Bundesamt
für Sicherheit in der
Informationstechnik

Technische Richtlinie TR-02102-4 Kryptographische Verfahren: Empfehlungen und Schlüssellängen

Teil 4 – Verwendung von Secure Shell (SSH)

Version 2020-01

Änderungshistorie

Version	Datum	Beschreibung
2019-01	22.02.2019	Anpassung der Verwendungszeiträume, Empfehlung von Diffie-Hellman Gruppen aus RFC 8268
2020-01	31.01.2020	Anpassung der Verwendungszeiträume, Abkündigung von HMAC-SHA-1

Bundesamt für Sicherheit in der Informationstechnik
Postfach 20 03 63
53133 Bonn
E-Mail: TR02102@bsi.bund.de
Internet: https://www.bsi.bund.de

Inhaltsverzeichnis

Tabellenverzeichnis

1 Einleitung

Diese Technische Richtlinie gibt Empfehlungen für den Einsatz des kryptographischen Protokolls *Secure Shell (SSH).* Mit diesem Protokoll kann ein sicherer Kanal in einem unsicheren Netzwerk aufgebaut werden. Die gängigsten Anwendungen des SSH-Protokolls sind das Anmelden auf einem entfernten System (remote command-line login) und das Ausführen von Befehlen bzw. Applikationen auf entfernten Systemen.

Die vorliegende Technische Richtlinie enthält Empfehlungen für die zu verwendende Protokollversion und die kryptographischen Algorithmen als Konkretisierung der allgemeinen Empfehlungen in Teil 1 dieser Technischen Richtlinie (siehe [TR-02102-1]).

Diese Richtlinie enthält keine Empfehlungen für konkrete Anwendungen, keine Risikobewertungen sowie keine Angriffsmöglichkeiten, die sich aus Fehlern in der Implementierung des Protokolls ergeben.

Hinweis: Auch bei Beachtung aller Empfehlungen für die Verwendung von SSH können Daten in erheblichem Umfang aus einem kryptographischen System abfließen, zum Beispiel durch Ausnutzung von Seitenkanälen (Messung von Timing-Verhalten, Stromaufnahme, Datenraten etc.). Daher sollte der Entwickler unter Hinzuziehung von Experten auf diesem Gebiet mögliche Seitenkanäle identifizieren und entsprechende Gegenmaßnahmen umsetzen. Je nach Anwendung gilt dies auch für Fault-Attacken.

Hinweis: Für Definitionen kryptographischer Begriffe in diesem Dokument siehe das Glossar in [TR-02102-1].

2 Grundlagen

Das SSH-Protokoll besteht aus den drei Unter-Protokollen *Transport Layer Protocol*, *User Authentication Protocol* und *Connection Protocol*.

Das Transport Layer Protocol (siehe [RFC4253]) ermöglicht Serverauthentisierung, Verschlüsselung, Integritätssicherung und optional Datenkompression. Es setzt logisch auf dem TCP/IP-Protokoll auf.

Das User Authentication Protocol (siehe [RFC4252]) ist dafür da, den Benutzer gegenüber dem Server zu authentisieren. Es setzt auf dem Transport Layer Protocol auf.

Das Connection Protocol (siehe [RFC4254]) ist für die Erzeugung und Verwaltung logischer Kanäle innerhalb des verschlüsselten Tunnels zuständig. Es setzt auf dem User Authentication Protocol auf.

Für weitergehende Informationen über die Protokollarchitektur von SSH siehe [RFC4251].

Die umfangreiche Spezifikation des SSH-2-Protokolls (siehe Abschnitt 3.2) findet sich in folgenden RFCs:

- RFC 4250: The Secure Shell (SSH) Protocol Assigned Numbers (Januar 2006)
- RFC 4251: The Secure Shell (SSH) Protocol Architecture (Januar 2006)
- RFC 4252: The Secure Shell (SSH) Authentication Protocol (Januar 2006)
- RFC 4253: The Secure Shell (SSH) Transport Layer Protocol (Januar 2006)
- RFC 4254: The Secure Shell (SSH) Connection Protocol (Januar 2006)
- RFC 4256: Generic Message Exchange Authentication for the Secure Shell Protocol (SSH) (Januar 2006)
- RFC 4335: The Secure Shell (SSH) Session Channel Break Extension (Januar 2006)
- RFC 4344: The Secure Shell (SSH) Transport Layer Encryption Modes (Januar 2006)

Die folgenden RFCs enthalten Erweiterungen und Ergänzungen des SSH-Protokolls:

- RFC 4255: Using DNS to Securely Publish Secure Shell (SSH) Key Fingerprints (Januar 2006)
- RFC 4419: Diffie-Hellman Group Exchange for the Secure Shell (SSH) Transport Layer Protocol (März 2006)
- RFC 4432: RSA Key Exchange for the Secure Shell (SSH) Transport Layer Protocol (März 2006)
- RFC 4462: Generic Security Service Application Program Interface (GSS-API) Authentication and Key Exchange for the Secure Shell (SSH) Protocol (Mai 2006)
- RFC 4716: The Secure Shell (SSH) Public Key File Format (November 2006)
- RFC 4819: Secure Shell Public Key Subsystem (März 2007)
- RFC 5647: AES Galois Counter Mode for the Secure Shell Transport Layer Protocol (August 2009)
- RFC 5656: Elliptic Curve Algorithm Integration in the Secure Shell Transport Layer (Dezember 2009)
- RFC 6187: X.509v3 Certificates for Secure Shell Authentication (März 2011)
- RFC 6239: Suite B Cryptographic Suites for Secure Shell (SSH) (Mai 2011)
- RFC 6594: Use of the SHA-256 Algorithm with RSA: Digital Signature Algorithm (DSA): and Elliptic Curve DSA (ECDSA) in SSHFP Resource Records (April 2012)
- RFC 6668: SHA-2 Data Integrity Verification for the Secure Shell (SSH) Transport Layer Protocol (Juli 2012)
- RFC 8268: More Modular Exponentiation (MODP) Diffie-Hellman (DH) Key Exchange (KEX) Groups for Secure Shell (SSH) (Dezember 2017)

- RFC 8270: Increase the Secure Shell Minimum Recommended Diffie-Hellman Modulus Size to 2048 Bits (Dezember 2017)

3 Empfehlungen

Dieses Kapitel enthält Empfehlungen für die Verwendung des SSH-Protokolls. Diese beziehen sich auf die zu verwendenden kryptographischen Verfahren und die SSH-Versionen. Die vorliegende Technische Richtlinie enthält keine Konfigurationsanleitungen für konkrete Implementierungen des SSH-Protokolls, sondern grundsätzliche kryptographische Empfehlungen, die für alle SSH-Implementierungen verwendet werden können.

3.1 Allgemeine Hinweise

3.1.1 Verwendungszeiträume

Die Empfehlungen in dieser Technischen Richtlinie sind mit Verwendungszeiträumen versehen. Die Angabe der Jahreszahl bedeutet hierbei, dass das entsprechende Verfahren bis zum Ende des angegebenen Jahres empfohlen wird. Ist die Jahreszahl mit einem „+“-Zeichen gekennzeichnet, so bedeutet dies, dass dieser Verwendungszeitraum möglicherweise in einer zukünftigen Version dieser Technischen Richtlinie verlängert wird.

3.1.2 Sicherheitsniveau

Das Sicherheitsniveau für alle kryptographischen Verfahren in dieser Technischen Richtlinie richtet sich nach dem in Abschnitt 1.1 in [TR-02102-1] angegebenen Sicherheitsniveau. Es liegt zurzeit bei 100 Bit.

Hinweis: Ab dem Jahr 2023 wird ein Sicherheitsniveau von 120 Bit angestrebt. Als Übergangsregelung ist die Verwendung von RSA-basierten Signatur- und Verschlüsselungsverfahren mit einer Schlüssellänge ab 2000 Bit für das gesamte Jahr 2023 aber weiter konform zu dieser Richtlinie. Siehe dazu auch Abschnitt 1.1 in [TR-02102-1].

3.2 SSH-Versionen

Im Jahr 1995 wurde die erste Version des SSH-Protokolls von Tatu Ylönen, einem Forscher an der Helsinki University of Technology, entwickelt. Diese Version wird heute SSH-1 genannt. Im Jahr 2006 wurde eine überarbeitete Version des Protokolls als Internet Standard (RFC) durch die IETF verabschiedet; diese Version heißt SSH-2.

Es gelten folgende Empfehlungen für die Auswahl der SSH-Protokollversion:

- Die Verwendung von SSH-2 wird empfohlen.
- Der Einsatz von SSH-1 wird **nicht empfohlen**, da diese Protokollversion kryptographische Schwächen enthält.

3.2.1 Konformität zur SSH-Spezifikation

Die Spezifikation des SSH-Protokolls enthält kryptographische Algorithmen, die von standardkonformen Anwendungen unterstützt werden müssen. In der vorliegenden Technischen Richtlinie werden davon möglicherweise nicht alle empfohlen. Der Grund dafür ist, dass die SSH-Spezifikation (siehe Liste der RFCs in Kapitel 2) im Wesentlichen aus dem Jahr 2006 stammt und daher mittlerweile einige Verfahren aus dieser Spezifikation veraltet sind oder möglicherweise keine ausreichende Sicherheit mehr bieten.

Wenn eine Anwendung vollständig konform zur SSH-Spezifikation sein muss, dann sollte wie folgt vorgegangen werden: Die Anwendung sollte die im vorliegenden Dokument empfohlenen Algorithmen *und* die in der Spezifikation festgelegten Verfahren unterstützen, aber so konfiguriert sein, dass die hier empfohlenen (kryptographisch starken) Algorithmen mit hoher Priorität und die (möglicherweise

kryptographisch schwachen oder veralteten) Algorithmen aus der SSH-Spezifikation mit niedriger Priorität bzw. nach Möglichkeit nicht verwendet werden.

3.3 Schlüsseleinigung

Im Rahmen des SSH-Verbindungsaufbaus wird ein Schlüsselaustausch (Key Exchange) durchgeführt, um gemeinsame Sitzungsschlüssel für die Authentisierung und die Verschlüsselung zu erzeugen und auszutauschen.

Folgende Key Exchange Methods werden empfohlen:

Lfd. Nr.	Key Exchange Method	Spezifikation	Empfohlene Verwendung bis
1	`diffie-hellman-group-exchange-sha256`	Abschnitt 4.2 in [RFC4419]	2026+
2	`diffie-hellman-group14-sha256`	Kapitel 3 in [RFC8268]	2022
3	`diffie-hellman-group15-sha512`	Kapitel 3 in [RFC8268]	2026+
4	`diffie-hellman-group16-sha512`	Kapitel 3 in [RFC8268]	2026+
5	`rsa2048-sha256`	Kapitel 4 und 6 in [RFC4432]	2023
6	`ecdh-sha2-*`	Abschnitt 6.3 in [RFC5656]	2026+

Tabelle 1: Empfohlene Key Exchange Methods

Bemerkung zu Key Exchange Method Nr. 1:

Unter Verwendung der Bezeichnungen aus Kapitel 3 in [RFC4419]:

1. Die Länge der Primzahl p soll mindestens 2000 Bit betragen (siehe dazu auch Abschnitt 7.2.1 in [TR-02102-1] sowie [RFC8270]).

   Wichtiger Hinweis: Ab dem Jahr 2023 wird für p eine Mindestlänge von 3000 Bit empfohlen. Siehe auch Abschnitt 3.1.2.

2. Die Ordnung des Erzeugers g soll mindestens 2^{250} groß sein (siehe dazu auch Abschnitt 7.2.1 in [TR-02102-1]).

3. Gemäß Kapitel 3 in [RFC4419] soll p eine *Safe Prime* sein, das heißt mit $p=2q+1$ sind sowohl p als auch q prim.

Da p eine *Safe Prime* ist, gilt $p-1=2q$, das heißt die Ordnung des Erzeugers g kann nur 2 oder q sein. Beachtet man die Empfehlung in Punkt 2, so bleibt nur q als mögliche Ordnung übrig. Aufgrund der zusätzlichen Forderung in Punkt 3 (Safe Prime) ist die Bitlänge von q viel größer als in Punkt 2 mindestens empfohlen wird. Dieser Umstand ist zu akzeptieren, wenn die Implementierung konform zu [RFC4419], [TR-02102-1] und der vorliegenden Technischen Richtlinie sein soll.

Hinweis: Bei dieser Key Exchange Method muss SHA-256 auch für die key derivation pseudo-random function (PRF) verwendet werden.

Bemerkung zu Key Exchange Method Nr. 6:

Das „*“-Zeichen wird ersetzt durch den Bezeichner einer elliptischen Kurve aus Abschnitt 10.1 in [RFC5656]. Zurzeit werden die folgenden elliptischen Kurven empfohlen:

`nistp256, nistp384, nistp521`

Die zugehörige Hashfunktion aus der SHA-2-Familie muss in Abhängigkeit der Bitlänge der Kurve gemäß Abschnitt 6.2.1 in [RFC5656] gewählt werden.

Es ist vorgesehen, ebenfalls die Brainpool-Kurven zu empfehlen, sobald diese in einem Internet-Standard für das SSH-Protokoll standardisiert sind.

3.3.1 Key Re-Exchange

Es ist sinnvoll, das Schlüsselmaterial einer Verbindung nach einer bestimmten Zeit oder einer bestimmten Menge übertragener Daten zu erneuern, um einen Angriff auf die Sitzungsschlüssel zu erschweren. Bei SSH kann das Erneuern der Sitzungsschlüssel durch das Senden der Nachricht `SSH_MSG_KEXINIT` erreicht werden. Sowohl Client als auch Server können diesen Vorgang initiieren.

Es wird empfohlen, ein Key Re-Exchange gemäß Kapitel 9 in [RFC4253] durchzuführen, das heißt die Sitzungsschlüssel werden nach einer Stunde oder nach der Übertragung von einem Gigabyte (je nachdem, was zuerst eintritt) erneuert.

3.4 Verschlüsselungsalgorithmen

Während des Key Exchange einigen sich Client und Server auf einen Verschlüsselungsalgorithmus sowie einen gemeinsamen Verschlüsselungs-Schlüssel. Hierzu werden die folgenden Verschlüsselungsverfahren empfohlen:

Lfd. Nr.	Verfahren	Spezifikation	Empfohlene Verwendung bis
1	`AEAD_AES_128_GCM`	Abschnitt 6.1 in [RFC5647]	2026+
2	`AEAD_AES_256_GCM`	Abschnitt 6.2 in [RFC5647]	2026+
3	`aes256-cbc`	Abschnitt 6.3 in [RFC4253]	2024
4	`aes192-cbc`	Abschnitt 6.3 in [RFC4253]	2024
5	`aes128-cbc`	Abschnitt 6.3 in [RFC4253]	2024
6	`aes128-ctr`	Abschnitt 4 in [RFC4344]	2026+
7	`aes192-ctr`	Abschnitt 4 in [RFC4344]	2026+
8	`aes256-ctr`	Abschnitt 4 in [RFC4344]	2026+

Tabelle 2: Empfohlene Verschlüsselungs-Verfahren

Hinweis: Bei den Verfahren Nr. 1 und Nr. 2 ist durch den GCM-Modus schon eine MAC-Sicherung enthalten.

Bemerkung: Wenn möglich, sollten die Verfahren Nr. 1 und 2 aus obiger Tabelle eingesetzt werden, da es für den GCM-Modus beweisbare Sicherheitsaussagen bzgl. des Sicherheitsziels *Authentisierte Verschlüsselung* gibt. Für die Verfahren Nr. 3 bis 8 sind vergleichbare Sicherheitsgarantien nicht verfügbar, weil SSH die Verschlüsselung und Authentisierung im Encrypt-and-MAC-Modus (statt der beweisbar sicheren Encrypt-then-MAC-Reihenfolge) kombiniert.

3.5 MAC-Sicherung

Für die MAC-Sicherung werden die folgenden Verfahren empfohlen:

Lfd. Nr.	Verfahren	Spezifikation	Empfohlene Verwendung bis
1	`hmac-sha1`[1]	Abschnitt 6.4 in [RFC4253]	2019
2	`hmac-sha2-256`	Kapitel 2 in [RFC6668]	2026+
3	`hmac-sha2-512`	Kapitel 2 in [RFC6668]	2026+

Tabelle 3: Empfohlene Verfahren zur MAC-Sicherung

3.6 Server-Authentisierung

Der Server authentisiert sich gegenüber dem Client; dies läuft im Rahmen des Transport Layer Protocol ab. In Kapitel 7 von [RFC4253] wird dazu die *Explicit Server Authentication* beschrieben. Dabei enthalten die Key Exchange-Nachrichten eine digitale Signatur des Servers (oder einen anderen Nachweis), um dessen Authentizität zu beweisen. Der Client kann die Signatur mit dem Public Key des Servers überprüfen und somit die Authentizität des Servers feststellen.

Die Algorithmen für digitale Signaturen werden in [RFC4253] „Public Key Algorithms“ genannt (vgl. Abschnitt 6.6 in [RFC4253]).

Folgende Verfahren für die Server-Authentisierung werden empfohlen:

Lfd. Nr.	Verfahren	Spezifikation	Schlüssel-länge	Empfohlene Verwendung bis
1	`pgp-sign-rsa`[2]	Abschnitt 6.6 in [RFC4253] sowie Abschnitte 9.1 und 9.4 in [RFC4880]	2000 Bit	2020
2	`pgp-sign-dss`[2]	Abschnitt 6.6 in [RFC4253] sowie Abschnitte 9.1 und 9.4 in [RFC4880]	2000 Bit / 250 Bit	2022
3	`ecdsa-sha2-*`	Kapitel 3 in [RFC5656]	250 Bit	2026+
4	`x509v3-rsa2048-sha256`	Abschnitt 3.3 in [RFC6187]	2048 Bit	2020
5	`x509v3-ecdsa-sha2-*`	Abschnitt 3.4 in [RFC6187]	250 Bit	2026+

Tabelle 4: Empfohlene Verfahren für die Server-Authentisierung

Bemerkung: Die Verfahren Nr. 1 und Nr. 4 verwenden das Codierungs-Verfahren `EMSA-PKCS1-v1_5` (vgl. Abschnitt 9.2 in [RFC3447]). Daher der verkürzte Verwendungszeitraum.

Bemerkung zu Verfahren Nr. 2: Ab dem Jahr 2023 werden mindestens 3000 Bit empfohlen. Siehe dazu auch Abschnitt 3.1.2.

Bemerkung: Die Verfahren `ssh-dss`, `ssh-rsa` (vgl. Abschnitt 6.6 in [RFC4253]) sowie `x509v3-ssh-dss`, `x509v3-ssh-rsa` (vgl. Abschnitte 3.1 und 3.2 in [RFC6187]) verwenden bei der Signaturerstellung die Hashfunktion SHA-1 und werden daher nicht empfohlen[1]. Da die beiden erstgenannten Verfahren gemäß [RFC4253] erforderlich bzw. empfohlen sind, sei hier auf den Hinweis in Abschnitt 3.2.1 verwiesen.

1 Aufgrund von Angriffen gegen die Kollisionsresistenz-Eigenschaften von SHA-1 (siehe auch Abschnitt 1.4 und Bemerkung 13 in [TR-02102-1] sowie [SBK17, LP19]), muss darauf geachtet werden, ob die Kollisionsresistenz beim Einsatz von SHA-1 benötigt wird. Bei der Signaturerstellung ist dies der Fall. Bei der HMAC-Berechnung wird die Kollisionsresistenz zwar nicht benötigt, die Verwendung von SHA-1 sollte hier aber ebenfalls vermieden werden.

2 Dieses Verfahren wird nur in Verbindung mit einer zur Schlüssellänge passenden Hashfunktion aus der SHA-2-Familie empfohlen (vgl. Abschnitt 9.4 in [RFC4880]).

Bemerkung zu den Verfahren Nr. 3 und Nr. 5: Das „*“-Zeichen wird ersetzt durch den Bezeichner einer elliptischen Kurve aus Abschnitt 10.1 in [RFC5656]. Zurzeit werden die folgenden elliptischen Kurven empfohlen:

`nistp256`, `nistp384`, `nistp521`

Die zugehörige Hashfunktion aus der SHA-2-Familie muss in Abhängigkeit der Bitlänge der Kurve gemäß Abschnitt 6.2.1 in [RFC5656] gewählt werden.

Es ist vorgesehen, ebenfalls die Brainpool-Kurven für die Verfahren Nr. 3 und Nr. 5 zu empfehlen, sobald diese in einem Internet-Standard für das SSH-Protokoll standardisiert sind.

Für die Authentisierung innerhalb von Projekten des Bundes sind die Vorgaben der Technischen Richtlinie [TR-03116-4] in der jeweils aktuellen Fassung zu beachten.

3.7 Client-Authentication

Die Client-Authentisierung findet (im Gegensatz zur Server-Authentisierung) nicht im Transport Layer Protocol, sondern im User Authentication Protocol statt; dieses Protokoll setzt logisch auf dem Transport Layer Protocol auf.

Die wichtigsten Verfahren für die Client-Authentisierung sind:

- Public key authentication
- Password authentication
- Host-based authentication

Empfehlung: Für die Client-Authentisierung wird die „Public key authentication“ zusammen mit einem der Verfahren aus Tabelle 4, Abschnitt 3.6 empfohlen.

Anmerkung: Die Public key authentication muss gemäß [RFC4252] von jeder SSH-Implementierung unterstützt werden. Der dazu gehörige Authentication Method Name gemäß [RFC4250], Abschnitt 4.8 lautet „publickey“. Die Authentisierungs-Methode wird in Kapitel 7 von [RFC4242] beschrieben.

Für die Authentisierung innerhalb von Projekten des Bundes sind die Vorgaben der Technischen Richtlinie [TR-03116-4] in der jeweils aktuellen Fassung zu beachten.

4 Schlüssel und Zufallszahlen

4.1 Schlüsselspeicherung

Private kryptographische Schlüssel, insbesondere statische Schlüssel und Signaturschlüssel, müssen sicher gespeichert und verarbeitet werden. Dies bedeutet u. a. den Schutz vor Kopieren, missbräuchlicher Nutzung und Manipulation der Schlüssel. Eine sichere Schlüsselspeicherung kann z. B. durch die Verwendung entsprechend zertifizierter Hardware (Chipkarte, HSM) gewährleistet werden.

Ebenso müssen die öffentlichen Schlüssel von als vertrauenswürdig erkannten Stellen (Vertrauensanker) manipulationssicher gespeichert werden.

4.2 Umgang mit Ephemer-Schlüsseln

Wenn eine SSH-Verbindung durch ein Verschlüsselungsverfahren gesichert ist, muss sichergestellt werden, dass alle Ephemer-Schlüssel nach ihrer Verwendung unwiderruflich gelöscht werden, und keine Kopien dieser Schlüssel erzeugt wurden. Ephemer- bzw. Sitzungsschlüssel dürfen nur für *eine* Verbindung benutzt werden und sollten grundsätzlich nicht persistent abgespeichert werden.

4.3 Zufallszahlen

Für die Erzeugung von Zufallszahlen, z. B. für kryptographische Schlüssel oder die Signaturerzeugung, müssen geeignete Zufallszahlengeneratoren eingesetzt werden.

Empfohlen wird ein Zufallszahlengenerator aus einer der Klassen DRG.3, DRG.4, PTG.3 oder NTG.1 gemäß [AIS20/31], vgl. auch Kapitel 9 in Teil 1 dieser Technischen Richtlinie.

Literaturverzeichnis

ID	Referenz
AIS20/31	BSI: W. Killmann, W. Schindler, AIS 20/31 – A proposal for: Functionality classes for random number generators, 2011
LP19	G. Leurent, T. Peyrin: From Collisions to Chosen-Prefix Collisions – Application to Full SHA-1, EUROCRYPT 2019, Lecture Notes in Computer Science, vol. 11478, 2019
RFC3447	J. Jonsson, B. Kaliski: RFC 3447, Public-Key Cryptography Standards (PKCS) #1: RSA Cryptography Specifications Version 2.1, 2003
RFC4251	T. Ylonen, C. Lonvick: RFC 4251, The Secure Shell (SSH) Protocol Architecture, 2006
RFC4252	T. Ylonen, C. Lonvick: RFC 4252, The Secure Shell (SSH) Authentication Protocol, 2006
RFC4253	T. Ylonen, C. Lonvick: RFC 4253, The Secure Shell (SSH) Transport Layer Protocol, 2006
RFC4254	T. Ylonen, C. Lonvick: RFC 4254, The Secure Shell (SSH) Connection Protocol, 2006
RFC4344	M. Bellare, T. Kohno, C. Namprempre: RFC 4344, The Secure Shell (SSH) Transport Layer Encryption Modes, 2006
RFC4419	M. Friedl, N. Provos, W. Simpson: RFC 4419, Diffie-Hellman Group Exchange for the Secure Shell (SSH) Transport Layer Protocol, 2006
RFC4432	B. Harris: RFC 4432, RSA Key Exchange for the Secure Shell (SSH) Transport Layer Protocol, 2006
RFC4880	J. Callas, L. Donnerhacke, H. Finney, D. Shaw, R. Thayer: RFC 4880, OpenPGP Message Format, 2007
RFC5647	K. Igoe, J. Solinas: RFC 5647, AES Galois Counter Mode for the Secure Shell Transport Layer Protocol, 2009
RFC5656	D. Stebila, J. Green: RFC 5656, Elliptic Curve Algorithm Integration in the Secure Shell Transport Layer, 2009
RFC6668	D. Bider, M. Baushke: RFC 6668, SHA-2 Data Integrity Verification for the Secure Shell (SSH) Transport Layer Protocol, 2012
RFC8268	M. Baushke: RFC 8268, More Modular Exponentiation (MODP) Diffie-Hellman (DH) Key Exchange (KEX) Groups for Secure Shell (SSH), 2017
RFC8270	L. Velvindron, M. Baushke: RFC 8270, Increase the Secure Shell Minimum Recommended Diffie-Hellman Modulus Size to 2048 Bits, 2017
SBK17	M. Stevens, E. Bursztein, P. Karpman, A. Albertini, Y. Markov: The first collision for full SHA-1. CRYPTO 2017, Lecture Notes in Computer Science, vol. 10401, 2017
TR-02102-1	BSI: Technische Richtlinie TR-02102-1, Kryptographische Verfahren: Empfehlungen und Schlüssellängen, 2020
TR-03116-4	BSI: Technische Richtlinie TR-03116-4, Kryptographische Vorgaben für Projekte der Bundesregierung, Teil 4 – Kommunikationsverfahren in Anwendungen, 2020