Eine Sicherheitsarchitektur auf Basis digitaler Wasserzeichen und kryptographischer Ansätze

Artur Dappa[a], Jana Dittmann[b], Martin Steinebach[b,c], Claus Vielhauer[c]

[a] Deutsche Telekom, Technologiezentrum, Darmstadt, Deutschland
[b]GMD Forschungszentrum Informationstechnik, Darmstadt, Deutschland
[c]Platanista GmbH, Darmstadt, Deutschland

Zusammenfassung

Sicherheitsarchitekturen werden durch die steigende Akzeptanz von onlinebasierten, sicherheitskritischen Diensten immer wichtiger. Oft werden neue Dienste aufgebaut, und erst nachträglich Sicherheitsaspekte behandelt. Wir stellen anhand des Beispiels Ausweiswasserzeichen ein System vor, das von vornhinein als sicherheitskritisch identifizierbar ist. Es handelt sich um fälschungssichere Ausweise, deren Authentizität durch Wasserzeichen geprüft und über Onlineverbindungen bestätigt werden kann.

In Kapitel 1 stellen wir das Umfeld der hier diskutierten Sicherheitsarchitekturen vor und zeigen anhand der Beispiele Online-Shop, Nutzungsverfolgung und Ausweiswasserzeichen Einsatzgebiete und Anforderungen für diese. Kapitel 2 beschreibt die Sicherheitsarchitekturen. Es werden allgemein Sicherheitsdienste vorgestellt und kategorisiert und danach die Sicherheitsmechanismen digitale Wasserzeichen, digitale Signaturen, PKI, Zeitstempeldienst und Non-Repudiation im Detail erörtert. Es wird anhand des Beispiels Ausweiswasserzeichen gezeigt, wie Wasserzeichen und PKI zu Verbesserung der Sicherheit eingesetzt werden können.

In Kapitel 3 wird ein Konzept vorgestellt, bei dem die in Kapitel 2 vorgestellten Sicherheitsmechanismen in einen gemeinsamen implementierungstechnischen Rahmen gefassten werden. Dabei soll keine neue Architektur entworfen, sondern eine bereits bestehende um entsprechende sicherheitstechnische Merkmale erweitert werden. Dazu wird die Common Object Request Broker Architecture (CORBA) der Object Management Group (OMG) verwendet. Kapitel 4 zeigt die Richtung der weiteren Entwicklung. Es zeigt sich, dass einige Komponenten vor ihrer Umsetzung in CORBA noch weiter spezifiziert werden müssen.

1 Motivation

Das Internet galt lange Zeit als ein unsicheres, riskantes Medium für die Unterstützung von geschäftlichen Vorgängen, insbesondere wenn es sich um den Transfer von zu schützenden Informationen und Werten handelte. Mit seinen multimedialen und interaktiven Fähigkeiten wird es jedoch noch lange den Status einer „Leittechnologie“ für die Informationstechnik haben, so dass Normierungsgremien, IT-Industrie und große Teile der Wirtschaft nun einen Kraftakt vollbringen diesen Engpass zu überwinden.

Es scheint jedoch, dass die sicherheitstechnische „Aufrüstung“ der Internettechnologie an vielen anderen in der heutigen IT-Welt eingesetzten Technologien vorbeigeht. Nicht nur in den gebräuchlichsten Betriebssystemen und Netzwerkkomponenten, sondern auch in der für die Intranetnutzung konzipierte anwendungsnahe Software zeigen sich große Defizite in punkto

Sicherheit. Beispielsweise werden fast überall noch passwortbasierte Authentisierungsmechanismen für den Netzzugang eingesetzt, die in manchen Fällen noch schwach oder gar unverschlüsselt übertragen werden.

Zu einem bestimmten Typ der anwendungsnahen Software gehören die „Middleware"-Komponenten, mit deren Hilfe der Umstrukturierungsprozess der früheren Jahre von monolithischen zu dezentralen Systemen mit relationalen Daten realisiert werden konnte. Man erkennt aber auch hier den Trend, dass diese Technologien immer schneller gegen das Internet konvergieren und damit auch die Notwendigkeit, das „alte" Sicherheitskonzept der Middleware-Technologien auszutauschen.

Der vorliegende Beitrag stellt einige Sicherheitstechnologien vor, mit deren Hilfe die Sicherheitseigenschaften von Middleware-Komponenten weiter verbessert werden könnten. Dazu gehören insbesondere die aus der asymmetrischen Kryptographie stammenden Verfahren wie digitale Signatur und Austausch von öffentlichen Schlüsseln, aber auch spezielle Sicherheitstechniken für A/V-Datenströme, die auf dem digitalen Wasserzeichenverfahren beruhen. Am Beispiel einer objektorientierten Software-Architektur (CORBA) werden Integrationsansätze für die vorgestellten Sicherheitsverfahren motiviert, die in der Praxis noch als Insellösungen zur Verfügung stehen. Es werden auch Anwendungsszenarien vorgestellt, in denen durch neue Sicherheitsarchitekturen eine starke Verbesserung der Systemsicherheit gegeben ist. Es handelt sich hier um Online-Shops für digitale Waren, die Nutzungsverfolgung und Ausweiswasserzeichen. Ihnen allen ist gemeinsam, dass digitale Wasserzeichen als Schutzmaßnahme eingesetzt werden.

Im Online-Shop werden digitale Wasserzeichen eingesetzt, um den Copyrightanspruch des rechtmäßigen Inhabers auch nach der Übertragung in das System des Anwenders zu gewährleisten. Um das Recht des Urhebers zu schützen, sind verschiedene Typen von Wasserzeichen sinnvoll: Ein Copyrightwasserzeichen enthält Hinweise bezüglich des Copyrightinhabers. Ein Fingerprintwasserzeichen enthält Daten, anhand derer auf den Käufer geschlossen werden kann. Bei der Nutzungsverfolgung werden digitale Daten, die im Internet vorgefunden werden, für unterschiedliche Zwecke untersucht: a) Missbrauchsverfolgung b) Marktanalyse. Bei der Missbrauchsverfolgung geht es darum, eine illegale Nutzung von digitalem Material zu detektieren und eventuell Schritte zum Einstellen der Nutzung einzuleiten. Bei Marktanalysen ist die Verbreitung und die Nutzung digitaler Daten von Interesse. Neben der Verwendung von digitalen Wasserzeichen zur Authentifizierung des rechtmäßigen Urhebers und zur Identifizierung von illegalen Kopien, können digitale Wasserzeichen auch zur Integritätsprüfung herangezogen werden. Einen vielversprechenden Sicherheitsmechanismus können digitale Wasserzeichen bei der Überprüfung von Ausweisdokumenten bieten. Beispielsweise kann ein digitales Wasserzeichen im Passfoto Informationen über die gesamten Ausweisinformationen beinhalten und somit eine direkte Verknüpfung der lesbaren Daten und des abgebildeten Fotos herstellen, um Fälschungen vorzubeugen. Mit einfachen Mitteln, wie einer digitalen Kamera vor die der Ausweis gehalten wird, können das Wasserzeichen und die Ausweisinformationen überprüft werden. Entsprechende Technologien werden beispielsweise von DigiMark [Dig01] unter der Bezeichnung Mediabrigde vertrieben.

2 Sicherheitsarchitekturen

In diesem Kapitel wird ein Umriss der zu konzipierenden Sicherheitsarchitektur vorgestellt. Dazu wird ein abstrakter Sicherheitsdienst skizziert und in seine Komponenten zerlegt (Abschnitt 2.1). Im Abschnitt 2.2 werden die einzelnen Sicherheitsmechanismen beschrieben, die

den Kern eines Sicherheitsdienstes bilden. Abschnitt 2.3 enthält das Beispielszenario Ausweiswasserzeichen, welches bereits in der Motivation kurz angerissen worden ist. Es wird nun aus den drei Beispielen herausgegriffen und eingehend betrachtet. Wichtig sind dabei vor allem die notwendigen Sicherheitsmechanismen.

Mit Sicherheitsarchitekturen verbindet man in der Regel Standards wie IETF GSS-API, Intel/Open Group CDSA oder herstellerspezifische Architekturen wie Microsoft Crypto-API oder JCA. Diese Architekturen haben den Zweck, Software- und Hardware-Komponenten unter einer gemeinsamen API (Application Programmers Interface) einzubinden und plattformübergreifende Interoperabilität von Sicherheitskomponenten zu ermöglichen.

Aus der Perspektive eines Anwendungsentwicklers sind noch weitere Anforderungen wichtig:

?? Es sollen komplexe Sicherheitsfunktionen in Software-Systemen mit einem minimalen Aufwand realisiert werden.

?? Anwendungsentwickler, die nur das grundlegende kryptographisches Wissen haben, sollen schon in der Lage sein, die Programmierschnittstellen richtig anzuwenden.

Der hier verfolgte Ansatz orientiert sich an der Philosophie der Middleware-Technik, die anstrebt, dem Anwendungsprogrammierer das komplexe Kommunikationsverhalten von verteilten Systemen zu verbergen, aber auch für einen verlässlichen Kommunikationsablauf zu sorgen. Es wird also das Ziel verfolgt, middleware-basierte Dienstplattformen und Werkzeuge um solche Sicherheitsmerkmale zu erweitern, die für die neuen Anwendungen vom großen Nutzen sind (siehe Abschnitt 2.1 und 2.2).

2.1 Security Services

Dieser Abschnitt zeigt, um welche Sicherheitsmerkmale die Middleware Security Services erweitert werden sollen und beschreibt die funktionalen Anforderungen für das Design dieser Dienste.

Tabelle 1 zeigt die Schutzbedürfnisse der neuen Zielanwendungen (oberste Zeile) und die in Frage kommenden Sicherheitstechniken, die von der Middleware unterstützt werden sollen (linke Spalte). In der Tabelle wird ferner gezeigt, dass unterschiedliche Sicherheitsmechanismen zusammen ein Schutzbedürfnis erfüllen müssen. Insbesondere für das Bedürfnis „Schutz der Urheberrechte“ unterscheiden sich die Sicherheitstechnologien am meisten (Wasserzeichen-Verfahren, kryptographische Verfahren). Im Abschnitt 2.2 werden die Sicherheitsmechanismen einzeln vorgestellt und ihre gegenseitige Abhängigkeiten gezeigt.

Die Sicherheitsdienste können in drei Kategorien eingeteilt werden: Sicherheitsdienste für den Schutz von Online- bzw. Internettransaktionen (Spalten 1 bis 5), Sicherheitsdienste für den Schutz von Urheberrechten (Spalte 6) und Sicherheitsdienste für den Schutz von digitalen Inhalten oder Mediendaten.

Die primäre Aufgabe der Middleware ist, eine abstrakte und uniforme Anwendungsprogrammierschnittstelle und interne Schnittstellen für die flexible Einbindung von elementaren Funktionen, Algorithmen und Protokollen ähnlich dem CSP-Konzept anzubieten. Gefordert wird auch ein flexibles Security Policy Framework, das sich an die Sicherheitsbedürfnisse unterschiedlicher Organisationen anpassen lässt. Eine Security Policy setzt sich aus einer Menge von Regeln oder Anweisungen zusammen, die um bestimmte Sicherheitsziele zu wahren, vom IT-System einschließlich der Nutzer befolgt werden müssen. Abstrakt gesehen setzt sich ein Middleware Security Service aus einem Security Policy und User Credential Management

sowie dem Security Enforcement Subsystem. Letzteres beinhaltet die Sicherheitsmechanismen und zugehörige Administrationsschnittstellen. Eine weitere Forderung ist die Skalierbarkeit des Sicherheitssystems. Dazu bedient man sich des Domain-Konzeptes. Eine Domain soll eine Gruppe von sicherheitsrelevanten Systemressourcen repräsentieren, auf die die gleiche Security Policy zutrifft. Abbildung 1 stellt das Konzept eines Middleware Security Service schematisch dar.

	Integrationsebene		Nutzer-authentisierung	Daten-authentisierung	Integritätsschutz der Daten	Vertraulichkeit der Daten	Nichtabstreitbarkeit	Schutz der Urheberrechte	Integritätsschutz von Inhalten	Autorisierter Zugriff auf Inhalte
			1	2	3	4	5	6	7	8
X.509-Zertifikate	A/M	1	X	X	X	X	X	X	X	X
sichere Transportprotokolle	M	2	X	X	X	X	X			
dig. Signatur (anwendungsspezifisch)	A	3					X	X		
Zeitstempelmechanismen	A/M	4					X	X		
Archivierungsmechanismen	A/M	5					X	X		
Nichtabstreitbarkeitsbeweise	A/M	6					X			
digitale Wasserzeichen	A	7						X		
inhaltsbasierte Signatur	A	8							X	
Verschlüss. für spezifische Formate	A	9								X

Tabelle 1: Die Tabelle stellt dar, welche Schutzziele (Spalten 1 bis 8) anwendungsseitig verfolgt werden und welche Diensttypen (Zeilen 1 bis 9), dafür in Frage kämen. Ebenfalls sind die Integrationsebenen der Dienste angegeben: Anwendung (A) und Middleware (M).

2.2 Sicherheitsmechanismen

Hier werden die Mechanismen zur Bereitstellung von Sicherheit erörtert. Es handelt sich dabei um digitale Wasserzeichen, digitale Signaturen, PKI, Zeitstempeldienst und Non-Repudiation. Sowohl ihre prinzipielle Funktion als auch eventuell vorhandene Verfahrensparameter werden aufgezeigt.

2.2.1. Digitale Wasserzeichen

Generell verstehen wir unter einem digitalen Wasserzeichen ein transparentes, nicht wahrnehmbares Muster, welches in das Datenmaterial (Bild, Video, Audio, 3D-Modelle) mit einem Einbettungsalgorithmus unter Verwendung eines geheimen Schlüssels eingebracht wird.

Jeder Wasserzeichenalgorithmus nutzt steganographische Grundprinzipien und besteht in Analogie zur Steganographie aus:

- Einem Einbettungsprozess *E*: Watermark Embedding
- Einem Abfrageprozess/Ausleseprozess *R*: Watermark Retrieval

Das eingebettete Muster repräsentiert die eingebrachte Information. Typischerweise kann das Muster zwei Arten von Informationen darstellen: Entweder ein von einem Schlüssel abhängiges Muster zur Identifizierung des Urhebers/Autors/Senders oder kodierte Informationen.

Hier handelt es sich im Allgemeinen um Urheberdaten (zur Kennzeichnung der Urheberrechte), Kundendaten (zur Kennzeichnung von Kopien) oder Metadaten.

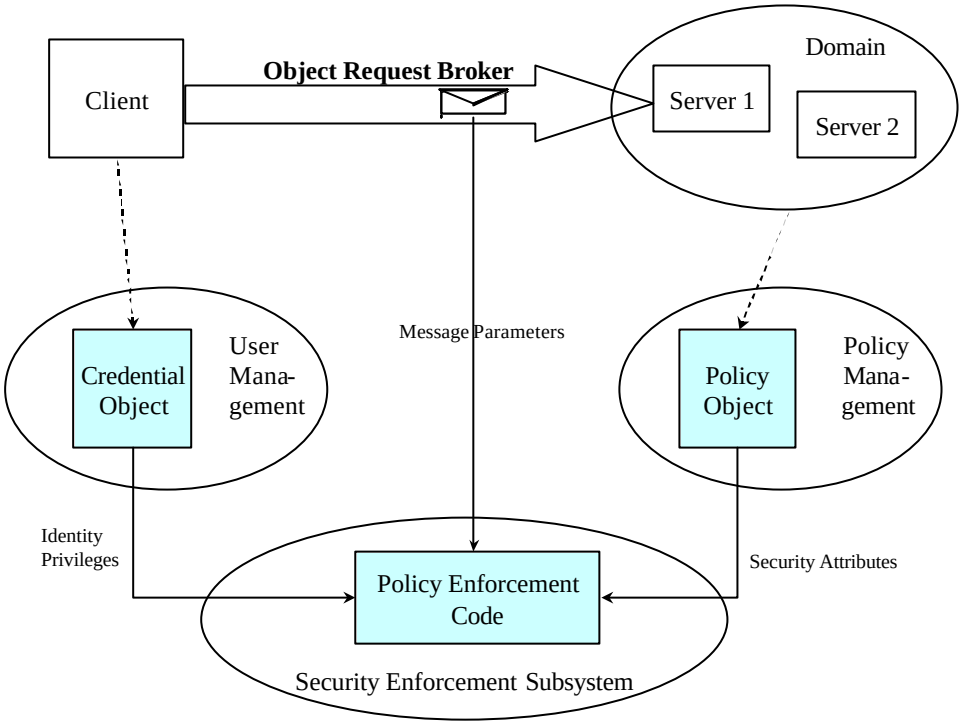

Abbildung 1: Schematische Darstellung eines Middleware Security Service. Das Policy und Credential Management liegt entweder in der Applikation oder in der Middleware, der Sicherheitsmechanismus kann in der Applikation, in der Middleware, oder in untergeordneten Protokollschichten liegen, und seine Administrationsschnittstelle in der Applikation oder in der Middleware.

Bei existierenden Wasserzeichenverfahren können wir folgende Anwendungsgebiete identifizieren [Ditt00]:

?? **Verfahren zur Urheberidentifizierung (Authentifizierung):** *Robust Authentication Watermark*

?? **Verfahren zur Kundenidentifizierung (Authentifizierung):** *Fingerprint Watermark*

?? **Verfahren zur Annotation des Datenmaterials**: *Caption Watermark, Annotation Watermark*

?? **Verfahren zur Durchsetzung des Kopierschutzes oder Übertragungskontrolle:** *Copy Control Watermark, Broadcast Watermark*

?? **Verfahren zum Nachweis der Unversehrtheit (Integritätsnachweis):** *Integrity Watermark oder Verification Watermark*

Jede Wasserzeichentechnik hat bestimmte Eigenschaften. Die wichtigste Eigenschaften eines Wasserzeichenverfahrens sind Robustheit, Nicht-Wahrnehmbarkeit, Security, Komplexität, Kapazität, Verifikation, Invertierbarkeit.

?? **Robustheit**: Robustheit bezeichnet die Widerstandsfähigkeit der in ein Datenmaterial eingebrachten Wasserzeicheninformation gegenüber zufälligen Veränderungen des Datenmaterials oder Medienverarbeitungen.

- ?? **Nicht-Wahrnehmbarkeit**: Die eingebrachte Information *W* ist nicht wahrnehmbar und somit transparent, wenn ein durchschnittliches Seh- bzw. Hörvermögen nicht zwischen markiertem Datenmaterial und Original unterscheiden kann.
- ?? **Security**: Diese Eigenschaft beschreibt im Gegensatz zur Robustheit die Sicherheit gegen gezielte (nicht-blinde) Angriffe auf das Wasserzeichen selbst.
- ?? **Komplexität**: Beschreibt den Aufwand, der erbracht werden muss, die Wasserzeicheninformation einzubringen und wieder auszulesen und ob zum Auslesen der Markierung das Originalbild benötigt wird.
- ?? **Kapazität**: Dieser Parameter misst, wie viel Information in das Original eingebracht werden kann.
- ?? **Geheime/öffentliche Verifikation**: Dieser Parameter sagt aus, ob nur der Urheber oder eine dedizierte Personengruppe das Wasserzeichen aufdecken können (geheim) oder ob die Verifikation öffentlich erfolgen kann bzw. soll.
- ?? **Invertierbarkeit**: Beschreibt die Möglichkeit das Wasserzeichen im Abfrageprozess aus dem Datenmaterial zu entfernen und das Original wieder zu rekonstruieren.

Wichtig ist vor allem auch eine Zuordnung der Wasserzeichenverfahren für verschiedene Anwendungsgebieten zu den Parametern. Abhängig von den Anforderungen der verschiedenen Anwendungen können Aspekte von Wasserzeichen eventuell als weniger relevant angesehen werden als andere. Genauso können für manche Anwendungen gewisse Parameter Grundvoraussetzung sein. Oft kann das Bewerten von Parametern auch erst bei direkter Kenntnis der Situation erfolgen, das verschiedene relevante Parameter sich gegenseitig hindern. So kann ein Wasserzeichen zum Urheberrechtsnachweis im Allgemeinen nicht gleichzeitig für Robustheit und Transparenz optimiert werden. Hier muss eine Entscheidung getroffen werden, welcher Parameter im Einzelfall wichtiger ist.

2.2.2 Digitale Signatur

Die digitale Signatur ist ein kryptographischer Mechanismus mit dem Identität von Personen[1] oder Integrität bzw. Authentizität von Daten nachgewiesen werden kann. Dieser Mechanismus wird meistens direkt in der Applikation (Beispiel Email) oder in Kryptoprotokollen eingesetzt (Beispiele SSL, IPSEC).

Das Prinzip der digitalen Signatur veranschaulicht Abbildung 2. Das Verfahren besteht aus einer Hash- und einer asymmetrischen Verschlüsselungsfunktion. Für die Signaturerzeugung braucht der Signierende seinen privaten Schlüssel des Verschlüsselungsalgorithmus (Signaturschlüssel), der Prüfer der Signatur benötigt den öffentlicher Schlüssel des Signierenden (Prüfschlüssel). Die Einbettung der Signatur in den Datenstrom (Signaturblock) wird durch entsprechende kryptographische Formatdienste realisiert, wie beispielsweise PKCS#7/CMS für S/MIME.

[1] anwendbar auch auf IT-Ressourcen wie z.B. Server-Rechner

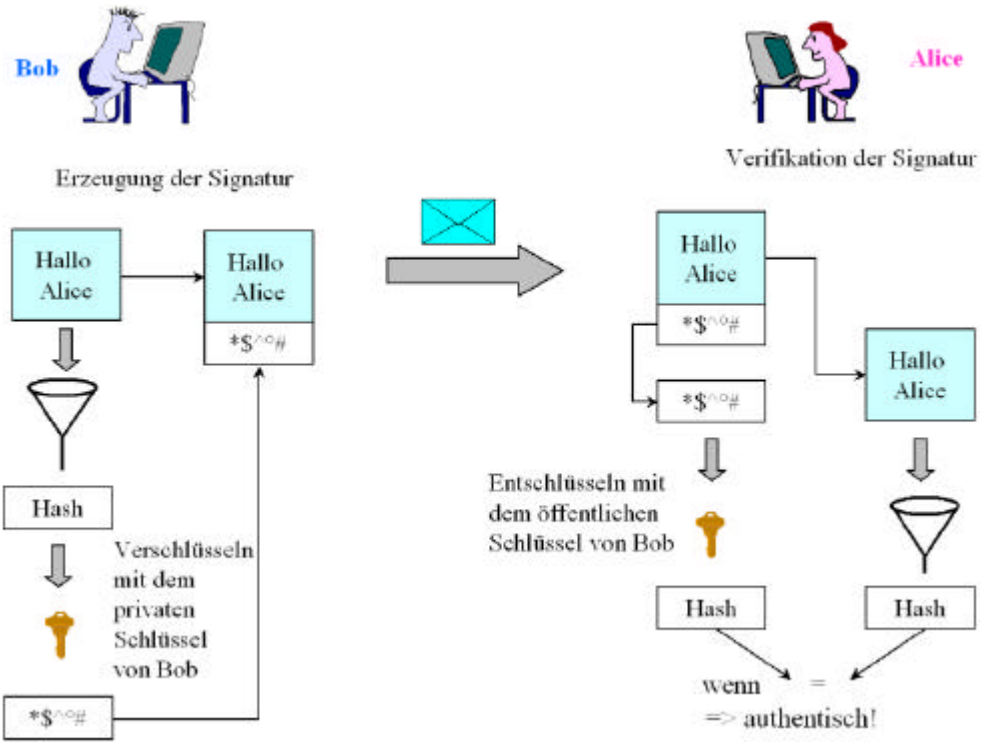

Abbildung 2: Prinzip der digitalen Signatur

Das in der Abbildung dargestellte Verifikationsschema betrifft nur die mathematische Relation zwischen der Signatur und dem Prüfschlüssel. Im Sinne eines vollständigen Beweises müsste noch die Authentizität des Prüfschlüssels nachgewiesen werden (siehe 2.2.3 PKI). Zu einem weiteren Element eines vollständigen Beweises gehört auch der Zeitstempel, der mit dem signierten oder zu signierendem Dokument mathematisch verknüpft werden muss (siehe 2.2.4 Zeitstempeldienst). Für bestimmte Anwendungen sind noch Berechtigungsnachweise (z.B. amtliche Zulassungen für Berufsstände) für die Ausführung der Signatur notwendig, die in Form von Attributzertifikaten bereitgestellt werden. Das signierte Dokument kann die Attributzertifikate direkt oder einen Verweis auf die letzteren enthalten[2].

2.2.3 PKI

Eine Public-Key-Infrastruktur (PKI) ist ein Konzept für das Management einer Vertrauensstruktur. Sie stellt den Anwendern integrierte Dienste für die Erzeugung, Verteilung und Verwaltung von Schlüsseln, Zertifikaten und Sperrlisten zur Verfügung. Die meisten heute verwendeten PKI-Technologien basieren auf X.509-Zertifikaten. Ein X.509-Zertifikat ist ein von einer vertrauenswürdigen Instanz digital signiertes Dokument (siehe 2.2.2 digitale Signatur), das u.a. Informationen bzgl. des Schlüsseleigentümers, des Zertifikatausstellers und der verwendeten kryptographischen Algorithmen beinhaltet. Ein Zertifikat dient somit als ein Authentisierungsnachweis für den Inhalt eines signierten Dokumentes (siehe 2.2.2 digitale Signatur) bzw. für den Halter des Signaturschlüssels.

Die Dienste einer PKI lassen sich gruppieren in

?? Zertifikatdienste

[2] Attributzertifikate werden weiter in dem Text nicht mehr betrachtet.

?? Schlüsselverwaltungs- und Archivierungsdienste

?? und Zeitstempeldienste

Diese Dienstgruppen werden von vertrauenswürdigen Instanzen (Trust Centers) bereitgestellt.

Die Kernfunktionen einer PKI sind die Zertifikatdienste, die der Certification Authority (CA) zugeteilt sind. Dazu gehören die Schlüsselpaargenerierung[3], Bearbeitung von Zertifikatanfragen, Zustellung oder Veröffentlichung von Zertifikaten, Überprüfung von Zertifikaten, Sperrung von Zertifikaten, sowie die Pflege und Verbreitung von Sperrlisten. Die Bearbeitung einer Zertifikatanfrage beinhaltet die Registrierung des Antragstellers basierend auf einer der jeweiligen CA-Policy entsprechenden Identitätskontrolle, sowie die Bindung des öffentlichen Schlüssels des Antragstellers mit seinem Namen, was die Erzeugung eines Public-Key-Zertifikates bedeutet.

Bei den Schlüsselverwaltungs- und Archivierungsdiensten handelt es sich um automatisierte Verfahren für die Erneuerung von Schlüsseln sowie für den Zugriff auf ältere Schlüsselversionen.

Die Zeitstempeldienste erbringen den Nachweis für den Zeitpunkt einer Aktion, z.B. für das Erstellen, Absenden oder Eintreffen eines Dokumentes (siehe 2.2.4 Zeitstempeldienst).

2.2.4 Zeitstempeldienst

Der Zeitstempeldienst ist eine Trusted Third Party (TTP), deren Dienst darin besteht, Daten mit einem bestimmten Zeitpunkt bzw. Zeitintervall in einer fälschungssicheren Weise zu verbinden. Zeitstempeldienste werden von Zertifizierungsstellen verwendet (siehe 2.2.3 PKI), um Gültigkeitsaussagen für Zertifikate im Falle einer Revokation treffen zu können und von Non-Repudiation-Diensten (siehe die 2.2.5 Non-Repudiation und 2.2.3 digitale Signatur).

Mit einem Zeitstempel kann man beweisen, dass die digitale Signatur vor einem bestimmten Zeitpunkt erstellt wurde (z.B. dem Revokationszeitpunkt des Zertifikates). Dadurch bewahrt eine digitale Signatur ihre Gültigkeit. Mit Zeitstempeln lässt sich auch die zeitliche Anordnung von mehreren Ereignissen nicht widerlegen (z.B. das Erstellen, Absenden und Eintreffen eines Dokumentes).

2.2.5 Non-repudiation

Tauschen sich zwei Parteien Nachrichten gegeneinander aus, so könnte der Sender bestreiten, eine bestimmte Nachricht abgeschickt zu haben. Er könnte behaupten, diese Nachricht wäre gefälscht worden. Ebenso könnte der Empfänger einer bestimmten Nachricht den Erhalt dieser Nachricht ableugnen. Zur Lösung von solchen Konflikten ist ein Non-repudiation Service notwendig. Dieser erzeugt und sammelt Beweise, mit deren Hilfe man eine neutrale Instanz überzeugen kann, dass eine bestimmte Person eine bestimmte Aktion ausgeführt hat.
In diesem Fall müsste der Non-repudiation Service beim Absenden einer Nachricht den Beweis erzeugen, dass der Sender die Nachricht erzeugt und abgeschickt hat (Evidence of Origination, Evidence of Submission). Ebenso müsste der Dienst beim Empfang einer Nachricht den Beweis erzeugen, dass der Empfänger die Nachricht erhalten und gelesen hat (Evidence of Receipt).

---

[3] ein Schlüsselpaar kann auch der Antragsteller selbst generieren und den Public Key der CA zusenden.

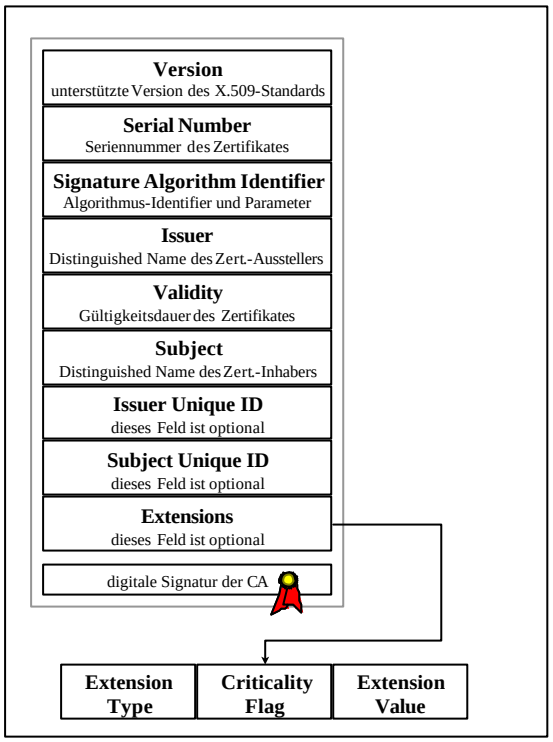

Abb. 3 Struktur eines X.509-Zertifikates

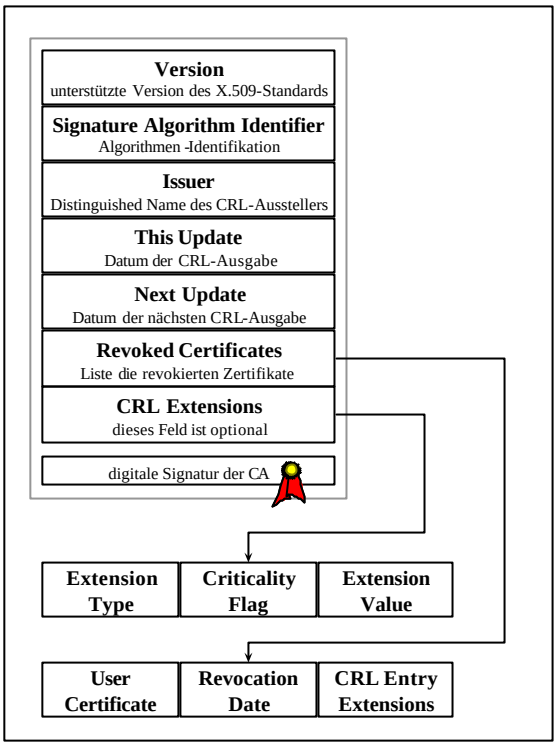

Abb. 4 Struktur einer X.509-Sperrliste (CRL)

Der Non-repudiation Service umfasst Mechanismen zum Erzeugen, Prüfen, Übermitteln und Archivieren von Beweisen[4]. Bestimmte Non-Repudiation-Mechanismen erfordern zusätzlich für die Beweisgenerierung und Verifikation eine Trusted Third Party (TTP). Im Streitfall werden die Beweise an eine autorisierte Schlichtungsinstanz eingereicht und von dieser überprüft.

2.3 Ausweiswasserzeichen: Ein Beispielszenario für Wasserzeichen und PKI

Digitale Wasserzeichen zur Überprüfung von Ausweisdokumenten können mit fragilen Verfahren erstellt werden, die ein Wasserzeichen zerbrechlich einfügen, um Manipulationen zu erkennen. Bei fragilen Wasserzeichen müssen die Aspekte der Veränderung von Mediendaten berücksichtigt werden. Bei Multimediadatenformaten kann sich die Syntax (der Bitstrom) verändern, ohne dass dadurch die Semantik beeinflusst wird – sei es beispielsweise durch Übertragungsfehler, Kompression oder Skalierung. Daher wird statt der Syntax der Daten ihre Semantik gesichert, um die Integrität der Bilddaten zu sichern. Um unerlaubte Manipulationen zu erkennen, müssen inhaltsbasierte Wasserzeichenverfahren verwendet werden [Ditt00]. Hier werden robuste Wasserzeichentechniken als Ausgangsbasis genommen.

Verwendet man robuste Verfahren, so ist man meist nicht in der Lage, Veränderungen am Bild zu erkennen. Im vorgestellten Szenario können robuste Wasserzeichenverfahren verwendet werden, wenn man die einzubettende Information vom Inhalt abhängig macht und somit eine Überprüfung der Integrität erfolgen kann. Dazu werden sollten digitale Signaturen verwendet werden.

---

[4] Bei Archivierung über lange Zeiträume, muss auf die Gültigkeitsdauer der bei der Beweisgenerierung verwendeten Kryptoalgorithmen geachtet werden.

Die Ansätze zum ID-Cardwasserzeichen von Digimarc und Signum klingen sehr vielversprechend. Da Ausweise einem hohen Potential an Angriffsversuchen unterliegen, muss evaluiert werden, welche Designanforderungen ein solches Wasserzeichen haben muss, um tatsächlich zusätzliche Sicherheit bieten zu können. Am Beispiel des Kopierangriffs diskutieren wir in diesem Abschnitt praktische Vorgehensmodelle von Angreifern und deren Erfolgsaussichten [DiBe01]. Die Kategorie der Protokollangriffe, vor allem der Kopierangriff (in der Literatur als Watermark Copy Attack, [KVH2000] beschrieben) eröffnet ein besonderes Risikopotential, da Angreifer daran interessiert sein werden, gültige Ausweiskopien mit gefälschtem Foto zu produzieren, um sich eine andere Identität zu verschaffen.

Betrachtet man das Ausweisszenario, so kann der Kopierangriff in blinde und nicht blinde Verfahren unterschieden werden [DiBe01]. Nicht-blinden Angriffen liegt das Originalpassfoto vor, welches nach derzeitig gängiger Praxis bei der Beantragung eines Personal- oder Firmenausweise vorgelegt wurde. Blinde Angriffe hingegen können nicht auf das Originalpassfoto zurückgreifen. Weiterhin können blinde und nicht blinde Verfahren in Mitwirkung und in ohne Mitwirkung des Ausweisinhabers unterschieden werden:

Nicht blinde Verfahren, Original oder dem Original sehr ähnliches Bild ohne Wasserzeichen liegt vor:

- Bei der Mitwirkung des Ausweisinhabers bei einem nicht blinden Verfahren ist davon auszugehen, das der Angreifer über das Original und das mit einem Wasserzeichen markierte Bild/Ausweisfoto verfügt.
- Ohne Mitwirken des Ausweisinhabers muss der Angreifer in Besitz des Originals kommen, wie zum Beispiel durch Diebstahl. Oder er versucht, den Ausweisinhaber in einer sehr ähnlichen Situation zu fotografieren und ein dem Original ähnliches nicht markiertes Bild zu erstellen.

Blinde Verfahren, Original liegt nicht vor:

- Bei Mitwirkung des Ausweisinhabers kann versucht werden das Original nachzustellen und entsprechend zu retuschieren, um den Kopierangriff zu starten.
- Ohne Mitwirkung des Ausweisinhabers kann mit Hilfe einer Fotomontage versucht werden, das Wasserzeichen zu kopieren und ein neues markiertes Bild zu konstruieren. Es eignen sich vorrangig moderne Bildbearbeitungsprogramme zum Retuschieren und Montieren des Bildmaterials der Zielperson. Mit Hilfe von einfachen Bildbearbeitungsmethoden wie Ausschnittsbildung, aber auch komplizierten Verfahren wie Morphing bieten gerade gängige Programme aus der Sharewareszene alle nötigen Hilfsmittel zur hochwertigen Bildmanipulation.

Betrachtet man das Vorgehen eines Kopierangriffs im Detail, so können die nicht-blinden Verfahren direkte Differenzbildangriffe durchführen, in dem Original und Wasserzeichenbild verglichen werden und die detektierte Differenz direkt auf ein drittes Bild übertragen wird.

Blinde Angriffe können die eingebrachte Wasserzeicheninformation nicht so einfach detektieren. [KVH2000] beschreibt ein Verfahren, welches auf Prädiktion beruht, siehe dazu Details in Endversion. Weiterhin ergeben sich jedoch durchaus sehr einfache Möglichkeiten, das Wasserzeichen durch spezielle Differenzbildungen zu kopieren, siehe folgendes Kapitel Testszenarien.

Wir haben Versuche zu Angriffen gegen Ausweiswasserzeichen durchgeführt. Diese zeigen, dass durch einfache Bildverarbeitungsoperationen das legitimierende Wasserzeichen von ei-

nem Ausweis auf eine Fälschung übertragen werden kann. Die Schwachstellen lassen sich auf das Szenario zurückverfolgen: Das Wasserzeichen muss einerseits zur Verifikation über die digitale Kamera aufgenommen und überprüft werden können, so dass eine generelle Robustheit gegenüber Druck und optischem Scan, leichten Größenänderungen (Skalierungen), Rotation und dem Kamera-Fischaugeneffekt gegeben sein muss. Andererseits soll das Wasserzeichen Veränderungen anzeigen. Beide Ziele können derzeit nur schwer gleichzeitig optimiert werden..

Ein weiteres Problem liegt darin, dass das Wasserzeichen eingebracht wird, ohne bildinhärente Eigenschaften zu berücksichtigen. Das in [Dit2000] vorgestellte SSP-Wasserzeichen, welches bildinhärente Eigenschaften nutzt, bietet Lösungsmöglichkeiten für den Kopierangriff. Ist das Wasserzeichen inhaltsabhängig, kann es zwar übertragen, beim Auslesen über den geänderten Inhalt aber nicht wieder gefunden werden. Allerdings muss das in [Dit2000] vorgestellte Verfahren an das Ausweiswasserzeichenszenario angepasst und hinsichtlich seiner Sicherheit überprüft bzw. erweitert werden.

Ebenso wichtig ist die Schlüsselproblematik: Ist es möglich, ein öffentlich verifizierbares Verfahren für Ausweiswasserzeichen zu entwickeln ? Hier wird eine Kombination mit einer PKI angestrebt. Es muss untersucht werden, wie hoch die Kapazität der Wasserzeichenverfahren ist, um eine digitale Signatur einbinden zu können und wie man dem Kopierangriff vorbeugt. Inhaltsbasierte Wasserzeichen könnten so eingesetzt werden, dass der Inhaltsauszug signiert und dann eingebettet wird. An dieser Stelle müssen die selben Kapazitätsbetrachtungen wie für das Direct Embedding erfolgen. Alternativ kann man mit inhaltsbasierten Signaturen arbeiten.

Um dem Kopierangriff vorzubeugen, kann man entweder das Wasserzeichenverfahren vom Inhalt des Bildes abhängig machen, oder die einzubettende Information. Im letzteren Fall bietet sich die Nutzung von sogenannten inhaltsbasierte Signaturen an. Grundsätzlich muss man bei der Echtheitsprüfung von Daten, der Verifizierung, zwei Konzepte unterscheiden: die vollständige Verifizierung und die inhaltsbasierte Verifizierung [DiSt99].

Bei der vollständigen Verifizierung werden die Daten als unveränderbare Nachrichten angesehen. Die zu prüfenden Daten werden nur als echt anerkannt, wenn keine Manipulationen vorliegen, d.h. die zu testenden Daten genau den Daten des Originals entsprechen. Eine praktische Umsetzung der vollständigen Verifizierung für Bilddaten lässt sich durch Benutzung des Wertes einer Einweg-Hash-Funktion des Urbildes erreichen. Dieser Hashwert dient dann, mit dem geheimen Schlüssel des Urhebers verschlüsselt, als digitale Signatur der Bilddaten und könnte mit einem Wasserzeichenverfahren eingebettet werden. Bei der Überprüfung wird aus der digitalen Signatur mit Hilfe des öffentlichen Schlüssels wieder der Hashwert der Originalnachricht ermittelt und mit dem Hashwert der zu testenden Daten verglichen. Werden Unterschiede festgestellt, wird das Testbild als manipuliert zurückgewiesen. Ein solches Verfahren ließe sich in Digitalkameras für den Überwachungsbereich oder Journalismus fest integrieren: Erzeugte Bilder dieser Kameras würden automatisch mit einer Kennung versehen, die deren Authentizität und Integrität bestätigt, solange keine Veränderungen an den Bildern vorgenommen werden [Fri1993].

Im Ausweiswasserzeichen-Szenario unterliegt der Ausweis jedoch einem Alterungsprozess und bei der Erstellung einer digital-analog-Wandlung sowie im Ausleseprozess einer analog-digital-Wandlung. Hierbei wird das Bildmaterial geändert und es treten Übertragungsfehler auf, so dass die digitale Signatur des Urhebers ungültig würde, da die Daten nur auf Originalzustand überprüft werden können. Diese Prozesse (Alterung, Analog-digital Wandlungen) än-

dern Bildpunktwerte aber nicht seinen Inhalt. Die Bedeutung eines Bildes ist nicht abhängig von der digitalen Repräsentation, sondern ausschließlich vom Bildinhalt. So werden Veränderungen der Daten, welche die Bildaussage nicht ändern als zugelassene Bildveränderungen eingestuft.

Bei einer Inhaltsverifizierung werden aus dem Bild Merkmale extrahiert, die für den Bildinhalt spezifisch sind. Inhaltsrepräsentierenden Bildmerkmale werden auch als Merkmalsvektoren [ChSh1996] bezeichnet. In der Auswahl dieser Vektoren liegt der Unterschied der verschiedenen Verfahren für digitale inhaltsbasierte Signaturen. Die Qualität dieser Verfahren lässt sich an der Robustheit gegen erlaubte Veränderungen und an der Empfindlichkeit gegen inhaltsverändernden Manipulationen feststellen. Bei der Generierung und der Überprüfung muss das Verfahren zur Merkmalsextraktion bekannt sein. Dazu müssen die verwendeten Inhaltsauszüge des Bildes robust genug gegen zugelassene Bildveränderungen sein. Ansätze zur inhaltsbasierten Signatur lassen sich unter [Cylin] finden. Weitere Kriterien sind die Berechnungs-Komplexität und die Länge der generierten Inhaltsauszüge.

3 Design von neuen Sicherheitsdiensten

Aus softwaretechnischer Sicht bieten sich mehrere Möglichkeiten an, die im vorigen Kapitel beschriebenen Sicherheitsmechanismen zu implementieren und in Anwendungen zu integrieren. Bereits existierende Applikationen können leicht mit Hilfe von Plug-ins erweitert werden, oder die fehlende Funktionalität wird einfach an den entsprechenden Ort der Kommunikationsstrecke in Form von Applets oder Servlets heruntergeladen. Auch die Middleware ermöglicht, fehlende Funktionalität in Applikationen nachträglich einzubauen, ohne die Struktur des Anwendungskodes verändern zu müssen. Bevorzugt werden insbesondere Komponentenmodelle wie XML/SOAP (Extended Markup Language/Simple Object Access Protocol) oder Enterprise Java Beans. Vor kurzem verabschiedete die Object Management Group die neue Standardversion des Common Object Request Broker Architecture (CORBA3), die u.a. das CORBA Component Model (CCM) beinhaltet [OMG], [CCM01]. CORBA hat als Middlewareplattform den Vorteil, daß die Implementierung der Funktionalität nicht auf eine Programmiersprache beschränkt ist. Ferner verfügt CORBA schon recht lange über ein Sicherheitskonzept, das schon vielfach in der Praxis angewendet wurde und dessen Stärken und Schwächen mittlerweile bekannt sind [ScLa98].

Dieses Kapitel ist folgendermaßen gegliedert. Im Abschnitt 3.1 wird der Einsatz der OMG-Standards als Referenzarchitektur motiviert. Abschnitt 3.2 erläutert kurz das CORBA-Modell. Im Abschnitt 3.3 werden die CORBA Security Services beschrieben. Im Abschnitt 3.4 werden die aktuellen sicherheitstechnischen Erweiterungen in CORBA aufgezeigt.

3.1 Motivation einer CORBA-basierten Sicherheitsarchitektur

CORBA vereinfacht die Entwicklung von verteilten Anwendungen, die auch von grösseren Dimensionen sein können. Dies trifft auch zu, wenn komplexe Sicherheitsfunktionen zu realisieren sind. Dafür sprechen die folgenden charakteristischen Eigenschaften der CORBA-Plattform:

- CORBA verfügt über ein generisches Objektmodell, das ein breites Spektrum von Applikationen unterstützt.

- Der gesamte Nachrichtenaustausch in einem CORBA-System erfolgt über die zentrale Komponente Object Request Broker (ORB), was die Komplexität der Sicherheitsproblematik reduziert.
- Die CORBA Security Services weisen sowohl vom Transportsystem unabhängige (z.B. Kerberos) als auch abhängige Sicherheitsmechanismen (z.B. SSL) auf.
- Die auf einer recht hohen Abstraktionsebene konzipierten Anwendungsschnittstellen der CORBA Security Services erlauben einen kompletten Austausch von einzelnen Sicherheitsmechanismen, so daß die Anwendungsarchitektur erhalten bleibt[5].

3.1 Das CORBA-Modell

Da es zu diesem Thema eine recht umfassende Literatur gibt, werden hier nur die wichtigsten Merkmale von CORBA erwähnt. Für Interessenten, die mehr über CORBA und CORBA-Programmierung verstehen wollen, werden im Literaturverzeichnis dieses Beitrags einige Ausgaben mit unterschiedlichen thematischen Schwerpunkten vorgeschlagen [HeVi99], [Bal00], [VoDu98], [VoRa99].

Grundlage der Standardisierungsaktivitäten der OMG ist die Object Management Architecture (OMA). Wesentliche Aspekte in OMA sind die Festschreibungen des Objektmodells, der Schnittstellen der Architekturkomponenten sowie der Syntax der Interface Definition Language (IDL).

Die zentrale Komponente in der Architektur ist der Object Request Broker (CORBA Core Services). Zusammen mit dem Language Mapping für die jeweilige Zielsprache der Anwendungsprogramme ermöglicht der ORB innerhalb eines verteilten Systems eine Clientanfrage (Request) an eine Objektimplementierung zu senden. Letztere ist in dem Server-Kode enthalten. Der ORB ist verantwortlich für das Lokalisieren der Objektimplementierung im Netz, das Übertragen der Daten zum Zielrechner und für die Zustellung des Requests an die Objektimplementierung. Unterscheiden sich die Darstellungsformate der Daten in der Hardware des Client- und Serversystems, so werden die Daten noch vor der Zustellung des Requests vom ORB in das lokale Darstellungsformat konvertiert. Der Request bzw. die vom Client spezifizierte Operation wird letzendlich von einem Objekt bearbeitet, indem die Argumente der Operation modifiziert, ein Resultat berechnet oder der Zustand des Objekts verändert wird. Erwartet der Client den Output der Operation, so erfolgt der Nachrichtentransfer ebenfalls über den ORB. Der ORB kann dem Client garantieren, daß die Operation ausgeführt wurde, es sei denn, die Operationssemantik war „best effort“. Konnte die Operation aufgrund eines aufgetretenen Fehlers in der Anwendung oder im ORB-System nicht erfolgreich zu Ende ausgeführt werden, liefert der ORB eine Ausnahmezustandsmeldung (Exception) zurück.

Neben dem Object Request Broker oder CORBA Core standardisiert die OMG drei weitere Dienstgruppen [OMG01]. Die Common Object Services oder auch CORBAservices genannt, stellen elementare betriebssystemähnliche Funktionen bereit, die ein virtuelles bzw. ein Objektnetz allgemein benötigt. Dazu zählen u.a. Naming, Life Cycle, Event, Transaction, Time, Trading und auch Security Services, die im nächsten Abschnitt vorgestellt werden. Die Gruppe CORBA Common Facilities stellen endnutzer-orientierte Dienste dar, die Gruppe CORBA

[5] gerinfügige Änderungen an den CORBA-Schnittstellen müssen jedoch vorgenommen werden, z.B. um Zertifikatattribute überprüfen zu können [LSAL01]

Domains enthält branchenspezifische Dienste (Finance, Healthcare, Telecommunications, Manufacturing, Business).

3.3 Die CORBA Security Services (CORBAsec)

Das Ziel der CORBA Security Services ist die Durchsetzung der Vertraulichkeit, Integrität und Verantwortlichkeit. Hierzu gehören Authentisierung des Client und Servers, Verschlüsselung und Integritätsprüfungen von Nachrichten, Zugriffskontrolle für den Aufruf von Objekten, Protokollierung von Systemaktionen, sowie die Erzeugung und Prüfung von Non-Repudiation-Beweisen. Das Sicherheitsmodell von CORBA ist verglichen mit dem Modell von SSL recht komplex. Die Gründe dafür sind einerseits der Umfang der Sicherheitsanforderungen an die CORBA Security Services [OM98] und andererseits die Komplexität des Objektinteraktionsmodells. In CORBA gibt es keine starren Client-Server-Beziehungen wie in den typischen SSL-basierten Kommunikationsszenarien. Ein CORBA-Objekt kann beide Rollen annehmen, so daß Operationsaufrufe sowohl an andere Objekte weitergeleitet werden, als auch in beiden Richtungen zwischen zwei interagierenden Objekten stattfinden können, z.B. wenn Callback-Operationen verwendet werden.

Nachfolgend werden die Hauptelemente des Sicherheitsmodells knapp erläutert. Ausführlichere Abhandlungen zu diesem Thema finden sich in [Bla00], [ScLa98], [La97], [La00]:

Credential Object: Credentials werden nach erfolgter Authentisierung beim lokalen Zugang erzeugt und enthaltenen die gesamte Identitäts- und Privileginformation eines Principals. Die Credential-Objekte dienen als Informationsquellen für nachfolgende Sicherheitsoperationen, wie Aufbau einer sicheren Session, Zugriffskontrollentscheidungen, Erzeugung von Audit Records oder Non-Repudiation-Beweisen.

PrincipalAuthenticator: Dieses Authentisierungsobjekt ist die Schnittstelle zur Anwendung (z.B. Login-Fenster), prüft die Nutzereingaben (Name, Passwort) und erzeugt das Credential-Objekt. Dieses Objekt bietet interne Schnittstellen zum Funktionscode des Sicherheitsmechanismus.

Security Context Object: Das Security Context Object repräsentiert auf jeder Seite die sichere Client-Server-Assoziation[6] und führt die vereinbarten Maßnahmen zum Schutz der Kommunikation aus (z.B. Verschlüsselung der Nachrichten).

Objektreferenz (IOR): Die Objektreferenz spielt bei der Objektkommunikation in CORBA die zentrale Rolle. Mit der Objektreferenz ist jedes Objekt eindeutig identifizierbar und kann mit Hilfe des ORB lokalisiert werden. Der Methodenaufruf eines Objektes erfolgt dann in der gleichen Weise wie bei lokalen Objektsystemen. Wird nun ein Objekt in einer Object Domain des Servers erzeugt, so wird auch die Security Policy-Information dieser Domain in der IOR festgehalten. Der Client kann anhand der IOR feststellen, welche Sicherheitsvorgaben von der Gegenseite verlangt werden.

***Security Policies und Domains*:**

Security Policies (siehe Abschnitt 2.1) können sich auf das System des Client oder des Servers beziehen. Diese werden in Policy-Objekten einer Security Domain verwaltet. Darüber hinaus definiert die CORBA Policies, die sich nur auf eine aktive Kommunikationsverbindung zwischen einem Client und einem Server-Objekt beziehen. Diese sind in der Objektrefe-

[6] es beinhaltet u.a. die Session Keys

renz eines jeden Objektes enthalten. Startet ein Client über den ORB-Mechanismus einen Objektaufruf, so kommt der Objektaufruf erst dann zustande, wenn der Policy-Enforcement-Mechanismus des ORB dies zulässt.

Policy Enforcement:

Das Policy Enforcement System setzt sich aus einem zentralen Objekt zusammen, das die Entscheidungslogik verkörpert. Bei der Entscheidungsfindung hat es den Zugriff auf die relevanten Informationsquellen wie Policy-Objekte, IORs, Context Setup Token, Credentials oder Zugriffskontrollisten.

Sicherheitsmechanismen:

Der Funktionscode der Sicherheitsmechanismen ist in den Objekten des CORBA Security Service gekapselt und nach außen unsichtbar. Der CORBA-Standard überlässt es den Produktherstellern, welche Mechanismen sie implementieren sollen, definiert aber Konformitäts- und Interoperabilitätskriterien. Angaben zu konkreten Sicherheitsmechanismen in der Spezifikation beziehen sich nur auf Authentisierung und Nachrichtenschutz, wobei Sicherheitsmechanismen wie Kerberos, SPKM, SESAME und SSL angesprochen werden. Der interne Zugriff auf den Funktionscode der Sicherheitsmechanismen erfolgt – außer bei SSL – über die GSS-API[7].

3.4 Erweiterungen der CORBA-Sicherheitsplattform:

Implementierung der SSL-Funktionalität

Da SSL ein transportabhängiges Sicherheitsprotokoll ist, das z.Z. noch nicht über GSS-API eingebunden ist[8], haben Produkthersteller von CORBA SSL über proprietäre Schnittstellen in die CORBA Security Services eingebunden. Sie bedienten sich der sogenannten „Interceptoren“. Die Interceptoren des Security Service können im Nachrichtenpfad des ORB den Aufruf und die Rückgabe der Operationen „abfangen“ und entsprechend der auf Client- oder Server-Seite herrschenden Policies den Aufruf bzw. die Ausführung der Operationen beeinflussen.

Eine andere Möglichkeit bieten „Pluggable Protocols“ des ORB. Mit deren Hilfe kann man das unter dem ORB befindliche Transportprotokoll durch ein anderes ersetzen. Damit ließe es sich bewerkstelligen, dass ein auf TCP basierter ORB dann auf SSL aufsetzen würde.

Anbindung der PKI-Funktionalität

Um die SSL-basierte ORB-Kommunikation auf Zertifikatbasis zu ermöglichen, ist eine PKI notwendig, die mindestens Zertifikate ausstellt und verifiziert. Die Anbindung kann

a) direkt im Funktionscode des SSL-Mechanismus, oder

b) auf Anwendungsebene über CORBA Interfaces

erfolgen. Da die Verwendung von Zertifikaten nicht nur für SSL, sondern auch für andere Applikationen, wie Wasserzeichen-basierte Authentisierungsmechanismen (siehe oben) vorgesehen ist, ist eine Anbindung auf CORBA-Ebene vorteilhafter. Das OMG-Konsortium hat dazu Ende des Jahres 1999 eine Schnittstellenspezifikation veröffentlicht, die den Status einer

---

[7] CORBA Security Service Architecture benutzt dazu eine eigene objektbasierte Schnittstelle, die auf GSS-API aufbaut.

[8] im GLOBUS-Projekt wurde dieser Ansatz vorgenommen

Revised Submission trägt und von dem australischen Forschungslabor DSTC Ltd. stammt. Die CORBA-Schnittstellen der PKI stellen sogenannte Wrapper für verschiedene Standardformate und Protokolle dar, so dass auch hier die CORBA-Applikationen nicht direkt auf den Funktionscode der PKI-Mechanismen zugreifen. Abbildung 5 zeigt dazu die Grundstruktur der PKI. Das Objekt *RegistrationAuthority* bearbeitet Anfragen bzgl. Zertifikatausstellung und Revokation, darüber hinaus führt es das Key Update und das Recovery von archivierten Schlüsseln aus. Das Objekt *CertificateAuthority* erbt die Schnittstelle von *RegistrationAuthority* und liefert zusätzlich CA-Zertifikate, CRLs und Objektreferenzen zu *CertificateStatusResponder* und *Repository*. Das Objekt *CertificateStatusResponder* führt Zertifikatstatusprüfungen aus, die sowohl Revokationslisten (CRLs) als auch Online-Verifikationsdienste (z.B. PKIX OCSP) unterstützen. Das Objekt *Repository* bietet Abfragedienste bzgl. unterstützter Verzeichnisschemata und Managementoperationen des Zertifikatauskunftdienstes.

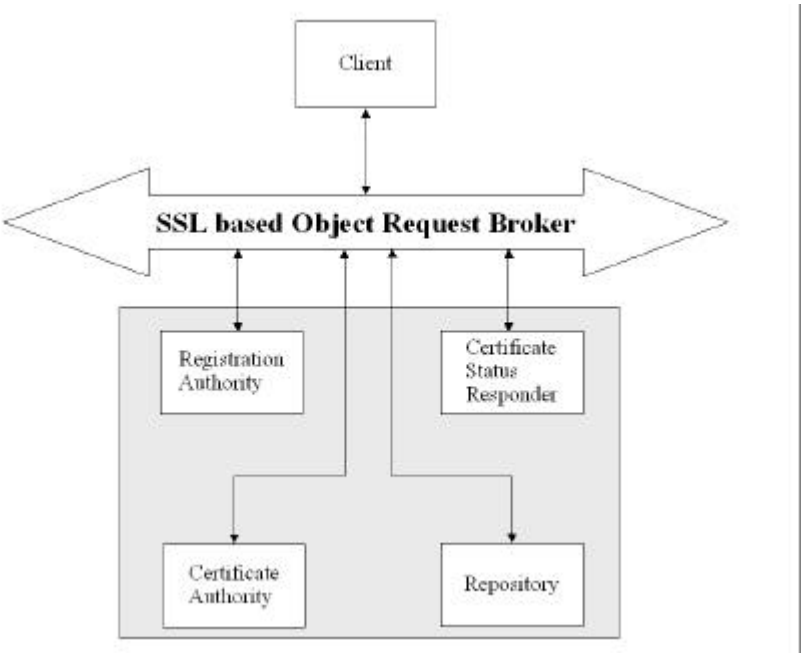

Abbildung 5: Grundstruktur einer PKI in einer CORBA-Umgebung

Es gibt eine Reihe von kommerziellen ORBs, die bereits das SSL-Protokoll unterstützen. Dazu gehören u.a. Inprise VisiBroker, Orbix von IONA und Dascom IntraVerse CORBA. Das Open Source-Produkt MICO [MICO], wurde kürzlich um Security Level 2 gemäß der CORBAsec-Version 1.7 erweitert (MICOsec) [ScLa00]. Der SSLIOP-Implementierung in MICO liegt OpenSSL/SSLeay zugrunde. MICO zusammen mit MICOsec erlaubt auch Anforderung, Überprüfung und Revokation von X.509-Zertifikaten auf der Basis der DSTC-Spezifikation, wozu zusätzlich eine Implementierung eines C++-Wrappers zur Anbindung von PKI-Algorithmen und kleine Erweiterungen in den CORBAsec-Schnittstellen (Repräsentation der CORBA-Sicherheitsattribute) notwendig waren[9] [LSAL01]. Schließlich war es auch möglich, die beiden Bibliotheken auf das Handheld iPaq H3600 von Compaq unter Linux erfolgreich zu portieren und testen, was neue Möglichkeiten eröffnet, Benutzerschnittstellen und sichere Applikationen auf der Basis des CORBA-Standards zu entwickeln [LSAL01].

[9] derzeit als eine prototypische Anwendung von MICO/MICOsec implementiert

4 Zusammenfassung und Ausblick

Wir stellen neue Konzepte und Methoden zu Verbesserung der Sicherheit beim elektronischen Handel vor. Dabei liegt das Hauptaugenmerk auf die Verwendung und Erweiterung bereits vorhandener Sicherheitsarchitekturen. Als Beispielszenarien werden Online-Shop, Nutzungsverfolgung und Ausweiswasserzeichen betrachtet, wobei das Ausweiswasserzeichen ausführlich diskutiert wird. Dazu werden allgemeine Sicherheitsszenarien analysiert und in ihre Bausteine zerlegt: Digitale Wasserzeichen, digitale Signaturen, PKI, Zeitstempeldienst und Non-repudiation werden identifiziert und vorgestellt. Die in den Beispielszenarien beteiligten Sicherheitskomponenten werden in CORBA übertragen. Der CORBA Security Service wird vorgestellt, Schwerpunkte sind die vorhandenen Sicherheitsmechanismen und deren notwendigen Erweiterungen: Die Implementierung der SSL-Funktionalität und die Anbindung von PKI-Funktionaltät. Dabei bietet es sich an, Digitale Signatur und Digitales Wasserzeichen als Applikationsobjekte (CORBA-Klienten) für Front-ends in Sprachen wie C/C++ oder Java zu implementieren, die lediglich auf die Dienste der CORBA-Plattform zugreifen, selbst aber keine Dienste im CORBA-System anbieten.

Nach diesen grundlegenden Überlegungen muss nun die praktische Umsetzung angegangen werden. Nächstes Ziel sind Implementierung und Evaluierung der beschriebenen Abläufe. Erst mit den daraus entstehenden Ergebnissen kann eine Beurteilung über Umsetzbarkeit und Sicherheit der vorgeschlagenen Abläufe abgegeben werden.

5 Literatur

[Bal00] Balen, H., Distributed Object Architectures with CORBA, Cambridge University Press, 2000, ISBN 0521654181

[Bla00] Blakey, B. (1999). CORBA Security. Addison-Wesley.

[CCM01] http://www.ditec.um.es/~dsevilla/ccm/

[ChSh1996] Chang, Shih-Fu und Schneider, Marc: A Robust Con-tent Based Digital Signature for Image Authentication, Proceedings of the International Conference on Image Processing, Lausanne, Switzerland, September 1996

[Cylin] http://www.ctr.columbia.edu/~cylin/watauth/sari.html

[DiBe01] Dittmann, Jana; Beier, Olaf: Ausweiswasserzeichen: Angriffspotential in Theorie und Praxis, eingereicht beim BSI-Kongress 2001

[Dig01] Digimarc, MediaBridge, www.digimarc.com

[DiSt99] Dittmann, Jana, Steinmetz, Arnd, Steinmetz, Ralf: Content-based Digital Signature for Motion Pictures Authentication and Content-Fragile Watermarking. In: IEEE Multimedia Systems '99, Int. Conference on Multimedia Computing and Systems, June 7 - 11, Florence, Italy, Vol. 1, pp. 209 - 213, ISBN 0-7695-0253-9, 1999

[Ditt00] Dittmann, Jana: Digitale Wasserzeichen, Springer Verlag, ISBN 3 - 540 - 66661 - 3, 2000.

[Fri1993] G. Friedman, *The trustworthy digital camera: Restoring credibility to the photographic image*, IEEE Transactions on Consumer Electronics, vol.39, pp 905-910, November 1993

[HeVi99] Henning, M., Vinoski, S., Advanced CORBA Programming with C++, Addison Wesley Professional Computing Series, 1999, ISBN 0-201-37927-9

[Ke00] Kehr, R., Untersuchung der Einsetzbarkeit mobiler Endgeräte für einen sicheren CORBA Object Request Broker, Dez. 2000

[KVH2000] M. Kutter, S. Voloshynovskiy, A. Herrigel: Watermark Copy Attack, n: Proceedings of SPIE: Security and Watermarking of Multimedia Contents II, 24 - 26 January, San Jose, California, USA, Vol. 3971, ISBN 0-8194-3589-9, pp. 371 - 381, 2000.

[La00] Lang, U. und Schreiner, R. (2000). "Flexibility and Interoperability in CORBA Security". Elsevier.

[La97] Lang, U. (1997). CORBA Security – Security Aspects of the Common Object Request Broker Architecture. M.Sc. Information Security 1996/1997.

[LSAL01] Lang, U., Schreiner, R., Alireza, A., Lorang, G., Eine Open-Source Implementierung der CORBA Sicherheitsdienste, 7. Deutscher IT-Sicherheitskongress, 15.01.2001

[MICO] http://www.mico.org

[OM00] OMG (2000). The Common Object Request Broker Architecture and Specification.

[OM98] OMG (1998). CORBA Security Services Specification.

[OM99] OMG(1999). CORBA Security Services Draft Version 1.7.

[OMG] http://www.omg.org

[OMG01] http://www.omg.org/technology/documents/formal/

[RPP99] Römer, K., Puder, A. and Pilhofer, F. (1999). MICO is CORBA, An Open Source CORBA 2.3 Implementation. Morgan Kaufman Publishers.

[ScLa00] Schreiner, R. und Lang, U. (2000). MICOSec User's Guide.

[ScLa00a] Schreiner, R. und Lang, U. (2000). MICO Reference Manual

[ScLa98] Lang, U. und Schreiner, R. (1998). Schutz und Trutz, Sicherheit in CORBA-basierten Systemen. iX 10/1998.

[VoDu98] Vogel, A.., Duddy, K., JAVA Programming with CORBA, Snd Edition, John Wiley & Sons, Inc., 1998, ISBN 0-471-24765-0

[VoRa99] Vogel, A.., Rangarao, M., Programming with Enterprise Java Beans, JTS and OTS], John Wiley & Sons, Inc., 1999, ISBN 0-471-31972-4