

Fail-Safe-Konzept für FlexiPKI

Michael Hartmann¹ · Sönke Maseberg²

¹Technische Universität Darmstadt
hartmann@cdc.informatik.tu-darmstadt.de

²GMD-SIT Darmstadt
maseberg@darmstadt.gmd.de

Zusammenfassung

Public Key Kryptographie basiert auf mathematischen Problemen, von denen nicht bekannt ist, ob sie wirklich schwierig sind. Aus diesem Grund wird in diesem Artikel eine Public Key Infrastruktur vorgestellt, die im Schadensfall ihre Funktionsfähigkeit behält, die repariert werden kann und in der zuvor erzeugte digitale Signaturen ihre Beweiskraft behalten.

1 Einleitung

Public Key Infrastrukturen (PKIs) stellen die Basis für sichere elektronische Kommunikation dar. Digitale Signaturen und Verschlüsselungen gewährleisten Authentizität, Integrität und Vertraulichkeit übertragener Daten. Digitalen Signaturen kommt eine besondere Bedeutung zu, nämlich in den Zertifikaten. Erst durch Zertifikate und Zertifikatsketten kann Vertrauen in einer PKI realisiert werden [PKIX]. Verschlüsselungen sind erst auf Basis einer PKI möglich und werden hier nicht betrachtet.

Die Sicherheit in einer PKI hängt entscheidend von der Güte der eingesetzten Komponenten ab, d.h. den kryptographischen Komponenten, wie Signatur-Algorithmus und Hashfunktion, und den einsatzspezifischen Komponenten, den Schlüsseln und Parametern. Das Bundesamt für Sicherheit in der Informationstechnik (BSI) stellt nach §17 Abs. 2 Satz 1 SigV [SigV97] eine Übersicht über die Algorithmen und dazugehörigen Parameter, die zur Erzeugung von Signaturschlüsseln, zum Hashen zu signierender Daten oder zur Erzeugung und Prüfung digitaler Signaturen als geeignet anzusehen sind, zusammen, die die Regulierungsbehörde für Telekommunikation und Post (RegTP) im Bundesanzeiger veröffentlicht [BA99] [BSI] [RegTP]. Zum Beispiel werden

- 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

als kryptographisch geeignet eingestuft.

Dass eine Komponente kompromittiert wird, kann dennoch nicht vollständig ausgeschlossen werden, denn es gibt keine beweisbar sicheren Signatur-Algorithmen oder Hashfunktionen und es gibt kryptographisch ungeeignete Schlüssel. Tritt nun ein Schaden ein, so ist die Sicherheit

der PKI nicht mehr vollständig gegeben, d.h. die elektronische Kommunikation kann unsicher sein. Ein bestehender Mechanismus in einem solchen Fall ist die Revokation - also die Sperrung von Zertifikaten. Wie schnell diese Revokation Wirkung zeigt, hängt von der Sicherheitspolitik (Security Policy) ab, die vorschreibt, ob beim Verifizieren oder auch beim Signieren die zur Signatur gehörenden Zertifikate verifiziert werden, ob dazu Sperrlisten (Certificate Revocation Lists - CRLs) [HFPS00] konsultiert werden, wie oft diese Sperrlisten aktualisiert werden oder ob der Status eines Zertifikats aktuell via Online Certificate Status Protocol (OCSP) [AAM+99] erfragt wird. Zwei Probleme bleiben:

- Durch Sperrung betroffener Zertifikate wird die Funktionsfähigkeit der PKI eingeschränkt. Die Wiederherstellung der PKI kann je nach Schadensfall die Entwicklung, Produktion, Verteilung und Installation neuer Komponenten nach sich ziehen, was Zeit und Geld kosten würde.
- Durch Kompromittierung des Signatur-Algorithmus können auch vor einem Schaden erzeugte digitale Signatur ihre Beweiskraft verlieren.

Ziel ist deshalb die Entwicklung einer Public Key Infrastruktur, in der trotz Schadensfall der sichere Austausch kompromittierter Komponenten möglich ist und in der digitale Signaturen mit einer zusätzlichen Komponente so erweitert werden, dass sie ihre Beweiskraft nicht verlieren.

Als Grundvoraussetzung müssen technische Komponenten so flexibel in die PKI integriert sein, dass ein Austausch überhaupt möglich ist. Dieser Philosophie folgt die flexible Public Key Infrastruktur, die am Lehrstuhl von Prof. Buchmann [BuRT00] im Projekt 'FlexiPKI' entsteht. Der nächste Schritt ist, die FlexiPKI derartig zu konfigurieren und zu erweitern, dass die genannten Probleme gelöst werden können.

2 Idee

Die Grundidee besteht darin, in eine Public Key Infrastruktur mehrere, voneinander unabhängige kryptographische und einsatzspezifische Komponenten einzubauen, so dass im Falle der Kompromittierung einer Komponente andere Teile der PKI weiterhin sicher funktionieren und dazu beitragen können,

- kompromittierte Komponenten sicher auszutauschen und
- elektronische Dokumente mehrfach zu signieren (Konzept der 'multiplen digitalen Signaturen').

Diese Idee basiert auf der Annahme, dass die Wahrscheinlichkeit für das gleichzeitige Eintreten mehrerer Schadensfälle, sehr gering ist. Da grundsätzlich niemand auftretende Schadensfälle für die Zukunft voraussagen kann, müssen die folgenden Annahmen und Einschränkungen gemacht werden: Es wird angenommen, es sei unmöglich, weder

- sämtliche zugrundeliegenden mathematischen Basisprobleme zu lösen,
- neue effiziente Lösungsverfahren zu entwickeln, noch
- Rechner mit so starker Leistung zu bauen,

dass alle kryptographischen Algorithmen plötzlich und ohne Vorwarnzeit unsicher werden. D.h. es wird angenommen, es besteht keine Möglichkeit mehr zu reagieren, um etwa Daten anderweitig zu sichern oder kryptographische Algorithmen und Parameter anzupassen.

Wenn also ein Fehler auftritt, soll dieser sicher abgefangen werden - ein sogenanntes Fail-Safe Konzept. Der Begriff Fail-Safe wird definiert als eine "programmgesteuerte Beendigung eines Prozesses bei Erkennen eines Hardware- oder Software-Fehlers. Das Verfahren dient dem Schutz des Prozesses, der noch ordnungsgemäß bis zu einem definierten Zustand geführt wird." [Pohl89]

3 Um ein Fail-Safe-Konzept erweiterte FlexiPKI

Zur Lösung der angesprochenen Probleme wird im folgenden eine Public Key Infrastruktur beschrieben, die auf der FlexiPKI basiert und um ein Fail-Safe-Konzept erweitert ist.

Es wird zunächst die Konfiguration der PKI beschrieben, und anschließend die normale und die erweiterte Funktionalität dargestellt. Bei der Realisierung neuer Funktionalitäten wird, auch um eine größtmögliche Interoperabilität mit bestehenden PKI-Systemen zu gewährleisten, auf bestehende Standards aufgesetzt und nur dort neu definiert, wo Änderungen oder Erweiterungen unabdingbar sind.

3.1 Konfiguration

Es wird angenommen, es stehen zwei¹ verschiedene, praktisch einsetzbare Signatur-Algorithmen zur Verfügung. Ein Signatur-Algorithmus bezeichne die Kombination aus Hashfunktion, Formatierungsalgorithmus und den eigentlichen Algorithmen zur Signatur-Erzeugung und -Verifikation [MeOV97]. Die beiden verschiedenen Signatur-Algorithmen sollen untereinander keine identischen Komponenten aufweisen, um das Ausmaß eines Schadens zu begrenzen.

Die hier konfigurierte PKI besteht aus zwei voneinander unabhängigen, parallel zu nutzenden 'Teil-PKIs' PKI^A und PKI^B .

Tab.1: Komponentenverteilung für CA und CH

	CA		CH	
	PKI^A	PKI^B	PKI^A	PKI^B
Signatur-Algorithmen	$sign^A$	$sign^B$	$sign^A$	$sign^B$
Sicherheitsanker			puK_{CA}^A	puK_{CA}^B
Schlüsselpaare	(puK_{CA}^A, prK_{CA}^A)	(puK_{CA}^B, prK_{CA}^B)	(puK_{CH}^A, prK_{CH}^A)	(puK_{CH}^B, prK_{CH}^B)
zugehörige Zertifikate	$cert_{CA}^A$	$cert_{CA}^B$	$cert_{CH}^A$	$cert_{CH}^B$
Verzeichnis	$\{ cert_{CA}^A, cert_{CA}^B, cert_{CH}^A, cert_{CH}^B, CRL \}$			

Es wird eine zweistufige² PKI-Hierarchie angenommen, die aus Zertifizierungsinstanz (Certification Authority - CA) und Teilnehmern (Certificate Holder - CH) besteht. Jeder verfügt je-

¹ Die Idee ist nicht auf zwei Signatur-Algorithmen beschränkt. Sie lässt sich auf m , $m \geq 2$, Verfahren erweitern und bietet dadurch eine skalierbare Sicherheit. Aus Praktikabilitätsgründen beschränken wir uns hier allerdings auf zwei Verfahren.

² Die Idee lässt sich auf beliebige PKI-Hierarchien verallgemeinern.

weils über zwei Signatur-Algorithmen, zwei Schlüsselpaare und zwei zugeordnete Zertifikate. Jeder CH verfügt darüber hinaus über zwei Sicherheitsanker der CA. [ArTu00] Der Verzeichnisdienst hält alle Zertifikate und die Sperrliste CRL bereit. Tabelle 1 zeigt die Komponentenverteilung im Detail. Zu PKI^A gehören alle mit 'A' gekennzeichneten Komponenten: $sign^A$, prK^A_{CA} , puK^A_{CA} , prK^A_{CH} , puK^A_{CH} , $cert^A_{CA}$ und $cert^A_{CH}$. Entsprechendes gilt für PKI^B .

3.2 Normale Funktionalität der FlexiPKI

PKI^A und PKI^B bieten alle Funktionalitäten einer PKI:

- X.509- und Attributszertifikate [ISO96]
- CRL-Zertifikats-Sperrlisten [HFPS00]
- Registrierung, Personalisierung, Initialisierung, Zertifizierung
- Schlüsselmanagement
- Revokation
- Betriebsprotokolle wie Lightweight Directory Access Protocol (LDAP) oder Online Certificate Status Protocol (OCSP) [BoHR99] [AAM+99]
- Kryptographische Protokolle wie Cryptographic Message Syntax (CMS), die in PKCS#1, S/MIME oder Secure Sockets Layer (SSL) integriert sind. [CMS][PKCS1][SMIME]

3.3 Multiple digitale Signaturen

Das angesprochene offene Problem, dass eine digitale Signatur im Schadensfall ihre Aussagekraft verlieren könnte, kann nun gelöst werden. Unter den Annahmen aus Kapitel 2 werden nicht beide Teil-PKIs gleichzeitig kompromittiert. Die Idee ist, PKI^A und PKI^B zur Signaturerzeugung zu nutzen und ein Dokument doppelt zu signieren. In Anlehnung an Cryptographic Message Syntax (CMS) hätten diese multiplen digitalen Signaturen folgende Form (Tab. 2) [CMS]:

Tab.2: Multipel signiertes Dokument in Anlehnung an CMS

1	Signatur_Info
2	Dokument D
3	optional Zertifikate
4	optional Sperrlisten
5	$sign_info^A$
6	Signatur ^A , erzeugt aus $sign^A$, prK^A und D
7	optional $sign_info^B$
8	optional Signatur ^B , erzeugt aus $sign^B$, prK^B und D

CMS bietet bereits die Möglichkeit, Dokumente mehrfach zu signieren, so dass keine Erweiterungen oder Änderungen nötig sind.

3.4 Funktionalität im Schadensfall

Angenommen, ein Schaden – die Kompromittierung einer beliebigen Komponente eines Signatur-Algorithmus – tritt ein. Ohne Beschränkung der Allgemeinheit sei PKI^A als unsicher anzusehen. Ausgangspunkt ist das Bekanntwerden, dass eine in der PKI verwendete Komponente kompromittiert wurde. Diese Information muss der zuständigen CA zugänglich gemacht werden. Die CA prüft den Schadensfall und leitet - bei einem realen Schaden - die weiteren Aktionen ein.

Das erste Ziel ist, Zertifikate cert^A kompromittierter Schlüssel nicht mehr anzuerkennen. Dazu existieren bereits Mechanismen: Betroffene Zertifikate werden revoziert und je nach Security Policy werden beim Verifizieren oder Signieren Sperrlisten konsultiert oder OCSP-Anfragen getätigt. Unter den Annahmen aus Kapitel 2 ist PKI^B vom Schaden nicht betroffen und steht vollständig zur Verfügung. Die Funktionalität der PKI gilt es nun in vollem Umfang wiederherzustellen. Die dazu notwendigen Schritte sind aus Tabelle 3 ersichtlich:

Tab.3: Arbeitsschritte im Kompromittierungsfall

Schritt	Fein	Grob
1	Schadensfall wird dem CA-Administrator bekannt	INFORMIEREN
2	Schadensfall wird bei CA registriert und geprüft	
3	CA: Sperren betroffener Zertifikate	
4	CA: Sperrlisten zusätzlich verbreiten	
5	Kompromittierte Komponenten deaktivieren	REPARIEREN
6	Neue Komponenten laden	
7	Schlüssel-Erzeugung	
8	CA: Zertifizierung	
9	Applikationsdatenpflege	RE-SIGNING

3.4.1 Informieren

Die CA wird über die Kompromittierung einer verwendeten Komponente informiert, sie prüft den Schaden und initiiert - bei einem real existierenden Schaden - die folgenden Aktionen, um Sperrlisten und Verzeichnisse zu aktualisieren. (vgl. Abb.1).

Die Sperr-Information zum Aktualisieren der Sperrliste in einem Verzeichnis erfolgt über LDAP- bzw. OCSP-Formate. Um das Bottleneck zu vermeiden, das entsteht, wenn viele CH mit der CA kommunizieren wollen, soll die Verteilung dieser Sperr-Information über weitere Kanäle möglich sein. Eine Möglichkeit ist, die Information über Mails, Mailattachments, Printmedien, Disketten oder das Internet zu verbreiten. Die Information soll eine für den CH

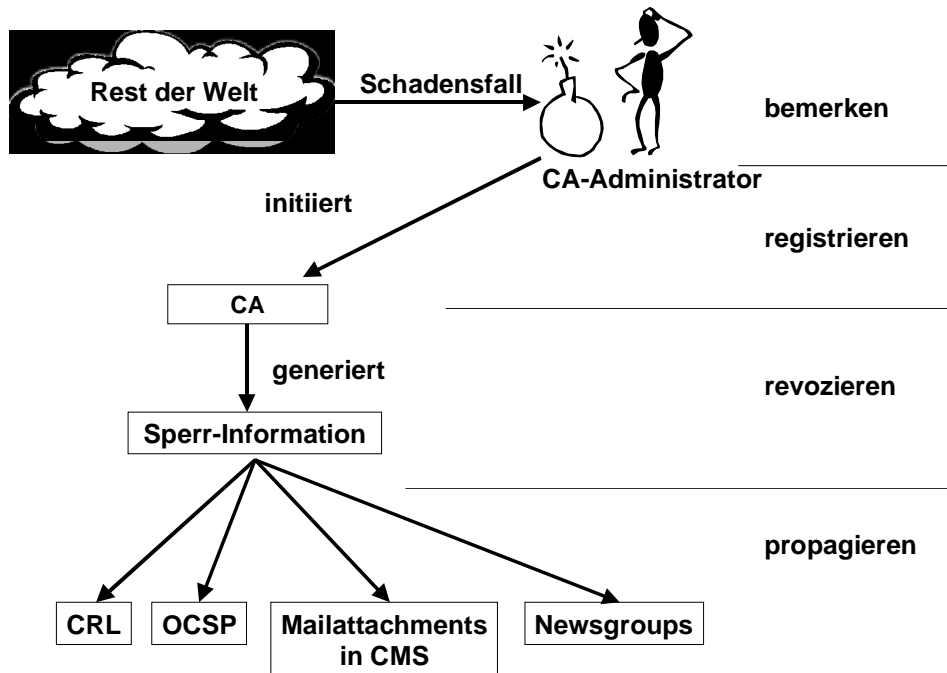


Abb.1: Informationsfluss beim Auftreten eines Schadensfalles

lesbare Nachricht sein, an die aktualisierte Sperrlisten angehängt sind. Die Information soll alle notwendigen Daten enthalten, aber nicht größer als nötig sein, damit es auch noch in den Personal Security Environments der CH mit geringer Rechenleistung und Speicherkapazität - wie z.B. Chipkarten - abgearbeitet werden kann. CMS stellt ein geeignetes Format zur Verfügung (vgl. Tabelle 4). Die Auswertung in den Clients erfolgt dann entweder automatisch oder halbautomatisch, wenn der Benutzer die Sperrlisten dem Client zur Verfügung stellen muss. Bei der Auswertung prüft der Client die beiden Signaturen und zeigt - nach erfolgreicher Verifikation - den Info_String an, um den Benutzer zu informieren.

Tab.4: Aufbau des Info_Object Datentyps auf CMS-Basis

1	Signatur_Info	
2	Info_String	Information über Schadensfall im Klartext; für Benutzer lesbar, um ihn über die aktuellen Geschehnisse zu informieren
3	Sperrlisten	
4	sign_info ^A	Multiple digitale Signatur, d.h. auch die Signatur, die eine Komponente mit dem kompromittierten OID verwendet! Andernfalls könnte ein Angreifer, der etwa PKI ^A erfolgreich attackiert hat, PKI ^B sperren!
5	Signatur ^A , erzeugt aus sign ^A , prK ^A und D	
6	sign_info ^B	
7	Signatur ^B , erzeugt aus sign ^B , prK ^B und D	

3.4.2 Reagieren

Über die zuvor beschriebenen Verteilungs-Möglichkeiten propagiert die CA ein Objekt des Datentyps `Component_Update` (siehe Tab. 5), um kompromittierte Komponenten zu deaktivieren und neue Komponenten zu laden. Der Client muss im Falle einer Chipkarte als PSE, diese über einen Kartenleser ansprechen. Existierende Management-Protokolle wie das Certificate Management Protocol (CMP) müssen dazu erweitert werden. [CMP]

Tab.5: Aufbau des `Component_Update` Datentyps

1	Signatur_Info	
2	Info_String	Information über Inhalt des Updates; für Benutzer lesbar
3	relevantID	Identifizier der Applikation, für die dieses Update gedacht ist (Mailtool, AID der Chipkarten-Applikation,...)
4	Info_Code	Info_Code mit für den Client interpretierbaren Informationen über Schaden und einzuleitende Aktionen
5	Neue_OID	OID der neuen Komponente
6	BINARY	Code neuer Komponente
7	optional Zertifikate	z.B. CV-Zertifikate, um in Chipkarte Rechte zu setzen
8	sign_info ^A	Multiple digitale Signatur, d.h. auch die Signatur, die eine Komponente mit dem kompromittierten OID verwendet! Andernfalls könnte ein Angreifer, der etwa PKI ^A erfolgreich attackiert hat, PKI ^B deaktivieren!
9	Signatur ^A	
10	sign_info ^B	
11	Signatur ^B	

Der Client vergleicht nun zuerst seinen eigenen Identifier mit dem in `Component_Update[3]` (`Component_Update[x]` bezeichnet die Zeile x aus Tabelle `Component_Update`) aufgeführten, um festzustellen, ob diese neue Komponente überhaupt für ihn gedacht ist. Bei Nicht-Übereinstimmung bricht er den Komponentenaustausch mit einer Fehlermeldung ab. Sonst vergleicht er alle aktiven Verfahren mit den in `Component_Update[4]` gelisteten und merkt diese bei Übereinstimmung zur Deaktivierung vor. Anschließend werden die beiden Signaturen `SignaturA` und `SignaturB` verifiziert, und bei positivem Verifikationsergebnis werden die zur Deaktivierung vorgemerkten Verfahren deaktiviert und die entsprechenden Komponenten gelöscht. Danach wird die neue Komponente eingebunden und als aktiv gekennzeichnet (vgl. Abb. 3).

Nachdem die neue Komponente integriert wurde, müssen eventuell neue Schlüssel generiert und neue Zertifikate ausgestellt werden. Dies ist abhängig von der ausgetauschten Komponente und der angewandten CA-Policy. Bei einer neuen Hashfunktion müssen z.B. keine neuen Schlüssel generiert werden, im Gegensatz zum Austausch eines Signatur-Algorithmus. Die Protokolle zum Einbringen der Schlüssel und der Zertifikatsverteilung in das PSE unterscheiden sich voneinander, wenn die Schlüsselgenerierung im PSE oder in der CA stattfindet. In dem Fall, dass das PSE die neuen Schlüsselpaare generiert, muss sicher gestellt werden, dass der öffentliche Schlüssel authentisch an die CA übermittelt wird, so dass diese dann das ent-

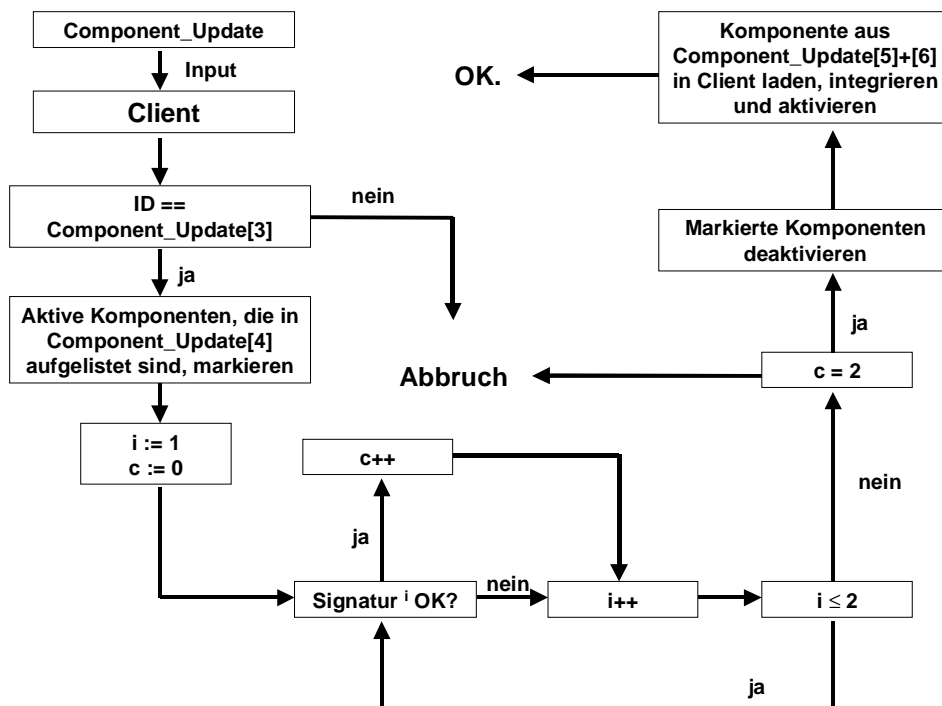


Abb. 3: Austausch einer Komponente im Client

sprechende Zertifikat ausstellen kann. Erzeugt die CA die neuen Schlüsselpaare, dann müssen öffentlicher und privater Schlüssel authentisch und vertraulich an das richtige PSE übertragen werden. Da trotz Schaden eine funktionierende Teil-PKI PKI^B zur Verfügung steht, ist dies zu bewerkstelligen.

3.4.3 re-signing

Nachdem die volle Funktionalität der PKI wieder hergestellt wurde, gilt es nun, die Datenbestände der einzelnen Applikationen zu pflegen. D.h. Dokumente, die zwar multiple digitale Signaturen verwenden, bei denen allerdings eine Komponente kompromittiert wurde, müssen re-signiert werden, um eine kontinuierliche Gültigkeit zu gewährleisten. Die Details sind dabei sehr applikationsspezifisch und je nach zu pflegendem Datenbestand sind unterschiedliche gesetzliche Vorschriften einzuhalten. Wie die Pflege der entsprechenden Datenbestände genau erfolgen muss, kann daher nicht mehr Teil dieses Artikels sein und muss gesondert betrachtet werden.

4 Kosten/Nutzen-Überlegungen

Eine hier konfigurierte FlexiPKI mit Fail-Safe-Konzept und den Teil-PKIs PKI^A und PKI^B benötigt gegenüber einer 'normalen' PKI den folgenden Mehraufwand:

- CA, CH-Client und Verzeichnisse benötigen jeweils doppelt soviel Ressourcen, was aber relativ unkritisch ist, weil diese PKI-Komponenten über beliebig skalierbare Ressourcen verfügen können.

- Ebenso muss die CH-Chipkarte doppelt sowiele kryptographische und einsetzspezifische Komponenten enthalten, was wegen der beschränkten Ressourcen kritisch werden kann.
- Zertifikate und Sperrlisten, sowie Registrierung, Initialisierung, Personalisierung, Zertifizierung, Schlüsselmanagement, Revokation, LDAP- und OCSP-Anfragen und SSL werden nicht verändert und sind ohne Mehraufwand in einer FlexiPKI mit Fail-Safe-Konzept zu betreiben.
- Erweiterte Funktionalität: Schutz vor Verlust der Beweiskraft digital signierter Dokumente durch Einsatz von multiplen digitalen Signaturen. Aktivitäten im Signatur- und Verifikationsprozess, die im CH-Client ablaufen, sind unkritisch. Sobald eine Chipkarte involviert wird, verlängern sich die benötigten Zeiten grob, da doppelt sowiele Berechnungen durchgeführt werden müssen. Der Transport im Netz ist unkritisch, weil eine Nachricht im CMS-Format durch multiple Signaturen nur unwesentlich wächst. Die Codierung einer Nachricht in CMS mit multiplen Signaturen (RSA mit 1024 Bit und ECDSA mit 160 Bit Schlüssellänge) benötigt im Vergleich zur selben Nachricht, codiert in CMS mit einfacher Signatur (RSA mit 1024 Bit Schlüsseln), lediglich 78 zusätzliche Byte.
- Minderaufwand im Schadensfall: Reparatur-Maßnahmen sind nicht unerheblich, aber wesentlich niedriger einzuschätzen als eine neue Entwicklung, Produktion, Zertifizierung und Roll-Out neuer Komponenten. Effizienzsteigerungen und Optimierungen sind bewußt aus diesem Grund noch nicht berücksichtigt worden.

Der Mehraufwand im normalen Betrieb ist also kalkulierbar. Der Aufwand im Schadensfall hingegen wesentlich geringer.

5 Beispiel

In diesem Beispiel wird eine 'kleine' PKI, bestehend aus Certification Authority und zwei Certificate Holdern Alice und Bob betrachtet. Genutzt wird

- RSA mit 1024 Bit Schlüssellänge und RIPEMD-160 als Hashfunktion, sowie
- ECDSA mit 160 Bit Schlüssellänge und SHA-1 als Hashfunktion

Alice und Bob wollen ihre e-Mails signieren. Beide verfügen über je einen Computer mit einem Internet-Anschluss, einem Mailprogramm und einem Kartenleser für die Kommunikation mit der Signatur-Chipkarte. (vgl. Abb. 4)

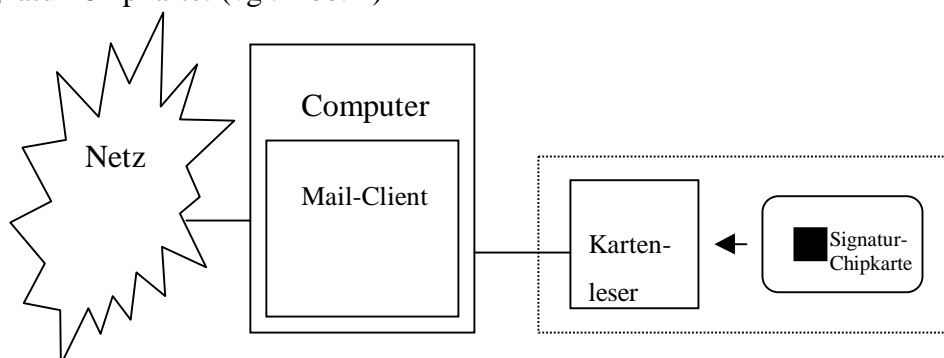


Abb. 4: Alices bzw. Bobs Computer-Ausstattung

Alice und Bob signieren ihre e-Mails multipel. Dazu steuert das Mailprogramm die Chipkarte mit ISO-Kommandos an, um den Hashwert, den der Client berechnet hat, signieren zu lassen:

1. MSE³ <RSA> und PSO: COMPUTE DS⁴ <Hashwert>
2. MSE <ECDSA> und PSO: COMPUTE DS <Hashwert>

Eines Tages gelingt es plötzlich, RSA mit 1024 Bit Schlüssellänge zu kompromittieren. Das BSI teilt mit, dass von nun an 2048 Bit zu benutzen seien. Die CA sperrt daraufhin alle mit RSA zertifizierten Schlüssel, in dem CRL-Sperrlisten und Verzeichnisse durch LDAP- und OCSP-Formate aktualisiert werden. Je nach Gültigkeitsmodell reicht eine Sperrung des CA-Zertifikats (cert_{CA}^A). Zusätzlich verschickt die CA an Alice und Bob eine e-Mail in CMS mit einer für Alice und Bob lesbaren Information über den Schaden und mitgelieferten aktualisierten Sperrlisten, die vom Mailclient ins System integriert werden. Alice und Bob können aufgrund der funktionierenden PKI^B weiterhin miteinander kommunizieren und ihre vorherige Kommunikation als nach wie vor beweiskräftig auffassen.

Zur Reparatur sendet die CA jeweils an Alices und Bobs Client ein Component_Update-Objekt (entsprechend Tabelle 5) mittels Certificate Management Protocol (CMP). Der Mailclient interpretiert die Nachricht und weiss, was zu tun ist:

1. Das mitgelieferte CV-Zertifikat wird an die Chipkarte geliefert, so dass die Rechte für die folgenden Aktionen gesetzt sind.
2. Kompromittierte Komponenten zu sign^A werden deaktiviert.
3. Neue Komponenten zu $\text{sign}^{A_{\text{neu}}}$ werden geladen, installiert und aktualisiert.

In diesem Beispiel findet die Schlüsselerzeugung in der Karte statt. Der Mailclient löst die Erzeugung mit GENERATE PUBLIC KEY PAIR aus, liest den soeben generierten öffentlichen Schlüssel $\text{puK}_{CH}^{A_{\text{neu}}}$ - etwa mittels READ BINARY - aus und lässt ihn mit PKI^B signieren: MSE <ECDSA> und PSO: COMPUTE DS < $\text{puK}_{CH}^{A_{\text{neu}}}$ >. Schlüssel und Signatur schickt der Client zur CA, die diesen öffentlichen Schlüssel auf Basis der Signatur zertifizieren kann.

Wichtig ist, dass jeder CH den neuen Sicherheitsanker der CA ($\text{puK}_{CA}^{A_{\text{neu}}}$) authentisch erhält. Dazu schickt die CA mittels CMP den Schlüssel $\text{puK}_{CA}^{A_{\text{neu}}}$, die Signatur sign^B (" $\text{puK}_{CA}^{A_{\text{neu}}}$ ", prK_{CA}^B) und ein CV-Zertifikat, so dass der Mailclient den Sicherheitsanker authentisch in die Chipkarte installieren kann.

Am Ende steht eine reparierte PKI^{A_{neu}} zur Verfügung.

6 Fazit

Das hier vorgestellte Konzept stellt die Funktionsfähigkeit einer PKI bei einem aufgetretenen Schadensfall sicher, gestattet die Reparatur im laufenden Betrieb und ermöglicht dem Anwender die auf seine Bedürfnisse angepasste Verwendung von multiplen digitalen Signaturen.

³ MSE steht für MANAGE SECURITY ENVIRONMENT

⁴ PSO: COMPUTE DS steht für PERFORM SECURITY OPERATION: COMPUTE DIGITAL SIGNATURE

Literatur

- [AAM+99] C. Adams, R. Ankney, A. Malpani, M. Myers, S. Galperin: Internet X.509 Public Key Infrastructure - Online Certificate Status Protocol - OCSP. Request for Comments 2560, 1999.
- [ArTu00] A. Arsenault, S. Turner: Internet X.509 Public Key Infrastructure - PKIX Roadmap. Internet Draft, 2000.
- [BA99] Bundesanzeiger Nr. 213 - Seite 18.638 vom 11. November 1999. <http://www.regtp.de>
- [BoHR99] S. Boeyen, T. Howes, P. Richard: Internet X.509 Public Key Infrastructure - Operational Protocols - LDAPv2. Request for Comments 2559, 1999.
- [BSI] Bundesamt für Sicherheit in der Informationstechnik. <http://www.bsi.de>
- [BuRT00] J. Buchmann, M. Ruppert, M. Tak: FlexiPKI - Realisierung einer flexiblen Public-Key Infrastruktur. In: P. Horster: Systemsicherheit, Vieweg, 2000, S. 309-314.
- [CMP] Network Working Group, "Internet X.509 Public Key Infrastructure - Certificate Management Protocols". Request for Comments 2510. März 1999.
- [CMS] Network Working Group, "Cryptographic Message Syntax". Request for Comments 2630, Juni 1999.
- [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.
- [HFPS00] R. Housley, W. Ford, W. Polk, D. Solo: Internet X.509 Public Key Infrastructure - Certificate and CRL Profile, Internet Draft, 2000.
- [MeOV97] Alfred J. Menezes, Paul C. van Oorschot, Scott A. Vanstone, "Handbook of applied cryptography", CRC Press, Boca Raton, 1997.
- [PKCS1] RSA Laboratories: Public-Key Cryptography Standards #1: RSA Cryptography Standard. Version 2.1, 1999, <http://www.rsalabs.com/pkcs/pkcs-1>
- [PKIX] Internet Engineering Task Force (IETF) Working Group: Public Key Infrastructure (X.509) (pkix). <http://www.ietf.org/html.charters/pkix-charter.html>
- [RegTP] Regulierungsbehörde für Telekommunikation und Post. <http://www.regtp.de>
- [SigV97] Verordnung zur digitalen Signatur (Signaturverordnung - SigV), 1997. <http://www.regtp.de>
- [SMIME] Internet Engineering Task Force (IETF) Working Group: S/MIME Mail Security (smime). <http://www.ietf.org/html.charters/smime-charter.html>
- [Pohl89] Hartmut Pohl: Lexikon Sicherheit der Informationstechnik A-Z. Datakontext-Verlag, Köln, 1998.