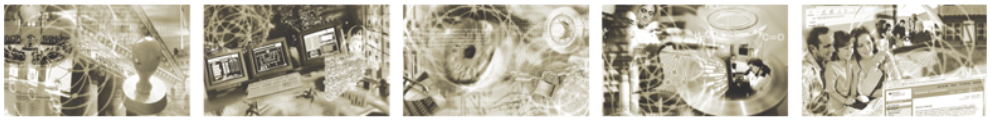
Bundesamt
für Sicherheit in der
Informationstechnik



Technische Richtlinie TR-02102-2

Kryptographische Verfahren: Empfehlungen und Schlüssellängen

Teil 2 – Verwendung von Transport Layer Security (TLS)

(Version 2016-01)

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 *Transport Layer Security (TLS).* Es dient der sicheren Übertragung von Informationen in Datennetzwerken, wobei insbesondere die Vertraulichkeit, die Integrität und die Authentizität der übertragenen Informationen geschützt werden können.

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

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

Hinweis: Auch bei Beachtung aller Vorgaben für die Verwendung von TLS können Daten in erheblichem Umfang aus einem kryptographischen System abfließen, z. B. 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

Transport Layer Security (TLS), früher bekannt als Secure Socket Layer (SSL), ermöglicht die sichere Übertragung von Informationen aus der Anwendungsschicht (z. B. HTTPS, FTPS oder IMAPS) über TCP/IP-basierte Verbindungen (insbesondere das Internet).

Bevor Daten übermittelt werden können, muss eine (gesicherte) Verbindung zwischen den zwei Verbindungspartnern (Client und Server) aufgebaut werden. Dieser Vorgang heißt *Handshake* und ist ein wichtiger Bestandteil des TLS-Protokolls. Hierbei werden zwischen Client und Server vereinbart:

1. Kryptographische Verfahren zur *Datenverschlüsselung*, *Integritätssicherung, Schlüsseleinigung* und ggf. zur (ein- oder beidseitigen) *Authentisierung.* Diese Verfahren werden durch die *Cipher-Suite* festgelegt (siehe Abschnitt 3.3).
2. Ein gemeinsames Geheimnis, das *premaster secret*. Aus diesem wird (von beiden Verbindungspartnern) das *master secret* erzeugt, aus welchem wiederum die Sitzungsschlüssel für den Integritätsschutz und die Verschlüsselung abgeleitet werden.

Hinweis: Das TLS-Protokoll erlaubt auch Verbindungen, die nicht oder nur einseitig authentisiert sind (Beispiel: HTTPS-Verbindungen sind üblicherweise nur serverseitig authentisiert). Daher sollten Systementwickler darauf achten, ob eine weitere Authentisierung in der Anwendungsschicht erforderlich ist (Beispiel: Authentisierung eines Homebanking-Benutzers durch Anforderung eines Passwortes). Bei Anforderung besonders kritischer Operationen sollte dabei grundsätzlich eine Au-

thentisierung durch Wissen und Besitz (Zwei-Faktor-Authentisierung) erfolgen, die sich unter Ausnutzung kryptographischer Mechanismen auch auf die übertragenen Daten erstrecken sollte.

3 Vorgaben

3.1 Allgemeine Hinweise

3.1.1 Verwendungszeiträume

Die Vorgaben und Empfehlungen in dieser Technischen Richtlinie sind jeweils mit einem maximalen Verwendungszeitraum versehen. Die Angabe der Jahreszahl bedeutet, dass das entsprechende Verfahren bis zum Ende des angegebenen Jahres eingesetzt werden kann. Ist die Jahreszahl mit einem „+“-Zeichen gekennzeichnet, so besteht die Möglichkeit einer Verlängerung des Verwendungszeitraums.

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 mindestens 100 Bit.

3.1.3 Schlüssellängen bei EC-Verfahren

Die Schlüssellängen bei Verfahren, die auf elliptischen Kurven (EC) basieren, sind – im Vergleich zum Sicherheitsniveau von RSA – in dieser Technischen Richtlinie etwas größer gewählt worden, um einen Sicherheitsspielraum für die EC-Verfahren zu erreichen (vgl. Abschnitt 3.6). Für die Begründung und weitere Erläuterungen siehe Bemerkung 4 in Kapitel 3 in [TR-02102-1].

3.2 SSL/TLS-Versionen

Das SSL-Protokoll existiert in den Versionen 1.0, 2.0 und 3.0, wobei die Version 1.0 nicht veröffentlicht wurde. TLS 1.0 ist eine direkte Weiterentwicklung von SSL 3.0 und wird in [RFC2246] spezifiziert. Des weiteren gibt es das TLS-Protokoll in den Versionen 1.1 und 1.2, welche in [RFC4346] und [RFC5246] spezifiziert werden.

Empfehlungen für die Wahl der TLS-Version:

- Grundsätzlich soll TLS 1.2 eingesetzt werden.
- TLS 1.1 wird **nicht mehr empfohlen** (siehe dazu Abschnitt 3.3.2).
- TLS 1.0 wird **nicht empfohlen.**
- SSL v2 ([SSLv2]) und SSL v3 ([SSLv3]) werden **nicht empfohlen** (siehe auch [RFC6176]).

3.3 Cipher-Suiten

Eine Cipher-Suite spezifiziert die zu verwendenden Algorithmen für

- die Schlüsseleinigung (und ggf. Authentisierung),

- die Nutzdaten-Verschlüsselung (Stromchiffre oder Blockchiffre inkl. Betriebsmodus), und
- eine Hashfunktion für die Integritätssicherung (HMAC-Algorithmus) der Datenpakete und für die Verwendung als Pseudozufallszahlengenerator (ab TLS 1.2).

Eine vollständige Liste aller definierten Cipher-Suiten mit Verweisen auf die jeweiligen Spezifikationen ist verfügbar unter [IANA].

3.3.1 Empfohlene Cipher-Suiten

Grundsätzlich wird empfohlen, nur Cipher-Suiten einzusetzen, die die Anforderungen an die Algorithmen und Schlüssellängen aus [TR-02102-1] erfüllen.

Es wird die Verwendung der folgenden Cipher-Suiten empfohlen (vgl. [RFC5246] und [RFC5289]):

	Schlüsseleinigung und -authentisierung		***Verschlüsselung***	***Betriebs-modus***	***Hash***	***Verwendung bis***
TLS_	ECDHE_ECDSA_	WITH_	AES_128_	CBC_ GCM_	SHA256	2022+
			AES_256_	CBC_ GCM_	SHA384	2022+
	ECDHE_RSA_	WITH_	AES_128_	CBC_ GCM_	SHA256	2022+
			AES_256_	CBC_ GCM_	SHA384	2022+
	DHE_DSS_[1]	WITH_	AES_128_	CBC_ GCM_	SHA256	2022+
			AES_256_	CBC_	SHA256	2022+
				GCM_	SHA384	2022+
	DHE_RSA_[1]	WITH_	AES_128_	CBC_ GCM_	SHA256	2022+
			AES_256_	CBC_	SHA256	2022+
				GCM_	SHA384	2022+

Tabelle 1: Empfohlene Cipher-Suiten mit Forward Secrecy

Sofern die Verwendung von Cipher-Suiten mit Forward Secrecy nicht möglich ist[2], können auch die folgenden Cipher-Suiten eingesetzt werden (vgl. [RFC5246] und [RFC5289]):

1 Da einige gängige Implementierungen von DH(E) in TLS zurzeit nur 1024 Bit unterstützen, sei hier auf Abschnitt 7.2.1 in [TR-02102-1] verwiesen, in welcher eine Mindestgröße von 2000 Bit für dieses Verfahren empfohlen wird.

2 Forward Secrecy (auch Perfect Forward Secrecy, kurz PFS) bedeutet, dass eine Verbindung auch bei Kenntnis der Langzeit-Schlüssel der Kommunikationspartner nicht nachträglich entschlüsselt werden kann. Bei der Verwendung von TLS zum Schutz personenbezogener oder anderer sensibler Daten ist Forward Secrecy grundsätzlich notwendig.

	Schlüsseleinigung und -authentisierung		***Verschlüsselung***	***Betriebs-modus***	***Hash***	***Verwendung bis***
TLS_	ECDH_ECDSA_	WITH_	AES_128_	CBC_ GCM_	SHA256	2022+
			AES_256_	CBC_ GCM_	SHA384	2022+
	ECDH_RSA_	WITH_	AES_128_	CBC_ GCM_	SHA256	2022+
			AES_256_	CBC_ GCM_	SHA384	2022+
	DH_DSS_	WITH_	AES_128_	CBC_ GCM_	SHA256	2022+
			AES_256_	CBC_	SHA256	2022+
				GCM_	SHA384	2022+
	DH_RSA_	WITH_	AES_128_	CBC_ GCM_	SHA256	2022+
			AES_256_	CBC_	SHA256	2022+
				GCM_	SHA384	2022+

Tabelle 2: Empfohlene Cipher-Suiten ohne Forward Secrecy

Sofern zusätzliche vorab ausgetauschte Daten in die Schlüsseleinigung einfließen sollen (*Pre-Shared Key*), bietet TLS die Verwendung entsprechender Cipher-Suiten (vgl. [RFC5489]). Es wird die Verwendung von Cipher-Suiten empfohlen, bei der neben dem Pre-Shared Key weitere ephemerale Schlüssel oder ausgetauschte Zufallszahlen in die Schlüsseleinigung eingehen. Die Verwendung von `TLS_PSK_*` (d. h. ohne zusätzliche ephemerale Schlüssel/Zufallszahlen) wird *nicht* empfohlen, da bei diesen Cipher-Suiten die Sicherheit der Verbindung ausschließlich auf der Entropie und der Vertraulichkeit des Pre-Shared Keys beruht.

	Schlüsseleinigung und -authentisierung		*Verschlüsselung*	*Betriebs-modus*	*Hash*	*Verwendung bis*
TLS_	ECDHE_PSK_	WITH_	AES_128_	CBC_	SHA256	2022+
			AES_256_		SHA384	2022+
	DHE_PSK_	WITH_	AES_128_	CBC_	SHA256	2022+
				GCM_		2022+
			AES_256_	CBC_	SHA384	2022+
				GCM_		2022+
	RSA_PSK_	WITH_	AES_128_	CBC_	SHA256	2022+
				GCM_		2022+
			AES_256_	CBC_	SHA384	2022+
				GCM_		2022+

Tabelle 3: Empfohlene Cipher-Suiten mit Pre-Shared Key

Hinweis: Die Cipher-Suiten TLS_RSA_PSK_* in Tabelle 3 bieten *keine* Forward Secrecy, alle anderen Cipher-Suiten aus Tabelle 3 bieten Forward Secrecy.

3.3.2 Übergangsregelungen

Abweichend zu obigen Vorgaben und den Empfehlungen in Teil 1 dieser Technischen Richtlinie kann in *bestehenden* Anwendungen als Hashfunktion für die Integritätssicherung mittels HMAC übergangsweise noch SHA-1 eingesetzt werden (d. h. Cipher-Suiten der Form *_SHA). Unabhängig von der in Tabelle 4 angegebenen *maximalen* Verwendung wird eine schnellstmögliche Migration zu SHA-256 bzw. SHA-384 und TLS 1.2 empfohlen.

Hinweis: Da TLS 1.1 die Hashfunktion SHA-1 als Komponente für die Signaturerstellung verwendet (und keine Unterstützung der SHA-2-Familie bietet), wird diese Protokoll-Version nach der Ankündigung von SHA-1 nicht mehr empfohlen.

Der Verschlüsselungsalgorithmus RC4 in TLS weist erhebliche Sicherheitsschwächen auf. Seine Verwendung wird daher nicht empfohlen.

Abweichung	*Verwendung maximal bis*	*Empfehlung*
SHA-1 zur HMAC-Berechnung und als Komponente der PRF[3]	2018+	Migration zu SHA-256/-384
SHA-1 als Komponente für die Signaturerstellung[3]	2015	Migration zu SHA-256/-384
TLS 1.0 **bei Bestandssystemen** zusammen mit Schutzmaßnahmen geeigneten Schutzmaßnahmen gegen Chosen-Plaintext-Angriffe auf die CBC-Implementierung wie oben beschrieben(z.B. BEAST)	2014	Migration zu TLS 1.2 mit AES
RC4 als Verschlüsselungsfunktion	2013	

Tabelle 4: Übergangsregelungen

3.4 Weitere Hinweise und Empfehlungen zu TLS

3.4.1 Session Renegotiation

Es wird empfohlen, *Session Renegotiation* nur auf Basis von [RFC5746] zu verwenden. Durch den Client initiierte Renegotiation sollte vom Server abgelehnt werden.

3.4.2 Verkürzung der HMAC-Ausgabe

Die in [RFC6066] definierte Extension „truncated_hmac“ zur Verkürzung der Ausgabe des HMAC auf 80 Bit sollte *nicht* verwendet werden.

3.4.3 TLS-Kompression und der CRIME-Angriff

TLS bietet die Möglichkeit, die übertragenen Daten vor der Verschlüsselung zu komprimieren. Dies führt zu der Möglichkeit eines Seitenkanalangriffes auf die Verschlüsselung über die Länge der verschlüsselten Daten (siehe [CRIME]).

Um dies zu verhindern, muss sichergestellt werden, dass alle Daten eines Datenpakets von dem korrekten und legitimen Verbindungspartner stammen und keine Plaintext-Injection durch einen Angreifer möglich ist. Kann dies nicht sichergestellt werden, so darf wird empfohlen, die TLS-Datenkompression nicht zu verwenden.

3 Aufgrund von Angriffen gegen die Kollisionsresistenz-Eigenschaften von SHA-1 (siehe auch Abschnitt 1.4 und Bemerkung 12 in [TR-02102-1] sowie [SKP15]) sollte darauf geachtet werden, ob die Kollisionsresistenz beim Einsatz von SHA-1 benötigt wird. Bei der Signaturerstellung ist dies der Fall, daher wird SHA-1 nur bis Ende 2015 empfohlen. Bei der HMAC- und PRF-Berechnung wird die Kollisionsresistenz nicht benötigt, daher ist der maximale Einsatzzeitraum größer als bei der Signaturerstellung.

Es handelt sich hierbei um eine Ausnahmeregelung für bestehende Systeme, die der Tatsache geschuldet ist, dass erst ab TLS 1.2 mit Hilfe der `signature_algorithms`-Erweiterung die Möglichkeit besteht, dass Client und Server das Signaturverfahren und die Hashfunktion aushandeln können.

3.4.4 Der Lucky 13-Angriff

Lucky 13 ist ein Seitenkanalangriff (Timing) auf TLS, bei dem der Angreifer sehr geringe Zeitdifferenzen bei der Verarbeitung des Paddings auf Seiten des Servers ausnutzt. Für diesen Angriff ist es erforderlich, dass der Angreifer sehr genaue Zeitmessungen im Netzwerk machen kann. Er schickt manipulierte Chiffrate an den Server und misst die Zeit, die Server benötigt, um das Padding zu prüfen bzw. einen Fehler zu melden. Durch Netzwerk-Jitter können hier aber sehr leicht Messfehler bei der Zeitmessung entstehen, so dass ein Angriff grundsätzlich schwierig realisierbar erscheint, da der Angreifer im Netzwerk „sehr nahe“ am Server sein muss, um genau genug messen zu können.

Der Angriff kann abgewehrt werden, wenn

- Authenticated Encryption, wie z. B. AES-GCM (erst ab TLS 1.2 verfügbar), oder
- Encrypt-then-MAC (siehe auch nächster Abschnitt)

verwendet wird.

3.4.5 Die „Encrypt-then-MAC“-Erweiterung

Gemäß TLS-Spezifikation werden die zu übertragenen Daten zunächst mit einem Message Authentication Code (MAC) gesichert und dann mit einem Padding versehen; danach werden die Daten und das Padding verschlüsselt. Diese Reihenfolge („MAC-then-Encrypt“) war in der Vergangenheit häufig der Grund für Angriffe auf die Verschlüsselung, da das Padding nicht durch den MAC geschützt ist. Bei den sogenannten Padding-Oracle-Angriffen werden die verschlüsselten TLS-Pakete durch einen Man-in-the-Middle-Angreifer manipuliert, um die Verifikation des Paddings als Seitenkanal zu missbrauchen. Dies kann bspw. dazu führen, dass der Angreifer ein HTTPS-Sitzungs-Cookie entschlüsseln kann und somit die Sitzung des Opfers übernehmen kann.

In RFC 7366 wird die TLS-Erweiterung „Encrypt-then-MAC“ spezifiziert. Hierbei werden die zu übertragenen Daten zuerst mit einem Padding versehen, dann verschlüsselt und danach mit einem MAC gesichert. Damit sind Manipulationen des Paddings ausgeschlossen, da es auch durch den MAC gesichert ist.

Der Einsatz der TLS-Erweiterung „Encrypt-then-MAC“ gemäß RFC 7366 wird empfohlen, sobald geeignete Implementierungen zur Verfügung stehen.

Hinweis: Ab TLS 1.2 gibt es Cipher-Suiten mit Authenticated Encryption. Dabei werden Verschlüsselung und MAC-Sicherung kombiniert. Die oben beschriebenen Angriffe können durch den Einsatz von Authenticated Encryption ebenfalls abgewehrt werden. Ein Beispiel für die Verschlüsselung mit Authenticated Encryption ist die Kombination aus AES und Galois Counter Mode (AES-GCM).

Die Verwendung von Authenticated Encryption (ab TLS 1.2) ist eine Alternative zur oben genannten „Encrypt-then-MAC“-Erweiterung.

3.4.6 Die Heartbeat-Erweiterung

Die Heartbeat-Erweiterung wird in RFC 6520 spezifiziert; sie ermöglicht es, eine TLS-Verbindung über einen längeren Zeitraum aufrecht zu halten, ohne eine Renegotiation der Verbindung durchführen zu müssen. Durch den sogenannten Heartbleed-Bug ist es einem Angreifer möglich, bestimmte Speicherbereiche des Servers auszulesen, die möglicherweise geheimes Schlüsselmaterial enthalten. Dies kann zu einer vollständigen Kompromittierung des Servers führen, wenn der private Schlüssel des Servers bekannt wird.

Empfehlung: Es wird dringend empfohlen, die Heartbeat-Erweiterung nicht zu verwenden. Sollte es trotzdem erforderlich sein, so sollte sichergestellt sein, dass die verwendete TLS-Implementierung über Schutzmaßnahmen gegen den Heartbleed-Bug verfügt.

3.4.7 Die Extended Master Secret Extension

Um Angriffe wie z.B. den Triple Handshake-Angriff (siehe [BDF14]) abzuwehren, ist es sehr sinnvoll, weitere Verbindungsparameter in den TLS-Handshake einfließen zu lassen, damit unterschiedliche TLS-Verbindungen auch unterschiedliche Master Secrets (aus welchem die symmetrischen Schlüssel abgeleitet werden) benutzen.

In [RFC7627] wird die TLS-Erweiterung *Extended Master Secret* vorgeschlagen, die bei der Berechnung des „Extended“ Master Secrets einen Hashwert über alle Nachrichten des TLS-Handshakes mit einfließen lässt.

Der Einsatz der TLS-Erweiterung *Extended Master Secret* gemäß [RFC7627] wird empfohlen, sobald geeignete Implementierungen zur Verfügung stehen.

3.5 Authentisierung der Kommunikationspartner

Das TLS-Protokoll bietet die folgenden drei Möglichkeiten zur Authentisierung der Kommunikationspartner:

- Authentisierung beider Kommunikationspartner
- Nur serverseitige Authentisierung
- Keine Authentisierung

Die Notwendigkeit einer Authentisierung ist anwendungsabhängig. Bei der Verwendung von TLS im Web ist im Allgemeinen zumindest eine Authentisierung des Servers notwendig. Bei der Verwendung in geschlossenen Systemen (VPN o. ä.) ist zumeist eine beidseitige Authentisierung notwendig.

Für die Authentisierung bei der Verwendung in Projekten des Bundes sind die Vorgaben in [TR--03116-4] zu beachten.

3.6 Domainparameter und Schlüssellängen

Die Domainparameter und Schlüssellängen für

- statische Schlüsselpaare der Kommunikationspartner,
- ephemerale Schlüsselpaare bei der Verwendung von Cipher-Suiten mit Forward Secrecy, und
- Schlüsselpaare für die Signatur von Zertifikaten

müssen den Vorgaben in Teil 1 dieser Technischen Richtlinie [TR-02102-1] entsprechen. Es wird empfohlen, mindestens die folgenden Schlüssellängen zu verwenden:

Algorithmus	*Minimale Schlüssellänge*	*Verwendung bis*
Signaturschlüssel für Zertifikate und Schlüsseleinigung		
ECDSA	224 Bit	2015
ECDSA	250 Bit[4]	2022+
DSS	2000 Bit[5]	2022+
RSA	2000 Bit[5]	2022+
Statische Diffie-Hellman Schlüssel		
ECDH	224 Bit	2015
ECDH	250 Bit[4]	2022+
DH	2000 Bit[5]	2022+
Ephemerale Diffie-Hellman Schlüssel		
ECDH	224 Bit	2015
ECDH	250 Bit[4]	2022+
DH	2000 Bit[5]	2022+

Tabelle 5: Empfohlene Schlüssellängen

Hinweis: Ist ein Schlüsselpaar *statisch,* so wird dieses mehrfach für neue Verbindungen wiederverwendet. Im Gegensatz dazu bedeutet *ephemeral,* dass für jede neue Verbindung ein neues Schlüsselpaar erzeugt wird. Ephemerale Schlüssel müssen nach Verbindungsende sicher gelöscht werden, siehe Abschnitt 4.2.

3.6.1 Verwendung von elliptischen Kurven

Beim Einsatz von elliptischen Kurven werden stets kryptographisch starke Kurven über endlichen Körpern der Form F_p (p prim) empfohlen. Zusätzlich wird empfohlen, nur *named curves* (siehe [IANA]) einzusetzen, um Angriffe über nicht verifizierte schwache Domainparameter zu verhindern. Die folgenden *named curves* werden empfohlen:

- brainpoolP256r1, brainpoolP384r1, brainpoolP512r1 (vgl. [RFC5639] und [RFC7027])

Sollten diese Kurven nicht verfügbar sein, so können auch die folgenden Kurven eingesetzt werden:

- secp256r1, secp384r1

Hinweis: Gemäß den Empfehlungen in Tabelle 5 wird die Kurve secp224r1 für einen Einsatz nach 2015 nicht mehr empfohlen.

4 Hier werden 250 Bit (statt 256 Bit) festgelegt, um kleine Co-Faktoren bei elliptischen Kurven zu ermöglichen.

5 Für einen Einsatzzeitraum nach 2016 ist es sinnvoll, eine Schlüssellänge von 3000 Bit zu nutzen, um ein gleichartiges Sicherheitsniveau für alle asymmetrischen Verfahren zu erreichen. Es ist vorgesehen, ab dem Jahr 2017 die Eignungsdauer von RSA- und DH-Schlüsseln mit unter 3000 Bit Schlüssellänge nicht weiter zu verlängern. Eine Schlüssellänge von ≥ 3000 Bit wird damit voraussichtlich ab dem Jahr 2023 für kryptographische Implementierungen, die zu der vorliegenden Richtlinie konform sein sollen, verbindlich werden. Jede Schlüssellänge von ≥ 2000 Bits bleibt aber voraussichtlich für Systeme mit einer Lebensdauer bis 2022 konform zu der vorliegenden Richtlinie; es handelt sich dabei um die in der vorliegenden Richtlinie empfohlene Mindest-Schlüssellänge für RSA, DH und DSA. Weitere Informationen finden sich in den Bemerkungen 4 und 5 in Kapitel 3 in [TR-02102-1].

Bemerkung: Die Empfehlungen in dieser Technischen Richtlinie sind geeignet, um das in Abschnitt 3.1.2 genannte Sicherheitsniveau von zurzeit mindestens 100 Bit zu erreichen. Der Vorhersagezeitraum für die vorliegenden Empfehlungen beträgt 7 Jahre.

Geeignete Empfehlungen für deutlich größere Zeiträume, wie sie in anderen öffentlich verfügbaren Dokumenten zu finden sind, sind naturgemäß sehr schwierig, da zukünftige kryptographische Entwicklungen über längere Zeiträume nicht oder zumindest nicht präzise vorausgesagt werden können. In solchen Fällen umfassen diese Empfehlungen Parameter und Schlüssellängen, die über die in der vorliegenden Technischen Richtlinie hinausgehen können.

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 Ephemeralschlüsseln

Wenn eine Cipher-Suite mit Forward Secrecy verwendet wird, sollte sichergestellt werden, dass alle Ephemeralschlüssel nach ihrer Verwendung unwiderruflich gelöscht werden, und keine Kopien dieser Schlüssel erzeugt wurden. Ephemeral- bzw. Sitzungsschlüssel sollten nur für *eine* Verbindung benutzt werden und 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 [AIS 20/31], vgl. auch Kapitel 9 in Teil 1 dieser Technischen Richtlinie.

Literaturverzeichnis

[AIS 20/31] BSI: AIS 20/31 -- A proposal for: Functionality classes for random number generators

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

[TR-03116-4] BSI: TR-03116 Kryptographische Vorgaben für Projekte der Bundesregierung, Teil 4 - Kommunikationsverfahren im eGovernment

[IANA] IANA: http://www.iana.org/assignments/tls-parameters/tls-parameters.xhtml

[RFC2246] IETF: T. Dierks, C. Allen: RFC 2246, The TLS Protocol Version 1.0

[RFC4346] IETF: T. Dierks, E. Rescorla: RFC 4346, The Transport Layer Security (TLS) Protocol Version 1.1

[RFC5246] IETF: T. Dierks, E. Rescorla: RFC 5246, The Transport Layer Security (TLS) Protocol Version 1.2

[RFC5289] IETF: E. Rescorla: RFC 5289, TLS Elliptic Curve Cipher Suites with SHA-256/384 and AES Galois Counter Mode (GCM)

[RFC5489] IETF: M. Badra, I. Hajjeh: RFC 5289, ECDHE_PSK Cipher Suites for Transport Layer Security (TLS)

[RFC5639] IETF: M. Lochter, J. Merkle: RFC 5639, Elliptic Curve Cryptography (ECC) Brainpool Standard Curves and Curve Generation

[RFC5746] IETF: E. Rescorla, M. Ray, S. Dispensa, N. Oskov: RFC 5746, Transport Layer Security (TLS) Renegotiation Indication Extension

[RFC6066] IETF: D. Eastlake 3rd: RFC 6066, Transport Layer Security (TLS) Extensions: Extension Definitions

[RFC6176] IETF: S. Turner, T. Polk: RFC 6176, Prohibiting Secure Sockets Layer (SSL) Version 2.0

[RFC7027] IETF: M. Lochter, J. Merkle: RFC 7027, Elliptic Curve Cryptography (ECC) Brainpool Curves for Transport Layer Security (TLS)

[RFC7627] IETF: K. Bhargavan, A. Delignat-Lavaud, A. Pironti, A. Langley, M. Ray: RFC 7627, Transport Layer Security (TLS) Session Hash and Extended Master Secret Extension, 2015

[CRIME] J. Rizzo, Th. Duong: The CRIME attack, http://www.ekoparty.org/2012/thai-duong.-php

[BDF14] K. Bhargavan, A. Delignat-Lavaud, C. Fournet, A. Pironti, P.-Y. Strub: Triple Handshake and Cookie Cutters: Breaking and Fixing Authentication over TLS, IEEE Symposium on Security and Privacy, 2014

[SKP15] Marc Stevens, Pierre Karpman, Thomas Peyrin: Freestart collision for full SHA-1, 2015, in submission

[SSLv2] Netscape: Hickman, Kipp: "The SSL Protocol"

[SSLv3] Netscape: A. Frier, P. Karlton, P. Kocher: "The SSL 3.0 Protocol"