

Fail-Safe-Konzept für Public-Key-Infrastrukturen

Michael Hartmann

Technische Universität Darmstadt, hartmann@cdc.informatik.tu-darmstadt.de

Sönke Maseberg

Technische Universität Darmstadt, maseberg@cdc.informatik.tu-darmstadt.de
Fraunhofer - Institut für Sichere Telekooperation (SIT), maseberg@sit.fraunhofer.de

Zusammenfassung

Public-Key-Infrastrukturen sind von zentraler Bedeutung für eine sichere elektronische Kommunikation in offenen Netzen. Unternehmen, Behörden und Privatleute nutzen diese Technik in zunehmendem Maße und verlassen sich auf sie. Public-Key-Infrastrukturen bergen aber auch Risiken, weil die der Public-Key-Kryptographie zugrunde liegenden mathematischen Probleme nicht beweisbar sicher sind, und weil Implementierungsfehler nie ausgeschlossen werden können. Public-Key-Infrastrukturen beschränken sich heutzutage meist auf einige wenige kryptographische Verfahren, so dass das Auftreten eines Fehlers den Ausfall der Funktionsfähigkeit der Infrastruktur bewirken könnte. Dies würde beträchtliche wirtschaftliche und gesellschaftliche Folgen nach sich ziehen. Zur Vermeidung dieser Gefahren ist ein Ansatz, mehrere Kryptoverfahren zu nutzen und diese derart flexibel in eine Public-Key-Infrastruktur zu integrieren, dass ihr Austausch im laufenden Betrieb ermöglicht wird. In diesem Artikel wird eine Public-Key-Infrastruktur vorgestellt, die die Nutzung mehrerer Verfahren unterstützt, und ein Protokoll für den Austausch von Komponenten diskutiert.

1 Einleitung

Public-Key-Infrastrukturen (PKI) stellen eine Basis für sichere elektronische Kommunikation in verteilten Systemen dar. Einige Sicherheitsziele sind, die Authentizität, Integrität, Zurechenbarkeit, Vertraulichkeit und Verfügbarkeit übertragener Daten zu gewährleisten und eine Authentisierung von Benutzern oder Rechnern und eine Autorisierung von Aktionen zu ermöglichen. Zwei wesentliche kryptographische Techniken, um diese Ziele zu erreichen, sind digitale Signaturen und Verschlüsselungen. Digitale Signaturen sind unter gewissen Voraussetzungen ein Pendant zur handschriftlichen Unterschrift, was für e-commerce zwischen verschiedenen Unternehmen (Business-To-Business) oder zwischen Unternehmen und Privatleuten (Business-To-Customer) und für e-government zwischen Behörden und Bürgern wichtig wäre. Es gibt nationale und internationale Bemühungen um eine rechtliche Gleichstellung von handschriftlicher Unterschrift und digitaler Signatur. Digitale Signaturen sind auch geeignet, berechtigten Personen oder Rechnern einen realen oder virtuellen Zugangsschutz zu gestatten. Eine weitere Anwendung ist die Verschlüsselung vertraulicher Daten, zu der eine PKI das Schlüssel-Management zur Verfügung stellen kann.

Public-Key-Infrastrukturen basieren auf Public-Key-Kryptographie, die ihrerseits auf bislang ungelöste mathematische Probleme zurückgeht. Beispiele sind das Faktorisierungsproblem oder das Diskrete-Logarithmus-Problem, die die Basis für RSA resp. DSA darstellen. Es ist nicht bekannt, ob diese Probleme wirklich schwierig sind. Es gibt keine Beweise für die Sicherheit. Es ist nur bekannt, dass bei geeigneter Schlüssel- und Systemparameterwahl unter Berücksichtigung der heute verfügbaren Rechenleistung keine effizienten Algorithmen zum Faktorisieren großer Zahlen oder Berechnen diskreter Logarithmen bekannt sind.

Empfehlungen über kryptographisch geeignete Algorithmen, Schlüssel und Systemparameter zur Erzeugung von Signaturschlüsseln, zum Hashen zu signierender Daten oder zur Erzeugung und Prüfung digitaler Signaturen werden u.a. vom Bundesamt für Sicherheit in der Informationstechnik (BSI) zusammengestellt. Als geeignet anzusehen sind bis Mitte 2005 zum Beispiel die Hashfunktionen SHA-1 und RIPEMD-160, sowie die Signatur-Algorithmen RSA mit 1024 Bit, DSA mit 1024 Bit und ECDSA mit 160 Bit Schlüssellänge mit weiteren Parametern, [BSI]. Lenstra und Verheul haben eine Übersicht über voraussichtlich notwendige Schlüssellängen und Systemparameter bis zum Jahr 2050 veröffentlicht, [LeVe00].

Das Problem ist, dass sich diese Empfehlungen und Voraussagen nur auf das Wissen der Vergangenheit und Gegenwart beziehen, und dass Fortschritte bei der Entwicklung effizienter Algorithmen oder Implementierungsfehler in der Zukunft nicht auszuschließen sind. Einige Beispiele aus der Geschichte belegen dies: John Pollards Zahlkörpersieb verbesserte 1988 die Faktorisierungsalgorithmen deutlich, [LeLe93]. Dobbertin zeigte, dass die Hashfunktion MD4 nicht kollisions-resistent ist, [Dobb96]. Bleichenbacher präsentierte einen Angriff auf RSA mit PKCS#1-Padding, [Ble98]. Es gibt Implementierungen mit zu kurzen Schlüsseln. Chipkarten sind einer Reihe von logischen Angriffen ausgeliefert, die Reaktionen auf gezielte Fehler, den Stromverbrauch oder Berechnungszeiten analysieren, [Koch95], [KoJJ99].

Deshalb ist eine unerwartete Kompromittierung einer Klasse von Schlüsseln nicht auszuschließen. Es gibt zwei Arten von Gründen:

- Kompromittieren einer kryptographischen Komponente, wie Signatur-Algorithmus, Verschlüsselungs-Algorithmus, Hashfunktion oder Formatierung. Dieser Schaden betrifft alle Schlüssel zu diesem Verfahren.
- Kompromittieren von einsatzspezifischen Komponenten, wie Systemparametern oder einer Klasse von Schlüsseln zu einem Algorithmus. Die Klasse ist eine Teilmenge aller Schlüssel und kann insbesondere ein einzelner Schlüssel sein.

Von einer Kompromittierung könnten digitale Signaturen und Kryptogramme betroffen sein. Digitale Signaturen könnten ihre Authentizität, Integrität und Zurechenbarkeit und damit Beweiskraft verlieren, weil durch den plötzlichen Schaden keine Möglichkeit mehr bestehen würde, Daten anderweitig zu sichern oder kryptographische Verfahren und Parameter anzupassen. Insbesondere wären Zertifikate betroffen. Zertifikate stellen in einer PKI die Verbindung von öffentlichem Schlüssel und Teilnehmer her. Darüber hinaus könnte die Vertraulichkeit verschlüsselter Daten nicht mehr garantiert sein.

Es gibt Maßnahmen, wie bei Kompromittierung eines Schlüssels vorzugehen ist. Um die Sicherheit des Systems gewährleisten zu können und Benutzer vor einem Akzeptieren kompromittierter Schlüssel zu warnen, werden Zertifikate kompromittierter Schlüssel revoziert (d.h. widerrufen) und in Revokationslisten veröffentlicht. Benutzer können diese - auch Sperllisten genannten - Listen konsultieren oder den aktuellen Status eines Zertifikats abfragen, um zu entscheiden, ob eine Signatur gültig ist oder nicht. Als Gründe für eine Revokation nennt [Road] neben Änderungen in den Zertifikatsangaben: Schlüssel verloren, Schlüssel kompromittiert oder Schlüssel erscheint kompromittiert.

Die Revokation von Zertifikaten kann den Ausfall der Funktionsfähigkeit der PKI zur Folge haben: Das in einer PKI notwendige Vertrauen wird über den öffentlichen Schlüssel der Zertifizierungsinstanz (Certification Authority - CA), welcher der Zertifikatsinhaber (Certificate Holder - CH) vertraut, und durch Zertifikate und Zertifikatsketten gebildet. Werden Zertifikate

in dieser Kette revoziert, kann eine digitale Signatur nicht mehr als gültig validiert werden. Je nach Umfang der von einem Schaden betroffenen Zertifikate schränkt dies die Funktionsfähigkeit der PKI ein. Die Wiederherstellung der gesamten PKI kann dann die Entwicklung, Produktion, Verteilung und Installation neuer Komponenten nach sich ziehen, was Zeit und Geld kosten würde. Sind z.B. Chipkarten involviert, müssten diese erst bestellt und u.U. produziert werden, was gerade dann besonders lange dauern dürfte, wenn das Problem mehrere Infrastrukturen trifft. Dieses Problem ist bislang nicht gelöst.

Eine weitere Frage betrifft die Aussagekraft von digitalen Signaturen, die vor einem Schaden erzeugt wurden: Um die Rechtsverbindlichkeit einer digitalen Signatur bei Kompromittierung eines Schlüssels zu bewahren, können digitale Signaturen mehrfach angewendet werden, z.B. durch einen Zeitstempeldienst, [FJPP95], [Wohl00], [Time]. Notwendig ist allerdings, dass diese ergänzenden Signaturen mit unterschiedlichen Verfahren erzeugt wurden, was derzeit nicht immer garantiert werden kann.

Darüber hinaus kann eine Kompromittierung besonderer Schlüssel oder kryptographischer Verfahren die Revokationsmechanismen wirkungslos machen: Revokationslisten oder Statusantworten auf Revokationsanfragen sind signiert und verlieren ihre Aussagekraft, wenn gerade diese Schlüssel kompromittiert wurden. Wird ein Zertifizierungsschlüssel kompromittiert und sind deshalb neue Zertifikate mit fiktiven Seriennummern im Umlauf, die der Zertifizierungsinstanz nicht bekannt sind, so können diese nicht in Sperrlisten aufgeführt werden.

Die Analyse der existierenden Mechanismen zur Revokation eines Zertifikats zeigt, dass eine Revokation des in der hierarchisch angeordneten PKI obersten Schlüssels nicht vorgesehen ist: Es gibt keine Methoden, um dann eine Revokation zu authentisieren. In PKIX ist für solche Fälle eine out-of-band Kommunikation vorgesehen, also eine Kommunikation via Post, TV, Zeitungen, etc. Außerdem würde die Revokation eines in der PKI-Hierarchie weit oben angesiedelten Zertifikats die Revokation aller hierarchisch unterhalb liegenden Zertifikate implizieren, was die derzeitigen Revokationsmechanismen bei großen Public-Key-Infrastrukturen überfordern könnte.

Ein Kompromittieren von Schlüsseln könnte kurzfristige Verschlüsselungen enthüllen: Da ein Abhören oder Mitschnitt von Daten über offene Netze nicht verhindert werden kann, ist damit zu rechnen, dass verschlüsselte Daten von Angreifern aufbewahrt und nach hinreichend langer Zeit zu entschlüsseln sein werden. Da dieses Vorgehen nicht verhindert werden kann, richtet sich das Interesse auf relativ kurzfristige Verschlüsselungen. Gemeint sind Geheimnisse, die nach einiger Zeit sowieso öffentlich bekannt oder uninteressant werden. Diese kurzfristigen Verschlüsselungen könnten offenbart werden.

Zusammenfassend bleibt das offene Problem des Ausfalls der Funktionsfähigkeit der PKI und die bislang unbefriedigend gelösten Probleme des Verlustes der Beweiskraft digitaler Signaturen, der unzureichenden Revokationsmechanismen und des Verlustes der Vertraulichkeit kurzfristiger Verschlüsselungen.

Als Lösung wird in diesem Artikel über ein Fail-Safe-Konzept für Public-Key-Infrastrukturen berichtet, welches eine optimale Konfiguration und Funktionsweise einer PKI beschreibt, die im Schadensfall die weitere Funktionsfähigkeit und den sicheren Austausch kompromittierter Komponenten ermöglicht. Mit den vorgestellten Mitteln lassen sich digitale Signaturen, Revokationsinformationen und Verschlüsselungen mit zusätzlichen Komponenten derart erweitern, dass sie ihre Beweiskraft resp. kurzfristige Vertraulichkeit nicht verlieren. Fail-Safe definieren

wir als einen Prozess, der bei einem Schadensfall (Fail) in Aktion tritt, um die Funktionsfähigkeit zu erhalten und um den Schaden zu reparieren, d.h. das System sicher (Safe) weiterzuführen.

Der Artikel gliedert sich wie folgt. Abschnitt 2 beschreibt die Idee. Die zur Realisierung notwendigen technischen Eigenschaften werden in Abschnitt 3, die Beschreibung der PKI in Abschnitt 4 und ihre Funktionsweise in Abschnitt 5 beschrieben. In Abschnitt 6 werden Interoperabilitätsfragen diskutiert.

2 Idee

Die Grundidee besteht darin, in einer Public-Key-Infrastruktur mehrere, voneinander unabhängige kryptographische und einsatzspezifische Komponenten (also Verfahren und Schlüssel) flexibel einzusetzen und zu nutzen. Dadurch können im Falle der Kompromittierung einer Komponente andere Teile der PKI weiterhin sicher funktionieren, um die ununterbrochene Funktionsfähigkeit zu bieten. Darüber hinaus sollen kompromittierte Komponenten flexibel durch neue, sichere Komponenten ausgetauscht werden, um die PKI langfristig einsatzfähig zu halten und um sie dynamisch sich ändernden Sicherheitsanforderungen anpassen zu können. Die multipel vorhandenen Verfahren und Schlüssel können genutzt werden, um elektronische Dokumente mehrfach zu signieren (multiple digitale Signatur) und zu verschlüsseln (iterative Verschlüsselung). Deren sinnvolle Nutzung hängt von der spezifischen Anwendung ab. Multiple digitale Signaturen sollten in erster Linie dazu eingesetzt werden, um die Beweiskraft von signierten Dokumenten im Schadensfall sicherzustellen, wenn durch Revokationsmechanismen Zertifikate zurückgezogen und Signaturen aussagelos wurden. Durch Einsatz multipler Komponenten können auch Sperrinformationen trotz Schadensfall sicher übertragen werden. Zur effizienten Sperrung von mehreren Zertifikaten könnte das Gültigkeitsmodell vorgeschrieben werden, welches zur Validierung stets den gesamten Zertifizierungspfad berücksichtigt, oder es müssten Möglichkeiten geschaffen werden, in Revokationslisten ganze Bereiche von Zertifikats-Seriennummern zu sperren.

Dieser Idee liegt die Annahme zugrunde, dass die Wahrscheinlichkeit sehr gering ist, dass zwei auf verschiedenen mathematischen Problemen beruhende kryptographische Verfahren durch einen Schaden gleichzeitig kompromittiert werden. Grund für diese Annahme ist, dass Ideen zu effizienten Algorithmen auf den speziellen grundlegenden mathematischen Problemen basieren. Wenn sich diese Basisprobleme unterscheiden, können Ideen effizienter Algorithmen nicht zwangsläufig auf verschiedene Verfahren angewendet werden, [BuMa00].

3 Flexible Public-Key-Infrastrukturen

Als Grundvoraussetzung müssen technische Komponenten so flexibel in die PKI integriert sein, dass ein Austausch überhaupt möglich ist. Dieser Philosophie folgen die Arbeiten an der 'flexiblen Public-Key-Infrastruktur (FlexiPKI)' am Lehrstuhl von Prof. Buchmann, [BuRT00], [FlexiPKI]. In der FlexiPKI können kryptographische Komponenten flexibel genutzt werden. Zum Beispiel unterstützt die FlexiPKI neben den üblichen Verfahren wie RSA, DSA, ElGamal, SHA-1 oder MD5 auch Kryptographie mit elliptischen Kurven. Daneben wird an weiteren Kryptoverfahren geforscht, die auf anderen mathematischen Problemen basieren, wie z.B. der Zahlkörperkryptographie.

In den folgenden Abschnitten wird eine PKI beschrieben und ihre Funktionsweise erläutert, um die in Abschnitt 1 genannten Probleme optimal zu lösen. Diese PKI heisst 'Fail-Safe-PKI'.

4 Beschreibung der Fail-Safe-PKI

Die Beschreibung dieser Fail-Safe-PKI besteht aus unverändert übernommenen Standards, aus von Standards für diese Zwecke konfigurierten Teilen und aus neuen Elementen, die es noch nicht gibt.

Der weitverbreitete PKI-Standard Internet 'X.509 Public Key Infrastructure (PKIX)' von der 'Internet Engineering Task Force (IETF)' definiert eine Public-Key-Infrastruktur als eine Menge aus Beteiligten, Hardware, Software, Verfahren und Richtlinien (Policies), um auf Public-Key-Kryptographie basierende Zertifikate zu erstellen, handzuhaben, zu verwahren, zu verteilen und zu revozieren, [IETF], [PKIX], [Road].

Der Gestaltung der Fail-Safe-PKI wird folgendes Modell zugrunde gelegt, das sich beliebig verallgemeinern lässt:

- Die PKI ist als zweistufige Hierarchie aufgebaut, d.h. die PKI besteht nur aus Zertifizierungsinstanz und von ihr zertifizierten Zertifikatsinhabern.
- Die multiplen, voneinander unabhängigen kryptographischen Komponenten beschränken sich auf zwei Verfahren.
- Schlüssel sind anwendungsneutral, d.h. in den Zertifikaten werden ihnen keine Funktionen zugeordnet.

Eine PKI besteht aus fünf Typen von Beteiligten:

- Zertifizierungsinstanz (Certification Authority - CA), die Zertifikate herausgibt und revoziert. Als Zertifikate werden X.509- und Attributs-Zertifikate nach [ISO96] genutzt.
- Registrierungsinstanz (Registration Authority - RA), die für die Verbindung zwischen öffentlichem Schlüssel, Identitäten und Attributen der CH bürgt.
- Verzeichnis (Directory - DIR), das Zertifikate und Sperrlisten (Certificate Revocation List - CRL) speichert und verfügbar macht. In diesem Modell wird das Verzeichnis von der CA betrieben. In einem Trust-Center sind diese drei Instanzen vereinigt.
- Zertifikatsinhaber (Certificate Holder - CH), dem von der CA Zertifikate ausgestellt werden und der Dokumente digital signieren und verschlüsseln kann.
- Client, der digitale Signaturen und ihre Zertifikats-Pfade validiert, ausgehend von einem bekannten öffentlichen Schlüssel einer vertrauenden CA (dem Sicherheitsanker).

Da die Fail-Safe-PKI zwei voneinander unabhängige Signatur- und Verschlüsselungsverfahren samt entsprechender Infrastruktur mit Schlüsseln und Zertifikaten enthält, die parallel genutzt werden können, besteht die PKI quasi aus zwei voneinander unabhängigen Public-Key-Infrastrukturen, die hier mit PKI^A und PKI^B bezeichnet werden.

PKI^A besteht aus Signatur-Verfahren $sign^A$, Verschlüsselungs-Verfahren $encrypt^A$, privatem Schlüssel $prK^A_{\text{Beteiligter}}$ und öffentlichem Schlüssel $puK^A_{\text{Beteiligter}}$ mit zugehörigem Zertifikat $cert^A_{\text{Beteiligter}}$. Aus einem Dokument und privatem Schlüssel wird mit dem Signatur-Verfahren $sign^A$ die Signatur $sign^A(\text{Dokument}, prK^A_{\text{Beteiligter}})$ erzeugt. Ein vertrauliches Dokument kann mit dem Verschlüsselungs-Verfahren $encrypt^A$ und dem öffentlichen Schlüssel des Empfängers $puK^A_{\text{Empfänger}}$ zum Kryptogramm $encrypt^A(\text{Dokument}, puK^A_{\text{Empfänger}})$ verschlüsselt werden. Jedes Zertifikat $cert^A_{\text{Beteiligter}}$ beinhaltet Namen, öffentlichen Schlüssel und weitere Daten des Beteiligten und ist von der CA zertifiziert. Die CA nutzt dazu das Signaturverfahren zu PKI^A : $cert^A_{\text{Beteiligter}} = (C = \text{"Beteiligter, } puK^A_{\text{Beteiligter}}, \text{ weitere Daten"}, sign^A(C, prK^A_{CA}))$. Jeder CH verfügt als Sicherheitsanker über das Zertifikat $cert^A_{CA}$ mit dem öffentlichen Schlüssel puK^A_{CA} der CA, dem

er vertraut. Entsprechendes gilt für PKI^B. Die CA stellt darüber hinaus ein Verzeichnis der ausgegebenen Zertifikate und Sperrlisten CRL^A und CRL^B zur Verfügung. Die beiden Sperrlisten unterscheiden sich lediglich durch ihre Signaturen: CRL^A ist durch prK^A_{CA} und CRL^B durch prK^B_{CA} signiert. Ihre Inhalte sind gleich und beinhalten sowohl PKI^A- als auch PKI^B-Zertifikate.

Abbildung 1 zeigt, über welche Komponenten Zertifizierungsinstanz und Zertifikatsinhaber verfügen.

Zertifizierungsinstanz Certification Authority (CA)	Zertifikatsinhaber Certificate Holder (CH)
<u>PKI^A-Komponenten:</u> Signatur-Verfahren sign ^A Verschlüsselungs-Verfahren encrypt ^A CA-Schlüsselpaar prK ^A _{CA} , puK ^A _{CA} CA-Zertifikat cert ^A _{CA}	<u>PKI^A-Komponenten:</u> Signatur-Verfahren sign ^A Verschlüsselungs-Verfahren encrypt ^A CH-Schlüsselpaar prK ^A _{CH} , puK ^A _{CH} CH-Zertifikat cert ^A _{CH} Sicherheitsanker cert ^A _{CA} mit puK ^A _{CA}
<u>PKI^B-Komponenten:</u> Signatur-Verfahren sign ^B Verschlüsselungs-Verfahren encrypt ^B CA-Schlüsselpaar prK ^B _{CA} , puK ^B _{CA} CA-Zertifikat cert ^B _{CA}	<u>PKI^B-Komponenten:</u> Signatur-Verfahren sign ^B Verschlüsselungs-Verfahren encrypt ^B CH-Schlüsselpaar prK ^B _{CH} , puK ^B _{CH} CH-Zertifikat cert ^B _{CH} Sicherheitsanker cert ^B _{CA} mit puK ^B _{CA}
<u>Verzeichnis:</u> Zertifikate cert ^A _{CA} , cert ^B _{CA} , cert ^A _{CH} , cert ^B _{CH} Sperrlisten CRL ^A , CRL ^B	

Abb. 1: Komponentenverteilung für CA und CH

Die Registrierung in der RA, in der sich ein Teilnehmer ausweist, und die Initialisierung und Personalisierung in der CA entsprechen in der Fail-Safe-PKI dem PKIX-Standard. Die Zertifizierung samt Schlüssel-Management wird mehrfach ausgeführt. Cross-Zertifizierungen zwischen verschiedenen CAs können über PKI^A- und PKI^B-Zertifikate organisiert werden. Revokation von Zertifikaten wird nach [CertCRL] unterstützt. Je nach Anwendung nutzt ein Zertifikatsinhaber verschiedene Hard- und Software:

1. *CH nutzt Client* - Entweder verfügt der CH über einen auf ihn personalisierten Client, der die privaten und öffentlichen Schlüssel und Zertifikate in einer Software-PSE (Personal Security Environment - Persönliche Sicherheitsumgebung) bereithält, oder der CH nutzt einen für ihn anonymen Client. Beispiele: In einer Signatur-Anwendung (mit einer Signatur im Sinne einer handschriftlichen Unterschrift) ist der Client ein Mail-Client, wie etwa Netscape Messenger oder Microsoft Outlook. In einer Authentisierungs-Anwendung (mit einer Signatur im Rahmen einer Zugangskontrolle) ist der CH der Client in einer Client/Server-Authentisierung oder der Client am Tor.
2. *CH nutzt ICC* - Der CH hat eine Chipkarte - auch Smart Card oder Integrated Circuit Card (ICC) genannt. Zur Benutzung dieser Karte benutzt der CH einen auf ihn personalisierten oder anonymen Client. Beispiele: Die Chipkarte eines CH als Signatur- oder Zugangskontroll-Karte.

Protokolle zum Zugriff auf das DIR-Verzeichnis oder zum Abfragen des aktuellen Status eines Zertifikats entsprechen in der Fail-Safe-PKI den Standards 'Lightweight Directory Access Protocol (LDAP)' und 'Online Certificate Status Protocol (OCSP)', [LDAP], [OCSP]. Dabei sind Zugriffe auf CRL^A und CRL^B und auf mit PKI^A und PKI^B signierte OCSP-Antworten möglich.

Anwendungsprotokolle, wie 'Secure/Multipurpose Internet Mail Extensions (S/MIME)' für den sicheren Daten-Transfer via e-Mail oder das 'Transport Layer Security (TLS)' / 'Secure Socket Layer (SSL)' Protokoll für Authentisierungen, werden von der Fail-Safe-PKI unterstützt. Die in PKIX diskutierte 'Cryptographic Message Syntax (CMS)' findet sich - leicht abgewandelt - in dem von RSA Security Inc. propagierten 'Standard Public Key Cryptographic Standard #7 (PKCS#7)' wieder, der sich zunehmend durchsetzt und deshalb in der Fail-Safe-PKI genutzt wird, [SMIME], [TLS], [CMS], [PKCS7]. PKCS#7 bietet bereits die Möglichkeit multipler digitaler Signaturen und iterativer Verschlüsselungen.

Folgende Elemente einer PKI müssen neu geschaffen werden, weil sie noch nicht vorhanden sind:

- Innerhalb eines Trust-Centers wird die Funktion des Update Service geschaffen, der für den Austausch von kompromittierten Komponenten zuständig ist und die Kommunikation zwischen Trust-Center und CH herstellt.
- Management-Strukturen müssen vorhanden sein. Während bei Signatur-Anwendungen der Update Service betroffene CHs direkt ansprechen kann, ist bei einer Authentisierungs-Anwendung diese Möglichkeit nicht immer vorhanden. In einem solchen Fall muss die Nutzung von Revokationsmechanismen vorgeschrieben werden, so dass diese Mechanismen dann als Management-Kanal genutzt werden können. CRL und OCSP-Antworten bieten die Möglichkeit, Erweiterungen hinzuzufügen, die zu diesem Zweck ausgenutzt werden. Darüber hinaus müssen Erweiterungen in die Authentisierungsprotokolle integriert werden.
- Um den Austausch durchführen zu können, wird ein Update Management Protocol (UMP) definiert. Dieser Datentyp wird über einen Object Identifier (OID), der unterhalb der OID der Arbeitsgruppe von Prof. Buchmann angeordnet ist, weltweit eindeutig zuzuordnen sein. UMP enthält Informationen über den Schaden und einen Link zum Download des Codes. Der Vorteil liegt darin, dass das Trust-Center so die Verteilung der Datenmenge zeitlich strecken und die Kapazitäten von Speichern und Netzen optimal nutzen kann.
- Für den Fall, dass Chipkarten als Sicherheits-Token involviert sind, wird ein entsprechendes UPDATE COMPONENT Kommando kreiert.
- In Client und Chipkarte muss der Austausch von Komponenten kontrolliert werden. Die dazu notwendigen Security Conditions werden in einer Registry abgelegt.
- Zertifikats-Richtlinien (Certificate Policies - CPs) (siehe [CP]) werden dahin gehend erweitert, dass für eine konkrete, das Fail-Safe-Konzept nutzende Anwendung u.a. folgendes geregelt werden kann:
 - Nutzung multipler Signaturen ist nicht oder optional möglich, oder ist obligatorisch.
 - Nutzung iterativer Verschlüsselung ist nicht oder optional möglich, oder ist obligatorisch.
 - Regelungen zum Validieren von multiplen Signaturen, d.h. Umfang der Pfadvalidierung mit Prüfung multipler CRL- oder OCSP-Antworten während der Signatur-Erzeugung oder -Verifikation.
- Sperrlisten werden um zwei Interpretationen ergänzt: Zur Verifikation wird das Gültigkeitsmodell vorgeschrieben, das die Validierung des gesamten Zertifizierungspfads beinhaltet. Alternativ werden Sperrlisten um die Möglichkeit der Sperrung von Bereichen von

Zertifikats-Seriennummern erweitert. Da Seriennummern nur innerhalb einer CA eindeutig sind, könnten diese Nummern so verteilt werden, dass am Anfang stets eine Kennung des verwendeten Algorithmus steht, wodurch etwa eine Sperrung von Zertifikatsnummer 123.*n* implizit alle Zertifikate revozieren würde, die den Algorithmen 123 nutzen.

5 Funktionsweise der Fail-Safe-PKI

In der Fail-Safe-PKI können Zertifikatsinhaber den Funktionsumfang einer 'gewöhnlichen' PKI nutzen, d.h. e-Mails signieren und verifizieren, Dokumente ver- und entschlüsseln, Zertifikate und Sperrlisten laden, den Status eines Zertifikats abfragen oder Authentisierungen durchführen, oder mittels erweitertem Funktionsumfang digitale Signaturen multipel signieren oder Dokumente mehrfach iterativ verschlüsseln. Darüber hinaus bietet die Fail-Safe-PKI die Möglichkeit, Komponenten im laufenden Betrieb auszutauschen.

5.1 Erweiterter Funktionsumfang

Alice und Bob möchten ihre e-Mails doppelt signieren und doppelt iterativ verschlüsseln.

Alice will Bob eine signierte Nachricht zusenden und nutzt dazu ihren Mail-Client.

1. Alice erzeugt einen Text D .
2. Sie drückt in ihrem Mail-Client den entsprechenden Button "Signieren" und autorisiert diese Aktion mit ihrem Paßwort.
3. Je nach Policy oder Einstellung wird standardmäßig eine multiple oder eine einfache Signatur erzeugt, unter Berücksichtigung der in der Policy festgelegten Pflicht, Zertifikate zu validieren.
4. Der Mail-Client verschickt das folgende Datenpaket in einem PKCS#7-Format:
 - 4.1. Im Falle einer doppelten Signatur: $D, \text{sign}^A(D, \text{prK}^A_{\text{Alice}}), \text{sign}^B(D, \text{prK}^B_{\text{Alice}})$.
 - 4.2. Im Falle einer einfachen Signatur: $D, \text{sign}^A(D, \text{prK}^A_{\text{Alice}})$.

Bob erhält die signierte Nachricht in seinem Mail-Client. Automatisch wird das PKCS#7-Format interpretiert, die enthaltenen Signaturen werden auf die in der Policy festgelegte Weise validiert und die Nachricht wird Bob samt Prüfergebnis (" n von m Signaturen gültig", $n \leq m$, $m=1,2$) angezeigt. Dabei kann es ausreichen, bloß eine Signatur zu validieren und nur bei revozierten Zertifikaten im Pfad auf die zweite Signatur zurückzugreifen.

Verfügen Alice und Bob nicht über multiple Komponenten, so können beide trotzdem sicher miteinander kommunizieren, wenn beide mindestens ein identisches Verfahren unterstützen.

Alice will Bob eine verschlüsselte Nachricht zusenden und nutzt dazu ihren Mail-Client.

1. Alice erzeugt einen Text D .
2. Sie drückt in ihrem Mail-Client den entsprechenden Button "Verschlüsseln".
3. Je nach Policy oder Einstellung soll standardmäßig eine iterative oder eine einfache Verschlüsselung durchgeführt werden. Dazu werden die entsprechenden Zertifikate von Bob benötigt, die entweder abgespeichert sind oder über einen Verzeichnisdienst eingeholt werden. Der Mail-Client kann eine oder mehrere Verschlüsselungen ausführen, wenn Bob entsprechende Zertifikate besitzt, Alice auf diese zugreifen kann und Alices Client selbst über diese Verschlüsselungs-Verfahren verfügt.
4. Der Mail-Client verschickt das folgende Datenpaket in einem PKCS#7-Format:
 - 4.1. Bei einer iterativen Verschlüsselung: $D' = \text{encrypt}^A(\text{encrypt}^B(D, \text{puK}^B_{\text{Bob}}), \text{puK}^A_{\text{Bob}})$.
 - 4.2. Bei einer einfachen Verschlüsselung: $D' = \text{encrypt}^A(D, \text{puK}^A_{\text{Bob}})$.

Bob erhält die verschlüsselte Nachricht D' in seinem Mail-Client.

1. Automatisch wird das PKCS#7-Format interpretiert und die erste Verschlüsselung entschlüsselt: $D'' = \text{decrypt}^A(D', \text{prK}_{\text{Bob}}^A)$, nachdem Bob die Nutzung seines privaten Schlüssels autorisiert hat.
2. Ist D'' eine Klartext-Nachricht, so zeigt der Mail-Client diese an. Ist D'' hingegen ein Kryptogramm, so wird Schritt 1 wiederholt.

Bei einer Authentisierungs-Anwendung besteht im Vergleich zur Signatur-Anwendung meist keine Notwendigkeit für multiple digitale Signaturen oder iterative Verschlüsselungen, weil Daten nur kurzfristig für die Dauer der Kommunikation geschützt werden.

5.2 Austausch im Schadensfall

Angenommen, ein Schaden tritt ein und betrifft PKI^A . Dann werden betroffene Zertifikate aus PKI^A revoziert. Die Funktionsfähigkeit der PKI kann durch PKI^B aufrecht erhalten werden. Der Update Service des Trust-Centers leitet Maßnahmen ein, in denen abhängig vom Schaden kompromittierte Verfahren und Schlüssel durch neue Komponenten ersetzt werden. Abb. 2 zeigt einen groben Überblick, der im folgenden detaillierter diskutiert wird.

5.2.1 Phase 0: Schaden tritt ein

Angenommen, ein Schaden tritt ein, wodurch PKI^A kompromittiert wird. Mögliche Ursachen:

- Schaden S1: Signatur-Verfahren sign^A oder Verschlüsselungs-Verfahren encrypt^A kompromittiert oder fehlerhaft implementiert
- Schaden S2: CA-Schlüssel prK_{CA}^A als Sicherheitsanker kompromittiert
- Schaden S3: CH-Schlüssel prK_{CH}^A kompromittiert

5.2.2 Phase 1: Revokation

Das Trust-Center erfährt vom Schaden entweder durch Veröffentlichungen, durch das BSI, durch einen CH, durch eigene Erfahrungen oder auf eine andere Weise. Angenommen, der Schaden sei u.a. vom Trust-Center geprüft und als schwer wiegend und praktisch relevant eingestuft worden. Diese Gründe rechtfertigen eine Revokation aller betroffenen Zertifikate durch das Trust-Center, denn die in [Road] definierten Umstände, in denen ein Trust-Center ein ausgestelltes Zertifikat vor ihrem regulären Ablauf revoziert, enthalten den Grund "einer Kompromittierung oder vermuteten Kompromittierung des privaten Schlüssels". Und auch die Allgemeinen Geschäftsbedingungen von Trust-Centern sehen dies vor, etwa Deutsche Post Signtrust: "Signtrust sperrt das ausgestellte Zertifikat auch, wenn die den angewendeten Verfahren zugrunde liegenden Algorithmen gebrochen wurden", [eTrust]. Die Revokation eines Zertifikats beinhaltet das Entfernen des Zertifikats aus dem Verzeichnis der gültigen Zertifikate und das Eintragen des Zertifikats auf die CRL^A - und CRL^B -Sperrlisten, die von der CA signiert sind.

PKI^B funktioniert währenddessen uneingeschränkt weiter.

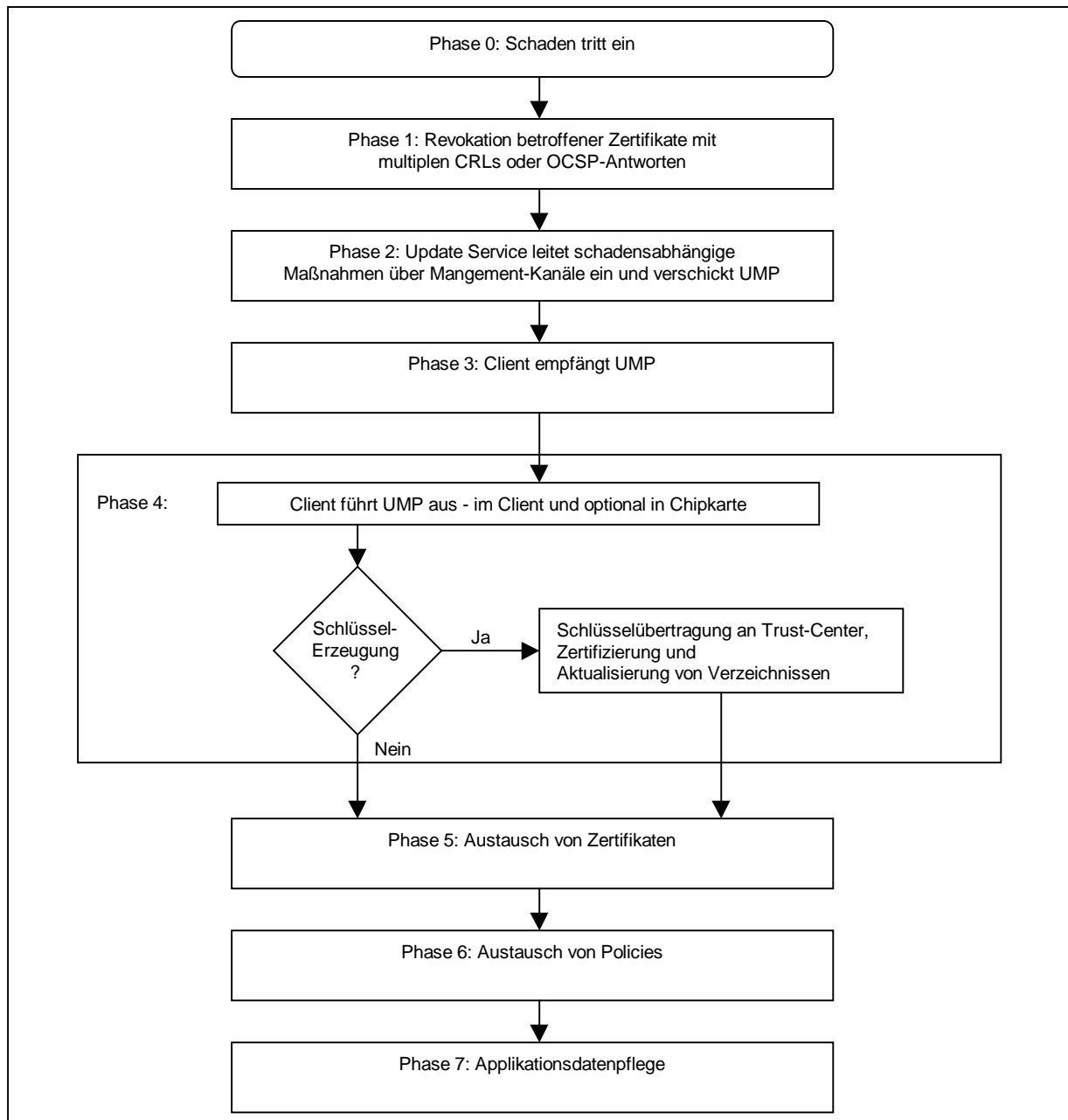


Abb. 2: Grob skizzierter Ablauf im Schadensfall

5.2.3 Phase 2: Update Service leitet schadensabhängige Maßnahmen ein

Schadensabhängig leitet der Update Service des Trust-Centers Maßnahmen ein, die sich zusammensetzen aus:

- Maßnahme M1 zu Schaden S1: Update von kryptographischen Verfahren
- Maßnahme M2 zu Schaden S2: Update des Sicherheitsankers (CA-Zertifikats mit CA-Schlüssel)
- Maßnahme M3 zu Schaden S3: CH-Schlüssel-Update

Dazu definiert der Update Service ein Update Management Protocol (UMP), signiert es multipl mit der kompromittierten und einer sicheren Komponente und verschickt es an alle betroffenen CHs. Grundprinzip einer Deaktivierung von Verfahren und Schlüsseln ist die Sicherung

dieser Aktion mit einer Signatur des kompromittierten Verfahrens, während eine Aktivierung stets mit der Signatur aus sicheren Komponenten abgesichert werden muss.

5.2.4 Phase 3: Client empfängt UMP

In einer Signatur-Anwendung empfängt ein Mail-Client das UMP dadurch, dass der Zertifikatsinhaber die Mail vom Update Service öffnet, das ein in PKCS#7 codiertes UMP enthält. Bei einer Authentisierungs-Anwendung muss der Client, der anderen Clients realen oder virtuellen Zugang zu Räumen oder Daten gewährt, von Zeit zu Zeit auf Revokationsinformationen zugreifen. In diese CRLs oder OCSP-Antworten kann ein UMP integriert werden.

Empfängt ein Client ein UMP, werden die multiplen Signaturen verifiziert. Sofern diese mathematisch korrekt sind, fährt der Client fort und springt von seinem normalen Betriebs-Modus in einen Update-Modus und führt UMP aus. Bei einem Mail-Client bleibt der Update-Modus solange bestehen, bis der Austausch erfolgt ist. Bei einem Client in einer Authentisierungs-Anwendung bleibt der Update-Modus hingegen solange bestehen, bis die Sperrinformationen kein UMP mehr enthalten. In dieser Zeit versucht er, in allen Authentisierungen bei seinem Gegenüber einen Austausch auszuführen.

5.2.5 Phase 4: Client führt UMP aus

Sind die multiplen Signaturen von UMP mathematisch korrekt, wird UMP analysiert. Sind Informationen für den Client enthalten und sind diese Aktionen noch nicht ausgeführt worden, werden sie nach einer Bestätigung durch den Benutzer nun ausgeführt. Der Benutzer kann die Aktion zeitlich verschieben, falls verschlüsselte Dokumente zuvor entschlüsselt werden müssen. Folgende Aktionen sind möglich:

- Verfahren deaktivieren und durch neue Verfahren ersetzen
- Sicherheitsanker ersetzen
- Das Kommando zum Generieren neuer Schlüsselpaare geben

Sind darüber hinaus Informationen für eine Chipkarte enthalten und ist eine Chipkarte an den Client angeschlossen, bei der die Aktionen noch nicht ausgeführt wurden, wird das Chipkarten-Kommando UPDATE COMPONENT mit den Informationen aus UMP an die Chipkarte übermittelt. Für Details zum technischen Austausch im Client und in der Chipkarte sei auf [FlexiPKI] und [HaMa01] verwiesen.

Werden in der persönlichen Sicherheitsumgebung (PSE), was als Software im Client oder in Hardware als Chipkarte realisiert sein kann, neue Schlüsselpaare erzeugt, so werden die öffentlichen Teile an den Client übertragen und mit dem noch vorhandenen sicheren Verfahren signiert. Der Client überträgt diesen neuen Schlüssel in einem PKCS#10-Format an das Trust-Center, wo sie nach Prüfung der Signatur und Güte des Schlüssels von der CA zertifiziert werden können, [PKCS10].

5.2.6 Phase 5: Austausch von Zertifikaten

Sind CH- und CA-Schlüssel ausgetauscht worden, müssen u.U. neue Zertifikate verteilt werden. Dazu existieren bereits Mechanismen. Zum Beispiel kann der Update Service ein neues Zertifikat an eine in PKCS#7 codierte Nachricht anhängen. Oder der CH erhält ein Zertifikat von einem Verzeichnis via LDAP. Da die Sicherheitsanker bei den Beteiligten etabliert sind, können all diese Zertifikate auf ihre Korrektheit verifiziert werden.

5.2.7 Phase 6: Austausch von Policies

Sind Verfahren ausgetauscht worden, müssen u.U. neue Policies verteilt werden. Die Verfahren dazu sind jedoch nicht sicherheitsrelevant und mit einer funktionierenden PKI zu bewerkstelligen.

5.2.8 Phase 7: Applikationsdatenpflege

Dokumente, die vor Eintritt eines Schadens multiple Signaturen aufwiesen und von denen jetzt eine ungültig ist, sollen durch eine Re-signing-Maßnahme wieder ihre ursprünglichen multiplen Signaturen erhalten. Vor dem Austausch von Komponenten entschlüsselte Kryptogramme können jetzt mit den neuen Komponenten wieder verschlüsselt werden.

6 Interoperabilität und Aufwand

Möchten Teilnehmer zweier verschiedener Public-Key-Infrastrukturen miteinander sicher kommunizieren, so geht dies nur, wenn die eingesetzten Verfahren, Zertifikate und Sicherheitsanker kompatibel sind. Verfügen beide Teilnehmer jeweils nur über Komponenten, die einer PKI^A entsprechen, so sollte es keine Probleme geben. Sind bei beiden Benutzern zusätzliche PKI^B-Komponenten verteilt, können sie sogar multipel signierte und iterativ verschlüsselte Dokumente austauschen. Falls nur ein Teilnehmer über multiple Komponenten verfügt, so können sie über die gemeinsam vorhandenen Komponenten kommunizieren - wenn auch ohne Schutz bei einem der beschriebenen Schadensfälle.

Eine Fail-Safe-PKI hat gegenüber einer FlexiPKI ohne Fail-Safe-Konzept natürlich einen erhöhten Bedarf an Speicherplatz, Rechenleistung und Ausführungszeit, weil die zusätzlichen Komponenten vorhanden sein müssen und Berechnungen ausführen sollen. Demgegenüber stehen die Vorteile im Schadensfall, wenn die Fail-Safe-PKI trotz Kompromittierung einer Komponente weiter funktioniert. Diese Vorteile wiegen umso schwerer, je bedeutender die PKI ist und je größer wirtschaftliche und gesellschaftliche Schäden wären. Die Entscheidung, welche Teile des Fail-Safe-Konzepts in einer dedizierten Anwendung genutzt werden, hängt vom ihrem Sicherheitsbedürfnis und der notwendigen Praktikabilität ab. Für eine konkrete Anwendung können folgende Funktionalitäten genutzt werden:

- Existenz multipler Komponenten zum Schutz vor einem Funktionsausfall der PKI.
- Existenz von Fail-Safe-Funktionalitäten, d.h. Komponenten, die das Update Management Protokoll interpretieren und ausführen können, und flexible Integration von Komponenten, die einen Austausch gestatten.
- Nutzung von mehrfach vorhandenen Revokationsmechanismen, d.h. CRL^A und CRL^B oder entsprechenden OCSP-Antworten, um im Schadensfall schnell und ohne Medienbruch Teilnehmer über den Schaden informieren zu können.
- Nutzung von erweiterten Ausdrücken zur Revokation mehrerer Zertifikate.
- Nutzung multipler digitaler Signaturen zum Schutz vor einem Verlust der Beweisbarkeit elektronischer Dokumente.
- Nutzung iterativer Verschlüsselungen zum Schutz vor einem Verlust der Vertraulichkeit elektronischer Dokumente.

Die Vorstellung ist, die zusätzlichen Fail-Safe-Komponenten in eine PKI zu integrieren, ohne die Funktionalität und Effizienz zu beeinträchtigen. Die Fail-Safe-PKI funktioniert wie eine gewöhnliche PKI auch und hält im Hintergrund Mechanismen bereit, die im Schadensfall einschreiten.

Zum Beispiel braucht eine Authentisierungs-Anwendung keine multiplen Signaturen oder iterativen Verschlüsselungen auszuführen: es werden default-mäßig PKI^A-Komponenten genutzt, während PKI^B-Verfahren, -Schlüssel und -Zertifikate ungenutzt bereit stehen. In Signatur-Anwendungen ist es nicht notwendig, dass jeder Teilnehmer jedes Dokument mehrfach signiert und jeder Teilnehmer multiple Signaturen vollständig validiert. Für eine beweiskräftige Signatur ist u.a. ein Zeitstempel nötig, so dass ein Zeitstempeldienst ein geeigneter Anwender multipler digitaler Signaturen sein kann.

7 Implementierung

In der Arbeitsgruppe von Prof. Buchmann wird seit einiger Zeit an der Implementierung der FlexiPKI gearbeitet. Die FlexiPKI wird bereits eingesetzt und weitere Einsätze sind geplant. Die Fail-Safe-Funktionalitäten werden derzeit implementiert - inklusive Austausch von Komponenten auf Java-Chipkarten.

8 Zusammenfassung

Werden die Gefahren und Risiken einer PKI realistisch eingeschätzt, die sich zwingend daraus ergeben, dass in vielen Anwendungen von einer verlässlichen Technik ausgegangen wird, diese aber nicht beweisbar sicher ist, dann bietet das Fail-Safe-Konzept für die FlexiPKI einen Ansatz zur Lösung. Dieser Ansatz ist so flexibel gestaltet, dass Interoperabilität zu bestehenden Standards und bestehenden Infrastrukturen gewährleistet wird.

Literatur

- [Blei98] Daniel Bleichenbacher: *Chosen Ciphertext Attacks against Protocols Based on the RSA Encryption Standard PKCS #1*. In: *Crypto '98*, LNCS 1462, Springer, 1998, Seiten 1-12.
- [BSI] Bundesamt für Sicherheit in der Informationstechnik. <http://www.bsi.de>
- [BuMa00] Johannes Buchmann, Markus Maurer, *Wie sicher ist die Public-Key-Kryptographie?*. In: Patrick Horster (Hrsg.): *Systemicherheit*, Vieweg, Braunschweig, 2000, S. 105-116.
- [BuRT00] Johannes Buchmann, Markus Ruppert, Markus Tak: *FlexiPKI - Realisierung einer flexiblen Public-Key Infrastruktur*. In: Patrick Horster (Hrsg.): *Systemicherheit*, Vieweg, Braunschweig, 2000, S. 309-314.
- [CertCRL] PKIX Working Group: *Internet X.509 Public Key Infrastructure - Certificate and CRL Profile*. Internet Draft, 2000.
- [CMS] Network Working Group: *Cryptographic Message Syntax*. RFC 2630, 1999.
- [CP] Network Working Group: *Internet X.509 Public Key Infrastructure - Certificate Policy and Certification Practices Framework*. RFC 2527, 1999.
- [Dobb96] Hans Dobbertin: *Cryptanalysis of MD4*. In: Dieter Gollmann (Ed.), *Fast Software Encryption: Third International Workshop*, LNCS 1039, Springer, Cambridge, 1996, Seiten 53-69.
- [eTrust] Deutsche Post: *"Allgemeine Geschäftsbedingungen für die eTrust-Dienstleistungen der Deutschen Post Signtrust"*. Januar 2001, www.signtrust.de

- [FJPP95] Hannes Federrath, Anja Jerichow, Andreas Pfitzmann, Birgit Pfitzmann: *Mehrseitig sichere Schlüsselerzeugung*. In: Patrick Horster (Hrsg.), *Trust Center*, Vieweg, Braunschweig, 1995.
- [FlexiPKI] Technische Universität Darmstadt, Fachbereich Informatik, Fachgebiet Kryptographie, Computeralgebra, Lehrstuhl Prof. Dr. Buchmann: *FlexiPKI*. <http://www.informatik.tu-darmstadt.de/TI/Forschung/FlexiPKI/Welcome.html>
- [HaMa01] Michael Hartmann, Sönke Maseberg: *SmartCards for the FlexiPKI Environment*. In: *Proceedings of Gemplus Developers Conference*, Paris, 2001, in Vorbereitung.
- [IETF] Internet Engineering Task Force. <http://www.ietf.org>
- [ISO96] ISO/IEC 9594-8 | ITU-T Recommendation X.509 (1996): *Final Text of Draft Amendments DAM 1 to ISO/IEC 9594-8 on Certificate Extensions, JTC 1/SC 21/WG 4 and ITU-T Q15/7*. April 1996.
- [Koch95] Paul C. Kocher: *Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and Other Systems*. In: Neal Koblitz (Ed.): *Crypto '96*. LNCS 1109, Springer, Berlin, 1996, Seiten 104-113.
- [KoJJ99] Paul C. Kocher, Joshua Jaffe, Benjamin Jun: *Differential Power Analysis*. In: *Crypto '99*. LNCS 1666, Springer, 1999, Seiten 388-397.
- [LeLe93] Arjen K. Lenstra, Hendrik W. Lenstra Jr. (Eds.): *The development of the number field sieve*. LNM 1554, Springer, Berlin, 1993.
- [LeVe00] Arjen K. Lenstra, Eric R. Verheul: *Selection Cryptographic Key Sizes*. In: *Proceedings of Public Key Cryptography 2000*, LNCS 1751, Springer, 2000, Seiten 446-465.
- [LDAP] PKIX Working Group: *Internet X.509 Public Key Infrastructure - Operational Protocols - LDAPv2*. RFC 2559, 1999.
- [OCSP] PKIX Working Group: *Internet X.509 Public Key Infrastructure - Online Certificate Status Protocol - OCSP*. RFC 2560, 1999.
- [PKCS7] RSA Laboratories: *Public-Key Cryptography Standards #7: Cryptographic Message Syntax Standard. Version 1.5*. 1993. <http://www.rsalabs.com/pkcs/pkcs-7>
- [PKCS10] RSA Laboratories: *Public-Key Cryptography Standards #10: Certification Request Syntax Standard. Version 1.7*. 2000. <http://www.rsalabs.com/pkcs/pkcs-10>
- [PKIX] IETF Working Group: *Public Key Infrastructure (X.509) (pkix)*. <http://www.ietf.org/html.charters/pkix-charter.html>
- [Road] PKIX Working Group: *Internet X.509 Public Key Infrastructure - PKIX Roadmap*. Internet Draft, 2000.
- [S/MIME] IETF Working Group: *S/MIME Mail Security (smime)*. <http://www.ietf.org/html.charters/smime-charter.html>
- [Time] PKIX Working Group: *Internet X.509 Public Key Infrastructure - Time Stamp Protocol (TSP)*. Internet Draft, 2001.
- [TLS] Network Working Group: *The TLS Protocol - Version 1.0*. RFC 2246, 1999.
- [Wohl00] Petra Wohlmacher: *Konzepte zur multiplen Kryptographie*. In: Patrick Horster (Hrsg.): *Systemsicherheit*, Vieweg, Braunschweig, 2000, S. 357-369.