2-512-cert-v01@openssh.com	[RFC-8332]
rsa-sha2-256	[RFC-8332]
rsa-sha2-256-cert-v01@openssh.com	[RFC-8332]

Hinweis zu ed448 und ed25519:

ed448 und ed25519 werden nicht (explizit) vom BSI empfohlen, aber bisher sind keine Schwachstellen bekannt, so dass sie als sicher einzustufen sind.

Quellen:

[RFC-5656] Elliptic Curve Algorithm Integration

[RFC-8709] Ed25519 and Ed448 Public Key Algorithms for the Secure Shell (SSH) Protocol

[RFC-8332] Use of RSA Keys with SHA-256 and SHA-512 in the Secure Shell (SSH) Protocol

Motivation: Bei einem Angriff kann gegebenenfalls unbemerkt ein Host imitiert werden.

Umsetzungsbeispiel: `sshd_config` enthält:

```
HostKeyAlgorithms ssh-ed448,ssh-ed25519,ssh-ed25519-cert-v01@openssh.com,sk-ssh-ed25519@openssh.com,sk-ssh-ed25519-cert-v01@openssh.com,ecdsa-sha2-nistp521,ecdsa-sha2-nistp521-cert-v01@openssh.com,ecdsa-sha2-nistp384,ecdsa-sha2-nistp384-cert-v01@openssh.com,ecdsa-sha2-nistp256,ecdsa-sha2-nistp256-cert-v01@openssh.com,sk-ecdsa-sha2-nistp256@openssh.com,sk-ecdsa-sha2-nistp256-cert-v01@openssh.com
```

Für diese Anforderung sind folgende Bedrohungen relevant:

- Unberechtigter Zugang zum System
- Unberechtigte Einsichtnahme oder Abgriff von Daten

ID: 3.50-51/8.0

Req 7	Logging für SSH muss aktiviert werden.
-------	--

Logging für SSH muss aktiviert werden. Es wird empfohlen die Einstellung "INFO" zu nutzen, um relevante Ereignisse zu protokollieren. Wenn notwendig, können auch höhere Level wie "VERBOSE" genutzt werden.

Motivation: Das Loggen von sicherheitsrelevanten Ereignissen ist eine Basisanforderung, um laufende Angriffe und bereits durchgeführte Angriffe zu erkennen. Dies ist die einzige Möglichkeit um geeignete Maßnahmen einzuleiten, um die Systemsicherheit zu erhalten oder wiederherzustellen. Log-Daten können zudem als Beweis genutzt werden, um rechtliche Schritte einzuleiten.

Umsetzungsbeispiel: `sshd_config` enthält:

```
SyslogFacility AUTH  
LogLevel INFO
```

Für diese Anforderung sind folgende Bedrohungen relevant:

- Abstreiten von durchgeführten Aktionen
- Unbemerkt durchführbare Angriffe

ID: 3.04-7/5.0

Req 8	SSH LoginGraceTime muss auf 120s oder weniger eingestellt sein. (OpenSSH Standardeinstellung)
-------	---

Der LoginGraceTime Parameter begrenzt das Zeitfenster für eine erfolgreiche Authentifizierung. Je länger dieses Zeitfenster ist, umso mehr offene nicht authentifizierte Verbindungen können bestehen. Die Anforderung beträgt maximal 120s, dies entspricht der OpenSSH Standardeinstellung.

Motivation: Eine angemessene Zeit für den LoginGraceTime Parameter schützt das System gegen nicht authentifizierte SSH-Verbindungen, die System-Ressourcen verschwenden.

Umsetzungsbeispiel: `sshd_config` enthält:

```
LoginGraceTime 120
```

Für diese Anforderung sind folgende Bedrohungen relevant:

- Störung der Verfügbarkeit

ID: 3.04-8/5.0

Req 9	SSH MaxAuthTries muss auf 6 oder niedriger gesetzt werden. (OpenSSH Standardeinstellung)
-------	--

Der MaxAuthTries Parameter legt die maximale Anzahl an Authentifizierungsversuchen für eine Verbindung fest. Dieser Werte muss auf 6 oder kleiner gesetzt werden, dies entspricht der OpenSSH Standardeinstellung.

Motivation: Dieser Parameter verringert das Risiko eines erfolgreichen Brute-Force-Angriffes gegen den SSH-Server.

Umsetzungsbeispiel: `sshd_config` enthält:
`MaxAuthTries 6`

Für diese Anforderung sind folgende Bedrohungen relevant:

- Störung der Verfügbarkeit

ID: 3.04-9/5.0

Req 10 Die Anmeldung von "root" muss für SSH deaktiviert werden.

Individuelle, eindeutige, persönliche Konten sind notwendig. Die permanente Nutzung von "root" ist nicht erlaubt. Um die entfernte Anmeldung mit "root" zu verhindern, muss dies für den SSH-Dienst deaktiviert werden.

Hinweis: Das notwendige Schutzniveau kann auch durch die ausschließliche Verwendung funktionaler Konten auf einem System erreicht werden. Zusätzlich muss sichergestellt sein, dass die SSH-Schlüssel der Konten über ein zentrales Nutzerverwaltungssystem (z. B. ZAM) für "root" verwaltet und durch ein Konfigurationsmanagement-System ausgerollt werden. Weiterhin muss der Zugriff über Jumpbox stattfinden, die personenbezogene Logins erfordern. Die Verwendung von SSH-Schlüsseln (und das Verbot von Passwörtern für den SSH-Zugang) ist obligatorisch.

Motivation: Die Nutzung von "root" für administrative Tätigkeiten im Normalfall stellt ein Risiko dar, weil "root" uneingeschränkte Rechte am System hat, standardmäßig auf dem System vorhanden ist und damit einfaches Ziel für Angriffe ist. Die Verwendung von individuellen, eingeschränkten Konten für wiederkehrende oder normale Tätigkeiten reduziert das Risiko. ("Prinzip der geringsten Privilegien" bzw. "Least Privilege Principle").

Umsetzungsbeispiel: `sshd_config` enthält:
`PermitRootLogin no`

Für diese Anforderung sind folgende Bedrohungen relevant:

- Abstreiten von durchgeführten Aktionen
- Unbemerkt durchführbare Angriffe

ID: 3.04-10/5.0

Req 11 SSH StrictModes muss aktiviert werden.

SSH StrictModes muss aktiviert werden. Dies stellt sicher, dass Besitzverhältnis und Dateirechte richtig bzw. sicher eingestellt sind bevor eine SSH-Verbindung aufgebaut wird.

Motivation: Dies verhindert die Verwendung von Dateien bzw. Ordnern im Zusammenhang mit OpenSSH mit zu großzügig gesetzten Berechtigungen.

Umsetzungsbeispiel: `sshd_config` enthält:
`StrictModes yes`

Für diese Anforderung sind folgende Bedrohungen relevant:

- Unberechtigter Zugang zum System
- Unberechtigte Einsichtnahme oder Abgriff von Daten
- Unberechtigte Modifikation von Daten
- Unberechtigte Nutzung von Diensten oder Ressourcen
- Abstreiten von durchgeführten Aktionen
- Unbemerkt durchführbare Angriffe

ID: 3.04-11/5.0

Req 12 Die Authentisierung am SSH-Server muss mittels SSH Public Keys erfolgen.

Authentisierung mit öffentlichem (public) und privatem (private) Schlüsseln muss für die SSH-Anmeldung verwendet werden.

Hinweis: Private Schlüssel von Personen müssen mittels einer Passphrase geschützt werden.

Motivation: Passwörter können mittels Phishing, Keylogger abgegriffen bzw. mittels Brute-Force-Angriff geknackt werden. Sie sind generell einfacher anzugreifen als private SSH-Schlüssel, besonders wenn Passphrases für private SSH-Schlüssel verwendet werden.

Umsetzungsbeispiel: `sshd_config` enthält:

```
AuthenticationMethods publickey
PubkeyAuthentication yes
```

Für diese Anforderung sind folgende Bedrohungen relevant:

- Störung der Verfügbarkeit
- Abstreiten von durchgeführten Aktionen
- Unbemerkt durchführbare Angriffe
- Angriffe motiviert und erleichtert durch Informationspreisgabe oder erkennbare Sicherheitsschwachstellen

ID: 3.04-12/5.0

Req 13 Authentisierung mittels Passwort muss für SSH deaktiviert werden.

Die Authentisierung muss mittels privatem und öffentlichem Schlüssel erfolgen. Daher muss die Anmeldung mit Passwort für SSH deaktiviert werden.

Bei neueren Linux-Distributionen kann "UsePAM yes" notwendig sein, um Hersteller-Support zu erhalten und damit das System wie erwartet funktioniert. In diesem Fall darf die PAM-Konfiguration die obligatorische Benutzerauthentifizierung nicht umgehen, unabhängig davon, ob es sich um eine Ein-Faktor-Authentifizierung für unprivilegierte Benutzer oder um 2FA für Administratoren handelt.

Systeme, welche an ein zentrales IAM angebunden sind, können es erfordern Authentifizierung mittels Kontoname und Passwort zu realisieren. In diesem Fall ist die Verwendung ausschließlich für zentral verwaltete Konten zulässig. Diese Abweichung kann beispielsweise mittels "Match" Konfiguration realisiert werden, siehe dazu "[Match Introduces a conditional block](#)". Privilegierte Konten eines zentralen IAM können dann beispielsweise durch "AuthenticationMethods password, publickey" eine 2FA-Authentifizierung realisieren.

Motivation: Passwörter können mittels Phishing, Keylogger abgegriffen bzw. mittels Brute-Force-Angriff geknackt werden. Sie sind generell einfacher anzugreifen als private SSH-Schlüssel, besonders wenn Passphrases für private SSH-Schlüssel verwendet werden.

Umsetzungsbeispiel: `sshd_config` enthält:

```
PasswordAuthentication no
KbdInteractiveAuthentication no
ChallengeResponseAuthentication no
UsePAM no
```

Für diese Anforderung sind folgende Bedrohungen relevant:

- Störung der Verfügbarkeit
- Abstreiten von durchgeführten Aktionen
- Unbemerkt durchführbare Angriffe
- Angriffe motiviert und erleichtert durch Informationspreisgabe oder erkennbare Sicherheitsschwachstellen

ID: 3.04-13/5.0

Req 14 SSH IgnoreRhosts muss aktiviert werden.

.rhosts ist ein Legacy-Mechanismus aus den früheren Protokollen rlogin und rsh, die es Benutzern ermöglichen, Hosts und Benutzer anzugeben, die vertrauenswürdig sind und sich ohne Angabe eines Kennworts anmelden dürfen. Dieser Mechanismus gilt aus Sicherheitssicht als schwach, da er sich für die Authentifizierung auf IP-Adressen und Hostnamen stützt, die gefälscht werden können. Daher muss es deaktiviert werden.

Motivation: Die Verwendung der ".rhosts"-Datei würde eine Anmeldung ohne Authentisierungsmerkmal wie ein SSH-Schlüsselpaar erlauben.

Umsetzungsbeispiel: `sshd_config` enthält:
`IgnoreRhosts yes`

Für diese Anforderung sind folgende Bedrohungen relevant:

- Unberechtigter Zugang zum System
- Unberechtigte Einsichtnahme oder Abgriff von Daten
- Unberechtigte Modifikation von Daten
- Unberechtigte Nutzung von Diensten oder Ressourcen
- Abstreiten von durchgeführten Aktionen
- Unbemerkt durchführbare Angriffe

ID: 3.04-14/5.0

Req 15 SSH HostbasedAuthentication muss deaktiviert werden.

Wie bei .rhosts lässt HostbasedAuthentication Anmeldungen ohne Angabe eines Authentifizierungsattributs zu. Daher muss es deaktiviert werden.

Motivation: Wenn eine vertrauliche Verbindung mit anderen System konfiguriert sind, so bedeutet ein erfolgreicher Angriff auch die Kompromittierung aller anderen so vertrauten Systeme.

Umsetzungsbeispiel: `sshd_config` enthält:
`HostbasedAuthentication no`

Für diese Anforderung sind folgende Bedrohungen relevant:

- Unberechtigter Zugang zum System
- Unberechtigte Einsichtnahme oder Abgriff von Daten
- Unberechtigte Modifikation von Daten
- Unberechtigte Nutzung von Diensten oder Ressourcen
- Abstreiten von durchgeführten Aktionen
- Unbemerkt durchführbare Angriffe

ID: 3.04-15/5.0

Req 16 Die Nutzung des SSH-Dienstes muss auf dedizierte Konten oder Gruppen eingeschränkt werden.

Für eine einfachere und sichere Administration ist es notwendig, dedizierte Konten oder Gruppen (empfohlen) für SSH zu nutzen.

Motivation: Die Nutzung dedizierter Konten oder Gruppen macht die Nutzung von SSH sicherer, wegen der Einschränkung des Zugriffs auf die notwendige Minimum. ("Prinzip der geringsten Privilegien")

Umsetzungsbeispiel: `sshd_config` enthält Optionen wie z. B. `AllowUsers`, `AllowGroups`, `DenyUsers`, `DenyGroups` um festzulegen, welchen Benutzer bzw. Gruppen der Login erlaubt ist. Siehe auch [manpage sshd_config](#).

Für diese Anforderung sind folgende Bedrohungen relevant:

- Unberechtigter Zugang zum System

ID: 3.04-16/5.0

Req 17 SSH MaxSessions muss auf 10 oder weniger eingestellt sein. (OpenSSH Standardeinstellung)

Der MaxSessions Parameter legt die maximale Anzahl gleichzeitig offener Shells, Logins oder Subsysteme (z. B. SFTP) fest. Dieser Werte muss auf 10 oder weniger eingestellt sein. dies entspricht der OpenSSH Standardeinstellung.

Motivation: Um ein System vor Überlast durch eine Übermenge von simultanen Verbindung zu schützen limitiert man die maximale Anzahl gleichzeitig offener Sessions.

Umsetzungsbeispiel: `sshd_config` enthält:
 MaxSessions 10

Für diese Anforderung sind folgende Bedrohungen relevant:

- Störung der Verfügbarkeit

ID: 3.04-17/5.0

Req 18 SSH Tunnel Devices müssen deaktiviert werden.

SSH kann genutzt werden, um andere Dienste zu tunneln. Für Managementdienste von Linux-Servern wird diese typischerweise nicht genutzt und kann deaktiviert werden.

Motivation: Die SSH-Tunnel-Funktion kann für Angriffe missbraucht werden um Verkehr zu einem böartigen Endgerät zu tunneln.

Umsetzungsbeispiel: `sshd_config` enthält:
 PermitTunnel no

Für diese Anforderung sind folgende Bedrohungen relevant:

- Unberechtigte Nutzung von Diensten oder Ressourcen

ID: 3.04-18/5.0

Req 19 SSH TCP Port Forwarding muss deaktiviert werden.

TCP Forwarding kann genutzt werden, um TCP-Verbindungen mittels SSH weiterzuleiten. Für Managementdienste von Linux-Servern wird diese typischerweise nicht genutzt und kann deaktiviert werden.

Hinweis: Diese Anforderung ist nicht für Jumphosts gültig!

Motivation: Regelwerke von Firewalls könnten umgangen werden, da mittels TCP Forwarding Tunnel über einen SSH-Server hinweg erzeugt werden können.

Umsetzungsbeispiel: `sshd_config` enthält:
 AllowTcpForwarding no
oder besser
 DisableForwarding yes

Für diese Anforderung sind folgende Bedrohungen relevant:

- Unberechtigte Nutzung von Diensten oder Ressourcen

ID: 3.04-19/5.0

Req 20 SSH Agent Forwarding muss deaktiviert werden.

SSH Agent Forwarding ermöglicht Authentisierungsanfragen an ein anderes System weiterzuleiten. Für Managementdienste von Linux-Servern wird diese typischerweise nicht genutzt und muss deaktiviert werden.

Hinweis: Diese Anforderung ist nicht für Jumphosts gültig!

Motivation: Die Deaktivierung dieser Funktion auf dem Server verhindert die Erzeugung eines Agent-Forwarding-Socket, wodurch Missbrauch vorgebeugt wird.

Umsetzungsbeispiel: `sshd_config` enthält:

`AllowAgentForwarding no`

oder besser

`DisableForwarding yes`

Für diese Anforderung sind folgende Bedrohungen relevant:

- Unberechtigter Zugang zum System
- Unberechtigte Einsichtnahme oder Abgriff von Daten
- Unberechtigte Modifikation von Daten
- Unberechtigte Nutzung von Diensten oder Ressourcen
- Abstreiten von durchgeführten Aktionen
- Unbemerkt durchführbare Angriffe

ID: 3.04-20/5.0

Req 21 SSH Gateway Ports müssen deaktiviert werden.

SSH Gateway Ports legen fest, ob ein entferntes Client-System sich auf weitergeleitete Ports verbinden kann. Für Managementdienste von Linux-Servern wird diese typischerweise nicht genutzt und kann deaktiviert werden.

Motivation: Regelwerke von Firewalls könnten umgangen werden, da mittels "GatewayPorts" Tunnel über einen SSH-Server hinweg erzeugt werden können.

Umsetzungsbeispiel: `sshd_config` enthält:

`GatewayPorts no`

oder besser

`DisableForwarding yes`

Für diese Anforderung sind folgende Bedrohungen relevant:

- Unberechtigte Nutzung von Diensten oder Ressourcen

ID: 3.04-21/5.0

Req 22 SSH X11 Forwarding muss deaktiviert werden.

X11 wird typischerweise auf Linux-Servern nicht verwendet. Die Weiterleitung von X11 über SSH muss deaktiviert werden.

Motivation: Diese Funktion stellt ein Sicherheitsrisiko für einen Server dar, wenn sie nicht in einer kontrollierten Art genutzt wird.

Umsetzungsbeispiel: `sshd_config` enthält:

`X11Forwarding no`

oder besser

`DisableForwarding yes`

Für diese Anforderung sind folgende Bedrohungen relevant:

- Unberechtigter Zugang zum System
- Unberechtigte Einsichtnahme oder Abgriff von Daten

- Unberechtigte Modifikation von Daten
- Unberechtigte Nutzung von Diensten oder Ressourcen
- Abstreiten von durchgeführten Aktionen
- Unbemerkt durchführbare Angriffe

ID: 3.04-22/5.0

Req 23 SSH PermitUserEnvironment muss deaktiviert werden.

Der SSH Option `PermitUserEnvironment` legt fest, ob kontospezifische Umgebungsvariablen von SSH ausgeführt werden. Die Funktion muss mittels Optionsargument "no" deaktiviert werden.

Motivation: Eine Anwendung von Umgebungsvariablen ermöglicht es konfigurierte Zugriffbeschränkungen von SSH zu umgehen.

Umsetzungsbeispiel: `sshd_config` enthält:
`PermitUserEnvironment no`

Für diese Anforderung sind folgende Bedrohungen relevant:

- Unberechtigte Nutzung von Diensten oder Ressourcen

ID: 3.04-23/5.0

Req 24 SSH PermitEmptyPasswords muss deaktiviert werden.

Mittels der Option "`PermitEmptyPasswords`" wird konfiguriert, ob ein Login über SSH mit einem Konto ohne Passwort möglich ist. Dies muss deaktiviert sein.

Motivation: Nicht berechnigte Benutzer können sich ggf. auf einen Server zugreifen, wenn die entfernte Anmeldung über SSH ohne Passwort möglich ist.

Umsetzungsbeispiel: `sshd_config` enthält:
`PermitEmptyPasswords no`

Für diese Anforderung sind folgende Bedrohungen relevant:

- Unberechtigter Zugang zum System
- Unberechtigte Einsichtnahme oder Abgriff von Daten
- Unberechtigte Modifikation von Daten
- Unberechtigte Nutzung von Diensten oder Ressourcen
- Abstreiten von durchgeführten Aktionen
- Unbemerkt durchführbare Angriffe

ID: 3.04-24/5.0

Req 25 Wenn SFTP aktiviert ist, muss der interne Server von SSH genutzt werden.

OpenSSH hat einen eigenen SFTP Daemon. Wenn SFTP genutzt werden soll, muss diese Funktion aktiviert und sicher konfiguriert werden.

Motivation: Es ist notwendig, den Open SSH SFTP Daemon zu nutzen, um eine einheitliche Konfiguration zu erreichen. Die ist notwendig, damit alle SSH-basierten Dienste das gleiche Sicherheitsniveau aufweisen.

Umsetzungsbeispiel: `sshd_config` enthält:
`Subsystem sftp internal-sftp`

Für diese Anforderung sind folgende Bedrohungen relevant:

- Unberechtigter Zugang zum System
- Unberechtigte Einsichtnahme oder Abgriff von Daten

- Unberechtigte Nutzung von Diensten oder Ressourcen

ID: 3.04-25/5.0

3. Zugriffsrechte & Schlüsselmateriale

Req 26	SSH-Konfigurationsdateien und private Anteile von SSH-Schlüsselmateriale müssen mit einer Maskierung von 0600 oder restriktiver geschützt sein. Öffentliche Schlüssel müssen eine Maskierung von 0644 oder restriktiver aufweisen .
--------	---

Die `sshd_config` und ggf. weitere Dateien unter `sshd_config.d/` beinhalten Konfiguration für den SSH-Daemon (`sshd`). Der SSH-Daemon wird meist mit root-Rechten ausgeführt und seine Konfiguration sowie das Schlüsselmateriale (`ssh_host_*_key` und `ssh_host_*_key.pub`) müssen angemessen geschützt werden.

Motivation: Konfiguration sowie Schlüsselmateriale müssen vor unautorisiertem Zugriff geschützt werden.

Umsetzungsbeispiel:

```
$ chown root:root /etc/ssh/sshd_config
$ chmod 600 /etc/ssh/sshd_config
$ chown root:root /etc/ssh/sshd_config.d/*
$ chmod 600 /etc/ssh/sshd_config.d/*
$ find /etc/ssh -xdev -type f -name 'ssh_host_*_key' -exec chown
root:root {} \;
$ find /etc/ssh -xdev -type f -name 'ssh_host_*_key' -exec chmod
600 {} \;
$ find /etc/ssh -xdev -type f -name 'ssh_host_*_key.pub' -exec
chown root:root {} \;
$ find /etc/ssh -xdev -type f -name 'ssh_host_*_key.pub' -exec
chmod 644 {} \;
```

Für diese Anforderung sind folgende Bedrohungen relevant:

- Unbemerkt durchführbare Angriffe
- Angriffe motiviert und erleichtert durch Informationspreisgabe oder erkennbare Sicherheitsschwachstellen

ID: 3.04-26/5.0

Req 27	Die Mindestschlüssellänge muss eingehalten werden. RSA-Schlüssel müssen min. 3072 Bit und ECDSA-Schlüssel müssen min. 256 Bit lang sein.
--------	--

Nutzerrollen: Betrieb, Entwicklung, Integration

Bei der Verwendung von (klassischen) asymmetrischen Algorithmen und Verfahren müssen mindestens die folgenden Schlüssellängen eingehalten werden ([TR-02102-1]):

- Verfahren basierend auf dem Faktorisierungsproblem wie RSA: 3000 Bit
- Verfahren basierend auf dem Diskreten-Logarithmus-Problem wie Diffie-Hellman (DH) oder Digital Signature Algorithm (DSA): 3000 Bit
- Verfahren basierend auf elliptischen Kurven (ECC) wie Elliptic Curve Diffie-Hellman (ECDH), Elliptic Curve Digital Signature Algorithm (ECDSA) oder Edwards-Curve Digital Signature Algorithm (EdDSA): 250 Bit

Hinweis zu zu PQC:

Wenn eine hybride Implementierung von klassischer und Post-Quanten-Kryptographie verwendet werden, dann muss der klassische Teil dieser Anforderung entsprechen.

Quelle:

[TR-02102-1] BSI-Technische Richtlinie: Kryptographische Verfahren - Empfehlungen und Schlüssellängen, Version 2025-01

Motivation: Die geforderten Schlüssellängen gelten bis mindestens 2030 als sicher.

ID: 3.04-28/5.0

Req 28 SSH Moduli kleiner als 3072 Bit dürfen nicht genutzt werden.

Die Datei `/etc/ssh/moduli` beinhaltet vorberechnete Parameter – sogenannte Moduli – für den Diffie-Hellmann Algorithmus. Es sind auch Moduli enthalten, die nicht lang genug sind, um gegen bekannte Angriffe resistent zu sein.

Um die Nutzung von Moduli-Werten kleiner als 3072 Bit zu verhindern, müssen diese aus der Datei `/etc/ssh/moduli` gelöscht werden. Die Spalte "size" in `/etc/ssh/moduli` listet den Index des höchstwertigen Bits (beginnend bei 0) auf, was bedeutet, dass ein 3072-Bit Primzahl, deren höchstes gesetztes Bit an Position 3071 liegt, als 3071 aufgeführt wird.

Quelle:

[TR 02102-4] BSI-Technische Richtlinie: Kryptographische Verfahren - Empfehlungen und Schlüssellängen - Teil 4 SSH, Version 2025-01

Motivation: Wenn der genutzte DH-Modulus zu kurz ist, ist der Austausch von Schlüsseln nicht ausreichend geschützt.

Umsetzungsbeispiel:

```
$ awk '$5 >= 3071' /etc/ssh/moduli > /etc/ssh/moduli.safe  
$ mv /etc/ssh/moduli.safe /etc/ssh/moduli
```

Für diese Anforderung sind folgende Bedrohungen relevant:

- Unberechtigter Zugang zum System
- Unberechtigte Einsichtnahme oder Abgriff von Daten
- Unberechtigte Modifikation von Daten
- Unberechtigte Nutzung von Diensten oder Ressourcen

ID: 3.50-47/8.0