H. Neuroth, A. Oßwald, R. Scheffel, S. Strathmann, M. Jehn (Hrsg.)

nestor Handbuch

Eine kleine Enzyklopädie der digitalen Langzeitarchivierung

Version 2.0

Kapitel 5.3
Praktische Sicherheitskonzepte

nestor

vwh

nestor Handbuch: Eine kleine Enzyklopädie der digitalen Langzeitarchivierung
hg. v. H. Neuroth, A. Oßwald, R. Scheffel, S. Strathmann, M. Jehn
im Rahmen des Projektes: nestor – Kompetenznetzwerk Langzeitarchivierung und Langzeitverfügbarkeit digitaler Ressourcen für Deutschland
nestor – Network of Expertise in Long-Term Storage of Digital Resources
http://www.langzeitarchivierung.de/

Kontakt: editors@langzeitarchivierung.de
c/o Niedersächsische Staats- und Universitätsbibliothek Göttingen,
Dr. Heike Neuroth, Forschung und Entwicklung, Papendiek 14, 37073 Göttingen

Die Herausgeber danken Anke Herr (Korrektur), Martina Kerzel (Bildbearbeitung) und Jörn Tietgen (Layout und Formatierung des Gesamttextes) für ihre unverzichtbare Unterstützung bei der Fertigstellung des Handbuchs.

Bibliografische Information der Deutschen Nationalbibliothek
Die Deutsche Nationalbibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Daten sind im Internet unter http://www.d-nb.de/ abrufbar.

Die Inhalte dieses Buchs stehen auch als Onlineversion (http://nestor.sub.uni-goettingen.de/handbuch/) sowie über den Göttinger Universitätskatalog (http://www.sub.uni-goettingen.de) zur Verfügung.
Die digitale Version 2.0 steht unter folgender Creative-Commons-Lizenz:
„Attribution-Noncommercial-Share Alike 3.0 Unported“
http://creativecommons.org/licenses/by-nc-sa/3.0/

Einfache Nutzungsrechte liegen beim Verlag Werner Hülsbusch, Boizenburg.
© Verlag Werner Hülsbusch, Boizenburg, 2009
www.vwh-verlag.de
In Kooperation mit dem Universitätsverlag Göttingen

Markenerklärung: Die in diesem Werk wiedergegebenen Gebrauchsnamen, Handelsnamen, Warenzeichen usw. können auch ohne besondere Kennzeichnung geschützte Marken sein und als solche den gesetzlichen Bestimmungen unterliegen.

Druck und Bindung: Kunsthaus Schwanheide

Printed in Germany – Als Typoskript gedruckt –

ISBN: 978-3-940317-48-3

URL für Kapitel 5.3 „Praktische Sicherheitskonzepte“ (Version 2.0):
urn:nbn:de:0008-20090811238
http://nbn-resolving.de/urn/resolver.pl?urn=urn:nbn:de:0008-20090811238

Gewidmet der Erinnerung an Hans Liegmann (†), der als Mitinitiator und früherer Herausgeber des Handbuchs ganz wesentlich an dessen Entstehung beteiligt war.

5.3 Praktische Sicherheitskonzepte

Siegfried Hackel, Tobias Schäfer und Wolf Zimmer

5.3.1 Hashverfahren und Fingerprinting

Ein wichtiger Bestandteil praktischer Sicherheitskonzepte zum Schutz der Integrität und Vertraulichkeit digitaler Daten sind Verschlüsselungsinfrastrukturen auf der Basis sogenannter kryptographisch sicherer Hashfunktionen. Mit Hilfe kryptographisch sicherer Hashfunktionen werden eindeutige digitale „Fingerabdrücke" von Datenobjekten berechnet und zusammen mit den Objekten versandt oder gesichert. Anhand eines solchen digitalen „Fingerabdrucks" ist der Empfänger oder Nutzer der digitalen Objekte in der Lage, die Integrität eines solchen Objektes zur prüfen, bzw. unautorisierte Modifikationen zu entdecken.

Hashfunktionen werden in der Informatik seit langem eingesetzt, bspw. um im Datenbankumfeld schnelle Such- und Zugriffsverfahren zu realisieren. Eine Hashfunktion ist eine mathematisch oder anderweitig definierte Funktion, die ein Eingabedatum variabler Länge aus einem Urbildbereich (auch als „Universum" bezeichnet) auf ein (in der Regel kürzeres) Ausgabedatum fester Länge (den Hashwert, engl. auch message digest) in einem Bildbereich abbildet. Das Ziel ist, einen „Fingerabdruck" der Eingabe zu erzeugen, die eine Aussage darüber erlaubt, ob eine bestimmte Eingabe aller Wahrscheinlichkeit nach mit dem Original übereinstimmt.

Da der Bildbereich in der Regel sehr viel kleiner ist als das abzubildende „Universum", können so genannte „Kollisionen" nicht ausgeschlossen werden. Eine Kollision wird beobachtet, wenn zwei unterschiedliche Datenobjekte des Universums auf den gleichen Hashwert abgebildet werden.

Für das Ziel, mit einer Hashfunktion einen Wert zu berechnen, der ein Datenobjekt eindeutig charakterisiert und damit die Überprüfung der Integrität von Daten ermöglicht, sind derartige Kollisionen natürlich alles andere als wünschenswert. Kryptographisch sichere Hashfunktionen H, die aus einem beliebig langen Wort M aus dem Universum von H einen Wert H(M), den Hashwert fester Länge erzeugen, sollen daher drei wesentliche Eigenschaften aufweisen:

1. die Hashfunktion besitzt die Eigenschaften einer effizienten Ein-Weg-Funktion, d.h. für alle M aus dem Universum von H ist der Funktionswert h = H(M) effizient berechenbar und es gibt kein effizientes Verfahren, um aus dem Hashwert h die Nachricht zu berechnen[14],
2. es ist - zumindest praktisch - unmöglich zu einem gegebenen Hashwert h = H(M) eine Nachricht M' zu finden, die zu dem gegebenen Hashwert passt (Urbildresistenz),
3. es ist - zumindest praktisch – unmöglich, zwei Nachrichten M und M' zu finden, die denselben Hashwert besitzen (Kollisionsresistenz).

Praktisch unmöglich bedeutet natürlich nicht praktisch ausgeschlossen, sondern bedeutet nicht mehr und nicht weniger, als dass es bspw. sehr schwierig ist, ein effizientes Verfahren zu finden, um zu einer gegebenen Nachricht M eine davon verschiedene Nachricht M' zu konstruieren, die denselben Hashwert liefert. Für digitale Objekte mit binären Zeichenvorräten Z = {0,1} lässt sich zeigen, dass für Hashfunktionen mit einem Wertbereich von 2n verschiedenen Hashwerten, beim zufälligen Ausprobieren von 2n/2 Paaren von verschiedenen Urbildern M und M' die Wahrscheinlichkeit einer Kollision schon größer als 50% ist.

Beim heutigen Stand der Technik werden Hashfunktionen mit Hashwerten der Länge n = 160 Bit als hinreichend stark angesehen.[15] Denn, selbst eine Schwäche in der Kollisionsresistenz, wie bereits im Jahre 2005 angekündigt[16], besagt zunächst einmal lediglich, dass ein Angreifer zwei verschiedene Nachrichten erzeugen kann, die denselben Hashwert besitzen. Solange aber keine Schwäche der Urbildresistenz gefunden wird, dürfte es für einen Angreifer mit einem gegebenen Hashwert und passendem Urbild immer noch schwer sein, ein zweites, davon verschiedenes Urbild zu finden, das zu diesem Hashwert passt.

Kern kryptographischer Hashfunktionen sind Folgen gleichartiger Kompressionsfunktionen K, durch die eine Eingabe M blockweise zu einem Has-

14 Obwohl die Ein-Weg-Funktionen in der Kryptographie eine wichtige Rolle spielen, ist nicht bekannt, ob sie im streng mathematischen Sinne eigentlich existieren, ihre Existenz ist schwer zu beweisen. Man begnügt sich daher zumeist mit Kandidaten, für die man die Eigenschaft zwar nicht formal bewiesen hat, für die aber derzeit noch keine effizienten Verfahren zur Berechnung der Umkehrfunktion bekannt sind.

15 Ein Rechner, der in der Lage ist, pro Sekunde den Hashwert zu einer Million Nachrichten zu berechnen, bräuchte 600.000 Jahre, um eine zweite Nachricht zu ermitteln, deren Hashwert mit einem vorgegebenen Hashwert der Länge 64 Bit übereinstimmt. Derselbe Rechner könnte allerdings in etwa einer Stunde irgendein Nachrichtenpaar mit gleichem Hashwert finden.

16 Schneier (2005)

hwert verarbeitet wird. Um Eingaben variabler Länge zu komprimieren, wendet man den Hashalgorithmus f iterierend an. Die Berechnung startet mit einem durch die Spezifikation des Hashalgorithmus festgelegten Initialwert $f(0):=I_0$. Anschließend gilt:

$$f(i) := K\,(f(i-1),M_i) \text{ mit } M = M_1, \ldots, M_n,\ i = 1, \ldots, n$$
$$H(M):= f(n) = h \text{ ist der Hashwert von } M$$

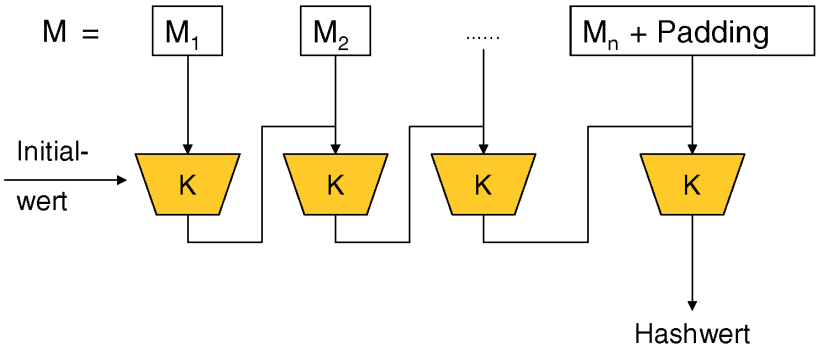

Abbildung 3: Allgemeine Arbeitsweise von Hashfunktionen (nach C. Eckert [17]*)*

Neben auf symmetrischen Blockchiffren, wie dem bereits 1981 durch das American National Standards Institute (ANSI) als Standard für den privaten Sektor anerkannten Data Encryption Standard (DES)[18], finden heute vor allem Hashfunktionen Verwendung, bei denen die Kompressionsfunktionen speziell für die Erzeugung von Hashwerten entwickelt wurden. Der bislang gebräuchlichste Algorithmus ist der Secure Hash Algorithm SHA-1 aus dem Jahre 1993.[19]

Der SHA-1 erzeugt Hashwerte von der Länge 160 Bits[20] und verwendet eine Blockgröße von 512 Bits, d.h. die Nachricht wird immer so aufgefüllt, dass die Länge ein Vielfaches von 512 Bit beträgt. Die Verarbeitung der 512-Bit Ein-

17 Eckert (2001)
18 Vgl. bspw. Schneier (1996)
19 Vgl. bspw. Schneier (1996)
20 Da nicht ausgeschlossen werden kann, dass mit der Entwicklung der Rechentechnik künftig auch Hashwerte von der Länge 160 Bit nicht mehr ausreichend kollisions- und urbildresistent sind, wird heute für sicherheitstechnisch besonders sensible Bereiche bereits der Einsatz der Nachfolger SHA-256, SHA-384 und SHA-512 mit Bit-Längen von jeweils 256, 384 oder 512 Bits empfohlen.

gabeblöcke erfolgt sequentiell, für einen Block benötigt SHA-1 insgesamt 80 Verarbeitungsschritte.

Merkle-Hashwertbäume

In der Kryptographie und Informatik werden Merkle-Bäume[21] eingesetzt, um über große Mengen von Daten oder Dokumenten einen zusammenfassenden Hashwert zu bilden.
Die Blätter eines Merkle-Baums sind die Hashwerte der Dokumente $H_i(d_i)$. Jeder Knoten im Baum wird als Hashwert $H(h_1 | h_2)$ seiner Kinder h_1 und h_2 gebildet. Dabei ist h_1 der Hashwert des Dokuments d_1 und h_2 der Hashwert des Dokuments d_2 und „|“ die Verkettung (Konkatenation) der beiden Hashwerte. Die Abbildung 4 zeigt ein Beispiel für einem binären Merkle-Hashbaum.

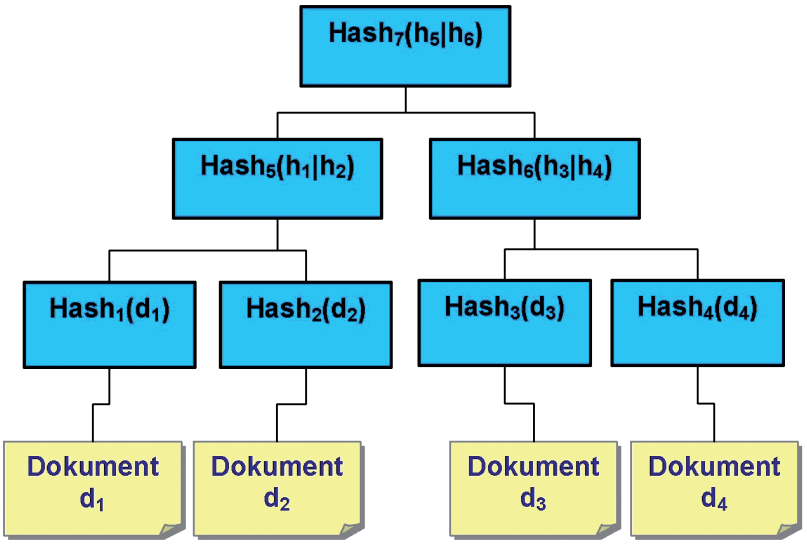

Abbildung 4: Binärer Merkle-Hashbaum.

Hash-Bäume können zum Nachweis der Integrität beliebiger Daten verwendet werden, die in oder zwischen Rechnern gespeichert, verarbeitet oder übertragen werden.

Die meisten Hash-Baum-Implementierungen sind binär, d.h. jeder Knoten besitzt zwei Kinder, es sind jedoch auch mehrere Kind-Knoten möglich.

21 Im Jahr 1979 von Ralph Merkle erfunden

Holt man für den obersten Hashwert (H_7 in Abbildung 4) einen Zeitstempeln ein, lässt sich beweisen, dass die Dokumente zum aktuellen Zeitpunkt existiert haben und seitdem nicht manipuliert wurden. Dieses Verfahren wurde im ArchiSig-Konzept[22] aufgegriffen und weiterentwickelt, um das Problem der nachlassenden Sicherheitseignung von Hash- und Signaturalgorithmen zu lösen. Dabei wird mit Hilfe von (akkreditierten) Zeitstempeln die vom deutschen Signaturgesetz geforderte Übersignatur von signierten Dokumenten vorgenommen.

5.3.2 Digitale Signatur

Elektronische Signaturen sind „Daten in elektronischer Form, die anderen elektronischen Daten beigefügt oder logisch mit ihnen verknüpft sind und die zur Authentifizierung im elektronischen Rechts- und Geschäftsverkehr dienen. Ihre Aufgabe ist die Identifizierung des Urhebers der Daten, d.h. der Nachweis, dass die Daten tatsächlich vom Urheber herrühren (Echtheitsfunktion) und dies vom Empfänger der Daten auch geprüft werden kann (Verifikationsfunktion). Beides lässt sich nach dem heutigen Stand der Technik zuverlässig am ehesten auf der Grundlage kryptographischer Authentifizierungssysteme, bestehend aus sicheren Verschlüsselungsalgorithmen sowie dazu passenden und personifizierten Verschlüsselungs-Schlüsseln (den so genannten Signaturschlüsseln) realisieren.

Die Rechtswirkungen, die an diese Authentifizierung geknüpft werden, bestimmen sich aus dem Sicherheitsniveau, das bei ihrer Verwendung notwendig vorausgesetzt wird. Dementsprechend unterscheidet das im Jahre 2001 vom deutschen Gesetzgeber veröffentlichte „Gesetz über Rahmenbedingungen für elektronische Signaturen und zur Änderung weiterer Vorschriften“[23], kurz Signaturgesetz (SigG), vier Stufen elektronischer Signaturen:

- „Einfache elektronische Signaturen“ gem. § 2 Nr. 1 SigG,
- „Fortgeschrittene elektronische Signaturen“ gem. § 2 Nr. 2 SigG,
- „Qualifizierte elektronische Signaturen“ gem. § 2 Nr. 3 SigG,
- „Qualifizierte elektronische Signaturen“ mit Anbieter-Akkreditierung gem. § 15 Abs. 1 SigG.

Mit Ausnahme der einfachen elektronischen Signaturen, denen es an einer verlässlichen Sicherheitsvorgabe völlig fehlt, wird das mit der Anwendung

22 Siehe www.archisig.de

23 BGBl I 876; BT-Drs 14/4662 und 14/5324

elektronischer Signaturen angestrebte Sicherheitsniveau grundsätzlich an vier Elementen festgemacht (§ 2 Nr. 2 SigG). Elektronische Signaturen müssen demnach

- ausschließlich dem Signaturschlüssel-Inhaber zugeordnet sein,
- die Identifizierung des Signaturschlüssel-Inhabers ermöglichen,
- mit Mitteln erzeugt werden, die der Signaturschlüssel-Inhaber unter seiner alleinigen Kontrolle halten kann und
- mit den Daten, auf die sie sich beziehen, so verknüpft sein, dass eine nachträgliche Veränderung der Daten erkannt werden kann.

Europaweit als Ersatz für die handschriftliche Unterschrift akzeptiert werden jedoch lediglich qualifizierte elektronische Signaturen. Für sie wird zusätzlich gefordert (§ 2 Nr. 3 SigG), dass sie

- auf einem zum Zeitpunkt ihrer Erzeugung gültigen qualifizierten Zertifikat beruhen und
- mit einer sicheren Signaturerstellungseinheit erzeugt werden.

Das Zertifikat übernimmt in diesem Fall die Authentizitätsfunktion, d.h. es bescheinigt die Identität der elektronisch unterschreibenden Person.[24] Sichere Signaturerstellungseinheiten sind nach dem Willen des Gesetzgebers Software- oder Hardwareeinheiten, die zur Speicherung und Anwendung des Signaturschlüssels dienen.[25]

Das Verfahren der digitalen Signatur basiert auf so genannten asymmetrischen kryptographischen Authentifizierungssystemen, bei denen jeder Teilnehmer ein kryptographisches Schlüsselpaar besitzt, bestehend aus einem ge-

---

24 Nach § 2 Nr. 6 SigG sind Zertifikate elektronische Bescheinigungen, mit denen Signaturschlüssel einer Person zugeordnet werden und die Identität einer Person bescheinigt wird. Für die Anwendung von Signaturverfahren von besonderer Bedeutung ist die Feststellung, dass „qualifizierte Zertifikate" nur auf natürliche Personen ausgestellt werden dürfen.

25 Das deutsche Signaturgesetz fordert, § 17 Abs. 1 SigG, dass sichere Signaturerstellungseinheiten vor unberechtigter Nutzung zu schützen sind. Nach § 15 Abs. 1 der Verordnung zur elektronischen Signatur (SigV) ist hierfür eine Identifikation „durch Besitz und Wissen oder durch Besitz und ein oder mehrere biometrische Merkmale" erforderlich. Da bislang keine Implementierungen biometrischer Verfahren bekannt sind, die die Anforderungen des Signaturgesetzes (vgl. Anlage 1 SigV) nachweislich erfüllen, werden für qualifizierte elektronische Signaturen in der Praxis immer Personal Identification Numbers (PIN) als Identifikationsdaten eingesetzt.

heimen privaten Schlüssel (private key, Kpriv) und einem öffentlichen Schlüssel (public key, Kpub).

Eine wesentliche Eigenschaft solcher asymmetrischer Authentifizierungssysteme ist, dass es praktisch unmöglich ist, den privaten Schlüssel aus dem öffentlichen Schlüssel herzuleiten, der öffentliche Schlüssel wird durch Anwendung einer sogenannten Einwegfunktion aus dem privaten Schlüssel berechnet. Der öffentliche Schlüssel kann daher in einem öffentlich zugänglichen Verzeichnis hinterlegt werden, ohne damit den privaten Schlüssel preiszugeben.

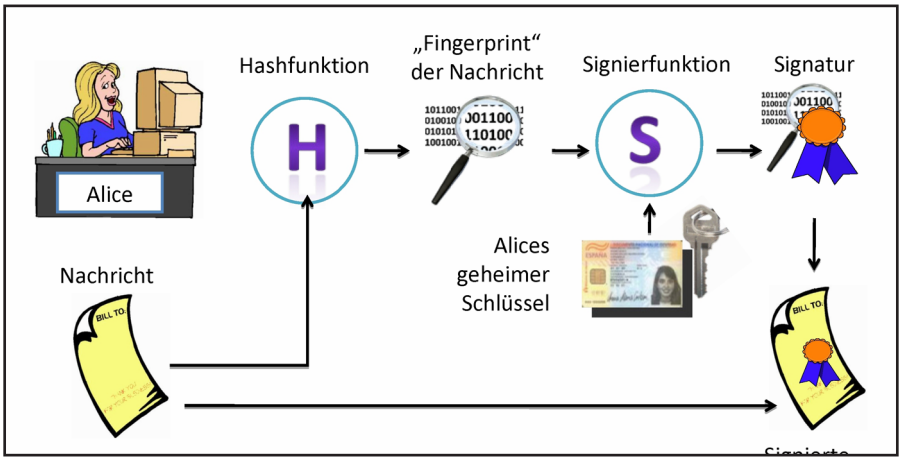

Abbildung 5: Digitale Signatur

Der Urheber, respektive Absender elektronischer Daten „unterschreibt“ nun seine Daten, indem er sie mit seinem geheimen, privaten Schlüssel verschlüsselt. Jeder, der die Daten empfängt, kann sie dann mit dem öffentlichen Schlüssel wieder entschlüsseln (s. Abbildung 5).

Unter der Voraussetzung, dass der öffentliche Schlüssel eindeutig und zuverlässig einer Person zugeordnet werden kann, bezeugt die Signatur folglich die Identität des Unterzeichners. Da die Signatur zudem das Ergebnis einer Verschlüsselungsoperation ist, sind die signierten Daten nachträglich auch nicht mehr veränderbar bzw. eine Änderung ist sofort erkennbar. Die Signatur kann auch nicht unautorisiert weiter verwendet werden, weil das Ergebnis der Verschlüsselungsoperation natürlich abhängig von den Daten ist. Geht man ferner davon aus, dass der private Signaturschlüssel nicht kompromittiert worden ist, kann der Absender der Daten die Urheberschaft auch nicht mehr zurückweisen, weil ausschließlich er selbst über den privaten Signaturschlüssel verfügt.

Technisch wäre natürlich eine Verschlüsselung der gesamten Daten (eines

Dokuments oder einer Nachricht) viel zu aufwändig. Aus diesem Grunde wird aus den Daten eine eindeutige Prüfsumme, ein Hashwert (s. dazu auch Kap. 5.3.1) erzeugt, dieser verschlüsselt („unterschrieben“) und den Originaldaten beigefügt. Der mit dem geheimen Schlüssel verschlüsselte Hashwert repräsentiert fortan die elektronische Signatur („Unterschrift“) der Originaldaten. Der Empfänger seinerseits bildet nach demselben Verfahren, d.h. mit demselben Hash-Algorithmus ebenfalls eine Prüfsumme aus den erhaltenen Daten und vergleicht sie mit der des Absenders. Sind die beiden Prüfsummen identisch, dann sind die Daten unverändert und stammen zuverlässig vom Inhaber des geheimen Schlüssels, denn nur er war in der Lage die Prüfsumme so zu verschlüsseln, dass sie mit dem zugehörigen öffentlichen Schlüssel auch entschlüsselt werden konnte.

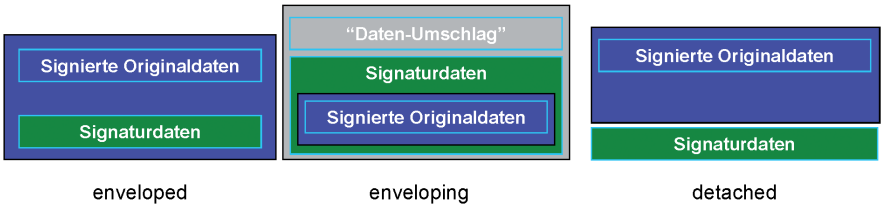

Abbildung 6: Hinzufügung der Signaturdaten

Die Hinzufügung der Signaturdaten zu den Originaldaten kann grundsätzlich auf folgende Weise geschehen (s. Abbildung 6):

- Enveloped („eingebettet“): die Signaturdaten sind als Element in den Originaldaten enthalten.
  Dieses Verfahren, auch als so genannte „Inbound-Signatur“ bezeichnet, wird vor allem bei der Signatur von PDF-Dokumenten und PDF-Formularen bspw. im Projekt ArchiSafe der Physikalisch-Technischen Bundesanstalt benutzt (s. a. Abb. 7).[26] Dabei werden die binären Signaturdaten direkt in das PDF-Dokument eingebettet und gemeinsam mit den Originaldaten im PDF-Format angezeigt. Mit dem neuen Adobe® Reader® (Version 8) ist der Empfänger der signierten Daten darüber hinaus imstande, unmittelbar eine Überprüfung der Integrität der angezeigten und signierten Daten vorzunehmen.
  Eingebettete Signaturen werden ebenso bei der Signatur von XML-Daten[27] verwendet und sollen zudem nun auch für den neuen XDOMEA

26 s. http://www.archisafe.de

27 1999 bis 2002 wurde der W3C-Standard für das Signieren von XML-Dokumenten am

Standard 2.0[28] spezifiziert werden. Da die Signatur eine binäre Zahlenfolge ist, lässt sie sich jedoch nicht direkt in ein XML-Dokument einbetten. Man codiert daher die binären Werte im Base64-Format (RFC 1521), um aus ihnen ASCII-lesbare Zeichen zu gewinnen. Die erhaltene Zeichendarstellung der Signatur findet sich schliesslich als SignatureValue in der XML-Signatur wieder[29].

- **Enveloping („umschließend"):** die Signaturdaten „umschließen" die Originaldaten. Diese Methode wird hauptsächlich für die Signatur von E-Mail Nachrichten oder reinen XML-Daten benutzt. Eine S/MIME Client-Anwendung, wie bspw. Microsoft Outlook, bettet in diesem Fall die Nachricht in einen signierten „Umschlag" ein.
- **Detached („getrennt"):** die Signaturdaten befinden sich außerhalb der Originaldaten in einer zusätzlichen, binären Signaturdatei. Diese Form, auch als „Outbound-Signatur" bezeichnet, wird standardmäßig für XML-Signaturen sowie die Signatur binärer Originaldaten eingesetzt. Ein separater Link in den Original-Daten oder zusätzlichen Beschreibungsdaten sorgt dann für die notwendige permanente Verknüpfung der Originaldaten mit den Signaturdaten.

Massachusetts Institute of Technology (MIT) entwickelt (XMLDSIG). Die XML Signatur Spezifikation (auch XMLDSig) definiert eine XML Syntax für digitale Signaturen. In ihrer Funktion ähnelt sie dem PKCS#7 Standard, ist aber leichter zu erweitern und auf das Signieren von XML Dokumenten spezialisiert. Sie findet Einsatz in vielen weiterführenden Web-Standards wie etwa SOAP, SAML oder dem deutschen OSCI. Mit XML Signaturen können Daten jeden Typs signiert werden. Dabei kann die XMLSignatur Bestandteil des XML Datenpakets sein (enveloped signature), die Daten können aber auch in die XML-Signatur selbst eingebettet sein (enveloping signature) oder mit einer URL adressiert werden (detached signature). Einer XML-Signatur ist immer mindestens eine Ressource zugeordnet, das heisst ein XML-Baum oder beliebige Binärdaten, auf die ein XML-Link verweist. Beim XML-Baum muss sichergestellt sein, dass es zu keinen Mehrdeutigkeiten kommt (zum Beispiel bezüglich der Reihenfolge der Attribute oder des verwendeten Zeichensatzes). Um dies erreichen zu können, ist eine so genannte Kanonisierung des Inhalts erforderlich. Dabei werden nach Maßgabe des Standards alle Elemente in der Reihenfolge ihres Auftretens aneinander gereiht und alle Attribute alphabetisch geordnet, so dass sich ein längerer UTF8-String ergibt (es gibt auch Methoden, die einen UTF16-String erzeugen). Aus diesem wird der eigentliche Hash-Wert gebildet beziehungsweise erzeugt man durch verschlüsseln den Signaturcode. So ist man wieder beim Standard-Verfahren für elektronische Signaturen (RFC 2437).

28 s. http://www.kbst.bund.de

29 Im Rahmen der Struktur eines XML-Dokuments lassen sich Subelemente explizit vom Signieren ausschliessen, so auch die Signatur selbst. Umgekehrt lassen sich beliebig viele Referenzen auflisten, die gemeinsam als Gesamtheit zu signieren sind.

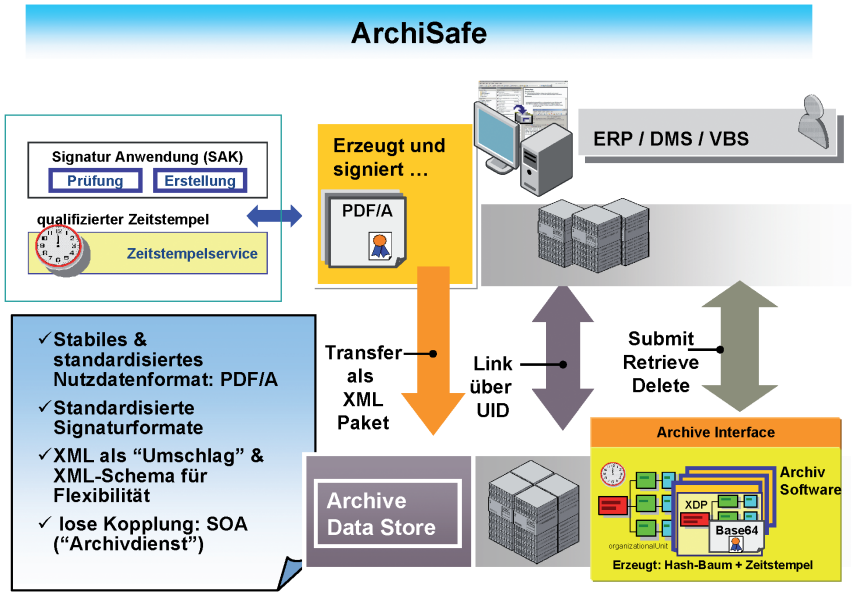

Abbildung 7: ArchiSafe – Rechts- und revisionssichere Langzeitspeicherung elektronischer Dokumente

Die Flexibilität der Hinzufügung von Signaturdaten zu Originaldaten basiert auf der als RFC 3852 – Cryptographic Message Syntax (CMS) im Juli 2004[30] durch die Internet Engineering Task Force (IETF) veröffentlichten Spezifikation sowie dem ursprünglich durch die RSA Laboratories veröffentlichten PKCS#7 (Public Key Cryptography Standard) Dokument in der Version 1.5. In beiden Dokumenten wird eine allgemeine Syntax beschrieben, nach der Daten durch kryptographische Maßnahmen wie digitale Signaturen oder Verschlüsselung geschützt, respektive Signaturdaten über das Internet ausgetauscht werden können. Die Syntax ist rekursiv, so dass Daten und Umschläge verschachtelt oder bereits chiffrierte Daten unterschrieben werden können. Die Syntax ermöglicht zudem, dass weitere Attribute wie z.B. Zeitstempel mit den Daten oder dem Nachrichteninhalt authentifiziert werden können und unterstützt eine Vielzahl von Architekturen für die Schlüsselverwaltung auf der Basis von elektronischen Zertifikaten.

30 Network Working Group (2004)

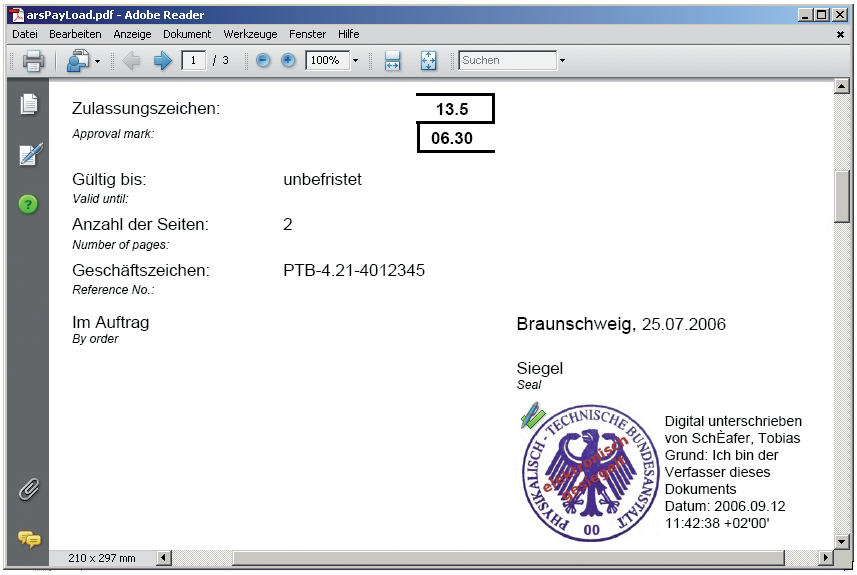

Abbildung 8: Digitale PDF-Signatur

Literatur

Eckert, Claudia (2001): *IT-Sicherheit: Konzepte – Verfahren – Protokolle*. München: Oldenbourg Wissenschaftsverlag

Network Working Group (2004): R. Hously: *Cryptographic Message Syntax (CMS) Request for Comment*s: 3852 http://www.ietf.org/rfc/rfc3852

Schneier, Bruce (1996): *Angewandte Kryptographie*, Bonn u.a.: Addison-Wesley Verl.

Schneier, Bruce (2005): *Schneier on Security. A blog covering security and security technology*. SHA-1 Broken (February 15, 2005) http://www.schneier.com/blog/archives/2005/02/sha1_broken.html