CCC Datenschleuder, März 2005

Hash Probleme

Rüdiger Weis

In den letzten Monaten wurde das lange Zeit vernachlässigten Gebiet der kryptographischen Hash-Funktionen verstärkt unter Feuer genommen. Hierbei zerbröselte faktisch die die gesamte MD4-Hash Familie. Diese bildet unter anderem die Grundlage für praktisch alle in Anwendung befindichen Digitaler Signaturverfahren. Besonders hart traf es die noch vielfach genutzten Hash-Funktionen MD4 und MD5, vor denen Kryptographen erst seit über 10 Jahren warnen. MD4 kann inzwischen als Übungsaufgabe auch mit Bleistift und Papier gebrochen werden. Auch SHA-0 die erste Version des Secure Hash Algorithm Standard ist inzwischen mit einem grösseren Fakultätsrechnerpool an einem Wochende brechbar. Die von der NSA ohne Veröffentlichung der Gründe modifizierte Version „SHA-1" ist höchstwahrscheinlich ebenso einer Gruppe um eine chinesische Mathematikerin zum Opfer gefallen. Dadurch ist eine neue Sicherheitsbewertung faktisch aller digitaler Signaturen notwendig geworden.

Kryptographische Hashfunktionen

Hash-Funktionen erzeugen für Eingaben (praktisch) beliebiger Länge einen kurzen, typischerweise zwischen 128 und 512 bit langen "fingerprint". Für viele kryptographische Anwendungen ist vor allem die Eigenschaft der *Kollisionsresistenz* von besonderer Bedeutung.

Kollisionsresitenz

Als *Kollision* bezeichnet man zwei verschiedene Nachrichten M und M', die auf den selben Hashwert

$$h(M)=h(M')$$

abgebildet werden.

Da es (mehr oder weniger) unendlich viele Eingaben aber nur endlich viele Hash-Werte gibt, ist die Existenz von Kollisionen nicht zu verhindern. Eine Hash-Funktion bezeichnet man dennoch als *kollisionsresistent*, wenn der Rechenaufwand, eine Kollision zu finden, so riesig ist, dass man diesen als *praktisch unmöglich* ansehen kann.

Aktuell gelten bei vielen Anwendern 2^{80} Rechenoperationen als untere Schranke dieser praktischen Unmöglichkeit. Schon wegen eines trivialen „Geburtstagsangriffes" muss eine Hash-Funktion für dieses Sicherheitsniveau also mindestens einen 160-bit Hashwert liefern. Schon aus diesem Grunde sollte von Hash-Funktionen wie MD4, MD5 und RIPEMD-128, welche lediglich 128 bit liefern abgeraten werden.

Eine Hash-Funktion, die nicht kollisionsresistent ist, sollte als *gebrochen* angesehen werden.

Integritätsprüfungen

Kryptographische Hash-Funktionen werden in vielen Bereichen der IT-Sicherheit eingesetzt, zum Beispiel bei der Integritätsprüfungen (z.B. tripwire). Dabei wird von als „harmlos" und „nützlich" eingestuften Programmen ein Hash-Wert abgespeichert. Vor Ausführung des Programms wird dessen Hashwert mit dem Referenz-Hashwert verglichen. Stimmen die beiden nicht überein, wurde das Programm manipuliert. Stimmen Soll- und Ist-Hash-Wert dagegen überein, dann solle es sich, um das Originalprogramm handeln - allerdings nur, wenn die Hash-Funktion kollisionsresistent ist.

Digitale Signaturen

Besondere Bedeutung haben Hash-Funktionen auch für *digitale Signaturen*: In der Praxis unterschreibt man in der Regel nicht die Nachricht, sondern ihren Hash-Wert („Hash-Then-Sign Paradigma").

Gelingt es jedoch, zwei kollidierende Nachrichten M und M' zu finden, gilt also h(M)=h(M'), dann ist eine gültige digitale Signatur für M auch eine gültig für M'.

Schwächen von Hash-Funktionen bezüglich der Kollisionsresistenz können die Rechtssicherheit von Verträgen in Frage stellen.

MD4 und MD5

Ron Rivest veröffentliche 1990 das Design für die 128bit Hash-Funktion MD4 [Ri90]. MD4 ist eine iterierende 3-ründige Hash-Funktion welche nach dem Damgard-Merkle Konstruktionsprinzip entwickelt worden ist. MD4 ist recht einfach implementierbar und besonders für die schnelle Verarbeitung auf 32-bit Prozessoren optimiert.
Da schon früh kryptographische Schwächen von MD4-Varianten aufgezeigt werden konnten, machte sich Ron Rivest bereits kurze Zeit später an die Entwicklung einer verstärkten Version.

1991 veröffentlichte Ron Rivest die Hash-Funktion MD5 [Ri91]. Neben einiger Modifikationen in der Kompressionsfunktion wurde eine zusätzliche 4. Runde spezifiziert. Hierdurch ist MD5 etwas langsamer als MD4. Wie MD4 liefert MD5 auch ledigliche einen 128 bit langen Message-Digest.

Bei einer 128 bit Hash-Funktion benötigt der bereits erwähnte triviale Geburtstagsangriff nur einen Rechenaufwand von 2^{64} Aufrufen der Hash-Funktion, selbst wenn die Hash-Funktion keine spezifischen Schwächen aufweist.

MD4 mit Papier und Bleistift angreifbar

Schon 1996 zeigte der damalige BSI Mitarbeiter Hans Dobbertin, wie man auf effiziente Weise eine Kollisionen für MD4 finden kann [Do96a]. Der Algorithmus von Dobbertin braucht nur kurze Zeit, um mit der Wahrscheinlichkeit 2^{-32} eine Kollision zu finden. Scheitert der Algorithmus, wird er wiederholt. Dobbertin benutzte für den Angriff einen einfachen PC.

Acht Jahre später wurde nicht einmal mehr ein PC gebraucht. Auf er Rump-Session der Crypto 2004 präsentierten die chinesischen Forscher Wang, Lai, Feng, Chen und Yu Angriffe auf MD4, die praktisch per Hand durchgeführt werden können.
Die neue Attacke findet eine Kollision mit einer Wahrscheinlichkeit von 2^{-6}-2^{-8} und einem überraschend niedrigem Aufwand von 2^{8} MD4 Berechnungen.

Die Autoren geben auch eine Anwendung ihrer Techniken auf RIPEMD an, welche mit einer Wahrscheinlichkeit von 2^{-17} eine Kollision mit einer Angriffskomplexität von 2^{19} findet.

MD5 gebrochen

Am 17. August 2004 zeigten Xiaoyun Wang, Dengguo Feng, Xuejia Lai und Hongbo Yu Kollisionen für MD5 auf. Der praktische Angriff benötigte nach ihren Angaben nur zwischen 15 Minuten eine Stunde auf einem Workstation Cluster.

Interessanterweise handelt sich es bei der vorgestellten Methode um einen spezielle differentiellen Angriff, welcher Differentiale bezüglich modulare Subtraktion verwendet. Diese Angriffsmethode findet nach Angaben der Autoren für MD4 eine Kollision Aufwand von 2^{23} MD4 Berechnungen, 2^{13} für HAVAL-128 [ZPS92] , 2^{33} für RIPEMD und 2^{62} für SHA-0.

Verwendunsstopp für MD4 und MD5

Angesichts der aktuellen Entwicklungen muss die **dringende Empfehlung** erteilt werden **MD5 und MD4 nicht mehr zu verwenden**.

Im Bereich von **digitalen Signaturen** sollte auf die Verwendung von MD4 und MD5 – **sofort** verzichtet werden. **Darüber hinaus sei eine Analyse möglicher Auswirkungen bezüglich bereits geleisteter Signaturen angeraten.**

Das US-amerikanische National Institute of Standards and Technology (NIST) William Burr, Manager security technology group, mahnte ebenfalls in einem Interview vom Februar 2005 insbesondere für die sensiblen Bereiche von Zertifikaten und Digitalen Signaturen einen sofortigen Verwendungsstop von MD5 an.

„*If by some chance you are still using MD5 in certificates or for digital signatures, you should stop*" [Ol05]

SHA-0 und SHA-1

Der SHA Algorithmus (inzwischen als SHA-0 bezeichnet) basiert auf einen von der „National Security Agency" (NSA) stammenden Design, dessen genaue Designprinzipien und Kriterien nicht veröffentlicht wurden.

Schon kurz nach seiner Veröffentlichung wurde dieser Standard mit dem Hinweis auf (unveröffentlichte) Sicherheitslücken modifiziert. Wörtlich heisst es im Standard [NI93]:

"*SHA-1 is a technical revision of SHA (FIPS 180). A circular left shift operation has been added to the specifications in section 7, line b, page 9 of FIPS 180 and its equivalent in section 8, line c, page 10 of FIPS 180. This revision improves the security provided by this standard.* " [NI93]

SHA-0 und SHA-1 lehnt sich nahe an den MD4 Hashalgorithmus an:

„The SHA-1 is based on principles similar to those used by Professor Ronald L. Rivest of MIT when designing the MD4 message digest algorithm ("The MD4 Message Digest Algorithm," Advances in Cryptology - CRYPTO '90 Proceedings, Springer-Verlag, 1991, pp. 303-311), and is closely modelled after that algorithm. " [NI93]

SHA-0 gebrochen

Nach dem Rückzug von SHA(-0) dauerte es nicht lange, bis erste Schwächen von dieser Hash-Funktion auch von den öffenlich forschenden und frei publizierenden Forschern aufgedeckt wurden. Im Jahr 2004 präsentierten dann mehreren Gruppen unabhängig voneinander gravierende Angriffe auf SHA-0.

SHA-1 wahrscheinlich ebenfalls gebrochen

Trotz diese wirklich klaren Signale zweifelten viele Industrievertreter, ob sich diese Angriffsmetoden auch auf SHA-1 übertragen lassen.

Mein Wettangebot an die Trusted Computing Group

„Ich persoenlich wuerde eher auf die Jungfraeulichkeit von Brittney Spears wetten, als auf die Sicherheit von SHA-1."

vom Januar 2005 wurde allerdings nicht angenommen.

Schließlich unterscheidet SHA-1 sich von SHA-0 ja in einer zusätzlichen extrem einfachen Operation, einer zyklischen Linksrotation eines 32-bit Wortes.
So liess der erfolgreiche Angriff nicht lange auf sich warten. Mitte Februar 2005 verbreiteteten die chinesischen Forscher Xiaoyun Wang, Yiqun Lisa Yin, and Hongbo Yu eine 3-seite Notiz, dass sie Kollisionen für SHA-1 mit einem Zeitaufwand von 2^{69} Aufrufen der Hash-Funktionberechnen können [Sc05].

In ihrer Ankünfigung veröffentlichen die Autoren weiterhin eine Kollission für SHA-0 und geben den Rechenaufwand hierfür mit 2^{39} an. Eine Kollission für SHA-1 mit einer auf 58 reduzierten Rundenzahl wurde mit einer Komplexität von 2^{33} gefunden.
Zum Redaktionsschluss dieses Artikels war die wissenschaftliche Diskussion über diese Arbeit allerdings noch nicht vollständig abgeschlossen.

Merkle-Damgard Design fragil

Faktisch alle in der Praxis eingesetzten Hash-Funktionen, wie MD4, MD5, SHA-0, SHA-1, SHA-XXX, RIPEMD, RIPEMD-160 und weitere orientieren sich in ihrem Design an den theoretischen Grundlagenarbeiten von Merkle [Me89] und Damgard [Da89].

Das *Merkle-Damgard Prinzip* besteht darin, eine Hash-Funktionfür beliebig lange Eingaben dadurch zu realisieren, dass man eine Mini-Hash-Funktion (die sogenannte „Kompressionsfunktion") für Eingaben fester Länge iteriert. Merkle und Damgard wiesen nach, dass die iterierte Hash-Funktion kollisionsresistent ist, wenn die Kompressionsfunktion ihrerseits kollisionsresistent ist.

Auf der Crypto 04 Konferenz präsentierte Joux [Jo04] jedoch einen *generischen Angriff*, der eine Schwäche des Merkle-Damgard Designprinzips aufdeckte.
Vereinfacht ausgedrückt, zeigte Joux, dass, *wenn* man Kollisionen für die Hash-Funktion (bzw. die Kompressionsfunktion) finden kann, es dann sogar leicht ist, *K-fache Kollisionen* für H zu berechnen. Der Aufwand hierfür ist nur logarithmisch.

Das Merkle-Damgard Design für Hash-Funktionen hat sich somit als ebenfalls als fragil erwiesen.

Stefan Lucks zeigte in [Lu04] Variationen des Merkle-Damgard Designs, die gegen den Joux-Angriff immun sind, und für die man sogar formal beweisen kann, dass alle generische Angriffe zum Finden K-facher Kollisionen scheitern .

Interessanter Weise verwenden einige Varianten der SHA-2 Familie ähnliche Techniken. Die macht die Geheimhaltung der genauen Designkriterien noch ärgerlicher.

SHA-1 und digitale Signaturen

SHA-1 ist noch für qualifizierte Signaturen nach dem Signaturgesetz zugelassen. Alternativen, wie beispielsweise die Hash-Funktionen des SHA-2 Standards, werden von signaturfähigen Smartcards oft noch nicht unterstützt . Wie sicher sind also heute Signaturen auf Basis von SHA-1?

Zunächst die „guten" Nachichten:

- Der Angriff der chinesischen Forscher impliziert unmittelbare keine Gefahr für Unterschriften, die in der Vergangenheit geleistet wurden. Es sei denn, Angreifer kannten den Angriff schon länger und habe ihn in der Vergangenheit bereits benutzt.
- Die Aufgabe, zu einer bereits unterschriebenen Nachricht M eine Nachricht M' zu finden, zu der die Unterschrift wegen einer Kollision H(M)=H(M') passt, ist schwieriger als die Aufgabe, eine beliebige Kollision zu finden.
- Der Rechenaufwand für den Angriff von Wang und Co-Autoren ist mit 2^{69} Aufrufen der Hash-Funktion recht gross, und nur von gut finanzierten und motivierten Organisationen zu leisten.

Nun zu den schlechte Nachrichten:

- Der Angriff von Wang und Co-Autoren lässt es bedrohlich erscheinen, Nachrichten zu unterschreiben, die von einer anderen Partei vorgelegt wurden.
- Die Erfahrung aus der Vergangenheit ist, dass Angriffe immer besser werden. Es wäre also wenig überraschend, wenn die Analyse von SHA-1 in den nächsten Monaten und Jahren noch verbessert und der Rechenaufwand für das Problem, Kollisionen für SHA-1 zu finden, noch weiter verringert werden würde.
- Die Hauptbedingung für den Angriff von Wang und Co auf SHA-0 besteht darin, dass die Nachrichten M und M' eine bestimmte Differenz aufweisen. Das lässt dem Angreifer große Freiheiten, die Wahl von M und M' einzuschränken, statt sie dem Zufall zu überlassen.

  Aus ähnlichen Gründen ist es auch Dobbertin bei MD4 gelungen, gezielte Kollisionen zu finden, konkret zwei Fassungen eines Kaufvertrages mit sehr unterschiedlichen Kaufpreisen [Do96a].

"At the price of $176,495 Alf Blowfish sells his house to Ann Bonida."
"At the price of $276,495 Alf Blowfish sells his house to Ann Bonida."

Zusammenfassend sei gesagt, dass im Moment bei der Verwendung von SHA-1 für digitale Signaturen **kein Grund zur sofortigen Panik** besteht.
Unabhängig jedoch davon, ob sich die Angaben der chinesischen Forscher sich bestätigen, **ist es an der Zeit, die Hash-Funktion SHA-1 durch Alternativen zu ersetzen**.

Die Angriffe auf sehr verwandte Hash-Funktionen, einschließlich SHA-0, sind zu eindrucksvoll, als das man SHA-1 noch lange vertrauen sollte.

Wir verweisen nochmals ausdrücklich auf die schon seit Jahren geäusserten Warnungen SHA-1 nicht innerhalb der geplanten Trusted Computing Initiativen zu verwenden. [WeB02 ,WB03,WB04,We04,WL05,...]

Gibt es Alternativen?

Keine praktischen Angriffe wurden bisher für die gestärkte RIPEMD Variante **RIPEMD-160** [DBP96] gefunden. Dennoch lassen die Angriffe auf die Vorversion RIPEMD und die Mitgliedschaft in der MD4 Familie zur Vorsicht raten. Eine 160-bit Hash-Funktion sollte ohnehin nur als temporäre Lösung für wenige Jahre betrachtet werden.

Als Alternative bieten sich beispielsweise **algebraische Kombinationen von Funktionen mit unterschiedlichem Basisdesigns** an (s .a. [We00]). Interessante Ansätze stellen unserer Meinung hierfür beispielsweise Tiger [AB96] und Whirlpool [BR00] aus dem europäischen NESSIE Projekt dar.

Der neue **NIST Standard SHA-2** definiert eine ganze Reihe von Funktionen mit längerer Hash-Wertlänge. Sie liefern zwischen 224 bit (SHA-224) und 512 bit (SHA-512) lange Hashwerte. Allerdings existieren für diese Funktionenfamilie erst sehr wenige Analysen [GH03].

Zudem gab es auch, im Gegensatz zum sehr erfolgreichen Wettbewerb um die Blockchiffre AES im Falle der Hash-Funktionen leider keinen Wettbewerb – vielmehr hat das NIST, wie bei SHA-0 und SHA-1, auf ein von der NSA stammendes Design zurückgegriffen, dessen Designprinzipien und -kriterien nicht veröffentlicht wurden.

Henri Gilbert und Helena Handschuh zeigten, dass die Funktionen gegen ein Reihe von gegen SHA-1 und MD4 mögliche Angriffstechniken sehr widerstandsfähig sind. Beunruhigend ist allerdings, dass modifizierte Versionen gravierend schwach sind – auch bei nur geringen Vereinfachung:

"*However we show that slightly simplified versions of the hash functions are surprisingly weak: whenever symmetric constants and initialization values are used throughout the computations, and modular additions are replaced by exclusive or operations, symmetric messages hash to symmetric digests.*" [GH03]

Für Kryptographen sind die Hash-Algorithmen der SHA-2 Familie damit unangenehm fragil – sehr kleine Änderungen der Hash-Funktion sollten eigentlich nur moderate Auswirkungen auf die Sicherheit der Hash-Funktion haben.

Dennoch kann die Verwendung der Funktionen des SHA-2 Standards im Vergleich zu SHA-1 die bessere Alternative sein. Diese Einschätzung sehen wir durch die momentane Abwesenheit von veröffentlichen Angriffen, die längeren Hashwerte und die etwas aufwendigeren Berechntngen motiviert.

Einige moderne Sicherheitslösungen wie GPG, heute bieten heute bereits SHA-256 als Option [GP05]. Cryptophone verwendet SHA-256 zur Erhöhnung der Robustheit unterschiedlich parametrisiert hintereinander [Cr03].

Mir persönlich gefällt allerdings auch für die SHA-2 Familie das gesamte Design der Kompressionsfunktion nicht. Es ist mir recht schleierhaft, wieso nicht besser verstandene Konstuktionsmethoden - orientiert etwa am besser verstandenen Design von Blockchiffrierern -verstärkt eingesetzt werden.

Zudem scheint es eine europäische Unart zu sein, Professoren-Stellen für Computersicherheit verstärkt mit Personen zu besetzten, welche nicht mal über kryptographische Grundkenntnisse verfügen. Es bleibt zu Hoffen, dass hier langsam ein Umdenken einsetzt, bevor auch die letzten praktisch eingesetzen Verfahren zertrümmert sind.

Es sei daher eine verstärkte Förderung der kryptographischen Forschung und die Ausschreibung eines Wettbewerbe um eine Standard-Hash-Funktion, analog zum AES-Wettbewerb um eine Standard-Blockchiffre, dringen angeraten.

„Doppelt genäht"

Eine unelegante aber kryptographisch robuste Methode, welche ohne grossen Aufwand anwendbar ist, ist die Verwendung mehrerer Hash-Funktionen.
Beispielsweise kann man einen Hash H durch das Aneinanderhängen des 128-bit,MD5-Hashes, des 160-bitten SHA-1 Hashes es bilden:

$$H(x):=(MD5(x)||SHA\text{-}1(x)).$$

Eine Kollision für H ist eine Kollision für *jede* der Hash-Funktionen, aus denen H zusammengesetzt wurde. Eine derartige „Kaskade" von Hash-Funktionen hat deshalb trivialer die Eigenschaft, dass der zusammengesetzte Hash mindestens so stark ist, wie der stärkste einzelne Hash.

Da bei RSA Signaturen ohnehin Schlüssellängen von 2048 bit oder mehr verwenden werden, ist genügend Raum für derartige Konstruktionen ohne dass eine zweite modulare Exponentiation notwendig wäre. (Dies träfe sogar für 1024 bit RSA Schlüssel zu, von deren Verwendung allerdings abgeraten werden sollte [WLB04].)
Eine derartige Randomisierung des Signaturinputs erhöht sogar die Robustheit des Signaturverfahrens im Vergleich zu immer noch sich in Verwendung befindlichen trivialen Signaturpaddingverfahren mit nur einem Hashwert.
Die Geschwindigkeitseinbussen erscheinen angesichts der deutlich aufwendigeren eigentlichen Signaturfunktion als vernachlässigbar.

Man kann sogar darauf hoffen, dass die Kaskade mehrerer Hash-Funktionen stärker ist als jede einzelne Hash-Funktion– die Kollisionen für die eine Hash-Funktion *sollte* doch keine Kollision für die andere Hash-Funktion darstellen.

Leider könnte diese Hoffnung auch trügerisch sein. Joux [Jo04] demonstrierte dies mit einer Anwendung seines Multi-Kollisions-Angriffes auf Kaskaden. Ein weiteres Problem ist, dass praktisch alle sich in Anwendung befindlichen Hashfunktionen auf den problematischen Bausteinen MD4 und Merkle/Damgard basieren.

Daher möchten wir die Verwendung von unterschiedlich konstruierten Funktionen innerhalb der Kaskade, beispielsweise

$$H'(x):=(SHA\text{-}256(x)||Whirlpool(x)||Tiger(x))$$

anraten [vgl .We00].

Literatur

[AB96] Anderson, R., Biham, E., "Tiger: A Fast New Hash Function", Fast Software Encryption - FSE'96, LNCS 1039, Springer-Verlag (1996), pp. 89—97.
[BR00] Barreto, P., Rijmen, V., "The Whirlpool Hashing Function", First open NESSIE Workshop, Leuven, Belgium, 13--14 November 2000.
[Cr03] Cryptophone FAQ http://www.gsmk.de/html/faq_en.html
[DBP96] Dobbertin, H., Bosselaers, A., Preneel, B., ``RIPEMD-160, a strengthened version of RIPEMD,'' Fast Software Encryption 1996, LNCS 1039, D. Gollmann, Ed., Springer-Verlag, 1996, pp. 71-82.
[Da89] Damgard, I. A design principle for hash functions. Crypto 89, LNCS 435, pp. 416—427.
[Do96a] Dobbertin, H., "Cryptoanalysis of MD4", Fast Software Encryption, 1996.
[Do96b] Dobbertin, H., "The Status of MD5 After a Recent Attack", CryptoBytes 2(2), 1996.
[D98] Dobbertin, H., "Cryptanalysis of MD4", Journal of Cryptology 11:4 (1998), pp. 253--271.
[GP05] GNU Privacy Guard, www.gnupg.org
[Ol05] Olsen, F., „NIST moves to stronger hashing", Federal Computer Week, Feb. 7, 2005
[Lu04] Lucks, S., Design Principles for Iterated Hash Functions, Cryptology ePrint Archive: Report 2004/253.
[NE03] NESSIE, „Portfolio of recoendatet cryptographic primitives", https://www.cosic.esat.kuleuven.ac.be/nessie/deliverables/decision-final.pdf
[NI93] National Institute of Standards and Technology (NIST), FIPS180-1, SECURE HASH STANDARD, http://www.itl.nist.gov/fipspubs/fip180-1.htm
[GH03] Henri Gilbert, H., Handschuh, H., "Security analysis of SHA-256 and sisters" in M. Matsui and R. Zuccherato, Eds., Selected Areas in Cryptography (SAC 2003), vol. 3006 of Lecture Notes in Computer Science, pp. 175-193, Springer-Verlag, 2004.
[Jo04] Joux, A., Multicollisions in iterated hash functions, application to cascaded constructions. Crypto 04, LNCS 3152, pp. 306--316.
[Me89] Merkle, R., One-way hash functions and DES. Crypto 89, LNCS 435, pp. 428--446.
[Sc05] Schneier, B., SHA-1 Broken, http://www.schneier.com/blog/archives/2005/02/sha1_broken.html
[Wea05] Wang, X., Lai, X., Feng, D., Chen, H., Yu, X., 'Cryptoanalysis for Hash Functions MD4 and RIPEMD', preprint, 2005.
[We00] Weis, R., "Protocols and Algotithms", PhD Thesis, 2000.
[WB02] Weis, R., Bogk, A., Trusted Comuting, Chaos Communication Congress, Berlin, 2002. http://www.cryptolabs.org/tcpa/WeisTCPAfutilePrinterfriendly.pdf
[WB03] Weis, R., Bogk, A., Trusted Computing, Chaos Communication Congress, Berlin, 2003.
[WB04] Weis, R., Bogk, A., Trusted Computing, Chaos Communication Congress, Berlin, 2004.
[We04] Weis, R., „Trusted Computing – Chancen und Risiken", DuD 11/04.
[WL05] Weis, R., Lucks, S.,"Hash-Funktionen gebrochen", DuD 03/05, 2005.
[WLB04] Weis, R., Lucks, S., Bogk, A., "Sicherheit von 1024-bit RSA-Schlüsseln gefährded", DuD, 2004.
[ZPS92] Y. Zheng, J. Pieprzyk, and J. Seberry, "HAVAL - a one-way hashing algorithm with variable length of output", Advances in Cryptology - Auscrypt'92, LNCS 718, Springer-Verlag (1993), pp. 83--104.