Schlanke Infrastrukturen für den digitalen Rechtsverkehr

Vorwärtssichere Verfahren für qualifizierte elektronische Signaturen
ISPRAT Studie

TECHNISCHE
UNIVERSITÄT
DARMSTADT

ISPRAT
Interdisziplinäre Studien zu Politik, Recht,
Administration und Technologie e.V.

Autoren

Johannes Braun
jbraun@cdc.informatik.tu-darmstadt.de

Andreas Hülsing
huelsing@cdc.informatik.tu-darmstadt.de

Alexander Wiesmaier
alex@wiesmaier.de

Technische Univeristät Darmstadt
Fachbereich Informatik
Kryptographie und Computeralgebra
Prof. Dr. Johannes Buchmann

Inhaltsverzeichnis

Abkürzungsverzeichnis **1**

1 Projektzusammenfassung **2**
1.1 Problemstellung und Ziel des Projektes . . . 2
1.2 Lösungsansatz . . . 2
1.3 Veröffentlichungen und Präsentationen . . . 3
1.3.1 Wissenschaftliche Publikationen . . . 3
1.3.2 Präsentationen . . . 4

2 Einleitung **5**
2.1 Motivation . . . 5
2.2 Problemstellung . . . 5
2.3 Zielsetzung . . . 6
2.4 Lösungsansatz . . . 6

3 Grundlagen **8**
3.1 Public Key Infrastrukturen . . . 8
3.1.1 Zertifikate . . . 8
3.1.2 Revokation . . . 9
3.1.3 Gültigkeitsprüfung . . . 9
3.2 Zeitstempeldienste . . . 10
3.3 Vorwärtssichere Signaturverfahren . . . 11
3.3.1 FSS und Revokation . . . 12
3.3.2 Schlüssel Update und Sicherheitskopien von Signaturschlüsseln . . . 13

4 Bedrohungs- und Anforderungsanalyse **14**
4.1 Bedrohungsanalyse und aktuelle Gegenmaßnahmen . . . 14
4.1.1 Kontinuierlicher technologischer Fortschritt . . . 14
4.1.2 Unvorhergesehener technologischer Fortschritt . . . 15
4.1.3 Schlüsselkompromittierung . . . 15
4.1.4 Schlüsselverlust . . . 16
4.1.5 Signaturabstreitung . . . 17
4.2 Anforderungsanalyse . . . 17
4.2.1 Formale Anforderungen . . . 17

4.2.2 Technische Anforderungen 19

5 Realisierung **21**

5.1 Kettenmodell mit FSS 21
5.2 Verbesserte Revokationshandhabe 23
5.3 Signaturverfahren 23
5.3.1 Sicherheit von XMSS 24
5.3.2 Frühwarnmechanismus 24
5.3.3 Redundanz durch Hash-Combiner 24
5.4 FSS für End-Entitäten 25
5.4.1 Online Protokoll 25
5.4.2 Gültigkeitstoken mit Novomodo 27

6 Sicherheitsanalyse und Validierung **30**

6.1 Anforderungserfüllung 30
6.2 Langfristige Sicherheit 30

Abkürzungsverzeichnis

CRL – Zertifikatsrevokationsliste (Certificate Revocation List)
FSS – Vorwärtssicheres Signaturverfahren (Forward Secure Siganture Scheme)
GMSS – Generalized Merkle Signature Scheme
HSM – Hardware Sicherheitsmodul
OCSP – Online Certificate Status Protocol
PKI – Public Key Infrastruktur
QES – Qualifizierte Elektronische Signatur
SigG – Gesetz zur digitalen Signatur
TLS – Transport Layer Security
XMSS – eXtended Merkle Signature Scheme
ZDA – Zertifizierungsdiensteanbieter

1 Projektzusammenfassung

1.1 Problemstellung und Ziel des Projektes

Ziel des Projektes war ein neuartiger Lösungsansatz zur Vereinfachung des Umgangs mit qualifizierten elektronischen Signaturen durch Verschlankung der benötigten Infrastrukturen: die vorwärtssichere, robuste Public Key Infrastruktur (PKI). Diese PKI erlaubt es den Nutzern Dokumente sicher und langfristig gültig zu signieren. Das Projekt hatte zwei zentrale Ziele:

- Gültigkeitserhalt der erstellten Signaturen unter verschiedenen Bedrohungen. Beispiele hierfür sind der kontinuierliche sowie der unvorhergesehene technologische Fortschritt, Schlüsselkompromittierungen oder nicht vertrauenswürdige Nutzer.

- Erfüllung der rechtlichen Anforderungen an qualifizierte elektronische Signaturen.

Das Projekt sollte dabei die bestehenden Lösungsansätze verbessern und vereinfachen. Damit sollte das Projekt einen Beitrag zur besseren Akzeptanz und zum breiteren Einsatz von elektronischen Signaturen leisten. Die erarbeiteten Lösungen sollten sich in bestehende Infrastrukturen integrieren lassen, um eine einfache Migration zu erlauben.

Die Laufzeit des Projektes betrug 12 Monate, von August 2011 bis August 2012. Im Folgenden gehen wir auf die wichtigsten Ergebnisse ein, die während dieser Zeit erzielt wurden. Wir fassen den Lösungsansatz kurz zusammen und geben einen Überblick über die wichtigsten Publikationen. Anschließend werden die Projektergebnisse im Detail dargestellt.

1.2 Lösungsansatz

Viele der heute verwendeten Lösungsansätze zur Abwehr von Bedrohungen im Umfeld elektronischer Signaturen benötigen Zeitstempel. Diese sind jedoch mit einer Reihe von Nachteilen bezüglich Effizienz und Aufwand verbunden. Die Verschlankung und Vereinfachung der benötigten Infrastruktur sollte daher durch den weitgehenden Verzicht auf Zeitstempel erreicht werden. Die Notwendigkeit von Zeitstempeln konnte zwar nicht komplett eliminiert werden, jedoch konnte deren Einsatz deutlich reduziert werden. Dadurch sind Zeitstempel lediglich bei langfristiger Archivierung, und auch hier nur in deutlich reduziertem Umfang, von Nöten. In der vorwärtssicheren PKI werden die Sicherheitsgarantien durch den Einsatz vorwärtssicherer Signaturverfahren erreicht. Zur Umsetzung haben wir das eXtended Merkle Signature Scheme (XMSS) [1] verwendet. Neben der Vorwärtssicherheit bietet XMSS als hash-basiertes Signaturverfahren die Möglichkeit, durch sogenannte Hash-Combiner auf effiziente Weise verschiedene

Verfahren zu kombinieren. Damit erreicht man Redundanz. Selbst der Bruch eines Verfahrens kann dann die Gültigkeit der Signaturen nicht gefährden.

Darüber hinaus haben wir die Vorteile einer vorwärtssicheren PKI auch im Falle einer Kompromittierung von Zertifizierungsdiensteanbietern (ZDA) detailliert betrachtet und gezeigt, dass hier Revokationsmechanismen verbessert werden können. Hierdurch kommt es trotz Revokation von ZDA Zertifikaten nicht zum Zusammenbruch einer solchen PKI. Daneben haben wir ein Protokoll erarbeitet, welches den Einsatz von zustandsbehafteten Signaturverfahren auf Nutzerseite auf sichere Weise ermöglicht. Der hier verfolgte Ansatz besteht darin, die Verfolgung des aktuellen Signaturindex online durch den ZDA zu handhaben. Damit wird erreicht, dass der aktuelle Schlüsselzustand selbst bei Verlust der Signaturkarte noch nachvollziehbar bleibt und somit Signaturabstreitungen verhindert werden können.

1.3 Veröffentlichungen und Präsentationen

1.3.1 Wissenschaftliche Publikationen

Kern des Projektes war die Umsetzung einer PKI mittels vorwärtssicheren Signaturverfahren (FSS). Dazu wurden die verschiedenen Gültigkeitsmodelle zur Signaturprüfung untersucht und gezeigt, dass das Kettenmodell die Vorteile von vorwärtssicheren Verfahren bestmöglich nutzt. Auf der anderen Seite wurde gezeigt, dass FSS erst eine effiziente Umsetzung des Kettenmodells ermöglichen. Die Ergebnisse, mit Fokus auf die Robustheit der beschriebenen PKI, wurden in Form eines Konferenzbeitrages auf der EuroPKI 2012, 9th European PKI Workshop: Research and Applications veröffentlicht. Der Beitrag wurd am 13./14. September 2012 auf der Konferenz in Pisa, Italien präsentiert.

Referenz:
Johannes Braun, Andreas Hülsing, Alexander Wiesmaier, Martin A. G. Vigil, and Johannes Buchmann. How to avoid the breakdown of public key infrastructures - forward secure signatures for certificate authorities. In: Proceedings of the 9th European PKI Workshop: Research and Applications - EuroPKI 2012, September 13-14, 2012.

Eine weitere Publikation beschäftigt sich gezielt mit der Realisierung des Kettenmodells im Hinblick auf qualifizierte elektronische Signaturen. Hier werden auch erste Ergebnisse aus einer Demonstrator Implementierung als Browser Plugin vorgestellt. Die Arbeit wird auf dem 13. Deutschen IT-Sicherheitskongress vom 14.-16. Mai 2013 in Bonn präsentiert.

Referenz:
Johannes Braun, Moritz Horsch, Andreas Hülsing. Effiziente Umsetzung des Kettenmodells unter Verwendung vorwärtssicherer Signaturverfahren. In: Proceedings zum 13. Deutschen IT-Sicherheitskongress, 14.-16. Mai 2013.

Ebenso wird eine weitere Arbeit auf dem 13. Deutschen IT-Sicherheitskongress vom 14.-16. Mai 2013 in Bonn vorgestellt, welche die Langzeitsicherheit von hash-basierten Signaturverfahren, insbesondere XMSS, behandelt. Hier werden Details zum allgemeinen Stand der Technik, der Performanz von XMSS und zur Umsetzung langzeitsicherer Signaturen mit Hilfe von Hash-Combinern gezeigt.

Referenz:
Andreas Hülsing, Johannes Braun. Langzeitsichere Signaturen durch den Einsatz hash-basierter Signaturverfahren. In: Proceedings zum 13. Deutschen IT-Sicherheitskongress, 14.-16. Mai 2013.

1.3.2 Präsentationen

Auf dem Vierten ISPRAT Wissenschaftstag am 5. Juni 2012 in St.Gallen wurde das Projekt in Form eines Poster-Vortrags mit dem Titel “Schlanke Infrastrukturen für den digitalen Rechtsverkehr: Vorwärtssichere Verfahren für qualifizierte elektronische Signaturen” vorgestellt.

Bei der Preisverleihung zum 4. Deutschen IT-Sicherheitspreis 2012 am 29. November 2012 in Darmstadt wurde unsere Einreichung in Form eines Poster-Vortrags präsentiert. Die Einreichung mit dem Titel “FSS4CA: Minimierung von Revokationsfolgen” ist eine der 10. Finalisten und baut auf Ergebnissen des hier beschriebenen ISPRAT Projektes auf.

2 Einleitung

2.1 Motivation

In den letzten Jahren wurden etwa mit Bürgerportalen im Internet oder mit elektronischen Formularen große Fortschritte im eGovernment erzielt. Dadurch wurde sowohl die Effizienz der Vorgänge als auch die Bürgernähe der Verwaltung gesteigert.

Elektronische Signaturen sind eine der wichtigsten Grundlagen um auch den Rechtsverkehr zwischen Bürger und Amt vollständig elektronisch abbilden zu können. Der Gesetzgeber hat mit der Einführung von qualifizierten elektronischen Signaturen und deren Gleichstellung mit handschriftlichen Unterschriften bereits die rechtlichen Grundlagen dafür geschaffen. Mit der Inbetriebnahme der entsprechenden Kommunikationsinfrastruktur hat der Staat auch die technischen Mittel bereitgestellt. Durch die Einführung des neuen Personalausweises ist spätestens in zehn Jahren auch jeder erwachsene Bundesbürger im Besitz der Technik zum Erstellen digitaler Signaturen.

Die Relevanz von qualifizierten elektronischen Signaturen wird im elektronischen Rechtsverkehr in den kommenden Jahren stark zunehmen. Sei es die elektronische Gesundheitskarte oder der elektronische Entgeltnachweis – beim Übergang zum elektronischen Rechtsverkehr sind qualifizierte Signaturen unabdingbar. Allerdings gestaltet sich der Einsatz rechtsgültiger elektronischer Unterschriften in der Praxis immer noch schwierig.

2.2 Problemstellung

Ursprung vieler Probleme ist, dass die Gültigkeit elektronischer Signaturen im Gegensatz zu handschriftlichen Unterschriften über längere Zeiträume nicht per se gegeben ist. Diese muss mit aktiven Maßnahmen erhalten werden. Gängige Maßnahmen um den Gültigkeitserhalt zu gewährleisten sind das Übersignieren (vgl. [2]) mit stärkeren Signaturverfahren bzw. Schlüsseln, das Aufbringen von Zeitstempeln (vgl. [3]) oder sogenannte Mehrfachsignaturen (vgl. [4]). Diese Maßnahmen sind jedoch mit erheblichem Verwaltungsaufwand, zusätzlichen unabhängigen Infrastrukturen und anderen Problemen verbunden, oder sie sind in heutigen Infrastrukturen schlicht nicht umsetzbar. Eng verknüpft mit dem Gültigkeitserhalt elektronischer Signaturen ist darüber hinaus das verwendete Modell zur Gültigkeitsprüfung. Gesetzlich wird hier das sogenannte Kettenmodell gefordert, welches bei aktuellen Implementierungen ebenfalls nicht ohne Zeitstempel und dem damit verbundenen Aufwand auskommt. Für Details sei hier auf Kapitel 4 verwiesen.

2.3 Zielsetzung

Das Ziel dieses Projekts ist die Entwicklung von Konzepten und Methoden, welche die Probleme beim Umgang mit qualifizierten elektronischen Signaturen lösen. Wichtig ist hierbei, dass dies unter Vereinfachung und Verschlankung der Prozesse und unter Benutzung vorhandener, standardisierter Infrastrukturen gelöst wird.

2.4 Lösungsansatz

Das genannte Ziel wird durch folgende Maßnahmen erreicht:

1. Benutzung von Signaturverfahren, deren Sicherheit mit schwächeren Sicherheitsannahmen gewährleistet ist. Dadurch wird die Häufigkeit von Übersignaturen sowie die Wahrscheinlichkeit einer unvorhergesehenen Kompromittierung reduziert.

2. Verwendung von Signaturverfahren, die selbst nach Bruch einzelner Sicherheitsannahmen die Sicherheit des Verfahrens gewährleisten und zusätzlich eine große Zahl möglicher Alternativen bieten.

3. Einsatz von Verfahren, die zusätzliche Zeitstempel unnötig machen. Das wird dadurch erreicht, dass der Erstellungszeitpunkt unfälschbar aus der Signatur abgelesen werden kann.

Konkret werden in diesem Projekt hash-basierte Signaturverfahren eingesetzt. Besonders attraktiv ist das bezüglich der Effizienz optimierte Generalized Merkle Signature Scheme (GMSS, vgl. [5]), bzw. dessen Weiterentwicklung eXtended Merkle Signature Scheme (XMSS, vgl [1]). Hash-basierte Signaturverfahren sind theoretisch sehr gut untersucht und entwickelt. Im Gegensatz zu gängigen Signaturverfahren basiert die Sicherheit hash-basierter Verfahren nicht auf zahlentheoretischen Annahmen, sondern auf der Sicherheit von Hashfunktionen. Jede sichere Hashfunktion liefert eine sichere Instanz von XMSS, welche unabhängig von allen anderen Instanzen ist. Damit steht eine große Zahl alternativer Verfahren mit reduzierten Sicherheitsannahmen zur Verfügung. Diese Flexibilität bei der Wahl der unterliegenden Sicherheitsannahme ist unter allen existierenden praxistauglichen Signaturverfahren einzigartig. Die existierenden Ansätze wurden bisher jedoch nicht bis in die Praxis vorangetrieben.

Durch die Abwesenheit weiterer Sicherheitsannahmen, die bei gängigen Signaturverfahren zusätzlich vorhanden sind, wird außerdem die Angriffsfläche für potentielle Angreifer minimiert. Der Zeitpunkt für eine notwendige Übersignatur wird allein aus dem Sicherheitsstatus der Hashfunktion bestimmt. Der Sicherheitsstatus zusätzlicher Annahmen spielt, anders als im Fall der gängigen Signaturverfahren, keine Rolle. Ein weiterer positiver Aspekt daran ist, dass das Sicherheitsniveau von Hashfunktionen im Laufe der Zeit wesentlich langsamer abnimmt

als das Sicherheitsniveau der zusätzlichen Annahmen gängiger Signaturverfahren. Durch den Einsatz hash-basierter Signaturen werden Übersignaturen seltener notwendig.

Auch Hashfunktionen können jederzeit durch neuartige Angriffsmethoden unsicher werden. Der Schutz gegen Sicherheitsverlust durch Bruch einzelner Sicherheitsannahmen wird durch den Einsatz von Hash-Combinern erreicht. Damit können beliebig viele Hashfunktionen kombiniert werden. Somit ist eine beliebig große Redundanz bzw. ein beliebig großer Sicherheitspuffer erreichbar. Dieses Projekt erarbeitet Empfehlungen bezüglich der Hash-Combiner, welche in XMSS zur Erreichung der entsprechenden Sicherheitsgarantien eingesetzt werden können.

Durch die Tatsache, dass jede XMSS Signatur einen fortlaufenden Index trägt, der auch nach Kompromittierung des Schlüssels unverfälschbar bleibt, kann jede Signatur eindeutig ihrem Erstellungsdatum zugeordnet werden. Hierzu führt XMSS einen internen Zustand, der sich mit jeder Signatur ändert. Das hat oben genannte Vorteile, birgt aber auch Herausforderungen für den praktischen Einsatz. Der Zustand muss korrekt geändert und gespeichert werden, um die Sicherheit des Verfahrens zu gewährleisten. Dieses Projekt erstellt Konzepte mit Anleitungen für den korrekten Einsatz von XMSS in der Praxis. Darüber hinaus wird gezeigt, wie vorwärtssichere Signaturverfahren eine PKI insgesamt sicherer machen können und gegen den Zusammenbruch aufgrund von Schlüsselkompromittierung von ZDA Schlüsseln schützen.

Die Beiträge dieses Projektes bieten Ansatzpunkte für eine Vereinfachung der Handhabe qualifizierter elektronischer Signaturen. Somit wird deren flächendeckender Einsatz, sowohl im eGovernment als auch im geschäftlichen oder privaten Umfeld, ermöglicht. Gleichzeitig dient das Projekt als Wegbereiter für einen entsprechenden Standardisierungsprozess und als Vorbereitung für hardwaregestützte Implementierungen des Verfahrens, z.B. auf Smartcards.

3 Grundlagen

Wir erläutern zunächst die relevanten Grundlagen. Nach einer Einführung in Public Key Infrastrukturen und die hier relevanten Konzepte erläutern wir Zeitstempeldienste und die damit verbundenen Nachteile. Anschließend geben wir einen Überblick über vorwärtssichere Signaturverfahren.

3.1 Public Key Infrastrukturen

Durch Zertifizierungsdiensteanbieter (ZDA) aufgespannte, hierarchische Public Key Infrastrukturen (PKI) sind eines der wichtigsten Konzepte im Bereich elektronischer Datensicherheit. Eine Public Key Infrastruktur unterstützt den Einsatz digitaler Signaturen durch das Bereitstellen von Zertifikaten [6]. Diese Zertifikate werden von den ZDAs ausgestellt und dienen zur Identifizierung des Inhabers eines privaten Schlüssels und somit zur Identifizierung des Urhebers einer digitalen Signatur. Die Echtheit der Zertifikate wird dabei durch die Signatur des ZDAs garantiert.

PKIs werden unter anderem bei der Authentifizierung mittels Transport Layer Security (TLS) [7], zur Absicherung von eMails [8] und bei qualifizierten elektronischen Signaturen (QES, vgl. 1999/93/EC [9]) verwendet. Dies macht PKIs zu einem kritischen Stützpfeiler der heutigen Infrastruktur für Internetanwendungen in allen Bereichen, von privater Nutzung über eCommerce bis hin zum eGovernment.

Bei hierarchischen PKIs gibt es im Allgemeinen einen sogenannten Wurzel-ZDA. Dieser dient als Vertrauensanker. Das heißt, der öffentliche Schlüssel des Wurzel-ZDAs muss jedem Teilnehmer bekannt sein und wird häufig in einem selbst signierten Zertifikat bereitgestellt. Der Wurzel-ZDA stellt Zertifikate für untergeordnete ZDAs (Sub-ZDA) aus. Die Sub-ZDAs stellen ihrerseits Zertifikate für untergeordnete ZDAs oder für End-Entitäten aus. End-Entitäten können die zertifizierten Schlüssel zum elektronischen Signieren, zur Authentisierung oder zur Verschlüsselung einsetzen. Eine solche hierarchische PKI ist beispielhaft in Abbildung 3.1 dargestellt.

3.1.1 Zertifikate

Die Bindung zwischen öffentlichem Schlüssel und dessen Inhaber ist entweder sichergestellt bis der Gültigkeitszeitraum eines Zertifikates endet, oder das jeweilige Zertifikat revoziert wird. Der Gültigkeitszeitraum eines Zertifikats wird über ein Anfangs- (*NotBefore*) und ein Enddatum (*NotAfter*) im Zertifikat festgelegt. Weitere Details zu X.509 Zertifikaten sind in RFC 5280 [6] zu finden.

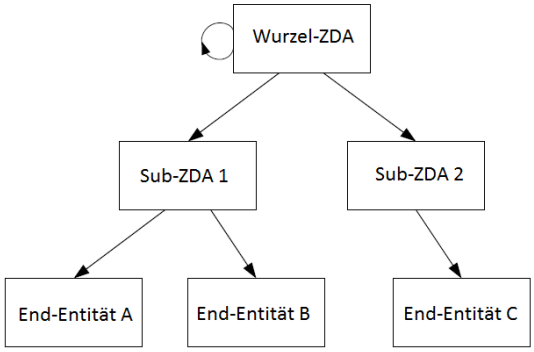


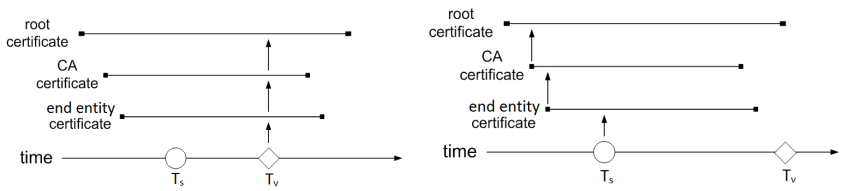
Abbildung 3.1: Hierarchische PKI

3.1.2 Revokation

Revokation bedeutet, dass ein Zertifikat innerhalb des Gültigkeitszeitraumes für ungültig erklärt wird. Dies ist beispielsweise bei Kompromittierung des Schlüssels, bei organisatorischen Veränderungen oder beispielsweise bei Namensänderungen des Schlüsselinhabers notwendig. Die Revokation wird durch den ZDA durchgeführt, der das Zertifikat ausgestellt hat. In der Praxis werden hierfür hauptsächlich Zertifikatsrevokationslisten (CRL) [6] und das Online Certificate Status Protocol (OCSP) [10] eingesetzt. Revokation ist ein Mechanismus, der es ermöglicht, das Gesamtsystem nach einen Schadens- bzw. Fehlerfall wieder in einen sicheren Zustand zu überführen.

3.1.3 Gültigkeitsprüfung

Um eine Signatur auf Gültigkeit zu prüfen, wird zunächst die mathematische Korrektheit der Signatur überprüft. Dabei wird getestet, ob eine Signatur tatsächlich mit einem bestimmten Schlüssel erstellt wurde. Anschließend wird geprüft, ob die Bindung zwischen Schlüssel und dessen Inhaber intakt ist. Hierfür überprüft der Verifizierende die Zertifikatskette vom Zertifikat des Signaturerstellers bis hin zum Wurzel-ZDA. Diesem muss der Verifizierende vertrauen. Sind beide Prüfungen erfolgreich, so wird eine Signatur als gültig angesehen.



(a) Schalenmodell (b) Kettenmodell

Abbildung 3.2: Gültigkeitsmodelle

Das Schalenmodell (Abb. 3.2a) gemäß RFC 5280 [6] ist der aktuelle Internetstandard zur Gültigkeitsprüfung von Zertifikatsketten. In diesem Modell wird die Gültigkeit der zu prüfenden Signatur und jedes involvierten Zertifikats zum aktuellen Zeitpunkt der Überprüfung T_v betrachtet. Ist eines der involvierten Zertifikate ungültig oder revoziert, so ist die zu prüfende Signatur ungültig. Die Überprüfung ist also unabhängig vom Erstellungszeitpunkt T_s der zu prüfenden Signatur.

Wird nun eines der Zertifikate revoziert oder endet dessen Gültigkeitszeitraum, so werden alle Zertifikate, die dieses in Ihrer Vertrauenskette haben, ungültig. Jede Authentifizierung mit einem solchen Zertifikat schlägt fehl und alle erstellten Signaturen werden ungültig. Dies widerspricht jedoch dem Ziel die Gültigkeit qualifizierter elektronischer Signaturen langfristig zu erhalten, vorausgesetzt sie wurden gültig (d.h. während dem Gültigkeitszeitraum des entsprechenden Zertifikats vom rechtmäßigen Schlüsselinhaber) erstellt.

Der Einsatz des Kettenmodells (Abb. 3.2b) verhindert das obige Szenario. Bei einer Revokation werden nur Signaturen und Zertifikate ungültig, die nach der Revokation mit dem entsprechenden Schlüsselpaar erzeugt wurden. Im Kettenmodell wird für die zu prüfende Signatur, bzw. jedes involvierte Zertifikat geprüft, ob es zum jeweiligen Erstellungszeitpunkt gültig war. Die Überprüfung ist also abhängig vom Erstellungszeitpunkt T_s der zu prüfenden Signatur sowie den Erstellungszeitpunkten der Zertifikate, jedoch unabhängig vom Überprüfungszeitpunkt T_v. Zur sicheren Umsetzung des Kettenmodells wird eine chronologische Ordnung von Signaturen und Revokationen benötigt. Bisherige Lösungen basieren auf Zeitstempeln durch vertrauenswürdige Dritte gemäß RFC 3161 [3]. Die Nachteile und der durch Zeitstempel entstehende Mehraufwand (vgl. Abschnitt 3.2) haben bisher einen breiten Einsatz des Kettenmodells verhindert. Für QES ist das Kettenmodell jedoch gesetzlich vorgeschrieben (siehe Kapitel 4). Das macht die Verwendung von QES kompliziert und aufwendig.

3.2 Zeitstempeldienste

Neben der Ausstellung von Zertifikaten können PKIs die Bindung von Datenobjekten an einen bestimmten (Erstellungs-) Zeitpunkt ermöglichen. Zeitstempeldiensteanbieter (TSA) erstellen hierfür Zeitstempel gemäß RFC 3161 [3] für ein gegebenes Datenobjekt. Dies kann beispielsweise ein Zertifikat oder eine elektronische Signatur sein. Ein Zeitstempel wird erzeugt, indem das Datenobjekt und eine vertrauenswürdige Zeitangabe zusammen durch den TSA mit dessen privatem Schlüssel signiert wird. Eine Zeitstempelinfrastruktur ist dabei gleichermaßen aufgebaut wie eine normale PKI. Oft werden Zeitstempeldienste auch direkt von ZDAs angeboten. Wie bereits erwähnt sind Zeitstempel ein häufig eingesetztes Mittel, um eine PKI gegen verschiedene Bedrohungen abzusichern. Hier ergeben sich jedoch eine Reihe von Nachteilen.

Zunächst sind Zeitstempel auch nur Signaturen. Daher gelten für diese die selben Bedrohungen wie für Signaturen an sich. Um einen echten Mehrwert zu erreichen, ist es daher erforderlich, dass die Signatur und der Zeitstempel strikt unabhängig voneinander sind. Damit wird Redundanz erzeugt und somit ein gleichzeitiges ungültig werden verhindert. Das bedeutet jedoch, dass neben der Verwendung eines zweiten, unabhängigen Signaturverfahrens auch eine separate Zeitstempelinfrastruktur parallel zur Zertifizierungsinfrastruktur betrieben werden muss.

Außerdem bedeuten Zeitstempel einen deutlichen Performanzoverhead. Zum einen muss bei der Signaturerstellung zusätzlich ein Zeitstempel angefordert und erstellt werden, was im Allgemeinen eine Onlineverbindung zum TSA erforderlich macht. Zum anderen muss bei der Verifikation zusätzlich zur eigentlichen Signatur auch die Zeitstempelsignatur verifiziert werden. Dies führt mindestens zu einer Verdopplung des Aufwands, da die Verifikation des Zeitstempels mit einer eigenen Zertifikatskette verbunden ist. Zusätzlich steigt die Signatur- und Zertifikatsgröße um die Größe des jeweiligen Zeitstempels.

3.3 Vorwärtssichere Signaturverfahren

Die Idee vorwärtssicherer Signaturverfahren (FSS) wurde von Anderson auf der Eurocrypt 97 [11] vorgestellt und später von Bellare und Miner formalisiert [12]. Zusammengefasst bedeutet Vorwärtssicherheit, dass auch nach einer Schlüsselkompromittierung alle zuvor erstellten Signaturen gültig bleiben.

Bei FSS wird die Gültigkeitsdauer eines Schlüsselpaares in mehrere Perioden, bspw. T, aufgeteilt. Die Aufteilung wird im Zertifikat festgelegt. Diese Perioden können auf unterschiedliche Arten definiert sein. Entweder basierend auf Zeit, sodass bspw. jede Periode einen Tag lang dauert oder basierend auf der Anzahl der erstellten Signaturen, sodass bspw. eine Periode nach jeweils einer, zehn, hundert usw. Signaturerstellungen endet. Generell ist aber jede erdenkliche Einteilung möglich, welche im jeweiligen Szenario zweckmäßig erscheint. Die Länge der einzelnen Perioden kann sich dabei durchaus unterscheiden.

Wie bei traditionellen Signaturverfahren gibt es für jedes Schlüsselpaar einen öffentlichen Schlüssel (pk) für den gesamten Gültigkeitszeitraum. Bei einem FSS existieren jedoch viele geheime Schlüssel $sk_1, \ldots, sk_T$, sodass jeder Periode genau ein Schlüssel zugeordnet ist. Das ist in Abbildung 3.3 dargestellt.

Zur Schlüsselerzeugung wird bei einem FSS zusätzlich die Anzahl der Perioden T benötigt. Um eine Nachricht zu signieren, wird dann jeweils der aktuelle private Schlüssel verwendet. Die Signatur enthält den Index des verwendeten Schlüssels bzw. der Periode. Dieser Index fließt wiederum in die Gültigkeitsprüfung der Signatur ein. Eine Signatur ist gültig, wenn sie unter dem öffentlichen Schlüssel und dem angegebenen Index als gültig ausgewertet wird. Am Ende

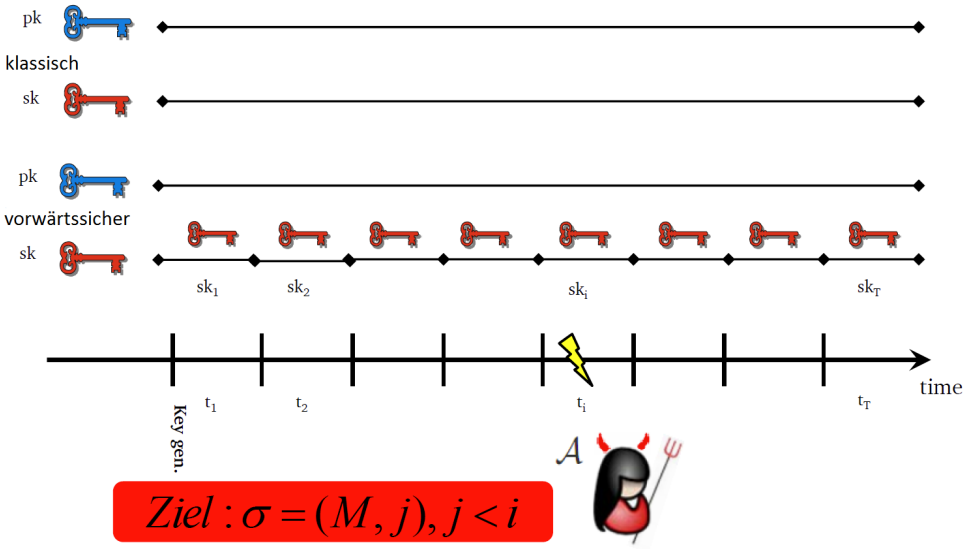


Abbildung 3.3: vorwärtssicheres Signaturverfahren

einer jeden Periode wird der private Schlüssel für die Folgeperiode mittels eines Updatealgorithmus erneuert. Dieser wird entweder manuell durch den Nutzer oder automatisch durch das Verfahren aufgerufen.

Ein FSS bietet zusätzlich zur Vorwärtssicherheit die selben Sicherheitsgarantien wie herkömmliche Verfahren. Selbst wenn ein Angreifer den Nutzer dazu bringen kann Nachrichten zu signieren, kann er trotzdem selbständig keine Signaturen fälschen. Die Vorwärtssicherheit gibt darüber hinaus die Garantie, dass ein Angreifer, welcher den aktuellen privaten Schlüssel sk_i der Periode i lernt, keine Signaturen für Perioden $j < i$ erstellen kann. Für formale Definitionen sei hier auf [12] verwiesen.

3.3.1 FSS und Revokation

Die Vorwärtssicherheit erlaubt eine feingranulare Handhabe von Revokation. Wird beispielsweise ein Schlüssel kompromittiert, garantiert die Vorwärtssicherheit, dass vor der Kompromittierung erstellte Signaturen vom rechtmäßigen Schlüsselinhaber erstellt wurden. Daher müssen diese nicht als ungültig angesehen werden. Da jede Signatur den Index der jeweiligen Periode enthält, kann ein Schlüssel einfach ab einer bestimmten Periode durch Bekanntgabe des entsprechenden Index revoziert werden. Dadurch wird ein Zertifikat nicht als prinzipiell ungültig gehandhabt, sondern nur für jene Perioden ab der Periode der Kompromittierung.

Ist also der Index c der Periode bekannt, in der die Kompromittierung passiert ist, so beinhaltet die Revokation diesen Index c. Signaturen mit Index i bleiben damit gültig, wenn $i < c$ gilt und sind ungültig für $i \geq c$.

3.3.2 Schlüssel Update und Sicherheitskopien von Signaturschlüsseln

In heutigen Publik Key Infrastrukturen werden im allgemeinen Sicherheitskopien von Signaturschlüsseln angelegt. Dadurch wird der Verlust der Schlüssel, beispielsweise durch technische Fehlfunktionen, verhindert. Damit kann vermieden werden, dass im Verlustfall ein neuer Schlüssel erstellt, zertifiziert und der öffentliche Schlüssel verteilt werden muss. Bei FSS ist eine solche Sicherheitskopie jedoch mit einigen Sicherheitsrisiken behaftet. Die Vorwärtssicherheit von FSS ist nur garantiert, wenn der aktuelle Schlüssel am Ende der jeweiligen Periode durch den neuen Schlüssel ersetzt wird. Der vorhergehende Schlüssel muss jeweils unwiderruflich gelöscht werden. Eine Sicherheitskopie kann daher die Vorwärtssicherheit gefährden, wenn das Update dieser Kopie nicht synchron durchgeführt wird. Ein Angreifer, dem eine nicht aktualisierte Sicherheitskopie in die Hände fällt, könnte Signaturen vorhergehender Perioden fälschen. Daher ist zu empfehlen, auf Sicherheitskopien der privaten Schlüssel zu verzichten. Andernfalls muss mittels organisatorischer Maßnahmen sichergestellt werden, dass ein synchrones Update der verwendeten Schlüssel und der Sicherheitskopien erfolgt bzw. in den Zeitspannen zwischen den Updates der Sicherheitskopie eine Kompromittierung ausgeschlossen ist. Bei einem ZDA ist dies beispielsweise denkbar, wenn die Sicherheitskopie offline in einem Hardware Sicherheitsmodul (HSM) gespeichert ist und das Update täglich durchgeführt und nach dem vier (oder mehr) Augen Prinzip überwacht wird.

Abgesehen vom Update der Sicherheitskopie muss auch das Update des aktiven Schlüssels immer rechtzeitig und korrekt durchgeführt werden. Bei dem von uns eingesetzten Signaturverfahren XMSS (siehe auch 5.3) ist das Schlüsselupdate in den Signaturalgorithmus integriert und wird somit automatisiert aufgerufen. Erfolgt die Signaturerstellung auf einer sicheren Signaturerstellungseinheit wie einer Smartcard oder einem HSM, so ist die Aktualität des Schlüssels garantiert. Die vor allem für End-Entitäten interessante Smartcard Implementierung wurde von uns bereits erstellt [13]. Eine für beispielsweise ZDAs relevante HSM Implementierung ist derzeit in Arbeit.

4 Bedrohungs- und Anforderungsanalyse

Im Folgenden werden zunächst die verschiedenen Bedrohungen im Umfeld von PKI und elektronischen Signaturen beschrieben. Danach werden aus den Bedrohungen und den gesetzlichen Vorgaben die entsprechenden Anforderungen an die zu entwickelnde PKI abgeleitet.

4.1 Bedrohungsanalyse und aktuelle Gegenmaßnahmen

Im Umfeld von Public Key Infrastrukturen bestehen verschiedene Bedrohungen, welche die Sicherheit der auf Basis dieser PKI erstellten Signaturen gefährden und deren Verfügbarkeit beeinflussen. Diese Bedrohungen werden hier aufgelistet und beschrieben. Außerdem werden die aktuell verwendeten Gegenmaßnahmen aufgezeigt und bewertet.

4.1.1 Kontinuierlicher technologischer Fortschritt

Durch Fortschritte in der Kryptoanalyse und steigende Verfügbarkeit von Rechenkraft werden potentielle Angreifer über die Zeit stärker. Damit werden alle heute bekannten Signaturverfahren über die Zeit unsicherer. Die Sicherheit von Signaturverfahren bzw. der Aufwand für einen Angreifer hängt vom verwendeten Sicherheitsparameter (oft die Schlüssellänge) ab. Der technologische Fortschritt impliziert, dass jede Wahl eines Sicherheitsparameters für ein Signaturverfahren nach einer gewissen Zeit unsicher wird. Das heißt, es wird einem Angreifer möglich entweder den Schlüssel zu gewinnen und zu verwenden, oder aber Signaturen ohne Kenntnis des Schlüssels zu fälschen.

Ab diesem Zeitpunkt sind alle mit diesem Signaturverfahren erstellten Signaturen als ungültig zu betrachten, da wie bei der Schlüsselkompromittierung nicht zweifelsfrei festgestellt werden kann, ob die betroffene Signatur vom Eigentümer des Schlüsselpaares oder von einem Angreifer erstellt wurde. Daher müssen die Sicherheitsparameter regelmäßig an den aktuellen Stand der Technik angepasst werden (vgl. [14]).

Ist abzusehen, dass ein für bereits erstellte Signaturen verwendeter Sicherheitsparameter unsicher wird, so ist das aktuelle Vorgehen, alle damit signierten Dokumente überzusignieren (vgl. [2]). Das bedeutet, dass das Dokument mitsamt der noch gültigen alten Signatur mit einem als länger sicher geltenden Verfahren erneut signiert werden muss. Hierbei wird entweder ein anderes Signaturverfahren oder das bisherige Verfahren mit einem neuen Sicherheitsparameter, wie zum Beispiel einer größeren Schlüssellänge, verwendet.

Dies erfordert jedoch den Betrieb einer zusätzlichen vertrauenswürdigen Infrastruktur und bringt den entsprechenden Aufwand mit sich, wie bereits für Zeitstempel beschrieben. Außer-

dem müssen die Gefährdungen für die jeweils eingesetzten Verfahren immer rechtzeitig und korrekt eingeschätzt werden. Dies kann nicht garantiert werden. Ein Fehler bei der Einschätzung hat jedoch fatale Folgen. Sobald der Gültigkeitsverlust eingetreten ist, besteht keine Möglichkeit die Gültigkeit wiederherzustellen. Daher muss sehr pessimistisch vorgegangen werden. Dies führt dazu, dass der Aufwand durch die entsprechenden Maßnahmen häufiger entsteht, als der technologische Fortschritt dies tatsächlich erfordern würde.

4.1.2 Unvorhergesehener technologischer Fortschritt

Ein weiteres Problem stellen unvorhergesehene Entwicklungen dar. Beispiele sind ein plötzlicher Durchbruch in der Kryptoanalyse oder beim Bau von Quantencomputern. In diesem Fall werden entweder Sicherheitsparameter viel schneller unsicher als erwartet, oder im schlimmsten Fall wird das Signaturverfahren als solches unsicher. Während im Fall des kontinuierlichen technologischen Fortschritts in der Regel ausreichend Zeit bleibt um Gegenmaßnahmen zu ergreifen, ist dies bei unerwarteten Entwicklungen nicht gegeben. Vorhandene Signaturen können mit einem Schlag unsicher werden und es bleibt keine Zeit sie rechtzeitig überzusignieren. Dann sind die erstellten Signaturen ein für alle Mal wertlos. Dafür genügt es bereits, dass sich eine einzige, dem Signaturverfahren zugrundeliegende Sicherheitsannahme als nicht länger haltbar erweist.

Der aktuelle Lösungsvorschlag für dieses Problem ist die Verwendung sogenannter Mehrfachsignaturen (vgl. [4]). Dabei werden die Dokumente parallel mit verschiedenen Signaturverfahren signiert, deren Sicherheit auf gänzlich unterschiedlichen Sicherheitsannahmen beruht. Allerdings ist das technisch in den gegenwärtigen Infrastrukturen nicht realisierbar. Dieses Problem ist also aktuell in der Praxis ungelöst. Zudem gibt es derzeit nur eine sehr geringe Auswahl an eingesetzten Signaturverfahren, die alle auf sehr ähnlichen Sicherheitsannahmen beruhen. Mit jeder nicht länger haltbar gewordenen oder gebrochenen Annahme reduzieren sich die möglichen Alternativen signifikant. Quantencomputer beispielsweise machen alle heutzutage in der Praxis eingesetzten Signaturverfahren unsicher und damit unbrauchbar.

4.1.3 Schlüsselkompromittierung

Schlüsselkompromittierung bedeutet, dass der private Schlüssel eines Schlüsselpaares einem Angreifer zur Verfügung steht. Hier kann zwischen Zugriff auf den Schlüssel und vollständige Kompromittierung unterschieden werden. Im ersten Fall hat ein Angreifer zeitweise Zugriff auf die Signaturerstellungseinheit, auf welcher der private Schlüssel gespeichert ist (bspw. Zugriff auf das Hardwaresicherheitsmodul eines ZDA oder Zugriff auf die Signaturkarte einer End-Entität durch Verlust oder Diebstahl). Der Angreifer kann solange der Zugriff besteht Signatu-

ren für beliebige Dokumente erstellen. Im Fall der vollständigen Kompromittierung lernt der Angreifer den privaten Schlüssel und kann diesen uneingeschränkt verwenden. In beiden Fällen bedeutet das, dass die Gültigkeit der mit dem kompromittierten Schlüssel erstellten Signaturen nicht mehr zweifelsfrei festgestellt werden kann. Da es bei klassischen Signaturverfahren nicht nachvollziehbar ist, ob eine Signatur vor oder nach der Kompromittierung erstellt wurde, kann keine Signatur zweifelsfrei dem Schlüsselinhaber bzw. dem Angreifer zugeordnet werden.

Darüber hinaus ist im Umfeld von hierarchischen PKIs zu unterscheiden, ob ein ZDA Schlüssel oder der Schlüssel einer End-Entität kompromittiert wurde. Beim Schlüssel einer End-Entität sind durch die Kompromittierung lediglich die eigens erstellten Signaturen betroffen. Dahingegen sind bei ZDA Schlüsseln sämtliche untergeordneten Zertifikate betroffen und damit alle auf diesen Zertifikaten beruhenden Signaturen. Im Falle der Kompromittierung eines Wurzel-ZDAs kann dies zu einem Zusammenbruch der gesamten PKI führen (vgl. [15]).

Die aktuelle Lösung dieses Problems ist das Aufbringen eines Zeitstempels (vgl. [3]) auf die signierten Dokumente. Damit gehen jedoch die bereits beschriebenen Nachteile von Zeitstempeln einher. Zudem beruhen Zeitstempel selbst auf digitalen Signaturen und verschieben damit das Problem nur von den Dokumentensignaturen auf die Zeitstempelsignaturen. Für Zeitstempeldienste bestehen die selben Gefährdungen wie für ZDAs. Betrachtet man also die Kompromittierung eines Zeitstempeldienstes, so werden alle mit dem betroffenen Schlüssel erstellten Zeitstempel ungültig.

Im Falle von ZDAs werden im allgemeinen Logdateien geführt, um Signaturen bzw. die Erstellung von Zertifikaten zeitlich zu verankern und nachvollziehbar zu machen. Wird jedoch in die Server eines ZDA eingebrochen, so kann nicht in jedem Fall garantiert werden, dass die Logdateien nicht ebenso vom Angreifer manipuliert wurden, um den Einbruch zu verschleiern. Dies war beispielsweise bei der Kompromittierung von DigiNotar der Fall [16].

4.1.4 Schlüsselverlust

Bei Schlüsselverlust ist ein zur Signatur verwendeter privater Schlüssel für den Eigentümer nicht mehr zugreifbar. Dies kann durch (un-)beabsichtigte Löschung des Schlüssels, Hardwaredefekte, Verlust der Signaturkarte oder ähnliches entstehen. Zunächst bedeutet dies lediglich, dass keine neuen Signaturen erstellt werden können. Bisherige Signaturen sind jedoch nicht betroffen. Im Falle von ZDA Schlüsseln bedeutet dies bspw. keine Neuausstellung von Zertifikaten. Problematischer wird dieser Fall, wenn nicht absolut sichergestellt werden kann, dass der Schlüssel tatsächlich vernichtet wurde und möglicherweise einem Angreifer in die Hände fallen kann (bspw. Verlust der Signaturkarte). Kann dies nicht zweifelsfrei festgestellt werden, ist aus Sicherheitsgründen von einer Kompromittierung des Schlüssels und den damit verbundenen Folgen auszugehen.

4.1.5 Signaturabstreitung

Signaturen, die vom Schlüsselinhaber erzeugt wurden, sind ununterscheidbar von Signaturen von einem Angreifer, der den Schlüssel kompromittiert hat. Deswegen kann ein Schlüsselinhaber prinzipiell Signaturen abstreiten, indem er behauptet, sein Schlüssel sei kompromittiert worden.

Die heutige Lösung ist hier ebenfalls das Aufbringen von Zeitstempeln und das Unterbinden von zurückdatierten Revokationen. Eine Signatur wird dann als gültig betrachtet, wenn diese einen Zeitstempel mit einem Datum vor der Veröffentlichung der entsprechenden Revokation trägt. Das verhindert zwar das Abstreiten erstellter Siganturen, ist jedoch mit den bereits genannten Nachteilen von Zeitstempeln verbunden. Darüber hinaus ist fraglich, ob ein Schlüsselinhaber immer in der Lage ist, eine Kompromittierung rechtzeitig zu bemerken und die Revokation auszulösen.

4.2 Anforderungsanalyse

Um qualifizierte elektronische Signaturen sicher und effizient umsetzen zu können, ergeben sich aus den beschrieben Bedrohungen konkrete Anforderungen. Daneben bestehen noch gesetzliche Anforderungen, die speziell im Falle von QES erfüllt werden müssen.

4.2.1 Formale Anforderungen

QES - gesetzliche Vorgaben

In Deutschland sind QES in Gesetzten und Verordnungen verankert (vgl. dazu [4]). Die Rahmenbedingungen für elektronische Signaturen wurden in Deutschland im ”Gesetz zur digitalen Signatur” (SigG) [17] – kurz: Signaturgesetz – und der dazugehörigen Signaturverordnung (SigV) 1997 festgelegt. Diese wurden 2001 mit dem neuen Signaturgesetz [18] und einer neuen Signaturverordnung [19] novelliert. Damit wurde die EU Richtlinie ”über gemeinschaftliche Rahmenbedingungen für elektronische Signaturen” [9] in nationalem Recht umgesetzt. Elektronische Signaturen gibt es demnach in drei verschiedenen Ausprägungen:
Gemäß [18, §2 Begriffsbestimmungen] gilt:
”Im Sinne dieses Gesetzes sind

1. ‘elektronische Signaturen’ Daten in elektronischer Form, die anderen elektronischen Daten beigefügt oder logisch mit ihnen verknüpft sind und die zur Authentifizierung dienen,

2. ‘fortgeschrittene elektronische Signaturen’ elektronische Signaturen nach Nummer l, die

   a) ausschließlich dem Signaturschlüsselinhaber zugeordnet sind,

b) die Identifizierung des Signaturschlüsselinhabers ermöglichen,

c) mit Mitteln erzeugt werden, die der Signaturschlüsselinhaber unter seiner alleinigen Kontrolle halten kann, und

d) mit den Daten, auf die sie sich beziehen, so verknüpft sind, dass eine nachträgliche Veränderung der Daten erkannt werden kann,

3. ‘qualifizierte elektronische Signaturen’ elektronische Signaturen nach Nummer 2, die

a) auf einem zum Zeitpunkt ihrer Erzeugung gültigen qualifizierten Zertifikat beruhen und

b) mit einer sicheren Signaturerstellungseinheit erzeugt werden. [...]”

”Qualifizierte elektronische Signaturen” erfordern also das Kettenmodell zur Gültigkeitsprüfung und eine sichere Signaturerstellungseinheit für die Signaturerzeugung. Hinweise wie eine Sichere Signaturerstellungseinheit zu realisieren ist, gehen aus der Siganturverordnung [19] hervor: Es muss ”die Korrektheit der digitalen Signatur zuverlässig geprüft und zutreffend angezeigt” werden. Eine sichere Signaturerstellungseinheit kann also beispielsweise eine Smartcard, welche den privaten Schlüssel enthält, in Verbindung mit einem entsprechenden Kartenleser sein.

Die Gleichstellung von QES und handschriftlichen Unterschriften ist schließlich durch weitere Gesetze erreicht worden [4].

Langfristige Gültigkeit

Die Gültigkeit von elektronischen Signaturen hängt vom angewendeten Gültigkeitsmodell ab. Da Zertifikate zwangsläufig ein Ablaufdatum haben, werden Signaturen nach dem Standard Gültigkeitsmodell – dem Schalenmodell – per se ab diesem Zeitpunkt ungültig (siehe Abschnitt 3.1). Da qualifizierte Signaturen als Äquivalent zur handschriftlichen Unterschrift jedoch für immer gültig bleiben müssen, wird hier ein anderes Gültigkeitsmodell – das Kettenmodell – gesetzlich gefordert (siehe Abschnitt 4.2.1). Die Erfordernis der langfristigen Gültigkeit ergibt sich auch direkt aus der Betrachtung von Anwendungsszenarien für rechtsgültige Signaturen, wie beispielsweise Vertrags- oder Testamentunterzeichnung. Hier wird direkt ersichtlich, dass solche Signaturen langfristig auch über mehrere Jahrzehnte hinweg gültig bleiben müssen.

Weiterhin erlaubt das Kettenmodell Signaturen, welche gültig erstellt wurden, auch nach der Revokation eines Zertifikates als gültig zu validieren.

4.2.2 Technische Anforderungen

Revokationsmechanismus

Die Schlüsselkompromittierung kann – selbst bei ZDAs – nicht vollständig ausgeschlossen werden. Eine kürzliche Studie [20] und öffentlich bekannt gewordene Vorfälle, z.B. bei Comodo [21], StartCom [22] und DigiNotar [16], zeigen, dass ZDAs kompromittiert und betrügerische Zertifikate von Angreifern verwendet werden. Die übliche und unvermeidliche Reaktion auf die Entdeckung eines kompromittierten oder gefälschten Zertifikats ist dessen Revokation durch den entsprechenden ZDA.

Gleiches gilt für den Schlüsselverlust. Während bei ZDAs Schlüsselverlust im allgemeinen dessen Zerstörung aufgrund technischer Defekte oder Ähnlichem bedeutet, kann dies bei End-Entitäten durch Verlust der Signaturkarte geschehen. In diesem Fall wird eine Revokation notwendig.

Daraus lässt sich die Anforderung eines funktionierenden und verlässlichen Revokationsmechanismus ableiten. Hiermit wird das System bei Kompromittierung oder Schlüsselverlust wieder in einen sicheren Zustand überführt.

Verhinderung von Signaturabstreitung

Aus der inhärenten Möglichkeit Signaturen abzustreiten ergibt sich die Anforderung eines Verhinderungsmechanismus. Die Abstreitungsmöglichkeit widerspricht dem Grundsatz rechtsgültiger Signaturen. Gerade bei solchen Signaturen kann es Anreize zur Signaturabstreitung durch den Siganturersteller geben, um beispielsweise vorher unterzeichnete Verträge für ungültig zu erklären.

Frühwarnmechanismen

Da wie erwähnt technischer Fortschritt nicht problemlos vorhergesagt werden kann, bzw. unvorhergesehene Entwicklungen auftreten können, gilt es Mechanismen in das System zu integrieren, welche als Frühwarnsystem fungieren können. Damit soll erreicht werden, dass stets ausreichend Zeit vorhanden ist, um die entsprechenden Gegenmaßnahmen einzuleiten. Damit kann zusätzlich unnötiger Aufwand, der aus ungenauen Vorhersagen entsteht, vermieden werden.

Verzicht auf Zeitstempel

Aufgrund der gravierenden Nachteile von Zeitstempeln und Zeitstempeldiensten, soll bei der Lösung weitestgehend auf Zeitstempel verzichtet werden.

Integration in bestehende Infrastrukturen

PKIs stellen ein etabliertes Konzept dar, das sich grundlegend bewährt hat. Der Aufbau komplett neuer Infrastrukturen würde einen erheblichen Implementierungs- und Migrationsaufwand verursachen, der mit hohen Kosten verbunden ist. Das entwickelte Verfahren soll sich daher mit möglichst geringen Änderungen in etablierte Infrastrukturen integrieren lassen.

5 Realisierung

Der Lösungsansatz besteht in der effizienten Umsetzung einer vorwärtssicheren PKI mittels vorwärtssicherer Signaturverfahren (FSS). Hierfür verwenden wir XMSS [1]. Durch die von FSS gebotenen Sicherheitsgarantien kann auf Zeitstempel verzichtet werden, solange das FSS sicher ist. Das Kettenmodell kann damit effizient implementiert werden.

Durch die Auswahl von XMSS sind weitere Mechanismen möglich, welche zusätzlich vor dem unsicher werden des Verfahrens schützen und somit die Notwendigkeit von Zeitstempeln zum Gültigkeitserhalt sehr lange hinauszögern können. So wird der Zusatzaufwand, der mit Zeitstempeln einhergeht, weitestgehend vermieden.

Gemäß den Anforderungen lässt sich unsere Lösung einfach in bestehende PKIs gemäß X.509 [23] integrieren. Die einzelnen Komponenten einer PKI werden nicht verändert. Daher wird hier auf eine genaue Beschreibung verzichtet und es werden lediglich die Unterschiede zu bestehenden PKIs aufgezeigt. Zertifizierungshierarchien, -prozesse und Revokationsmechanismen basierend auf OCSP und CRLs bleiben dabei weitestgehend unverändert. Darüber hinaus zeigen wir, wie die Revokationshandhabung verbessert werden kann und beschreiben ein Verfahren, welches es erlaubt FSS durch End-Entitäten sicher zu handhaben.

5.1 Kettenmodell mit FSS

Hier erläutern wir kurz, wie das Kettenmodell mittels FSS umgesetzt werden kann. Hierbei nutzen wir die Möglichkeit der in Abschnitt 3.3.1 beschriebenen, feingranularen Revokation mittels FSS aus.

Beim herkömmlichen Kettenmodell werden absolute Signaturerstellungszeitpunkte benötigt, wie sie im Falle von Zeitstempeln vorliegen. Zur sicheren Implementierung des Kettenmodells sind diese jedoch nicht zwangsläufig erforderlich. Durch die chronologische Ordnung von Signaturen mittels FSS ergeben sich relative Zeitabschnitte. Daher fließt bei der Gültigkeitsprüfung der Zertifikatskette nur der jeweils verwendete Index mit ein. Da die Revokationsprüfung auch die relativen Zeitabschnitte der FSS verwendet, kann bei der Gültigkeitsprüfung entschieden werden, ob ein Zertifikat zum Signaturzeitpunkt revoziert war. Die Prüfung expliziter Zeitpunkte entfällt in diesem angepassten Modell. Hierdurch wird die Umsetzung effizienter. Gleichzeitig wird auch das Problem Uhren zu synchronisieren gelöst. Angriffe auf die korrekte Gültigkeitsprüfung von Zertifikaten durch manipulierte Systemuhren sind nicht möglich. Wie in Abb. 5.1 zu sehen ist, beeinflusst die Revokation des Wurzel-ZDA-Zertifikats nach dem Index t_2 eine danach erfolgte End-Entitäten Signatur nicht, da das entsprechende ZDA-Zertifikat mit dem Index

t_1 erstellt wurde. Auch das Ende von Gültigkeitszeiträumen der Zertifikate hat keinen Einfluss auf die Gültigkeit von gültig erstellten Signaturen.

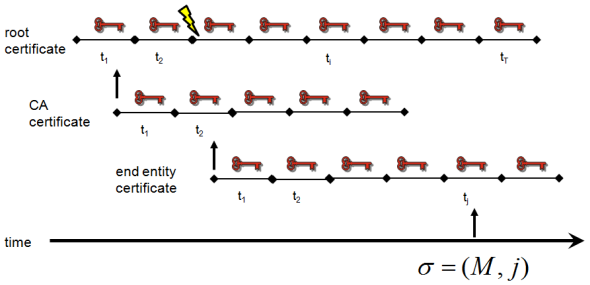


Abbildung 5.1: Kettenmodell mittels FSS

Sollen – bspw. zur Überprüfung von Gültigkeitszeiträumen oder um das Datum einer Vertragsunterzeichnung bereitzustellen – die tatsächlichen Ausstellungszeitpunkte einer Signatur explizit gemacht werden, so ist dies über das Einfügen einer Zeitmarke möglich. Hierfür fügt der Schlüsselinhaber der Nachricht vor der Signaturerstellung eine Zeitangabe hinzu, die den Signaturerstellungszeitpunkt angibt. Eine potenziell gefälschte Signatur wird über den verwendeten, von der Zeitmarke unabhängigen, Signaturindex erkannt. Daher sind Zeitmarken als vertrauenswürdige Angaben des legitimen Schlüsselinhabers zu sehen. Entweder ist der entsprechende Index revoziert oder es ist garantiert, dass die Zeitmarke vom legitimen Schlüsselinhaber erstellt wurde.

ZDAs sind von vornherein als vertrauenswürdig zu sehen. Damit sind auch die von ZDAs erstellten Zeitmarken vertrauenswürdig. Bei End-Entitäten ist dies nicht a priori gegeben. Diese könnten zwar falsche Zeitmarken einfügen, jedoch nur in einem gewissen Rahmen. Anderenfalls käme es zu Unstimmigkeiten mit der chronologischen Ordnung der Signaturen. Wie das Handhaben von FSS und die Vermeidung von falschen Revokationsinformationen durch End-Entitäten realisiert werden kann zeigen wir im Abschnitt 5.4.

Wir gehen davon aus, dass sowohl alle beteiligten ZDAs, als auch die End-Entitäten FSS verwenden. Die Umsetzung des Kettenmodels mittels FSS bedarf somit lediglich den Austausch der von ZDAs und End-Entitäten eingesetzten Signaturverfahren. Daneben muss der Pfad-Prüfalgorithmus aus RFC 5280 leicht angepasst werden. Diese Anpassungen sind jedoch sehr gering, da sich das Kettenmodell bei FSS durch unsere Umsetzung implizit aus der Prüfung der Indizes während der mathematischen Korrektheitsprüfung der Signaturen ergibt.

Eine detaillierte Behandlung und der Vergleich der Gültigkeitsmodelle, sowie formale Definitionen und verschiedene Möglichkeiten das Kettenmodell mit FSS umzusetzen sind in [15] zu finden. In dieser Veröffentlichung beschreiben wir auch die genauen Änderungen am Pfad-Prüfalgorithmus.

5.2 Verbesserte Revokationshandhabe

Da mit jeder FSS Signatur ein chronologisch fortlaufender Index verbunden ist, ist eine sehr schlanke Bereitstellung von Revokationsinformationen durch Publikation des kleinsten, nicht vertrauenswürdigen Index möglich. Auf der Clientseite kann darüber hinaus für jedes installierte ZDA-Zertifikat der größte, als nicht-revoziert bekannte Index (nach einer online Revokationsprüfung) lokal gespeichert werden. Kleinere Indizes gelten damit ebenfalls als nicht-revoziert und müssen nicht erneut geprüft werden.

Da vorherige Signaturen nicht betroffen sind, können Zertifikate bei *begründetem Verdacht* vorsorglich revoziert werden. Im Gegensatz zum aktuellen Vorgehen – erst bei evidenter Kompromittierung zu revozieren – wird so der Gefährdungszeitraum weiter stark reduziert. Nach einer Revokation muss lediglich ein neues Zertifikat ausgestellt werden.

Betrachtet man ZDA-Zertifikate im Speziellen, so ist die Verfügbarkeit der PKI bei Revokation von ZDA-Zertifikaten aufgrund der feingranularen Revokationsmöglichkeit nicht gefährdet. Darüber hinaus entsteht kein Aufwand durch die Erneuerung aller legitim ausgestellten Zertifikate. Anstatt einzelne gefälschte Zertifikate zu revozieren, kann daher bei einem Sicherheitsvorfall das betroffene ZDA-Zertifikat problemlos revoziert werden. Dies macht alle potenziell gefälschten Zertifikate direkt ungültig, auch wenn diese zum Revokationszeitpunkt unbekannt sind.

Hierdurch wird die Sicherheit deutlich erhöht und die Benutzung gefälschter Zertifikate wird wirkungsvoll verhindert. Insbesondere entfällt die genaue Prüfung von Logdateien zur Identifizierung gefälschter Signaturen oder Zertifikate im Falle von ZDAs. Lediglich der Kompromittierungszeitpunkt muss festgestellt werden, was eine sehr schnelle Reaktion ermöglicht. Wie dies für End-Entitäten realisiert werden kann, zeigen wir in Abschnitt 5.4.

5.3 Signaturverfahren

Für die Realisierung des Konzepts wird XMSS [1] eingesetzt. Dabei handelt es sich um eine Weiterentwicklung von GMSS [24]. Es bietet Frühwarnmechanismen und die einfache Kombination verschiedener zugrunde liegender Primitive, die auf unterschiedlichen Sicherheitsannahmen beruhen. Damit wird Redundanz bezüglich der Sicherheitsanforderungen geschaffen.

XMSS ist das erste effiziente FSS, das nur minimale Sicherheitsanforderungen benötigt und beweisbar sicher ist. Es ist trotz der Vorwärtssicherheit so schnell wie RSA und ECDSA und bietet vergleichbare Parametergrößen. Ein Zeitabschnitt endet bei XMSS mit jeder Signatur. Das heißt, nach jeder Signatur ändert sich der geheime Schlüssel. Dieses Schlüsselupdate wird automatisch durch den Signaturalgorithmus ausgeführt.

5.3.1 Sicherheit von XMSS

XMSS ist ein hash-basiertes Signaturverfahren und daher resistent gegen Quantencomputer-gestützte Angriffe. Minimale Sicherheitsanforderungen bedeuten, dass eine sichere Variante von XMSS existiert, solange es überhaupt ein sicheres Signaturverfahren gibt. Es kann mit beliebigen Hashfunktionen realisiert werden. Es ist sicher, solange diese Hashfunktionen den schwachen Sicherheitsanforderungen *Zweites Urbild Resistenz* und **Pseudozufälligkeit** genügen. Mit XMSS kann die Wahrscheinlichkeit eines plötzlichen Bruchs des Signaturverfahrens effizient minimiert werden. Wie dies realisiert werden kann, wird in den folgenden Abschnitten beschrieben.

5.3.2 Frühwarnmechanismus

Kryptographische Hashfunktionen müssen stärkeren Sicherheitsanforderungen als Zweites Urbild Resistenz und Pseudozufälligkeit genügen. Die wichtigste geforderte Eigenschaft ist Kollisionsresistenz. Die Erfahrung mit MD-5 und SHA-1 zeigt, dass für etablierte Hashfunktionen in aller Regel zuerst Angriffe gegen die starken Sicherheitsannahmen gefunden werden. In den bisherigen Fällen ließen sich diese ersten Angriffe nicht auf die schwächeren Sicherheitsannahmen ausdehnen. Ist ein Angriff gegen eine starke Sicherheitsannahme bekannt, so bleibt ausreichend Zeit, die verwendete Hashfunktion zu wechseln. Solche Angriffe auf starke Sicherheitsannahmen können damit als Frühwarnmechanismus gesehen werden.

5.3.3 Redundanz durch Hash-Combiner

Um die Sicherheit vergleichbar zu Mehrfachsignaturen auf verschiedenen mathematischen Problemen zu basieren, können so genannte Hash-Combiner (siehe bspw. [25]) eingesetzt werden. Diese ersetzen die verwendete Familie von Hashfunktionen. Hash-Combiner verwenden je benötigter Eigenschaft zwei Funktionsfamilien, sodass die Kombination die jeweilige Eigenschaft erhält, solange mindestens eine der eingesetzten Funktionsfamilien diese Eigenschaft erfüllt. Für die von XMSS benötigten Eigenschaften – Pseudozufälligkeit und Zweites Urbild Resistenz – gibt es allgemein anerkannte Hash-Combiner (vgl. bspw. [25]).

Zweites Urbild Resistenz

Seien M der Input einer Funktion und K der verwendete Schlüssel. Hat man nun zwei Zweites Urbild resistente Hashfunktionen h_1, h_2, so garantiert der Konkatenations-Combiner $H_{||}^{h_1,h_2}(M) = (h_1(M)||h_2(M))$ die Zweites Urbild Resistenz, solange entweder h_1 oder h_2 diese Eigenschaft besitzt.

Pseudozufälligkeit

Der XOR-Combiner $H_{\oplus}^{f_1,f_2}(M) = (f_1(K,M) \oplus f_2(K,M))$ dagegen garantiert die Pseudozufälligkeit, solange eine der Funktionsfamilien f_1, f_2 pseudozufällig ist.

Integration in XMSS

Da diese Hash-Combiner selbst Hashfunktionen sind, können sie auf einfache Weise in XMSS statt der normalen Hashfunktionen verwendet werden. Dadurch müssen nicht zwei verschiedene Signaturverfahren eingesetzt werden, um die Sicherheit auf zwei verschiedenen mathematischen Problemen aufzubauen. Man muss in diesem Fall lediglich zwei verschiedene (Hash-) Funktionen verwenden, welche auf verschiedenen Problemen bzw. Konstruktionsprinzipien beruhen. Zwar entspricht die Laufzeit von jedem der Hash-Combiner der Summe der Laufzeiten der eingesetzten (Hash-) Funktionen, die Ausgabe des XOR-Combiners hat jedoch die selbe Größe wie die Ausgabe einer einzelnen Hashfunktion. Da diese Ausgabegröße für die Signaturgröße maßgeblich ist, wächst damit die Signaturgröße nur leicht. Zusätzlich sei angemerkt, dass sich Hash-Combiner beliebig schachteln lassen. Somit lassen sich theoretisch beliebig viele Funktionen verknüpfen.

5.4 FSS für End-Entitäten

Während es für ZDAs verhältnismäßig einfach ist den Schlüsselzustand (den aktuell gültigen Signaturindex) nachzuverfolgen, gibt es bei End-Entitäten hier einige Schwierigkeiten zu überwinden. ZDAs können beispielsweise den Index in einer sicheren Logdatei (z.B. als Ausdruck auf Papier) speichern und sind darüber hinaus als vertrauenswürdig anzusehen. Es ist also davon auszugehen, dass ZDAs bei Revokation den korrekten Index benennen.

End-Entitäten haben in der Regel kein solches Log zur Verfügung. Der Index wird zwar auf der Signaturkarte verwaltet, steht jedoch bei Verlust dieser nicht mehr zur Verfügung. Eine einfache Lösung wäre es den Index jeweils anzuzeigen, und zu verlangen, dass die End-Entität sich den Index merkt. Das ist jedoch fehleranfällig und nicht in jedem Szenario umsetzbar. Dazu kommt, dass es keine Kontrollmöglichkeit bei Kartenverlust gibt. Eine nicht vertrauenswürdige End-Entität kann damit nicht vom Widerruf erstellter Signaturen abgehalten werden. Daher schlagen wir organisatorische Maßnahmen vor, um diese Probleme zu lösen.

5.4.1 Online Protokoll

Das Problem, korrekte Revokationsinformationen zu garantieren und einer End-Entität zu ermöglichen den Status seines Schlüssels nachzuverfolgen, lösen wir mit einem Online Protokoll. Hierbei baut die End-Entität nach jedem Signaturvorgang eine Verbindung zum ZDA auf (oder

einem anderen vertrauenswürdigen Dritten, welcher den Revokationsstatus verwaltet) und meldet den neuen Index nach Signaturerstellung. Das Vorgehen kann damit gerechtfertigt werden, dass heutige Computer nahezu immer online sind. Da der ZDA in diesem Fall den Signaturindex überwacht, kann der Schlüsseleigentümer keine erstellten Signaturen widerrufen. Er ist lediglich in der Pflicht, eine Kompromittierung entsprechend zu melden. Ist der Siganturschlüssel auf einer Smartcard gespeichert, welche auch zur Signaturerstellung verwendet werden muss, so entspricht es der gängigen Praxis davon auszugehen, dass diese bspw. bei einem Verlust der Karte den Schlüssel zumindest solange vor Missbrauch schützt, bis der Verlust bemerkt wird.

Nachdem der Index gemeldet wurde, erhält der Signaturersteller ein Token vom ZDA, welcher zur Bestätigung des Loggings dient. Diesen Token nennen wir Gültigkeitstoken. Er dient zusätzlich als Beweis, dass es für den verwendeten Index keine Revokation gibt. Wenn die End-Entität nun ihren Schlüssel revozieren muss, meldet sie das lediglich dem ZDA. Dieser prüft das Log für den letzten verwendeten Index und revoziert das Zertifikat von diesem Index an, bzw. stellt ab diesem Zeitpunkt keine Gültigkeitstoken für höherwertige Indizes aus.

Abbildung 5.2 zeigt das Protokoll. In Schritt (a) signiert der Schlüsselinhaber Dokument m mit seinem Schlüssel. Diese Signatur wird zum ZDA geschickt. Der ZDA (Schritt (c)) prüft die Signatur und den Index. Falls diese korrekt sind und der Schlüssel nicht revoziert ist, wird das Gültigkeitstoken erstellt. Durch die Signaturüberprüfung wird sichergestellt, dass keine falschen Indexinformationen an den ZDA geschickt werden können und so bspw. Gültigkeitstoken für noch nicht verwendete Indizes angefordert werden können. Das Token wird zurück an den Signierer geschickt. Danach übergibt der Signierer die Signatur gemeinsam mit dem Gültigkeitstoken an den Verifizierer. Dieser kann nun (vollständig offline) die Signatur sowie das Gültigkeitstoken verifizieren. Wenn beides korrekt ist, akzeptiert er die Signatur als gültig. Es sei angemerkt, dass das Gültigkeitstoken auch vom Verifizierer angefordert werden könnte.

	ZDA		Signierer		Verifizierer
(a)			$\sigma = sign(sk, m)$		
(b)		$\xleftarrow{\sigma}$			
(c)	$verify(pk, \sigma) = b$				
(d)	if (b= true) ∧ pk is not revoked: gen proof ϕ				
(e)		$\xrightarrow{\phi}$			
(f)				$\xrightarrow{\sigma,\phi}$	
(g)					$verify(\sigma, \phi) = b$
(h)					if (b= true): accept

Abbildung 5.2: Online Protokoll

Da das Gültigkeitstoken durch den Index mit einer Signatur eindeutig verknüpft ist und bestätigt, dass nicht revoziert wurde, kann es auch in zukünftigen Gültigkeitsprüfungen ohne weitere online Anfragen genutzt werden. Diese Bindung ist der Hauptunterschied zu einer OCSP An-

frage, welche ebenso die Gültigkeit von Zertifikaten bzw. eines Schlüssels bestätigt, jedoch bei jeder Signaturprüfung erneut ausgeführt werden muss. Aufgrund der Vorwärtssicherheit des Signaturverfahrens kann ein Gültigkeitstoken für Index i darüber hinaus als Token für alle vorhergehenden Signaturen verwendet werden. Damit ist es auf einfache Weise möglich solche Anfragen an den ZDA für mehrere Signaturen zu aggregieren, indem nur ein Token für den höchsten Index angefragt wird.

Das Gültigkeitstoken kann als einfaches Paar (öffentlicher Schlüssel, Index), welches vom ZDA signiert wird, realisiert werden. Der Nachteil an einem solchen, signierten Gültigkeitstoken ist, dass bspw. Delegation nicht ohne weiteres realisiert werden kann, da der Signaturschlüssel des ZDAs benötigt wird. Darüber hinaus muss die Tokensignatur zusätzlich geprüft werden, die Tokengröße ist nicht optimal und die Signaturerzeugung ist verhältnismäßig aufwendig. Daher verwenden wir die Merkle Hashbaum Variante des Novomodo Systems, um die Token zu generieren. Zunächst erläutern wir jedoch kurz eine zusätzliche Erweiterung, die durch das Online Protokoll möglich wird.

Rückbestätigte Signaturerzeugung

In das hier verwendete System kann ein zusätzlicher Bestätigungsprozess für die Signaturerzeugung integriert werden. Bevor der ZDA einen Gültigkeitstoken erstellt und veröffentlicht, kann der ZDA eine Bestätigung vom Schlüsselinhaber über einen unabhängigen Kanal einholen. Eine Möglichkeit dies umzusetzen sind mobile Transaktionsnummern wie vom Onlinebanking bekannt. Eine vergleichbare Anwendung haben wir beispielsweise in [26] für den neuen Personalausweis entwickelt. Dieses Vorgehen ermöglicht das Erkennen von Schlüsselkompromittierungen, da der Schlüsselinhaber mittels eines unabhängigen Kanals über die Schlüsselverwendung informiert wird. Zusätzlich kann das zu signierende Dokument auf dem Smartphone angezeigt werden. Dafür müsste das Dokument lediglich in Schritt (b) des Online Protokolls (siehe Abbildung 5.2) zusätzlich zur Signatur an den ZDA übertragen werden. Der zusätzliche Bestätigungsprozess ermöglicht das Lockern bestimmter Sicherheitsanforderungen, das heißt Kartenleser ohne Pinpad und Display zu erlauben, oder die Sicherheit des Gesamtsystems zu verbessern. Dadurch kann beispielsweise die sichere Verwendung von Smartphones als Kartenleser ermöglicht werden.

5.4.2 Gültigkeitstoken mit Novomodo

Zunächst erklären wir zum besseren Verständnis wie das zugrundeliegende Novomodo System [27] funktioniert.

Bei Novomodo werden Hashketten verwendet. Der Endwert dieser Hashketten wird im jeweiligen Zertifikat angegeben und mit diesem signiert. Der ZDA speichert für jedes Zertifikat den Anfangswert der Hashkette. Anstatt ein Zertifikat zu revozieren, bestätigt ein ZDA dessen

Gültigkeit aktiv in regelmäßigen Abständen. Hierzu veröffentlicht er die Vorgängerwerte in der Hashkette, beginnend mit dem Vorgängerwert des Endwerts im Zertifikat. Um die Korrektheit des veröffentlichten Wertes zu prüfen, kann ein Verifizierer aus diesen Token durch einfaches Hashen den Endwert der Hashkette errechnen und mit demjenigen im Zertifikat abgleichen. Die Berechnung der Gültigkeitstoken (und die Durchführung des Online Protokolls) kann damit leicht delegiert werden, indem die entsprechenden Startwerte der Hashketten an den jeweiligen vertrauenswürdigen Dritten übertragen werden.

Anstelle von Hashketten können auch Merkle Hashbäume verwendet werden, was den Verifikationsaufwand signifikant verbessert. In diesem Fall wird die Wurzel des Merkle Baumes in das Zertifikat eingefügt und vom ZDA signiert. Wenn ein Zertifikat in Periode i gültig ist, so veröffentlicht der ZDA das Blatt an Position i zusammen mit den Geschwisterknoten auf dem Pfad zur Wurzel. QuasiModo Bäume [28] erlauben es die Baumgröße weiter zu reduzieren und ermöglichen im Durchschnitt kürzere Pfade zur Wurzel, indem auch innere Knoten als Token verwendet werden. Für Standard Merkle Bäume sind jedoch höchst effiziente Baumtraversierungsalgorithmen verfügbar [1].

Daher setzen wir einen Merkle Baum, wie in Abbildung 5.3 gezeigt, zur Erzeugung der Gültigkeitstoken ein. Die Gültigkeitstoken (die Vorgängerknoten der Blätter) des Revokationsbaumes können pseudozufällig erzeugt werden, vergleichbar mit dem Vorgehen bei XMSS [1]. Eine Anwendung der Hashfunktion liefert die Blätter des Baumes, aus welchen der Merkle Baum berechnet wird. Der Baumtraversierungsalgorithmus von [1] kann dabei verwendet werden um den Berechnungsaufwand gleichmäßig über alle Perioden zu verteilen.

In Periode eins wird beispielsweise $c =$ (Blatt 1) zusammen mit dem Pfad zur Wurzel, welcher grau markiert ist, veröffentlicht. Um das Gültigkeitstoken zu verifizieren wird zunächst das Blatt $h(c)$ berechnet. Dann wird der Pfad im Merkle Baum rekonstruiert und die Berechnung anhand der Wurzel, welche im Zertifikat veröffentlicht wurde, überprüft.

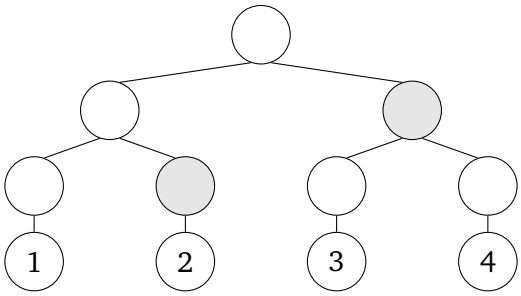


Abbildung 5.3: Revokationsbaum

Um eine feingranulare Revokation zu ermöglichen, verknüpfen wir jeden Signaturindex mit einem Gültigkeitstoken. Der ZDA veröffentlicht den $i$ten Gültigkeitstoken wenn ein Gültigkeitsbeweis für die Signatur mit Index i angefordert wird. Um Verzögerungen zu vermeiden, kann

eine bestimmte Anzahl von Gültigkeitstoken vorberechnet und gespeichert werden, wodurch der Aufwand bei der Onlineanfrage auf einen Speicherzugriff reduziert wird.

6 Sicherheitsanalyse und Validierung

Hier verifizieren wir die Anforderungserfüllung bezüglich des deutschen SigG und QES. Anschließend betrachten wir den langfristigen Gültigkeitserhalt von elektronischen Signaturen, die mit dem zuvor beschriebenen Verfahren mittels FSS erstellt wurden.

6.1 Anforderungserfüllung

Die gesetzlichen Anforderungen an QES wurden in Abschnitt 4.2.1 dargelegt. Diese sind

- Kettenmodell zur Gültigkeitsprüfung: Diese Anforderung wird direkt durch die Umsetzung des Kettenmodells mittels FSS (vgl. Abschnitt 5.1) erfüllt.
- Sichere Signaturerstellungseinheit: Diese Anforderung kann wie bei klassischen Signaturverfahren mittels Smartcard und sicherem Kartenleser (mit Pinpad und Display) erfüllt werden. Eine Implementierung von XMSS auf einer Smartcard wurde bereits an unserer Forschungsgruppe umgesetzt [13]. Daneben haben wir ein Verfahren vorgeschlagen, welches die sichere Signaturerstellungseinheit mittels eines vertrauenswürdigen Dritten und einem unabhängigen Kanal realisiert.

Darüber hinaus ist es problemlos möglich alle Sicherheitsanforderungen, die mit herkömmlichen Signaturverfahren erreicht werden können, gleichermaßen umzusetzen. Das beschriebene Online Protokoll ermöglicht zusätzlich Nichtabstreitbarkeit ohne Zeitstempeldienste. Existenzgarantien werden über die Vorwärtssicherheit erreicht.

Die Absicherung gegen plötzlichen technologischen Fortschritt erfolgt über die durch Hash-Combiner in XMSS integrierte Redundanz bezüglich der Sicherheitsbausteine. Mehrfachsignaturen können daher vermieden werden.

6.2 Langfristige Sicherheit

Bezüglich der langfristigen Sicherheit ist zum einen das Gültigkeitsprüfmodell wichtig. Hier sind die Anforderungen entsprechend dem Kettenmodell umgesetzt.

Auf der anderen Seite ist hier der kontinuierliche technologische Fortschritt besonders relevant. Auch vorwärtssichere Verfahren unterliegen dem bekannten Sicherheitsverfall über den Zeitverlauf. Ist also davon auszugehen, dass das eingesetzte Verfahren mit den verwendeten Parametern unsicher wird, besteht die Lösung weiterhin im Übersignieren. Allerdings können die Intervalle, in denen solche Übersignaturen notwendig werden, mit dem hier entwickelten Verfahren und durch den Einsatz von XMSS deutlich vergrößert werden. Durch die benannten

Frühwarnmechanismen kann eine Übersignatur ausgelöst werden. Daher entfällt das Antizipieren von zukünftigen Entwicklungen. Sicherheit gegen Gültigkeitsverfall wird unabhängig von oft vagen Vorhersagen erreicht. Da der Bruch zweier unabhängiger Verfahren im selben Moment selbst bei kontinuierlichen Entwicklungen unwahrscheinlich ist, können Hash-Combiner auch hier zusätzliche Sicherheit bieten. So kann bspw. der Bruch der Kollisionsresistenz einer Hashfunktion bereits eine Übersignatur auslösen, während die zweite eingesetzte Hashfunktion noch als vollständig sicher betrachtet werden kann.

Zusammenfassend bedeutet dies, dass auch bezüglich der langfristigen Sicherheit die Häufigkeit von Übersignaturen bzw. Zeitstempeln und der dabei anfallende Overhead deutlich reduziert wird.

Werden darüber hinaus die von uns vorgeschlagenen Gültigkeitstoken eingesetzt, vereinfacht dies die Archivierung von Signaturen zusätzlich. Von einem vertrauenswürdigen ZDA erzeugt, kann damit ein Beweis über die “Nicht-Revokation” leicht gespeichert werden.

Literaturverzeichnis

[1] Johannes Buchmann, Erik Dahmen, and Andreas Hülsing. XMSS - a practical forward secure signature scheme based on minimal security assumptions. In Bo-Yin Yang, editor, *Post-Quantum Cryptography*, volume 7071 of *Lecture Notes in Computer Science*, pages 117–129. Springer Berlin / Heidelberg, 2011.

[2] BSI. Vertrauenswürdige elektronische langzeitspeicherung. Technical Report 03125, Bundesamt für Sicherheit in der Infromationstechnik, 2009.

[3] C. Adams, P. Cain, D. Pinkas, and R. Zuccherato. Internet X.509 Public Key Infrastructure Time-Stamp Protocol (TSP). RFC 3161 (Proposed Standard), August 2001. Updated by RFC 5816.

[4] Sönke Maseberg. *Fail-Safe-Konzept für Public-Key-Infrastrukturen*. Doctoral thesis, Technische Universität Darmstadt, 2002.

[5] Erik Dahmen. *Post-quantum signatures for today*. PhD thesis, Technische Universität Darmstadt, Feb 2009.

[6] D. Cooper, S. Santesson, S. Farrell, S. Boeyen, R. Housley, and W. Polk. RFC 5280: Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile. RFC 5280 (Proposed Standard), 2008.

[7] T. Dierks and E. Rescorla. The Transport Layer Security (TLS) Protocol — Version 1.2. RFC 5246, August 2008.

[8] B. Ramsdell and S. Turner. Secure/Multipurpose Internet Mail Extensions (S/MIME) Version 3.2 Message Specification. *RFC*, 5751, January 2010.

[9] The European Parliament and the Council of the European Union. Directive 1999/93/ec of the european parliament and of the council of 13 december 1999 on a community framework for electronic signatures. In *European Union law*. Dec. 1999. `http://eur-lex.europa.eu/smartapi/cgi/sga_doc?smartapi!celexapi!prod!CELEXnumdoc&numdoc=31999L0093&model=guichett`.

[10] M. Myers, R. Ankney, A. Malpani, S. Galperin, and C. Adams. X.509 Internet Public Key Infrastructure Online Certificate Status Protocol - OCSP. RFC 2560 (Proposed Standard), June 1999. Updated by RFC 6277.

[11] Ross Anderson. Two remarks on public key cryptology. In *Manuscript. Relevant material presented by the author in an invited lecture at the 4th ACM Conference on Computer and Communications Security, CCS*, pages 1–4. Citeseer, 1997.

[12] Mihir Bellare and Sara Miner. A forward-secure digital signature scheme. In Michael Wiener, editor, *Advances in Cryptology – CRYPTO' 99*, volume 1666 of *Lecture Notes in Computer Science*, pages 786–786. Springer Berlin / Heidelberg, 1999.

[13] Johannes Buchmann, Christoph Busold, and Andreas Hülsing. Forward secure signatures on smart cards. In *SAC2012 - Conference on Selected Areas in Cryptography*, University of Windsor, ontario, Canada, Aug 2012. to appear.

[14] Bundesnetzagentur. Bekanntmachung zur elektronischen signatur nach dem signaturgesetz und der signaturverordnung (übersicht Ã¼ber geeignete algorithmen). In *Bundesanzeiger*, volume 17, pages 383 – 397. 2010.

[15] Johannes Braun, Andreas Hülsing, Alexander Wiesmaier, Martin A. G. Vigil, and Johannes Buchmann. How to avoid the breakdown of public key infrastructures - forward secure signatures for certificate authorities. In *EuroPKI 2012, September 13-14*, Sep 2012. to appear.

[16] h online. Fake Google certificate is the result of a hack. `http://h-online.com/-1333728`, visited Nov. 2011.

[17] Bundesrepublik Deutschland. Gesetz zur digitalen signatur. In *Signaturgesetz - SigG*. Juni 1997.

[18] Bundesrepublik Deutschland. Gesetz zur digitalen signatur. In *Signaturgesetz - SigG*. Mai 2001.

[19] Bundesrepublik Deutschland. Verordnung zur elektronischen signatur. In *Signaturverordnung - SigV*. Nov. 2001.

[20] Peter Eckersley and Jesse Burns. The (Decentralized) SSL Observatory. Invited talk at 20th USENIX Security Symposium, August 2011 visited Jan. 2012. `http://www.usenix.org/events/sec11/tech/slides/eckersley.pdf`.

[21] Comodo. The Recent RA Compromise. `http://blogs.comodo.com/it-security/data-security/the-recent-ra-compromise/`, visited Nov. 2011.

[22] h online. Attack on Israeli Certificate Authority. `http://h-online.com/-1264008`, visited Nov. 2011.

[23] ITU-T. Recommendation X.509 (08/2005) - Information technology – Open Systems Interconnection – The Directory: Public-key and attribute certificate frameworks. Technical report, ITU-T, 2005.

[24] Johannes Buchmann, Erik Dahmen, Elena Klintsevich, Katsuyuki Okeya, and Camille Vuillaume. Merkle signatures with virtually unlimited signature capacity. In Jonathan Katz and Moti Yung, editors, *Applied Cryptography and Network Security*, volume 4521 of *Lecture Notes in Computer Science*, pages 31–45. Springer Berlin / Heidelberg, 2007.

[25] Marc Fischlin, Anja Lehmann, and Krzysztof Pietrzak. Robust multi-property combiners for hash functions revisited. In Luca Aceto, Ivan Damgård, Leslie Goldberg, Magnús Halldórsson, Anna Ingólfsdóttir, and Igor Walukiewicz, editors, *Automata, Languages and Programming*, volume 5126 of *Lecture Notes in Computer Science*, pages 655–666. Springer Berlin / Heidelberg, 2008.

[26] Johannes Braun, Moritz Horsch, and Alexander Wiesmaier. ipin and mtan for secure eid applications. In Guilin Wang Mark Ryan, Ben Smyth, editor, *8th International Conference on Information Security Practice and Experience (ISPEC 2012)*, number 7232 in LNCS, pages 259–276. Springer, Apr 2012. 10.1007/978-3-642-29101-2_18.

[27] Silvio Micali. Novomodo: Scalable certificate validation and simplified pki management. In *1st Annual PKI Research Workshop*, pages 15–25, 2002.

[28] Farid F. Elwailly, Craig Gentry, and Zulfikar Ramzan. Quasimodo: Efficient certificate validation and revocation. In Feng Bao, Robert H. Deng, and Jianying Zhou, editors, *Public Key Cryptography*, volume 2947 of *Lecture Notes in Computer Science*, pages 375–388. Springer, 2004.