

Veröffentlichung einer Übersicht gemäß § 17 Abs. 2 Satz 1 SigV über die Algorithmen und dazugehörige Parameter, die zur Erzeugung von Signaturschlüsseln, zum Hashen zu signierender Daten oder zur Erzeugung und Prüfung digitaler Signaturen als geeignet anzusehen sind, sowie den Zeitpunkt, bis zu dem die Eignung jeweils gilt:

Geeignete Kryptoalgorithmen gemäß § 17 (2) SigV

Allgemein

Die Sicherheit einer digitalen Signatur hängt primär von der Stärke der zugrunde liegenden Kryptoalgorithmen ab. Im Folgenden werden Kryptoalgorithmen genannt, die für digitale Signaturen mindestens für die **kommenden sechs Jahre** (d. h. bis Ende 2005) als geeignet anzusehen sind. Die bit-genauen Spezifikationen findet man in den entsprechenden Standards verschiedener Organisationen (ISO/IEC, NIST, IEEE usw.). Ebenso wie patentrechtliche Fragen und Definitionen der mathematischen Begriffe sind diese Spezifikationen nicht Gegenstand des vorliegenden Dokuments. Informationen hierzu findet man in der einschlägigen Literatur (Lehrbücher, Proceedings von Konferenzen etc.) und im Internet.

Hier werden nur die wichtigsten praxisrelevanten Algorithmen betrachtet, deren kryptographische Eigenschaften aufgrund der heute vorliegenden Ergebnisse langjähriger Diskussionen und Analysen am Besten eingeschätzt werden können.

Abweichend von den hier vorgeschlagenen Algorithmen können auch andere Verfahren eingesetzt werden, wenn deren Eignung nach Angaben des Bundesamtes für Sicherheit in der Informationstechnik (BSI) festgestellt wird und diese von der Regulierungsbehörde für Telekommunikation und Post im Bundesanzeiger veröffentlicht worden sind.

Die Liste der hier genannten Algorithmen ist als offen und vorläufig anzusehen. Sie wird gemäß der weiteren Entwicklung der kryptologischen Forschung und den Erfahrungen mit praktischen Realisierungen von Signaturverfahren aktualisiert und gegebenenfalls ergänzt werden.

Auf die Sicherheit einer konkreten Implementierung in Hard- und Software wird hier nicht eingegangen. Diese wird im Rahmen der Untersuchung nach SigG § 14 (4) festgestellt.

Wie in der Fassung von 1999 bereits angekündigt, wurde eine Vergrößerung der Parameter bei allen Verfahren für den Zeitraum ab 2005 notwendig. Bei der Festlegung der Werte für diese Parameter wurden die derzeit in der Wissenschaft als ausreichend angesehenen Vorgaben berücksichtigt, siehe z. B. [13].

1 Kryptographische Anforderungen

Ein Schema zur digitalen Signatur im Sinne des Gesetzes umfasst die folgenden Kryptoalgorithmen:

- Einen Algorithmus zum Hashen von Daten (einer Hashfunktion), der die zu signierenden Daten auf einen Hashwert, d. h. eine Bitfolge fester kurzer Länge reduziert. Signiert werden dann nicht die Daten selbst, sondern stattdessen jeweils ihr Hashwert,
- einen Signaturalgorithmus, der aus einem Signier- und einem Verifizieralgorithmus besteht. Der Signaturalgorithmus hängt ab von einem Schlüsselpaar, bestehend aus einem privaten (d. h. geheimen) Schlüssel zum Signieren (Erzeugen einer Signatur) und dem dazugehörigen öffentlichen Schlüssel zum Verifizieren (Prüfen) der Signatur, und
- ein Verfahren zur Erzeugung von Schlüsselpaaren für die einzelnen Teilnehmer.

Ein solches Schema ist sicher, wenn nur der Inhaber des privaten Schlüssels in der Lage ist, eine Signatur zu erzeugen, die vom Verifizieralgorithmus als gültig erkannt wird. Welche Anforderungen sich hieraus an die oben genannten kryptographischen Algorithmen ergeben, soll im Folgenden kurz beschrieben werden:

1.1 Hashfunktionen

Beim Signieren wird der Hashwert der zu signierenden Daten gewissermaßen wie ein 'digitaler Fingerabdruck' benutzt. Damit hierbei keine Sicherheitslücke entsteht, muss die Hashfunktion H folgendem Kriterium genügen

- H muss *kollisionsresistent* sein; d. h., es ist praktisch unmöglich, Kollisionen zu finden (Zwei verschiedene digitale Dokumente, die auf denselben Hashwert abgebildet werden, bilden eine Kollision).

Aus der Kollisionsresistenz von H folgt:

- H muss eine *Einwegfunktion* sein; d. h., es ist praktisch unmöglich, zu einem gegebenen Bitstring aus dem Wertebereich ein Urbild bzgl. H zu finden.

Die Existenz von Kollisionen ist unvermeidbar. Dies ist aber nur eine theoretische Aussage. Bei der praktischen Anwendung kommt es nur darauf an, dass es, wie oben verlangt, unmöglich ist, Kollisionen (bzw. Urbilder) zu *finden*.

1.2 Signaturalgorithmen

Niemand anderes als der Besitzer des Signierschlüssels darf in der Lage sein, Signaturen zu erzeugen. Insbesondere bedeutet dies, dass es praktisch unmöglich ist, den Signierschlüssel aus dem (öffentlichen) Verifizierschlüssel zu berechnen.

1.3 Schlüsselerzeugung

Die verschiedenen Signaturalgorithmen benötigen Schlüssel, die bestimmte Bedingungen erfüllen. In einigen Fällen kommen weitere Bedingungen hinzu, deren Nichtbeachtung zu Schwächen des jeweiligen Verfahrens führen könnte. Im Rahmen dieser Anforderungen müssen die Schlüssel zufällig erzeugt werden.

2 Vorschläge für geeignete Hashfunktionen

Man kann nach heutigem Kenntnisstand der Analyse von der langfristigen Sicherheit folgender beider 160-Bit Hashfunktionen der MD4-Familie ausgehen:

- RIPEMD-160 ([7], [3]),
- SHA-1 ([2], [3]).

Diese beiden Hashfunktionen sind (mindestens) in den **kommenden sechs Jahren**, d. h. bis Ende 2005, für die Anwendung bei digitalen Signaturen geeignet.

3 Vorschläge für geeignete Signaturalgorithmen

Im Jahr 1977 haben Rivest, Shamir und Adleman als Erste ein Verfahren zum Erzeugen digitaler Signaturen explizit beschrieben. Es handelt sich um das nach seinen Erfindern benannte RSA-Verfahren [9]. In 1984 hat El'gamal [8] einen anderen Signaturalgorithmus vorgeschlagen. Eine Variante dieses El'gamal-Verfahrens ist der 1991 vom National Institute of Standards and Technology (NIST) publizierte Digital Signature Standard (DSS) [1], der den

Digital Signature Algorithm (DSA) spezifiziert. Daneben gibt es Varianten des DSA, die auf Punktgruppen $E(K)$ elliptischer Kurven über endlichen Körpern K basieren, wobei $K = F_p$ ein endlicher Primkörper bzw. $K = F_{2^m}$ ein endlicher Körper der Charakteristik 2 ist.

Folgende Signaturalgorithmen sind geeignet:

1. RSA [9], [4]
2. DSA [1], [4]
3. DSA-Varianten, basierend auf elliptischen Kurven [1], [5], [10], [11]

Die Sicherheit der oben genannten Verfahren hängt jeweils zusammen mit

1. dem Faktorisierungsproblem für ganze Zahlen,
2. dem Diskreten-Logarithmus-Problem in der multiplikativen Gruppe eines Primkörpers F_p ,
3. dem Diskreten-Logarithmus-Problem in den Gruppen $E(F_p)$ bzw. $E(F_{2^m})$.

Um festzulegen, wie groß die Systemparameter bei diesen Verfahren gewählt werden müssen, um deren Sicherheit zu gewährleisten, muss man zum einen die besten heute bekannten Algorithmen zum Faktorisieren ganzer Zahlen bzw. zum Berechnen diskreter Logarithmen (in den oben genannten Gruppen) betrachten und zum anderen die Leistungsfähigkeit der heutigen Rechnertechnik berücksichtigen. Um eine Aussage über die Sicherheit für einen bestimmten zukünftigen Zeitraum zu machen, muss man außerdem eine Prognose für die beiden genannten Aspekte zugrunde legen. Solche Prognosen sind nur für relativ kurze Zeiträume möglich (und können sich natürlich jederzeit aufgrund unvorhersehbarer dramatischer Entwicklungen als falsch erweisen).

Im Folgenden bezeichnen wir mit der Bitlänge r einer Zahl x dasjenige r mit der Eigenschaft $2^{r-1} \leq x < 2^r$.

Die Sicherheit der einzelnen Verfahren ist (mindestens) für die **kommenden sechs Jahre**, d. h. bis Ende 2005, bei folgender Wahl der Parameter gewährleistet:

3.1 RSA

Der zugrunde liegende Modulus $n = pq$ (p und q Primzahlen) soll eine Bitlänge von mindestens 2048 haben. Für den Zeitraum **bis Ende 2000** reicht eine Minimallänge des Modulus von 768 Bit aus. Für den Zeitraum **bis Mitte 2005** reicht eine Minimallänge des Modulus von 1024 Bit aus.

Die folgende Tabelle fasst die minimalen Bitlängen zusammen.

Parameter\ Zeitraum	bis Ende 2000	bis Mitte 2005	bis Ende 2005
n	768	1024	2048

Die Primfaktoren p und q von n sollten ungefähr gleich groß sein, aber nicht zu dicht beieinander liegen, d. h.

$$\varepsilon_1 < |\log_2(p) - \log_2(q)| < \varepsilon_2$$

Als Anhaltspunkte für die Werte ε_1 und ε_2 werden hier $\varepsilon_1 \approx 0,5$ und $\varepsilon_2 \approx 30$ vorgeschlagen. Die Primfaktoren p und q werden unter Beachtung der genannten Nebenbedingungen zufällig und unabhängig voneinander erzeugt.

Der öffentliche Exponent e wird mit $ggT(e, (p-1)(q-1)) = 1$ gewählt. Der zugehörige geheime Exponent d wird dann berechnet, so dass $ed \equiv 1 \pmod{(p-1)(q-1)}$ gilt.

Bemerkungen:

- Die Forderung, dass p und q *starke* Primzahlen sein müssen (d. h. $p-1$ und $q-1$ haben große Primfaktoren etc.), erscheint im Hinblick auf die heute bekannten besten Faktorisierungsalgorithmen nicht mehr ausreichend begründet und daher verzichtbar.
- Der öffentliche Exponent kann zufällig gewählt werden. Auf der anderen Seite haben kleine öffentliche Exponenten den Vorteil, dass die Verifikation der Signatur sehr schnell durchgeführt werden kann.
- Der Hashwert muss vor der Anwendung des geheimen Exponenten auf die Bitlänge des Moduls formatiert werden. Diese Formatierungsverfahren sind dabei sorgfältig zu wählen, siehe [14]. Konkrete Verfahren werden hier nicht weiter behandelt. Die Realisierung eines Formatierungsverfahrens - z.B. die Form der Arbeitsteilung zwischen einer Chipkarte, auf der die Exponentiation mit dem geheimen Schlüssel durchgeführt wird, und dem Hintergrundsystem - muss aber im Rahmen der Prüfung technischer Komponenten nach SigG § 14 (4) untersucht werden.

3.2 DSA

In FIPS-186 wird für den Parameter p (p Primzahl) eine Bitlänge von mindestens 512 und höchstens 1024 verlangt. Gleichzeitig wird dort die Bitlänge des Parameters q auf 160 festgelegt.

Abweichend von FIPS-186 wird hier verlangt, dass für den Zeitraum **bis Ende 2005** die Bitlänge des Parameters p mindestens 2048 betragen soll. Für den Zeitraum **bis Mitte 2005** reicht eine Minimallänge des Parameters von 1024 Bit aus. Die Bitlänge des Parameters q soll weiterhin mindestens 160 betragen.

Die folgende Tabelle fasst die minimalen Bitlängen zusammen.

Parameter\ Zeitraum	bis Mitte 2005	bis Ende 2005
p	1024	2048
q	160	160

Bemerkungen:

- Zur Erzeugung von p und der weiteren Parameter siehe [1].
 - In FIPS-186 wird die Bitlänge des Parameters q auf genau 160 festgelegt. Dies erlaubt die Konstruktion von 'Kollisionen' im Sinne von [12] bei der Parametergenerierung. Diese Kollisionen haben jedoch in der Praxis keine Bedeutung. Wenn man dessen ungeachtet die Möglichkeit, diese Kollisionen konstruieren zu können, ausschließen möchte, muss man Bitlängen > 160 wählen.
- a) DSA-Varianten basierend auf Gruppen $E(F_p)$

Um die Systemparameter festzulegen, werden eine elliptische Kurve E und ein Punkt P auf $E(F_p)$ erzeugt, so dass folgende Bedingungen gelten:

- $\#E(F_p) = a \cdot q$ mit einer von p verschiedenen Primzahl q .
- $\text{ord}(P) = q$.
- $r_0 := \min(r : q \text{ teilt } p^r - 1)$ ist groß, konkret $r_0 > 10^4$.
- Die Klassenzahl der Hauptordnung, die zum Endomorphismenring von E gehört, ist mindestens 200.

Für den Zeitraum bis Ende 2005 soll die Bitlänge von p mindestens 192 betragen. Dabei sollte $p \approx q$. Auf jeden Fall ist sicherzustellen, dass die Bitlänge von q mindestens 160 Bit beträgt. Für den Zeitraum **bis Mitte 2005** reicht - ohne weitere Bedingung an p - eine Minimallänge des Parameters q von 160 Bit aus.

Bemerkung:

Die untere Abschätzung für r_0 hat den Sinn, Attacken auszuschließen, die auf einer Einbettung der von P erzeugten Untergruppe in die multiplikative Gruppe eines Körpers F_{p^r} beruhen. In der Regel (bei zufälliger Wahl der elliptischen Kurve) ist diese Abschätzung erfüllt, denn r_0 ist die Ordnung von $p \pmod{q}$ in F_q^* und hat deshalb im Allgemeinen sogar dieselbe Größenordnung wie q . Im Idealfall sollte r_0 explizit bestimmt werden, was allerdings die etwas aufwendige Faktorisierung von $q - 1$ voraussetzt. Demgegenüber ist $r_0 > 10^4$ wesentlich schneller zu verifizieren und wird in diesem Zusammenhang als ausreichend angesehen.

b) DSA-Varianten basierend auf Gruppen $E(F_{2^m})$

Um die Systemparameter festzulegen, werden eine elliptische Kurve E und ein Punkt P auf $E(F_{2^m})$ erzeugt, so dass folgende Bedingungen gelten:

- m ist prim.
- $E(F_{2^m})$ ist nicht über F_2 definierbar (d.h. die j -Invariante der Kurve liegt nicht in F_2)
- $\#E(F_{2^m}) = a \cdot q$ mit q prim.
- $\text{ord}(P) = q$.
- $r_0 := \min(r : q \text{ teilt } 2^{mr} - 1)$ ist groß, konkret etwa $r_0 > 10^4$.
- Die Klassenzahl der Hauptordnung, die zum Endomorphismenring von E gehört, ist mindestens 200.

Für den Zeitraum bis Ende 2005 soll m mindestens 191 betragen. Dabei sollte $2^m \approx q$. Auf jeden Fall ist sicherzustellen, dass die Bitlänge von q mindestens 160 Bit beträgt. Für den Zeitraum **bis Mitte 2005** reicht - ohne weitere Bedingung an m - eine Minimallänge des Parameters q von 160 Bit aus.

Bemerkungen:

- Die Bedingung, dass m prim ist, war in der Fassung dieses Papiers von 1999 nicht gefordert. Hintergrund für diese neue Bedingung sind die Überlegungen in [6].

- In Bezug auf die oben erwähnten 'Kollisionen' im Sinne von Vaudenay [12] gilt für die auf elliptische Kurven basierenden Verfahren dasselbe wie für DSA.
- Beim DSA und bei elliptischen Kurven könnte die Wahl bestimmter, ganz spezieller Parameter möglicherweise dazu führen, dass das Verfahren schwächer ist als bei einer zufälligen Wahl der Parameter. Unabhängig davon, wie gravierend man diese Bedrohung einschätzt, kann man dem „Unterschieben„ schwacher Parameter vorbeugend dadurch begegnen, dass bei der Konstruktion der Parameter eine praktische Einwegfunktion, d.h. eine der oben genannten Hashfunktionen angewandt wird und die Parameter zusammen mit einer nachvollziehbaren entsprechenden Berechnung übergeben werden. Konkrete Vorschläge findet man in [1] und [10].

4 Erzeugung von Zufallszahlen

Bei der Erzeugung von Systemparametern für Signaturalgorithmen und die Schlüsselgenerierung werden Zufallszahlen benötigt. Bei DSA-ähnlichen Signaturalgorithmen benötigt man bei jeder Generierung einer Signatur eine neue Zufallszahl.

Für diese Zwecke bieten sich als Zufallszahlengeneratoren Systeme an, die

- eine physikalische Rauschquelle, die beispielsweise auf elektromagnetischen, elektro-mechanischen oder quantenmechanischen Effekten beruht, und
- eine kryptographische (bzw. mathematische) Nachbehandlung des Primärausgangs

besitzen. Die Entnahme der Bits aus der physikalischen Rauschquelle sollte sich hinreichend gut durch ein stochastisches Modell beschreiben lassen. Das Primärausgang sollte, soweit dies technisch möglich ist, permanent einem angepassten statistischen Test unterzogen werden. Um die Entropie pro entnommenen Bit zu erhöhen, ist in der Regel eine mathematische Nachbehandlung des Primärausgangs notwendig. Diese Nachbehandlung sollte die Abhängigkeiten, die sich aus der physikalischen Rauschquelle bzw. aus seinem stochastischen Modell ergeben, berücksichtigen und auflösen.

Zur Schlüsselerzeugung sollte stets ein physikalischer Zufallszahlengenerator verwendet werden.

Eine aussagekräftige Bewertung eines Zufallszahlengenerators setzt umfassende Erfahrungen voraus. Das Bundesamt für Sicherheit in der Informationstechnik (BSI) verfügt über solche Erfahrungen, und es wird empfohlen, in diesem Zusammenhang auf das Know-how des BSI zurückzugreifen.

Wenn für andere Anwendungen (z. B. beim Signieren mit einer DSA-Variante) kein physikalischer Zufallszahlengenerator zur Verfügung steht, kommt als Alternative ein Pseudozufallszahlengenerator in Frage. Der innere Zustand des Pseudozufallszahlengenerators wird durch den sogenannten seed initialisiert. In jedem Schritt wird der innere Zustand erneuert und hieraus eine Zufallszahl abgeleitet. Der innere Zustand des Pseudozufallszahlengenerators muss gegen Auslesen (physikalisch, über Schnittstelle etc.) ebenso sicher geschützt sein wie die geheimen Signaturschlüssel. Denn mit Kenntnis des inneren Zustands könnte ein potentieller Angreifer zumindest alle zukünftig erzeugten Zufallszahlen mühelos bestimmen.

Jeder Pseudozufallszahlengenerator, der im Zusammenhang mit digitalen Signaturen genutzt wird, sollte ein K4-DRNG (Mechanismenstärke hoch) im Sinne von AIS 20, [7], sein. Qualitativ bedeutet dies:

- Es ist einem Angreifer nicht praktisch möglich, aus Kenntnis eines inneren Zustands Vorgängerzufallszahlen oder innere Vorgängerzustände zu errechnen, oder diese mit einer Wahrscheinlichkeit zu erraten, die nicht vernachlässigbar über der Ratewahrscheinlichkeit ohne Kenntnis des inneren Zustands liegt.

Wenigstens muss aber folgende Bedingung erfüllt sein: Jeder Pseudozufallszahlengenerator, der im Zusammenhang mit digitalen Signaturen genutzt wird, muss mindestens ein K3-DRNG (Mechanismenstärke hoch) im Sinne von AIS 20, [7], sein. Qualitativ bedeutet dies:

- Es ist einem Angreifer nicht praktisch möglich, zu einer ihm bekannten Zufallszahlenteilfolge Vorgänger oder Nachfolger dieser Teilfolge oder gar einen inneren Zustand zu errechnen, oder diese mit einer Wahrscheinlichkeit zu erraten, die nichtvernachlässigbar über der Ratewahrscheinlichkeit ohne Kenntnis der Teilfolge liegt.

Andernfalls muss das entsprechende Verfahren zur digitalen Signatur als potentiell unsicher angesehen werden.

Literatur

- [1] NIST: *FIPS Publication 186-2: Digital Signature Standard (DSS)*, Januar 2000.
- [2] NIST: *FIPS Publication 180-1: Secure Hash Standard (SHS-1)*, Mai 1995.
- [3] ISO/IEC 10118-3: *Information technology – Security techniques - Hash functions - Part 3: Dedicated hash functions*, 1998.
- [4] ISO/IEC 14888-3: *Information technology – Security techniques - Digital signatures with appendix – Part 3: Certificate-based mechanisms*, 1999.
- [5] IEEE P1363: *Standard specification for public key cryptography*.
- [6] S.D. Galbraith, N.P. Smart: *A Cryptographic Application of Weil Descent*, Cryptography and Coding, LNCS 1746, 1999.
- [7] *AIS 20: Funktionalitätsklassen und Evaluationsmethodologie für deterministische Zufallszahlengeneratoren*, Version 1, 2.12.99, www.bsi.bund.de
- [8] T. El'gamal: *A public key cryptosystem and a signature scheme based on discrete logarithms*, Crypto '84, LNCS, Band 196, S. 10 – 18, Springer-Verlag, 1985.
- [9] R. Rivest, A. Shamir, L. Adleman: *A method for obtaining digital signatures and public key cryptosystems*, Communications of the ACM, vol. 21 no. 2, 1978.
- [10] ANSI X9.62-1998: *Public Key Cryptography for the Financial Service Industry: The Elliptic Curve Digital Signature Algorithm (ECDSA)*
- [11] ISO/IEC 15946-2 (FCD): *Information technology – Security techniques – Cryptographic techniques based on elliptic curves – Part 2: Digital signatures*, 2000.
- [12] Serge Vaudenay: *Hidden collisions in DSS*, Crypto'96, LNCS, Band 1109, S. 83 – 88, Springer Verlag, 1996.
- [13] A.K. Lenstra, E.R. Verheul: *Selecting Cryptographic Key Sizes*, www.cryptosavvy.com
- [14] J.-S. Coron, D. Naccache and J. Stern: *On the Security of RSA padding*. In *Advances in Cryptology- Crypto '99*