Höhere Fachschule für Wirtschaftsinformatik Luzern

Diplomarbeit

3. Juli bis 11. September 1998

Online Registrierung mit Verteilung des Client Zertifikates für eine gesicherte Authentifizierung

On the Internet, nobody knows you're a dog

Bosshard Stefan

Referent:
Fischer Peter
Kastanienbaumstr. 54
6048 Horw
Tel. 041 340 23 37

Koreferent:
Staub Johannes
Bachtelweg 5
8052 Zürich
Tel. 01 301 30 10

Autor:
Bosshard Stefan
Hauptstrasse 26
8414 Buch am Irchel
Tel. 052 301 30 29

Management Summary

Wie kann ein Internetkunde authentifiziert und einfach verwaltet werden? Wie können wir einen geschlossenen Internetdienst mit geheimen Daten anbieten? Diese Frage taucht oft auf.

Da unsere Unternehmung in absehbarer Zeit diverse medizinische Stammdaten einer auserwählten Kundschaft online gegen Entgelt zur Verfügung stellen wird, stellte sich auch hier die Frage nach der besten Technik. Die passwortgeschützte Uebermittlung ist zu unsicher, also bleibt nur der Weg zum digitalen Zertifikat (englisch: Certificate), welches mit der Identitätskarte oder dem Passport vergleichbar ist.

Mit der Realisierung des Prototyps „Online Registrierung mit Verteilung des Client Zertifikates für eine gesicherte Authentifizierung“ zeige ich detailiert dessen Ablauf auf. Im Vorfeld wurde bereits die benötigte Software-Evaluation durchgeführt, mit welchen Komponenten der Prototyp realisiert wird. Diese wurden zu Beginn installiert und es folgte ein Block mit Dokumentenstudium und Internetdurchforstung, um sich das allgemeine Wissen über die Zertifizierung anzueignen. Als das Grundwissen und die abstrakte Reihenfolge dieser Technologie bekannt war, begann eine erste Testphase mit dem Anfordern eines Client-Certificates über die Testumgebung mit dem Protokoll HTTP. Dies löste beim Certificate Server ein Certificate aus und ohne grosse Probleme hatte ich nach kürzester Testzeit ein Certificate der MediData AG in meinem Browser installiert.

Die grössere Problematik erwies sich beim Einrichten des gesicherten, privaten Internetbereiches, sowie dem getrennten Administrations-Web, das eine Administrierung übers Web erlaubt. Auf Web-, Verzeichnis- oder Dateiebene kann nun entsprechend ein gesicherter Zugriff erfolgen. In diesem Fall wird jeweils das ausgestellte Client Certificate einem Account zugeteilt. D.h., sobald der Internetanwender eine entsprechende Internetseite anfordert, verlangt der Webserver das spezifische Certificate, bevor die Seite übers Netz freigeschaltet wird. Ist das Certificate noch gültig und die Authentifizierung erfolgreich, werden die Daten gesichert übertragen. Die ganze Verschlüsselung erfolgt mittels einem „unbekannten“ Algorithmus. Selbstverständlich macht diese Vorgehensweise nur Sinn, wenn jeder Certificate-Inhaber vorsichtig mit seinem digitalen Bytestrom umgeht und diesen keiner weiteren Drittperson aushändigt.

Wie mit dem Auftraggeber beschlossen, realisierte ich nun den Vorgang wie folgt: Die Registrierung wird in die Datenbank geschrieben. Der Kunde signiert das ausgedruckte Registrierungsformular und sendet dieses zusammen mit einer Kopie eines Ausweises an den Certificate Aussteller MediData AG. Bei Eintreffen dieser Papiere wird entschieden, ob dem Antragssteller der Zugang gewährt werden soll. Falls positiv, gibt der Administrator den Status für die Generierung des Certificates frei. Das Administrationstool wurde mit einer Scriptsprache entwickelt und kann übers Internet benutzt werden. Dieses Programm sendet dem Kunden ein EMail mit Angabe der entsprechenden Internetadresse um das Certificate abzuholen und es im Browser zu installieren.

Der Certificate-Inhaber muss nur noch einen kleinen Test starten, der das Certificate zu einem bestimmen Benutzer Account mappt. Dem Kunden ist ab diesem Zeitpunkt der Dienst freigeschaltet. Ist das Certificate einmal generiert und abgeholt, ist die nochmalige Ausstellung gesperrt.

Wie kann das Certificate wieder deaktiviert werden? Erstens wird beim Generieren eine Gültigkeitsdauer festgelegt, welche jedesmal automatisch überprüft wird. Zweitens hat der Administrator selbstverständlich die Möglichkeit, übers Tool das entsprechende Mapping zu löschen.

Das Wissen und Know How für eine gesicherte Authentifizierung ermöglicht nun, den Prototyp auszubauen. Es sollte nun möglich sein, in angemessener Zeit ein produktives Umfeld zu entwickeln und somit die Sicherheit zu erhöhen und den Verwaltungsaufwand bezüglich Berechtigungs-Vergabe zu reduzieren.

Vorwort

? Fragezeichen, dieses Symbol soll zeigen, wie ich mich fühlte, als ich erfuhr, was ich realisieren sollte.

Nach den ersten Vorschlägen für das Thema meines Vorgesetzten versuchte ich einige Ausweichmanöver und brachte Gegenvorschläge für Projekte, die mir vertrauter und realisierbarer schienen. Doch Johannes Staub liess nicht locker. Und so blieb mir nichts anderes übrig, als mich langsam an den Gedanken gewöhnen zu müssen, mich mit dieser noch wenig bekannten Materie auseinanderzusetzen. Nach dem Durcharbeiten verschiedener Konzepten und Beschreibungen von Fremdfirmen ging langsam der Sinn und Zweck dieses Prototypings hervor. Und ich begann das Thema immer mehr zu „lieben“.

Unzähliges Ausprobieren, Aendern, Nachlesen und viel Geduld vom Arbeitgeber und mir, erzielten dann das gewünschte Produkt. Und ich bin zur Erkenntnis gekommen, dass mir solche Aufgabenstellungen enorm viel Spass bereiten, denn was gibt es schöneres, als vor ein Problem gestellt zu werden, welches nicht nach Methode bzw. Verfahren „0815“ gelöst werden kann, sondern wo es noch ein wenig „Pioniergeist“ erfordert.

Die Zeit rannte davon in diesen zehn Wochen, und es gab Tage, da kam ich keinen Schritt weiter. Doch dafür war das Glücksgefühl viel höher, wenn man dann nach dieser Zeit das Zwischenziel erreichte, obwohl diese Schritte einen vielleicht nur ein Prozent näher ans Ziel brachten. Tja, und dann gab es die Tage, an denen man am liebsten wieder Vollzeitstudent gewesen wäre und man die warmen Sommertage am Vierwaldstättersee geniessen wollte. Life is going on...

An dieser Stelle möchte ich mich bei der Unternehmung MediData AG herzlich für die Aufnahme als Praktikant bzw. Werkstudent bedanken. Denn in der heutigen Zeit ist es nicht selbstverständlich, als praxisunerfahrener Mitarbeiter eingestellt zu werden. Ausserdem darf ich wohl sagen, dass ich in diesem Jahr viel dazu gelernt habe, sei dies im Informatik-, unternehmerischen- oder gar wirtschaftlichen Bereich und ich bin froh, dass ich mich schlussendlich wieder für den wohl momentan brisanten aber auch interessanten Arbeitsbereich im Gesundheitswesen entschieden habe.

Einen speziellen Dank möchte ich meinem Koreferenten Johannes Staub und meinem Büronachbarn Roman Arnold aussprechen, für die gute Zusammenarbeit und die zur Verfügung stellen der Infrastruktur, Ausbildung und vieles mehr! Und nicht vergessen möchte ich meine Eltern, Freundin und all diejenigen Personen, die meine Rechtschreibung überprüft haben.

Und zu guter Letzt ein Dankeschön an die Innerschweiz. Wieso? Weil es als Zürcher Weinländer einfach fantastisch ist, seine Freizeit in so einer schönen Landschaft zu verbringen und Energie aufzutanken für all die Bit und Bytes...

Diese neue Internetaera wird über kurz oder lang einschlagen. Also, viel Spass beim Lesen, und seien Sie sich immer bewusst: Ein völlig sicheres Netz ist unrealistisch, deshalb braucht es von allen Benutzern Fairness. Denn was wären wir Entwickler ohne dieses enorme Informationsnetz? Dieses Arbeitswerkzeug ermöglicht uns eine „weltweite“ Frage und Anwort Dienstleistung.

Mein Motto: Reden ist Silber, Entwickeln ist Gold!

Inhaltsverzeichnis

1 Portrait MediData AG

Mein Auftraggeber MediData AG mit Sitz in Luzern besteht aus Hauptaktionären der grössten Versicherer der Schweiz (Krankenkassen und Unfallversicherer). Gründungsgedanke im Jahre 1994 war die Realisierung des „Elektronischen Datenaustausches“ im Gesundheitswesen und die Sammlung, das Pflegen und zur Verfügung stellen von Stammdaten wie Tarife, Medikamentenpreise, Diagnosecodes und vieles mehr.

Unternehmensziel war und ist sicherlich, gemeinsam eine schnelle Kommunikation zwischen Leistungserbringer und Versicherer aufzubauen, Verwaltungskosten zu senken und die Papierflut zu stoppen. Nun, die Idee an sich ist hervorragend, nur die Umsetzung in einem der heute wohl schwierigsten „Wesen“ ist einiges problematischer!

Im Gegensatz zum elektronischen Datenaustausch, der unter anderem das Senden und Empfangen von Rechnungen und Kostengutsprachen ermöglicht, ist das Verwalten von Stammdaten in unserer Datenbank MediFrame einer grösseren Nachfrage gewidmet. Dieses Produkt wird in sogenannten Flat-Files dem Kunden zur Verfügung gestellt. Dieser kann die Daten in sein System importieren und benutzen. Selbstverständlich musste auch die Online-Abfrage übers World Wide Web ins Angebot aufgenommen werden und deshalb bietet das Unternehmen seit Sommer 1998 dieses Produkt einer geschlossenen Kundengruppe an.

1.1 Unternehmensziel

Das Unternehmensziel der MediData AG ist der Aufbau und Betrieb eines Branchenweiten Informationsnetzes im Gesundheitswesen, das die modernen Telekommunikationstechnologien allen Partnern verfügbar macht.

1.2 Tätigkeitsgebiete

Die Tätigkeitsgebiete sind:

- Elektronischer Datenaustausch zwischen allen Partnern im Gesundheitswesen in den Bereichen Finanzen (Faktura, Kostengutsprache, Zahlungsverkehr), Medizin (Labormeldungen, Berichte) und Logistik (Handel mit Medikamenten und medizinischen Hilfsmitteln)
- Einheitliche Versichertenidentifikation
- Datenbanken mit Tarifen und anderen Stammdaten des Gesundheitswesens
- Elektronische Unfallmeldung

1.3 Anwendungen für den Kunden

Produkte Anwendungen für den Kunden:

- Elektronische standardisierte Abrechnungen medizinischer Leistungen zwischen den rund 35'000 Leistungserbringern und Versicherern
- Elektronische Durchführung einer fallbezogenen Kostengutsprache zwischen Leistungserbringern und Versicherern
- Elektronische Uebertragung medizinischer Berichte und Resultate zwischen den einzelnen Gesundheitspartnern (z.B. Laborauftrag und –befund zwischen Arzt und Labor, oder Uebertragung von Arztberichten)
- Bezug aller im Gesundheitswesen notwendigen Stammdaten von einer zentral durch MediData geführten Datenbank (z.B. Tarife, Medikamentenliste, Zahlstellenregister, Diagnosecodes)
- Zugriff auf verschiedenste medizinische und administrative Online-Informationsangebote, die dem Kunden die alltägliche Arbeit erleichtern (Bsp. Stammdaten)
- Im Aufbau: Weitere Anwendungen wie Electronic Mail oder elektronische Diskussionsforen zur Erleichterung der elektronischen Kommunikation zwischen den einzelnen Partnern

In Tab. 1 sind die wichtigsten Kenndaten aufgelistet.

Tabelle 1: Kenndaten MediData AG (Stand: August 1998)

Was	Daten
Gründung	1. Oktober 1994
Rechtsform	Aktiengesellschaft
Standort	Suva Gebäude Rösslimatt, Luzern
Anzahl Mitarbeiter	18 Mitarbeiter Stellenprozent 1060 %
Aktionäre	• Christliche Soziale Schweiz (CSS) • Die Eidgenössische • Groupe Mutuel • Helsana Versicherung • Intras Krankenkasse • Konkordia • Oeffentliche Krankenkassen (Oekk) • Sanitas • Swica • Visana • Basler Vesicherung • Helvetia Patria • Mobiliar Versicherung • Schweiz. Unfall Versicherungsanstalt (SUVA) • Winterthur Versicherung • Zürich Versicherung

2 Aufgabenstellung

Ziel ist es, einerseits das Wissen und Know How über das Anfordern, Generieren und Zustellen eines Client-Certificates (Bytestrom bzw. eine Art Datei) für eine gesicherte Authentifizierung des Internetbenutzers oder allgemein übers Client-Certificate zu erarbeiten. Ausserdem soll aufgezeigt werden, wie die Verschlüsselung funktioniert, damit sich der Leser ein Bild machen kann, wie sicher diese Variante ist.

Andererseits soll von Grund auf in einer Testumgebung ein Prototyp für die „Online Registrierung mit Verteilung des Certificates für eine gesicherte Authentifizierung“ realisiert werden. Der Auftraggeber (MediData AG) will anhand des Prototyps und der Dokumentation (Diplomarbeit) den Prototyp weiterentwickeln. Nebenbei, der Prototyp soll nur hausintern in einem Testnetzwerk getestet werden können (Intranet).

2.1 Aufgabenabgrenzung

Die Hardware, d.h. der Testserver, wurde vor Beginn der Arbeit evaluiert, eingekauft und mit dem Betriebssystem und der üblichen Software installiert. Ebenfalls im voraus wurde die Softwareevaluation für die Komponenten erledigt, d.h. zu Beginn der Arbeit wurde bereits festgelegt, mit welcher Software der Prototyp entwickelt werden soll, jedoch mit der Option, bei Zwischenfällen kurzfristig während der Arbeit auf ein anderes Produkt zu wechseln. Deshalb wird in dieser Arbeit keine Gegenüberstellung von Produkten etc. aufgezeigt. Die Firma MediData AG hat sich vor einiger Zeit für die Produktegruppe von Microsoft entschieden, weil wir denken, dass sich unsere Kunden ebenfalls in diesem Bereich bewegen werden.

3 Problembeschreibung

Wie kann ein User, der das Internet nützt, authentifiziert werden und dies zu einem normalen Kosten Nutzen Aufwand? Wie weiss ich, wer sich da wirklich einloggt? Wie kann ich einzelne Internetseiten auch ausserhalb eines Intranets sichern? Diese Frage stellt man sich schon längere Zeit.

„The Internet has long fought a battle to maintain security of resources in the face of unlimited access from the outside world.“ [1]

3.1 Passwort-Verfahren

Das Passwort-Verfahren ist eine verbreitete Variante, die einfach zu administrieren ist. Sie ist aber auch unsicher, wird doch viel zu oft das „geheime“ Passwort aufgeschrieben. Sind die Passwörter zu einfach und zu kurz ist es ein Kinderspiel für einen Hacker. Ausserdem werden die Passwörter mit einer viel zu geringen Verschlüsselungstechnik übers Netz gesandt und viele Online Registrierungen werden ohne irgendeine Ueberprüfung vorgenommen und freigeschaltet.

3.2 Biometrisches Verfahren (Erkennung vom Lebenden)

Dieses Loginverfahren, sei dies durch Erkennung von Fingerabdruck, Auge, Stimme, Haar oder was auch immer, ist in der heutigen Zeit machbar aber viel zu teuer und zu empfindlich, um wirklich den Markt in absehbarer Zeit flächendeckend zu erobern.

3.3 Challenge-Response Protokolle

Das Problem, dass ein Passwort (oder ein geheimer Schlüssel allgemein) bei der Uebertragung abgehört werden kann, kann durch ein Challenge-Response Protokoll gelöst werden: Das System, welches das Passwort kennt, wählt einen Challenge-Wert zufällig (jedesmal anders) und verlangt vom Benutzer einen Wert (Response), der sich aus dem Challenge nur bei Kenntnis des Schlüssels berechnen lässt. Die Fähigkeit, den Responsewert korrekt zu berechnen, beweist die Kenntnis des Schlüssels (oder des Passwortes) [2].

3.4 Magnet- und Chipkarten

Auch dieser externe Zusatzkartenleser ist heute wohl noch umständlich und nicht ganz kostengünstig. Dennoch könnte sich in naher Zukunft diese Variante durchsetzen, denn die Kreditkarte ist wohlbekannt und weit verbreitet und jede Person gibt Acht auf diese Karte.

[1] Literatur (Goulde, zit. in [ECS], S. 501).

[2] Literatur [MAK]

3.5 Digitales Certificate

Was ist das? Kurz gesagt: Eine Zeichenlänge, welche Client-seitig im Browser oder in einer anderen Applikation installiert wird. Durch die „amtliche“ Ausstellung eines Client-Certificates gegen Vorweisung eines Ausweispapieres kann schon hier mal sortiert werden, wer eines erhält und wer nicht. Ausserdem verpflichtet sich der Inhaber, Sorge dazu zu tragen. Nun, der Trick liegt darin, dass beim Uebertagen des Certificates und des Public Keys, der wohl wichtigste Teil - der Private Key - nicht übers Netz gesandt wird und sich der Hacker somit mit nur einem Teil des Verschlüsselungscodes begnügen muss.
Ein weiterer Vorteil ist, dass dieses Vorgehen kostengünstig ist und keine zusätzliche Programme und Hardware benötigt. Doch Details und Informationen später!

4 Ausgangslage

4.1 System

Die MediData AG ist schon seit längerer Zeit im Internet unter http://www.medidata.ch präsent. Seit Mai 1998 läuft nun für eine geschlossene Usergruppe der Pilotbetrieb von „MediFrame Online“. Dieses Produkt ermöglicht dem Kunden, Tarife, Diagnosecodes etc. online abzufragen bzw. nachzuschlagen. Die Zugriffsberechtigung wird mittels UserID und Passwort-Eingabe verwaltet. Als Web Server benutzen wir Internet Information Server 4.0 von Microsoft auf einem Windows NT Server 4.0.

D.h., eine Certificate Generierung und Authentifizierung besteht nicht und das Wissen zu dieser Thematik fehlt.

4.2 Komponenten

Die Software- und Hardwareevaluation wurde im Vorfeld durchgeführt und abgeschlossen, jedoch mit der Option auf eine kurzfristige Umstellung auf ein Fremdprodukt. Die notwendigen Programme und Testversionen wurden ebenfalls schon vorzeitig bestellt oder heruntergeladen. Um den Prototyp zu entwickeln und zu testen, wurde mir ein Windows NT Server 4.0 eingerichtet und ans LAN angeschlossen. Als Clientmaschine wurde ein Personalcomputer mit Windows NT Workstation 4.0 gewählt.

5 Zielbeschreibung

5.1 System

Konkret sollte der Ablauf der Prototyp Registrierung in etwa folgendermassen aussehen:

Der Kunde soll sich online, sprich mit einem aktuellen Netscape Navigator Browser, registrieren lassen können.

Die übermittelten Daten werden serverseitig in eine Microsoft (MS) Access Datenbank eingetragen. Ein Mail gibt dem Kunden die URL bekannt, wo das Certificate abgeholt werden kann. Beim Abholen generiert der Certificate Server 1.0 von Microsoft ein Certificate, welches dem Kunden zugestellt und in den Browser installiert wird.

Ab diesem Zeitpunkt kann der Kunde unser „online Produkt“ produktiv nutzen, da eine sichere Authentifizierung stattfindet. Mittels einem Check des Client-Certificates, welches vom Server (Win NT 4.0 Server) kontrolliert wird, wird das Produkt freigeschaltet.

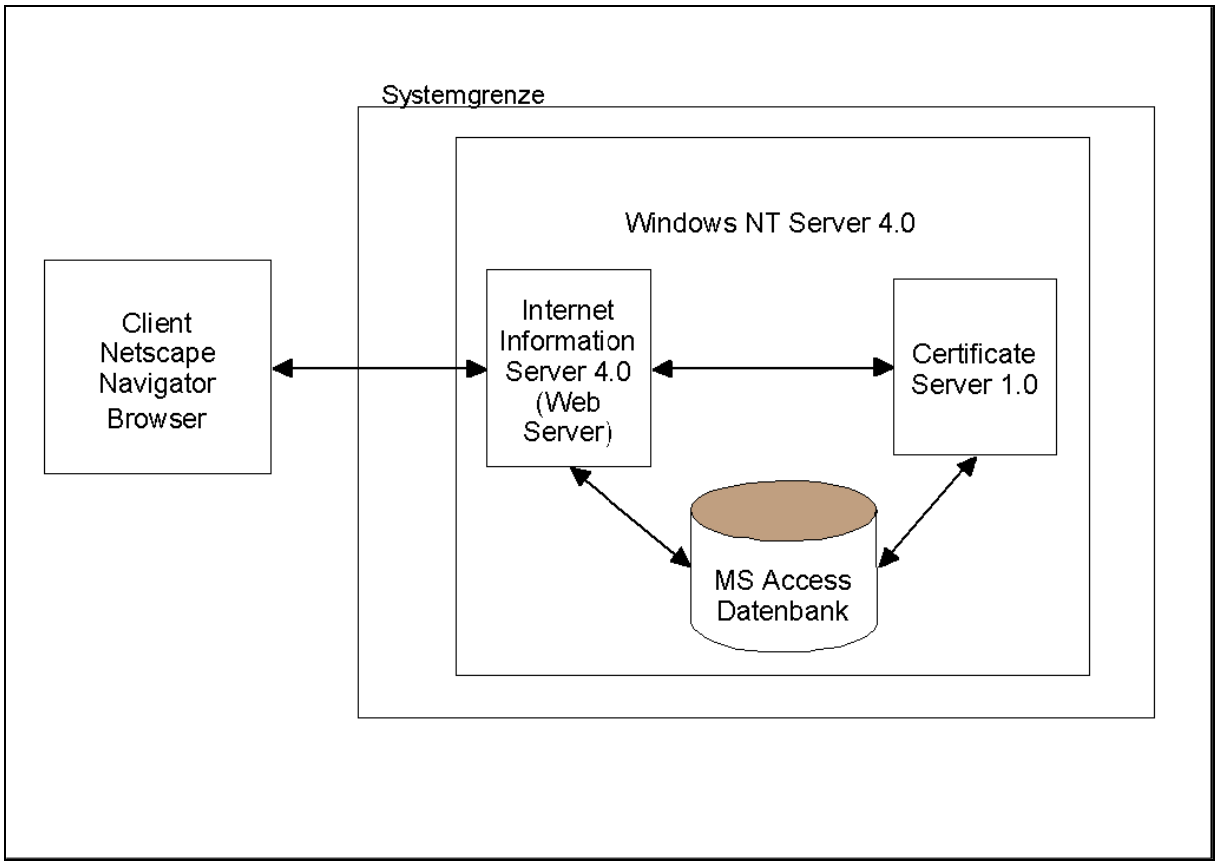

Abbildung 1: Prototyp

In Abb. 1 ist ersichtlich, welche Komponenten in dieser Arbeit näher beschrieben werden. Details zu den genauen Verbindungen folgen.

5.2 Komponenten

Folgende Komponenten wurden vor Beginn der Arbeit für den Prototyp und deren Entwicklung ausgewählt (Tab. 2 und 3).

Tabelle 2: Software Komponenten im Prototyp

Komponente	Produkt
Server	Windows NT Server 4.0 von MS
Web Server	Internet Information Server 4.0 von MS
Certificate Server	Certificate Server 1.0 von MS
Datenbank	Access Datenbank 97 von MS

Tabelle 3: Entwicklungskomponenten

Komponente	Produkt
HTML Editor	Frontpage Editor von MS und Texteditor UltraEdit32

6 Vorbereitung Prototyp

Um sich erst einmal in diesem Thema zurechtzufinden, musste ich mir vorgängig viel Fachwissen aneignen. Erst Dank Konzepten von verschiedensten Unternehmen und dem Internet, war es mir möglich, einen gewissen roten Faden zu finden.

6.1 Projektplan

Um das Projekt zeitlich und reihenfolgenmässig zu koordinieren, arbeitete ich mit einem Projektplan (siehe Tab. 4). Alle Begriffe und Abkürzungen werden in den einzelnen Kapiteln erklärt.

Tabelle 4: Projektplan

Nr	Beschreibung	Termin bis	Erledigt am
1	Wissen und Technik nachforschen, erarbeiten und dokumentieren.	Während ganzer Arbeit	15.08.1998
2	Auf Client (Win NT Workstation bestehend) zusätzlich folgende SW installieren: • PcAnywhere32 8.0 (Verbindung zu Host SRVWISWEB TCP/IP 192.168.1.9)	06.07.1998	06.07.1998
3	Hardware Testumgebung Server installieren mit folgenden Komponenten: • Windows NT Server (Name SRVWISWEB) • MS Internet Information Server 4.0 WebServer (Teil von Option Pack) • Certificate Server von MS (Teil von Option Pack) • PcAnywhere32 8.0	06.07.1998	06.07.1998
4	Konfigurieren des WEB Servers. Erste Tests im Intranet, ob der Zugriff von Client Browser auf Web SRVWISWEB funktioniert. http://srvwisweb .	07.07.1998	07.07.1998
5	Certificate Authority Certificate in Web Server installieren (IE 4.0).	08.07.1998	08.07.1998
6	Tests, mit den einzelnen HTML-Seiten, ob Zugriff verweigert wird oder nicht, wenn Client Browser kein Certificate besitzt. Ausserdem installieren von CA und Client-Certificates auf Client Browser.	07.07.1998	13.07.1998
7	Für Server Certificate Key Pair generieren lassen im Key Manager.	08.07.1998	08.07.1998
8	Test Client-Certificate von Swisskey herunterladen und probieren, mit diesem auf Web Server zu kommen http://www.swisskey.ch/webcert.html?	10.07.1998	14.07.1998

Nr	Beschreibung	Termin bis	Erledigt am
9	Ein virtueller öffentlicher Ordner und ein virtueller privater Ordner einrichten und dem privaten nur Zugriff mit einem Certificate gewähren. Für erste Tests.	13.07.1998	15.07.1998
10	Planung und Modellierung des Prototypes.	10.07.1998	15.07.1998
11	Zwei NT User Accounts eröffnen.	13.07.1998	15.07.1998
12	Client Certificate mappen auf Win NT Account und testen.	13.07.1998	15.07.1998
13	HTML Registrierungsformular plus Ausdruck erstellen – automatisch Certificate installieren.	17.07.1998	20.07.1998
14	Zwischenschritt: Daten in DB importieren, aber noch keine Certificate Generierung, dann manuelle Statusänderung in Access DB und Mitteilung an Kunde der Abhol-URL. Mit dieser und Passwort wird Certificate generiert und installiert.	17.07.1998	21.07.1998
15	Zusätzlich bei obengenanntem Vorgang Certificate auf Server abspeichern, wird für manuelles Mapping benötigt.	17.07.1998	22.07.1998
16	Bestehendes Mail Modul im Internet finden und testen.	20.07.1998	28.07.1998
17	Client-Certificate „Maschinenabhängig“ machen. Lösung Cookie.	28.07.1998	03.08.1998
18	Automatisches Win NT Mapping.	28.07.1998	04.08.1998
19	Entwickeln aller HTML- und ASP-Seiten für Certificate Request, Generierung etc.	31.07.1998	07.08.1998
20	Entwickeln aller HTML- und ASP-Seiten für Administrationsweb.	14.08.1998	21.08.1998
21	Schluss-Konfiguration beider Webs und Sicherheitseinstellung der einzelnen Directories.	17.08.1998	25.08.1998
22	Testphase und Anpassungen.	18.08.1998	28.08.1998
23	Review Arbeitgeber.	?	31.08.1998
24	Anpassungen und Schlusstest des Prototyps.	21.08.1998	02.09.1998
25	Dokumentation überarbeiten und abschliessen.	04.09.1998	10.09.1998

7 Certificate-Technologie

7.1 Grundidee

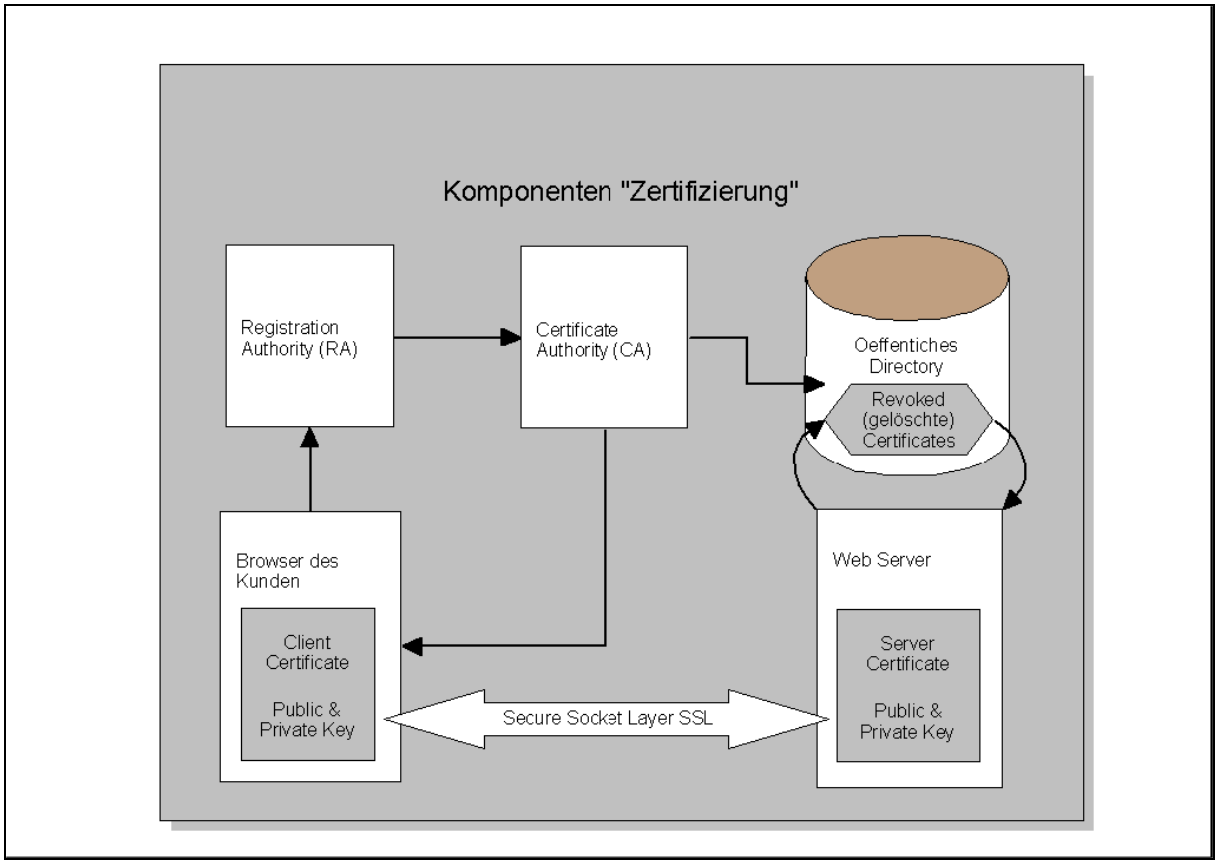

Abbildung 2: Komponenten „Zertifizierung“

In Abb. 2 sind grob alle Komponenten abgebildet, die bei der Zertifizierung Einfluss haben. Die Pfeilverbindungen symbolisieren in etwa, welche Komponente in einer Beziehung stehen.

Die Grundidee ist eine einmalige Benutzeranmeldung („Single-User-Log-on“) und ein einziges sich ausweisen, um sich auf den verschiedenen Web-, Mail-, Proxy-, Directory- und Catalog-Servern anzumelden. Systemverwalter müssen nun nicht mehr endlose Listen mit Benutzernamen und Passwörtern auf den einzelnen Servern pflegen, sondern konfigurieren einfach einen Server, der nur Certificates (Dies ist einfach erklärt – ein Bytestrom bzw. eine Datei, die entsprechende öffentliche Daten beinhaltet und wie ein „Passwort“ übermittelt wird) akzeptiert, die von einer bestimmten Stelle unterzeichnet wurden. Da die Ueberprüfung von Benutzernamen und Passwörtern nicht die zuverlässigste Methode ist, werden digital Certificates, die den Benutzer ausweisen, immer wichtiger für Unternehmungen und Privatkunden, die ihre Geschäfte online abwickeln. Certificates haben aber noch weitere Vorteile gegenüber der herkömmlichen Bestätigung durch Benutzername und Passwort: Es werden keine vertraulichen Informationen über das Netzwerk übertragen. Anders als Passwörter enthalten Certificates ausschliesslich öffentliche Informationen. Certificates, die über das Netzwerk übertragen werden, stellen also keine Sicherheitsbedrohung dar [3].

[3] Literatur [NCS]

7.1.1 Arten

Momentan spricht man von vier diversen digitalen Internet Certificates:

1. ***Certification authority certificate***
   Oberste Hierarchie aller Certificates. Die CA sind Organisationen, die ermächtigt sind, Certificates auszustellen. Darunter fallen auch unternehmensinterne CA's, die eigene Certificate Servers betreiben.

2. ***Server certificate***
   Damit kann sich der Hersteller bzw. Betreiber eines Internetauftrittes ausweisen und der Server kann sich gegenüber dem Client authentifizieren.

3. ***Personal certificate***
   Auch genannt Client-Certificate.

   Diese können weiter eingesetzt werden:
   - Web Browser benötigen Certificates für die Authentifizierung von Usern und für SSL Verbindungen.

   - Secure Electronic Transaction (SET) Standard benötigt Certificates für Authentifizierung von Kreditkarten-Transaktionen (diese Certificates können nicht wie gewöhnliche Client-Certificates verwendet werden. SET wurde absichtlich so spezifiziert, dass die Certificates nicht für die anderen Standard Internet-Kommunikationsprotokolle wie SSL oder S/MIME sondern nur für SET-Transaktionen verwendbar sind). Die SET-Certificates werden durch Organisationen zertifiziert, die eigentlich Kreditkartenherausgeber sind.

   - Der S/MIME Standard für „secure eMail" benutzt Certificates für die Authentifizierung von Sender und Empfänger von eMail.

   - EDIFACT Certificate.

4. ***Software publisher certificate***
   Wird verwendet für „signed Java Applets". Somit weiss der Internet Benutzer, wenn er eine HTML Seite mit einem Java Applet öffnet, wer diese Software entwickelt hat (Software [SW]-Unterzeichnung). Denn das Risiko ist erheblich und man hat keine grosse Sicherheitsüberprüfung [4].

   SW-Unterzeichnung: Unterzeichnet Certificates mit der digitalen RSA-Unterschrift, dem Industriestandard (RSA-Unterschrift mit MD5- oder SHA1-Hash). Unterstützt Unterschriftenschlüssel von max. 1024 Bits [5].

[4] Literatur [UBS]
[5] Literatur [NCS]

7.1.2 Einsatzbereich

- *Internet Banking*
  ⇨ Zahlungsaufträge online abwickeln

- *Closed User Groups*
  ⇨ Gewisse Internetseiten nur einem ausserwählten Kundenkreis zur Verfügung stellen

- *Sicheres EMail*
  ⇨ Nur der gewünschte Empfänger kann die verschlüsselte Nachricht lesen

- *Internet Zugang*
  ⇨ Oeffentlicher Zugang

- *E-Shopping*
  ⇨ Einkaufen online

- *E-Commerce allgemein*
  ⇨ Ganzes Umfeld im elektronischen Markt

7.2 Authentication

„Authentifizieren“ heisst beglaubigen, die Echtheit bezeugen.

Den Benutzer auffordern, sich ausweisen zu müssen, heisst Authentication.

Der Gebrauch von physischen Dokumenten, um (in der echten Welt) eine Authentifizierung zu erreichen, existiert schon seit längerer Zeit. Beispiel: wenn man einen Check für eine eben gekaufte Ware ausstellt, verlangt der Verkäufer ein Ausweispapier, wie zum Beispiel die Fahrerlizenz. Dieses Ausweispapier wird nun benutzt, um dem Verkäufer glaubhaft zu machen, dass ich wirklich derjenige bin, als den ich mich ausgebe. In diesem Fall vertraut der Verkäufer dem Staat, der dieses Papier mit den wohl gültigen Identitätsinformationen ausstellte! Ein anderes Beispiel ist die Benutzung des Passports beim Ueberschreiten der Landesgrenze. Der Zöllner vergleicht das Foto im Passport mit dem Passportinhaber und kann somit die Identität feststellen, da auch er dem Staat vertraut, dass richtige Angaben erfasst worden sind. Merke, dass in beiden Beispielen ein Level an Vertrauen gegenüber dem CA vorhanden sein muss [6].

Denn auch heutzutage ist in der „echten Welt“ eine 100 % Authentifizierung fast ausgeschlossen! Und deshalb können wir dies auch nicht von der „virtuellen“ Welt erwarten und wir müssen ein gewisses Mass an Vertrauen beisteuern, damit die komplexe Welt Internet auch in Zukunft noch „boomen“ wird.

[6] Literatur [MSS]

7.2.1 Vertrauen

Ich möchte kurz den Unterschied zwischen der Sicherheit der Uebermittlung und dem Vertrauen gegenüber dem Betreiber des Servers an einem Beispiel der Übertragung von Informationen wie Kreditkartennummern aufzeigen:

Um das Risiko zu reduzieren, dass ein Uebermittler auf die Kreditkarteninformationen zugreift, können sie ihre Kreditkartennummer auf einem Browserformular eintragen und es verschlüsselt via Internet über eine gesicherte Verbindung übertragen. Die heutige angebotene Verschlüsselungs-Technologie hilft, Geschäfts- und andere Transaktionen vor Entwendung und Betrug zu schützen, während Informationen über Internet-Computer übertragen werden.

Verschlüsselte Kommunikationen zerstreuen jedoch nicht alle Bedenken der Internet-Benutzer. Es ist beispielsweise wichtig, dass Sie das Gefühl haben, dem Serververwalter Ihre Kreditkartennummer anvertrauen zu können, bevor Sie eine Geschäftstransaktion vornehmen. Sicherheitstechnologien schützen zwar die Routen der Internet-Kommunikation, schützen Sie aber nicht vor unehrlichen oder unvorsichtigen Menschen, mit denen Sie eventuell Geschäfte abschliessen.

Genauso wie bei Telefongesprächen sind Sie sich vielleicht auch hier nicht sicher, dass niemand gehört hat, wie Sie Ihre Kreditkartennummer durchgegeben haben (Vertraulichkeit) und dass Ihr Gesprächspartner für die Firma arbeitet, von der Sie kaufen möchten (Authentizität), aber Sie müssen auch bereit sein, der Person und der Firma zu vertrauen [7].

[7] Literatur [NCSH]

7.3 Inhalt eines Certificates

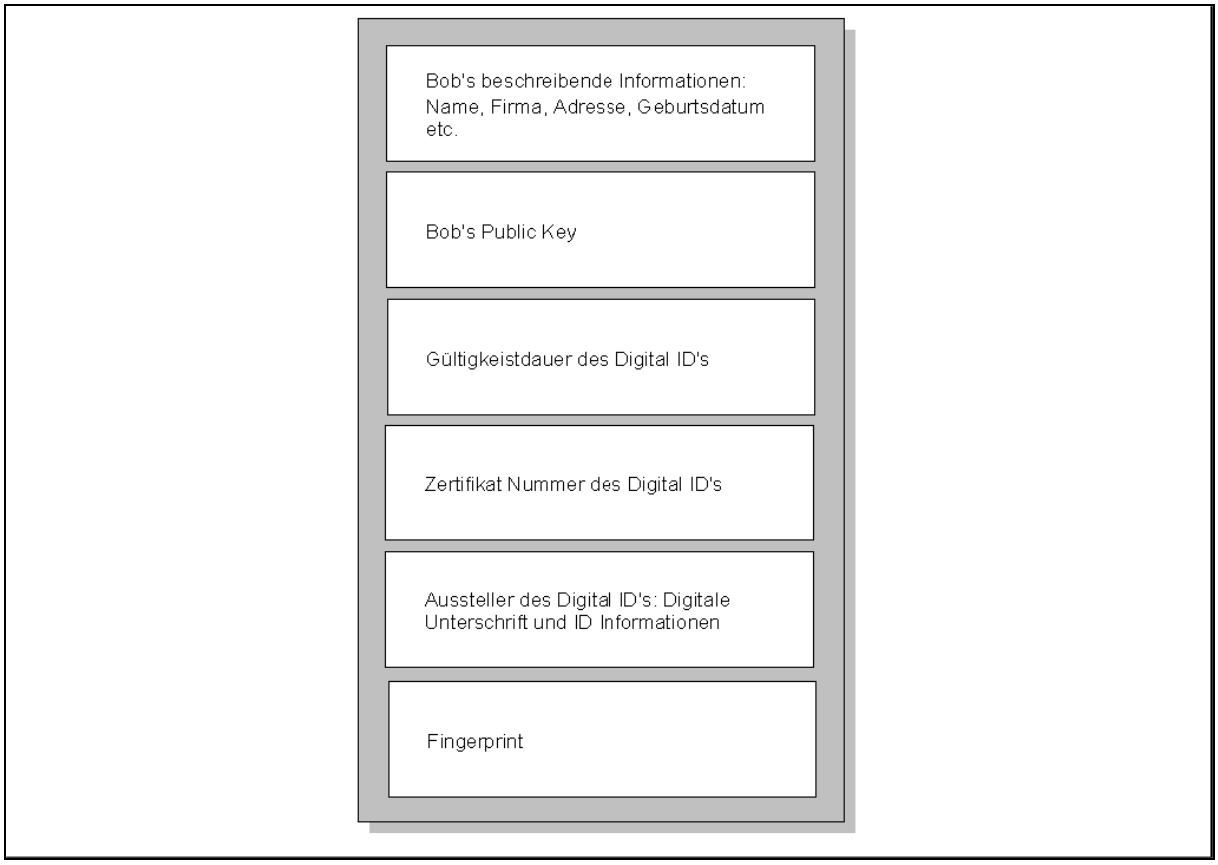

Abbildung 3: Zusammensetzung einer Digital-ID

Benutzer dieser Technologie hängen typischerweise ihren Public Key dem ausgehenden Dokument an, damit der Empfänger den Public Key nicht in einer öffentlichen Liste nachsehen muss. Aber wie kann der Empfänger sicher sein, dass dieser Public Key, oder ein anderer im öffentlichen Register, wirklich zu dieser Person gehört? Könnte nicht ein Hacker irgendeine Nachricht zusammen mit einem „geklauten“ Public Key versenden und sich so als Mister Bob ausgeben?

Die Lösung ist die Digital-ID (auch genannt Certificate), eine Art digitaler Passport oder Identitätskarte, welches selber von einem vertrauten Aussteller digital signiert wurde, Beispielsweise Swisskey. Die Abb. 3 zeigt den Inhalt einer Digital-ID (der Inhalt eines ITU-T X.509 Certificates hängt von der jeweiligen Certificate Klasse ab).

Jedesmal wenn eine Nachricht gesandt wird, wird die Digital-ID mitgeliefert. Der Empfänger der Nachricht benutzt zuerst die Digital-ID um den Public Key des Senders zu überprüfen, nachher wird der Public Key für die Ueberprüfung der Message selber benötigt.

Ein digitales Certificate, auch bekannt unter dem Namen Digital-ID, ist ein elektronisches „Abbild“ zu einem Passport oder einer Geschäftslizenz. Es ist eine Beglaubigung, ausgestellt von einer vertrauenswürdigen Organisation, damit Personen oder Organisationen elektronisch ihre Identität oder ihr Zugriffsrecht präsentieren können. Wenn ein CA (wie Swisskey) Digital-ID’s ausstellt, wird überprüft, ob der Antragssteller keine falsche Identität

vorweist. Nach dem gleichen Schema, wie wenn der Staat ein Passport ausstellt. Es ist ein offizielles „Gütesiegel“ für die Identität des Eigentümers. Wenn die CA ein digitales Certificate für eine Unternehmung ausstellt, setzt die CA ihr Name dahinter, damit man weiss, wer es ausgehändigt hat. Ob man dieser Organisation vertrauen kann? Dies hängt davon ab, ob irgendwelche Zwischenfälle aufgedeckt werden oder nicht!

Der Fingerprint ist ein digitaler Fingerabdruck eines öffentlichen Schlüssels (16 Zeichen lang). Damit ist es es auf einfache Weise möglich, die Korrektheit eines Schlüssels zu überprüfen. Beinahe jedes Client Certificate hat einen Fingerprint.

7.4 Funktionsweise

Die Funktionsweise der Digital-ID Technologie ist, dass kein geheimer Schlüssel ausgeteilt bzw. übers Netz gesandt wird, wie dies Beispielsweise bei der Passwortübergabe der Fall ist (d.h., der Administrator bzw. der Server und der Kunde wussten das Passwort). Aber dafür benützen diese denselben Schlüssel für die Ver- und Entschlüsselung. Ein Digital-ID benötigt ein Key Pair, welches eineindeutig ist weil es algorithmisch zueinander gehört. In anderen Worten, was mit einem Key verschlüsselt wird kann nur mit dem Besitz des anderen Teils des Paares wieder entschlüsselt werden.

Diese beiden Schlüssel des Key Pairs werden als Private und Public Key bezeichnet. Die Schlüssel werden auf der Client-Maschine generiert (selbstverständlich gibt es auch Key Generation Authority [KGA], die Key Pairs erstellen und mittels SmartCard oder anderen Datenträgern ausstellen) und installiert. Nur der Antragssteller hat auf den Private Key Zugriff (der Private Schlüssel wird durch ein Passwort geschützt). Der Public Key wird beim Senden des Certificates (normalerweise im Certificate enthalten) immer mitgeschickt. Kunden, welche mit mir kommunizieren möchten, greifen auf meinen Public Key zu und verschlüsseln die Nachricht damit. Ich bin dann der einzige, der diese Nachricht entschlüsseln kann. Da der Public Key alleine keinen Zugriff etc. verschafft, spielt es keine Rolle, wer alles Zugriff auf diesen Schlüssel hat.

Meine Digital-ID zeigt allen Kunden und Korrespondierenden, dass mein Public Key zu mir gehört. Anhand von Certificates lässt sich die Identität eines Certificate Inhabers überprüfen. Trauen Sie einem Certificate nur dann, wenn Sie dem Aussteller des Certificates vertrauen [8].

Der Sicherheitsschlüssel ist im Grunde genommen eine Datei. Sie kann jedoch nicht wie ein Dokument oder eine Textverarbeitungsanwendung geöffnet werden. Schlüssel entsprechen vielmehr magnetischen Abzeichen mit leistungsstarken Verschlüsselungs- und Entschlüsselungsfähigkeiten [9].

[8] Literatur [VER]
[9] Literatur [NCSH]

7.5 Public und Private Key

Wenn Sie Ihr Certificate von einem Certificate Aussteller erhalten, werden Public Keys und Private Keys in der Client Maschine erzeugt. Diese haben ausserhalb der USA eine Schlüssellänge von 512 Bit. In der USA ist eine Schlüssellänge von 1024 Bit erlaubt (grössere Schlüssellänge bietet mehr Sicherheitsschutz).

Ein Private Key ist ein Verschlüsselungscode, den der Browser generiert, wenn Sie ein Certificate von einem Certificate Aussteller erhalten. Der Browser speichert Ihren Private Key in Ihrer Schlüsseldatenbank und verwendet diesen zur Entschlüsselung von Informationen, die mit Ihrem Public Key verschlüsselt wurden.

Ein Public Key ist ein Verschlüsselungscode, den der Browser generiert und in Ihrer Schlüsseldatenbank speichert. Wenn Sie eine ausgehende Nachricht oder ein anderes Objekt unterschreiben, wird der Public Key zusammen mit Ihrer digitalen Unterschrift der Nachricht oder dem Objekt angefügt. Jede Person, die im Besitz Ihres Public Key ist, kann Ihre Nachrichten entschlüsseln und Nachrichten an Sie senden [10].

[10] Literatur [NCSH]

7.6 Einsatzart

7.6.1 Nachrichten Verschlüsselung

Unter Verschlüsselung versteht man die Codierung von Informationen mit Hilfe eines Public Key, der in einem Certificate enthalten ist, das Sie von einer anderen Person erhalten haben.

Wenn Sie eine ausgehende Nachricht (auch bekannt unter „Digital Envelopes“) verschlüsseln, verwenden Sie hierfür den Public Key des Empfängers. Nur der Empfänger kann diese Nachricht entschlüsseln. Das Certificate jedes Empfängers enthält genau einen Public Key.

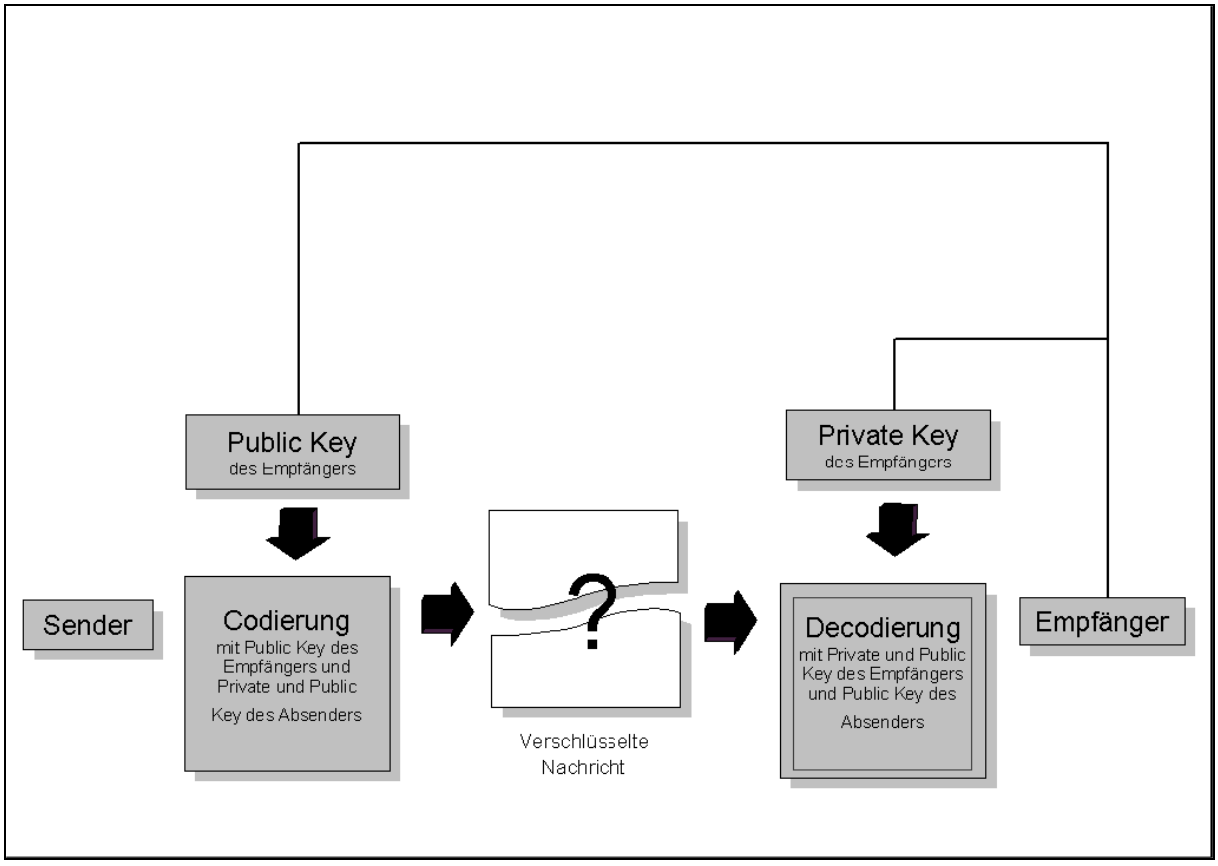

Abbildung 4: Funktionsweise der Public-Key-Verschlüsselung[11]

In gleicher Weise verwendet Ihr Kommunikationspartner Ihr Certificate für die Verschlüsselung von Nachrichten, die er an Sie schickt. Das Lesen verschlüsselter Nachrichten oder die Darstellung verschlüsselter WWW-Seiten ist ohne Entschlüsselung nicht möglich.

Abb. 4 illustriert, wie der Sender mittels Public Key des Empfängers und seinem Private Key die Nachricht verschlüsselt. Der Empfänger kann nun mit dem erhaltenen Public Key vom Absender und seinem eigenen Private Key die Nachricht entschlüsseln. Anmerkung: Private Keys werden nie übers Netz mitgeschickt, lediglich die Public Keys!

[11] Literatur [HAN], Seite 454

Als Entschlüsselung bezeichnet man das Decodieren empfangener verschlüsselter Nachrichten.

Wenn Sie eine verschlüsselte Nachricht empfangen oder die Verbindung zu einer verschlüsselten WWW-Seite herstellen, benutzen Sie Ihren Private Key zur Entschlüsselung und Darstellung der Nachricht bzw. der WWW-Seite.

Den Inhalt einer verschlüsselten Nachricht oder WWW-Seite können Sie ohne vorherige Entschlüsselung nicht darstellen.

Sie können keine Nachrichten oder WWW-Seiten entschlüsseln, wenn:

- diese mit einem anderen Certificate verschlüsselt wurden als dem, das mit Ihrem übereinstimmt.

- Sie mit einem anderen Computer arbeiten als mit dem, den Sie zur Beantragung Ihres Certificates verwendet haben (es sei denn, Sie haben Ihr Certificate von Ihrem ursprünglichen Computer exportiert und in den neuen Rechner importiert).

Ein Server kann eine WWW-Seite verschlüsseln, wenn er die Seite zu Ihrem Browser überträgt. Ihr Browser entschlüsselt die Seite, bevor diese dargestellt wird. Nachdem die Seite empfangen, entschlüsselt und dargestellt wurde, verbleibt sie im unverschlüsselten Format auf Ihrem Computer.

7.6.2 Digitale Unterschrift

Hashfunktion

Aus einer beliebig langen Originalinformation wird mit der Hashfunktion ein Hashwert - eine kurze, eindeutige Zeichenfolge erzeugt. Durch Vergleichen der Hashwerte vor und nach dem Versenden einer Information, lässt sich feststellen, ob ein digitales Dokument während der Übermittlung verändert wurde (siehe Abb. 5 und 6) [12].

Vorgang

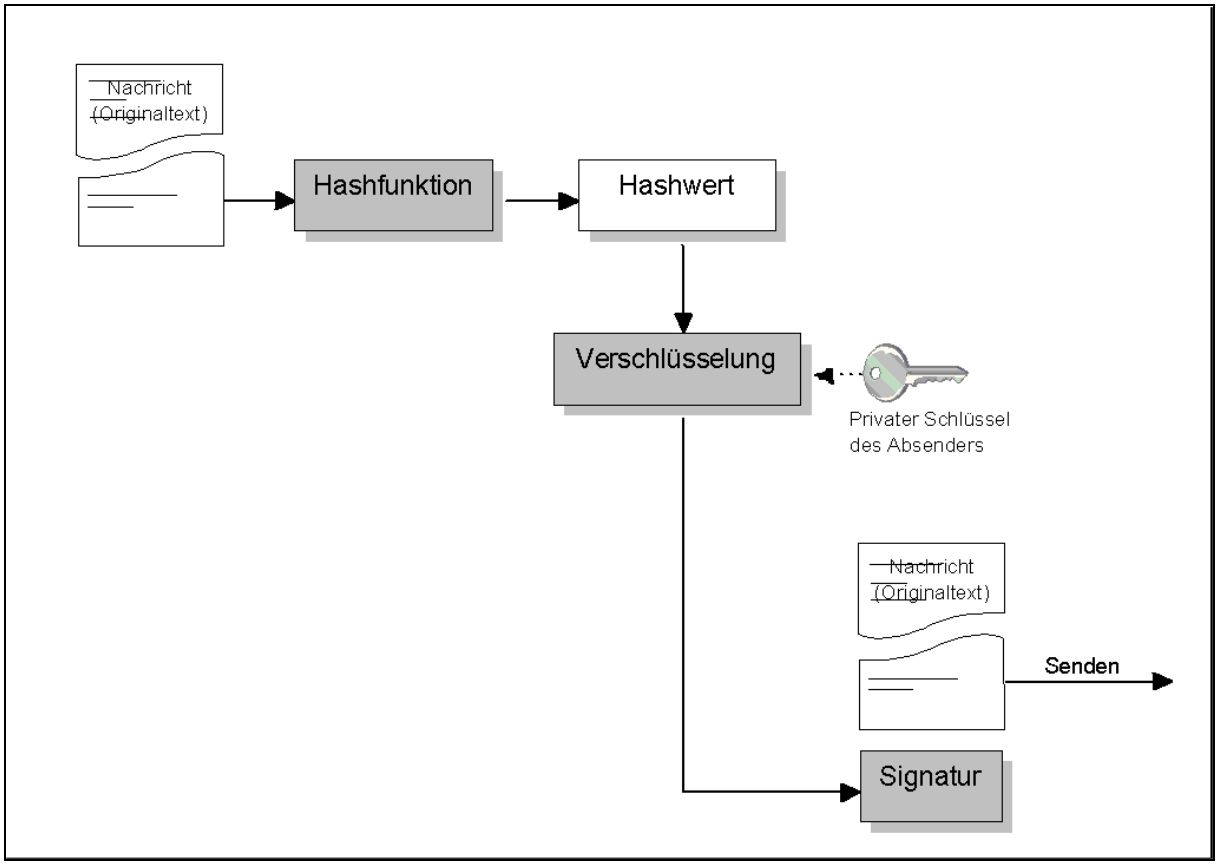

Abbildung 5: Erzeugung „Digitale Signatur“ beim Absender

Digital signatures können gebraucht werden, um Nachrichten verschlüsselt oder unverschlüsselt zu verschicken. Es geht nur darum, nachzuprüfen, ob die Nachricht durch eine unauthorisierte Person manipuliert wurde beim Uebertragen. Eine Message zu unterzeichnen ändert die Message nicht (d.h. es kann eine bereits verschlüsselte Message unterzeichnet werden), es generiert einfach eine digitale Unterschrift (String mit einem Teil des Textes als Inhalt) welche an den Text angehängt wird oder separat übermittelt wird (Abb. 5). Digital signatures werden mittels Verschlüsselung der Meldungszusammenfassung (bekanntester Algorithmus ist MD5-„Message Digest 5“) mit dem Private Key des Senders implementiert.

[12] Literatur [SWI]

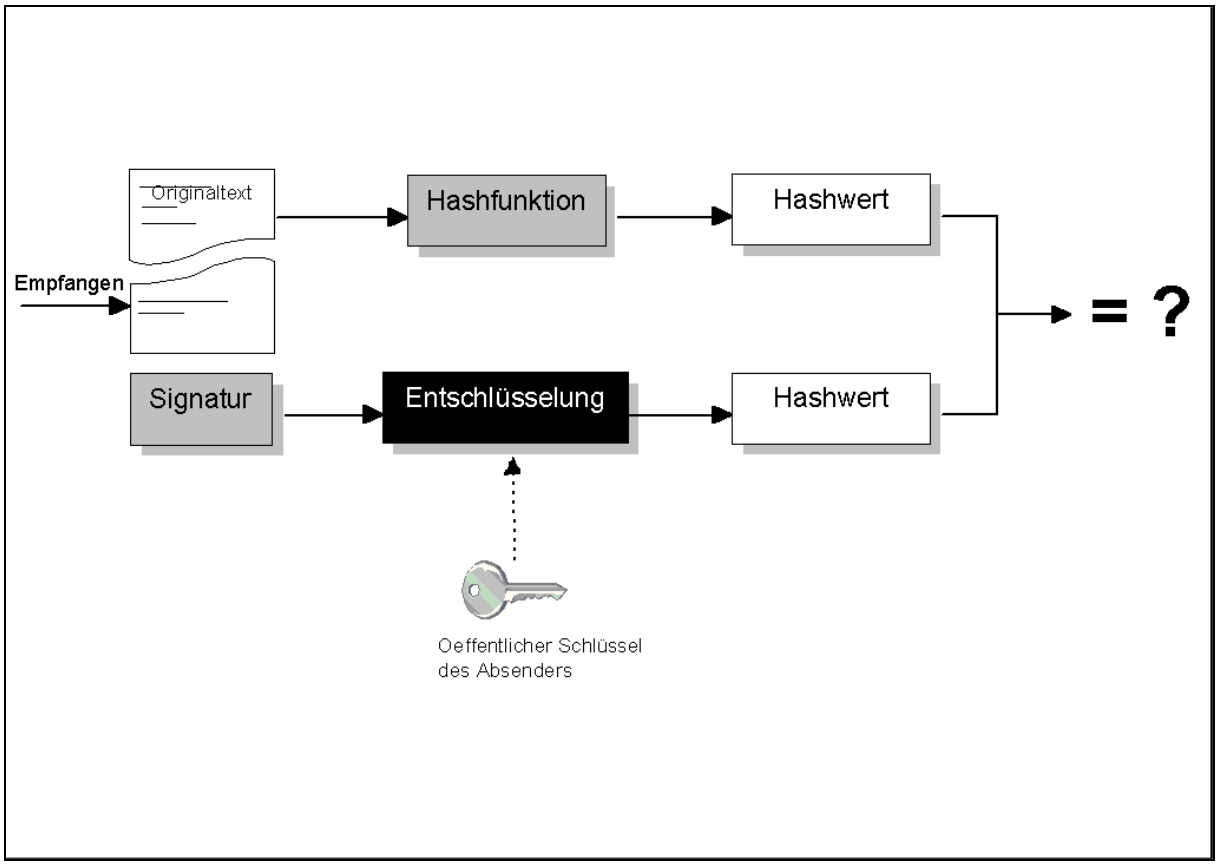

Abbildung 6: Entschlüsselung der digitalen Signatur beim Empfänger

Der Empfänger rechnet danach selber die Meldungszusammenfassung, entschlüsselt mit dem Public Key des Senders seine Meldungszusammenfassung und vergleicht beide Ergebnisse. Falls sie gleich sind, ist bewiesen, dass die Meldung unterwegs nicht manipuliert wurde und dass sich kein anderer als Sender ausgibt (Senderauthentifizierung). Dieser Prozess ist in Abb. 6 aufgeführt.

7.7 Certificate Authority (CA)

Certificate Aussteller nennt man die Unternehmen oder Organisationen, die Certificates ausgeben und autorisieren. Bei den Certificate Aussteller kann man eigene Certificates anfordern, aktualisieren und die Gültigkeit von Certificates überprüfen, sei dies als Privatperson oder als Unternehmung.

Sie können sich mit Hilfe der im Browser Fenster angezeigten Sicherheitsinformationen, Hinweise über die diversen Certificate Ausstellern einholen. Diese Certificates sind Standardmässig in den aktuellen Browsern vorhanden [13].

Die Aussagekraft eines Certificates hängt stark von der Vertrauenswürdigkeit der CA ab. Damit bezüglich den Zertifizierrichtlinien jederzeit eine hohe Qualität gewährleistet ist, muss das Zusammenspiel zwischen Registrierungsstelle und CA eines Landes bzw. der ganzen Welt reibungslos funktionieren [14].

[13] Literatur [NCSH]
[14] Literatur [SWK]

7.7.1 Certificate Authority Hierarchie

Bei Grossunternehmungen, welche in diverse Abteilungen über diverse Länder etc. unterteilt sind, ist das eigene Verwalten der Ressourcen im Intranet heutzutage unumgänglich. Jede Abteilung muss die Zulassung zu den Ressourcen im Intranet selber verwalten und dafür eignet sich die Ausstellung von Certificates am besten.

Damit diese Abteilungen überhaupt die Möglichkeiten haben, die Aufsicht um die Ausstellung und Verteilung der Certificates zu bewahren, wird diesen die Erlaubnis erteilt, selber als CA zu wirken, mit je einem eigenen CA Server. Doch aufgepasst, die „Mutterunternehmung“ muss ständig ein Auge auf diese CA’s halten!

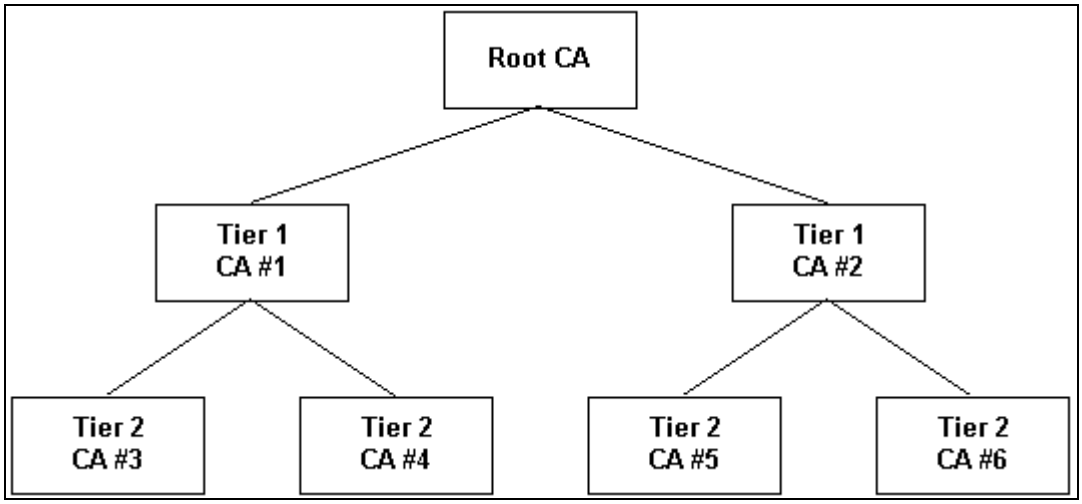

Abbildung 7: CA Server Hierarchie einer Grossunternehmung

Die CA Hierarchie beginnt mit einem ultimativen CA genannt Root. Die Root Authority zertifiziert CA Server in der eigenen Unternehmung um die Kompetenz in Sicherheit und Kontrolle zu verteilen. In Grossunternehmungen können diverse Hierarchie-Stufen von CA Server (multiple tiers of CA server) vorkommen. Siehe Abb. 7.

Wenn ein ausgestelltes Certificate von einem Tier 1 oder Tier 2 CA per Netzwerk übermittelt wird, muss der Empfänger überprüfen, ob der CA-Aussteller selber von einem über ihm liegenden zertifiziert wurde und dieser wiederum von einem über ihm etc., bis eine authentifizierte Verbindung zwischen dem Lower-level-CA und dem Root-CA besteht. Beispielsweise kann in Abb. 7 nachgeprüft werden, dass das CA #4 von CA #1 zertifiziert wurde und dieses wiederum vom Root-CA [15].

[15] Literatur [MSS]

7.7.2 Certificate Hierarchie

Um Digital-ID nutzbar zu machen, muss ein hohes Vertrauen bestehen zwischen User und Certificate bzw. von der Organisation zum Certificate. Dieses Vertrauen wird aufgebaut durch die Bildung von Digital-ID-Hierarchien, mit all den Mitgliedern einer Hierarchie, die zum selben Sicherheitsfaktor gehören. Digital-IDs werden nur Personen oder Organisationen ausgestellt, wenn Beweis der Identität feststeht. Verschiedene Hierarchien haben verschiede Sicherheitsfaktoren und legen den Umfang bzw. Grad fest, wie sich ausweisen und wie die Digital-IDs ausgestellt bzw. verteilt werden.

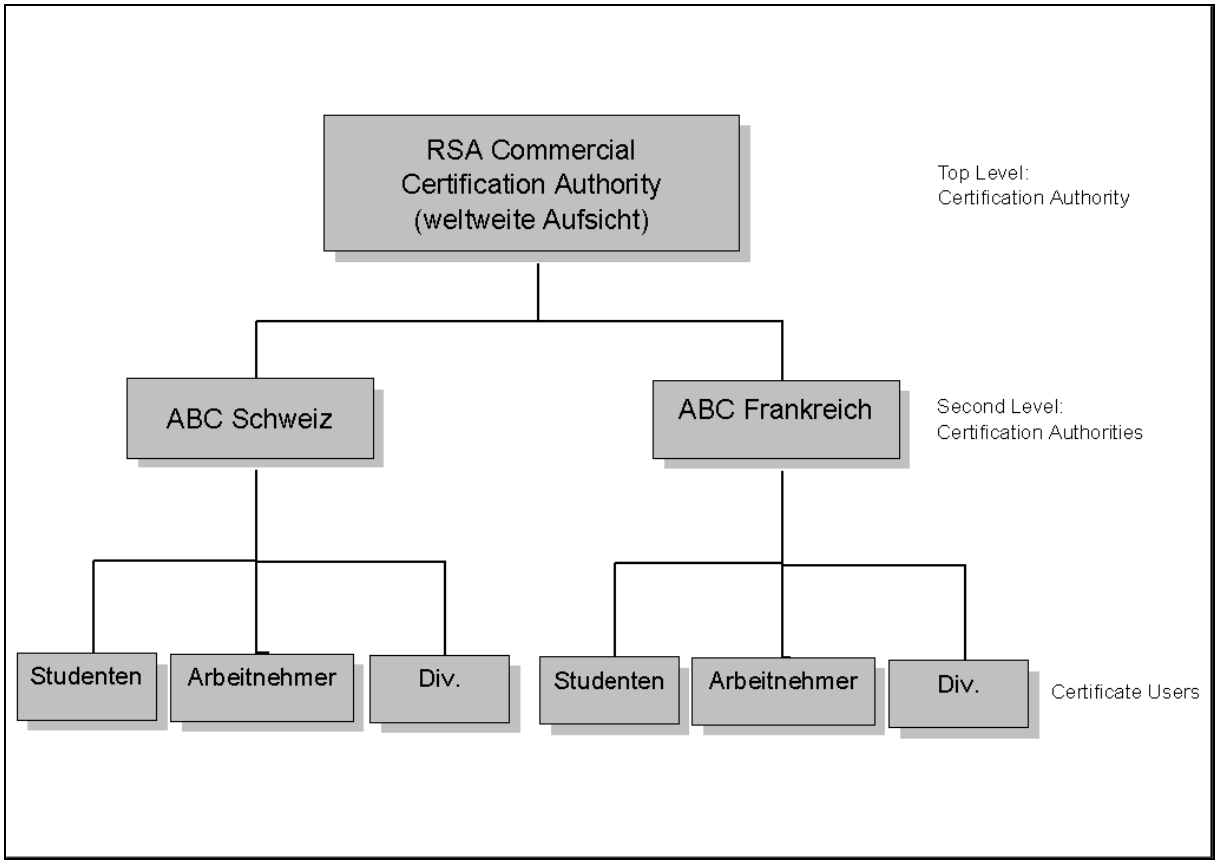

Abbildung 8: Certificate Hierarchie

Die CA führen diverse Digital-ID Hierarchien (Beispiel in Abb. 8). Die CA Aussteller haben eine hohe Sicherheitsgarantie gegenüber einem End-User Digital-ID und dem End-user vorzuweisen. Members von Rivest-Shamir-Aldeman RSA's CA Ausstellers werden eine hohe Sicherheitsstufe aufweisen. Dies wird normalerweise nicht der Fall sein zwischen zwei End-Usern, welche Members von zwei „lower-assurance hierarchies" sind, die miteinander mittels Digital-IDs kommunizieren. Ohne die Sicherheit, eine sauber geführte Digital-ID Hierarchie vorzufinden, hat der Gebrauch einer Digital-ID einen limitierten „Wert" [16].

[16] Literatur [VER]

7.8 Registration Authority (RA)

Für den Erhalt eines Certificates muss der zukünftige Inhaber persönlich bei einer Registrierungsstelle vorsprechen und sich mit einem amtlichen Dokument ausweisen. Bei Firmen werden zusätzliche Prüfungen im Handelsregister vorgenommen. So ausgestellte Certificates werden mit einem Ablaufdatum versehen und von der Zertifizierungsstelle unterschrieben. Die CA bestätigt damit, dass die Identität des Certificate Inhabers gemäss den gültigen Richtlinien überprüft wurde.

Die Aussagekraft eines Certificates hängt stark von der Vertrauenswürdigkeit der RA ab und sollte somit in allen Ländern nach dem gleichen Standard ablaufen und qualitativ „gesichert“ sein. Deshalb sind folgende Kriterien für die Bestimmung der RA entscheidend:

- Vertrauenswürdigkeit
- Erfahrung im Bereich Registrierung
- Bestehende technische und organisatorische Infrastruktur
- Flächendeckende Präsenz

Beispiel Schweiz:
Damit bezüglich der Zertifizierungsrichtlinien jederzeit eine hohe Qualität gewährleistet ist, werden diese Richtlinien durch Digisigna (Verein der Handelskammern der Schweiz und des Fürstentums Liechtensteins) erstellt und überwacht. In Zukunft werden Banken, Post und Handelskammern als Registrierstellen in der Schweiz auftreten [17].

7.9 Web Server

Eindeutige digitale Identifikation, genannt Server-Certificate, bilden die Basis der SSL „security features“ eines Web Servers. Server-Certificate, ausgestellt von einer vertrauenswürdigen Organisation, geben dem User die Möglichkeit, die Identitität der Web-Seite zu authentifizieren. Das Server-Certificate beinhaltet detaillierte Identifizierungs-Informationen, wie der Name der Organisation, welche den Web Server beansprucht, der Name des Zertfikats-Ausstellers und eine eindeutige Identifikation, genannt Public Key. Diese Informationen helfen dem User, die Authentifikation des Web Servers zu erhalten und eine gesicherte HTTP Connection zu „garantieren“.

Gibt es auf dem Web Server auch Möglichkeiten nur gewissen Client Certificates Zutritt zu gewähren? Ja. Bei den meisten aktuellen Web Server gibt es Zugriffsabfragen, danach kann ich gewisse Daten eines Certificates überprüfen und entscheiden.

7.9.1 Vorsichtsmassnahmen

Server-Verwalter müssen besondere Vorsichtsmassnahmen gegen Verletzungen der Datensicherheit ergreifen. Damit Ihre Informationen geschützt werden können, müssen die Server physisch geschützt und der Zugriff auf Softwarekennwörter und Private Keys überwacht werden.

[17] Literatur [SWK]

7.10 Browser

Die Certificate Funktionen sind in den beiden Browsern Netscape Navigator 3.0 und Internet Explorer 3.0 oder aktueller implementiert.

7.10.1 Sicherheitsvorkehrung

Die Browser öffnen eine Warnmeldung (Abb. 9) bei einer unbeabsichtigten Uebermittlung von privaten oder persönlichen Daten in einem ungesicherten Channel.

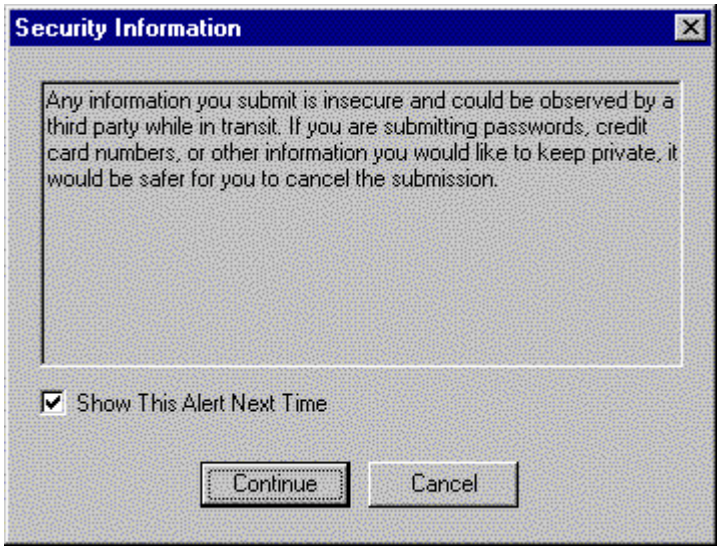

Abbildung 9: Browser - ungesicherte Kommunikation

Wenn hingegen ein User eine Credit Karten Nummer oder andere Informationen an eine Web Seite mit einer gültigen Digital-ID senden will und dies über eine SSL Verbindung läuft, wird selbstverständlich keine solche Meldung erscheinen, da alles in Ordnung ist.
Die Browser zeigen in der Statusbar ein Sicherheitszeichen an, wenn eine gesicherte Verbindung besteht.

Um Informationen über den Betreiber des Web Servers zu erhalten, der diese Web Seite zur Verfügung stellt, kann der Users das Server Certificate mittels einem „double click“ aufs Sicherheits-Icon öffnen und ansehen, siehe Abb. 10.

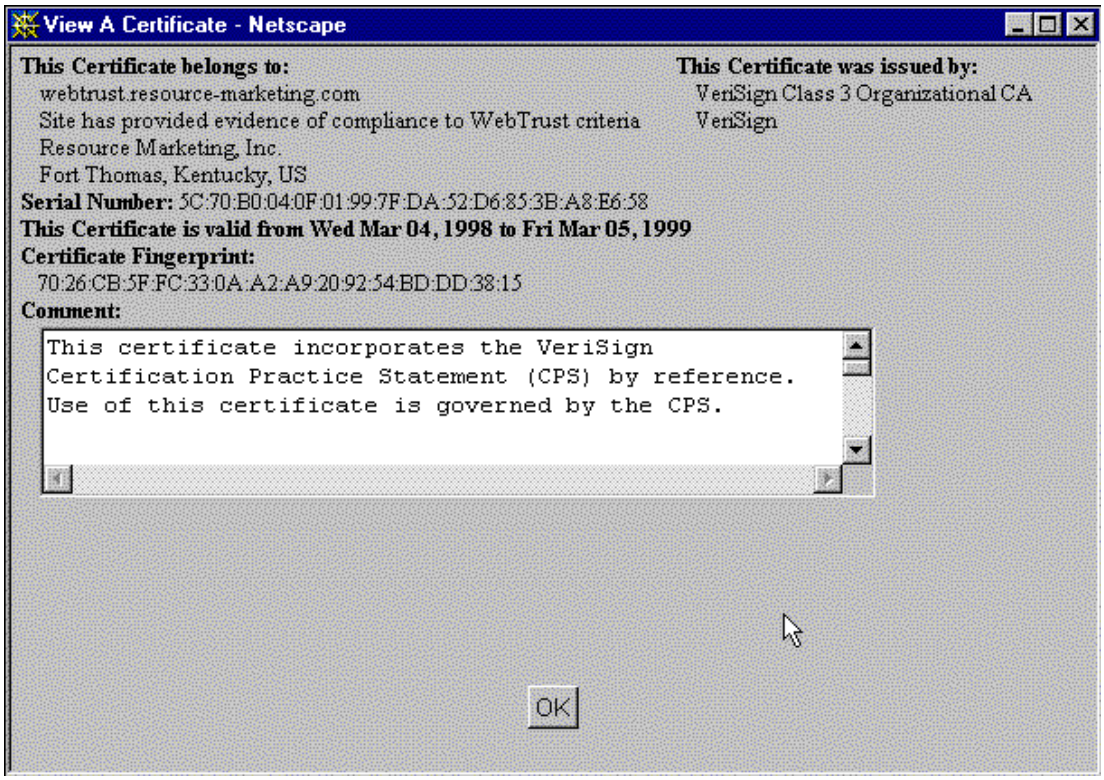

Abbildung 10: Browser - Server Certificate des Web Servers

Dieses Digital-ID (Abb. 10) gibt Ausschluss, dass dieses Seite wirklich zu „Resource Marketing“ gehört. Es gibt ebenso Ausschluss darüber, dass VeriSign dieses Server Certificate ausgestellt hat und für diese Seite bürgt.

Dieses Dialogfenster erscheint nur bei Seiten mit einem gültigen, bei einem CA ausgestellten Certificate, welchem wir vertrauen. Technisch ist die Meinung, das der Public Key des CA’s im Browser aufgelistet sein muss im Directory „of trustet roots“. VeriSign’s Public Key als Beispiel sind heutzutage zu standardmässig in beinahe allen Browsern enthalten.

Abbildung 11: Browser - nicht vertrauenswürdiger CA

Im Gegensatz, wenn eine Seite ein Certificate hat, welches von einem „untrusted“ CA ausgestellt wurde, erscheint Dialogfenster gem. Abb. 11.

Abbildung 12: Browser - Web Server vertraut gewissen CA' s nicht und somit auch deren Client Certficate nicht

Das gleiche gilt, wenn der Web Server unserem Certificate CA nicht vertraut, dann erscheint Dialog (Abb. 12).

Wenn man ein VeriSign Digital-ID auf dem Server installiert und SSL ermöglicht, sieht der Kunde (Client) anhand eines Sicherheitszeichens, dass in einem gesicherten Umfeld operiert wird. Internet Users wissen, dass sie somit dieser Seite vertrauen können und fühlen sich sicher um mit dem „Shopping" zu beginnen, dies als Beispiel [18].

Hinweis

Wenn im Browser-Fenster eine verschlüsselte Seite dargestellt wird, verbleibt die Seite im unverschlüsselten Format im Festplatten-Cache. Jede Person, die Zugriff auf den für den Browser reservierten Festplatten-Cache hat, kann den Inhalt dieser Seite lesen.

Sie können eine mit Certificate verschlüsselte WWW-Seite nicht auf jedem Computer entschlüsseln, sondern nur auf dem Computer, für den Ihr Certificate ausgestellt wurde bzw. auf welchem der User das Certificate installiert hatte. Wenn Sie einen anderen Computer verwenden, müssen Sie sich mit Ihrem Certificate Aussteller in Verbindung setzen, um ein neues Certificate für Ihren aktuellen Computer anzufordern oder allenfalls das Certificate aus dem Browser exportieren und importieren [19].

[18] Literatur [VER]
[19] Literatur [NCSH]

7.10.2 Certificate Verwaltung

Wo speichert der Browser all meine Certificates ab? Nun, der Browser hat eine spezielle Certificate Datenbank. Die Certificates kann zwar jeder User, der an meinem Computer arbeitet ansehen aber er kann diese nicht benutzen, da jeder Zugriff mittels einem Passwort bestätigt werden muss!

Client-Certificates sind im Browser durch ein User Passwort geschützt! D.h. jedesmal wenn ich den Browser neu starte und ein Certificate benötige, fragt mich der Browser nach dem Passwort und erst dann kann ein Certificate ausgewählt und benutzt werden!

Nun, die Frage ob es möglich ist, Certificates manuell zu exportieren und zu importieren liegt Nahe. Sicher ist dies gewährleistet, denn es gibt viele User, die vielleicht von mehreren Computern aus Zugriff auf ein und denselben Dienst haben möchten.

Sobald ich Beispielsweise ein Certificate auf eine Diskette exportiere, verlangt der Browser eine Passworteingabe plus Bestätigung, damit dieses nachher auch wieder Passwortgesichert in ein anderen Browser importiert werden kann.

Das Exportformat ist „Binary“ PKCS12!

Jedoch aufgepasst, die grosse Lücke momentan in einigen Browsern ist, dass jedermann bestehende Certificates aus der Datenbank löschen kann (keine vorherige Passwortaufforderung notwendig)!

7.11 Registrationsablauf

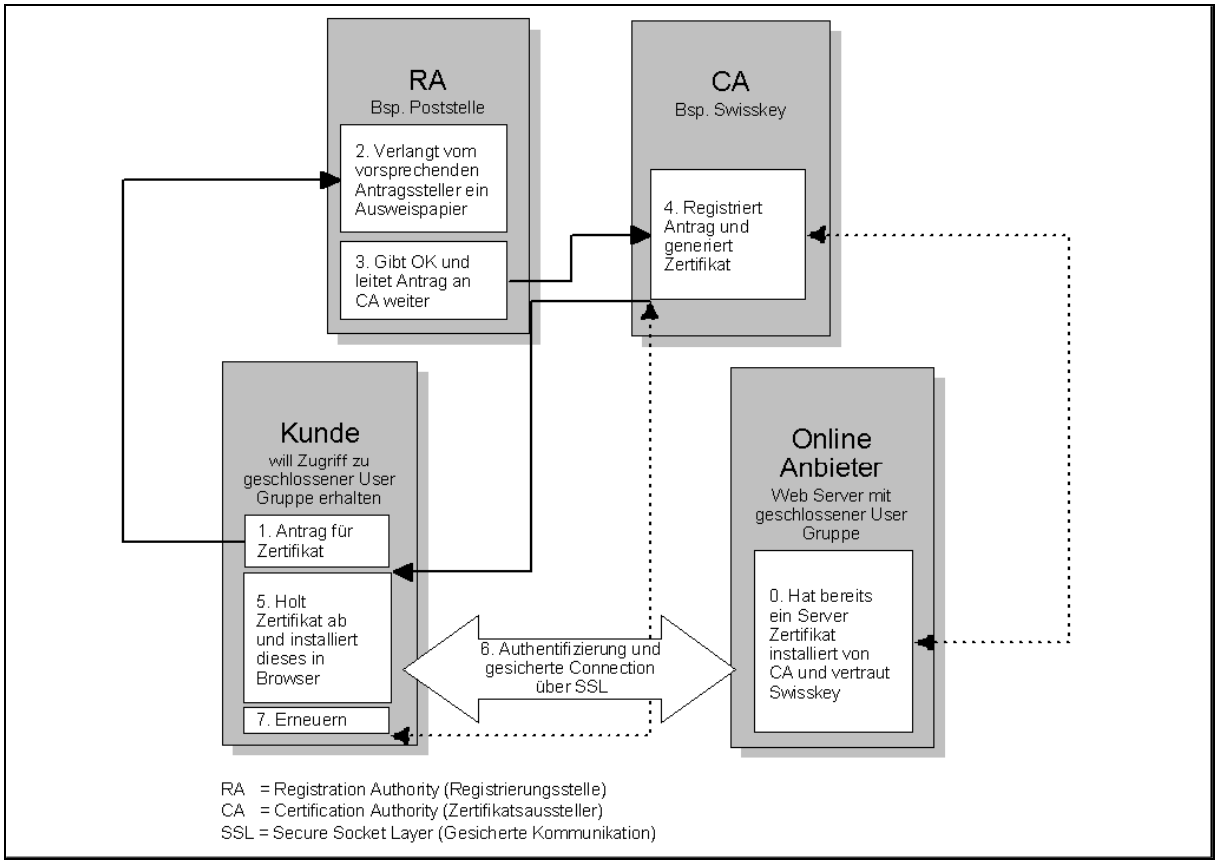

Abbildung 13: Anfordern eines Certificates plus Zusammenspiel aller Komponenten

Der Ablauf einer Client Certificate Registrierung sieht folgendermassen aus (gem. Abb. 13):

1. Kunde stellt Antrag für ein Client Certificate, sei dies schriftlich oder Online und muss dafür identifizierbare Informationen bekanntgeben (Bsp. Name, Adresse, Email etc.).
2. Im Falle eines „higher-level“ Certificates muss noch der Zwischenschritt bei der RA eingeschaltet werden. Der Kunde muss persönlich „vorsprechen“ und sich mit einem Ausweispapier ausweisen können.
3. Die Registrierungsstelle überprüft den Antrag und bestätigt dies. Der Antrag wird somit an die Certificate Authority weitergeleitet.
4. Die CA registriert Antrag und generiert ein Client Certificate im Certificate Server.
5. Der User kann nun das persönliche Certificate bei der CA abholen (übers Internet oder Speichermedium) und installiert dies im Browser. Zuvor jedoch generiert der Browser ein Key Pair für das neue Certificate.
6. Falls der Web Server dem Certificate Aussteller Swisskey vertraut und der Kunde dem Web Server selber auch vertraut kann eine gesicherte Connection (SSL) aufgebaut werden.
7. Da Standardmässig die Gültigkeitsdauer eines Certificates auf zwei Jahre begrenzt wird, muss dieses bei der zuständigen CA erneuert werden.

7.12 Diebstahl

Schützen Sie Ihren Computer und Ihre Certificates vor einem Zugriff durch unbefugte Personen. Jeder, der Zugriff auf Ihre Certificates oder die Schlüsseldatenbank und Ihren Rechner hat, kann Ihre Nachrichten entschlüsseln und ausgehende Nachrichten mit Ihrer Unterschrift unterzeichnen.

Grundsatz ist: Achten sie auf ihre Certificates, wie wenn sie auf ihren Schlüssel oder ihre Kreditkarte acht geben würden! Denn wenn diese gestohlen werden, kann auch mit diesen Unfug getrieben werden!

7.13 Archivierung

Um sicher zu gehen, dass Digital-IDs auch noch in zehn Jahren auf ihre Gültigkeit überprüft werden können, müssen die Public Keys und ihre Certificates archiviert werden. Am besten wäre ein öffentliches Directory, in welchem die Public Key Certificates während zehn Jahren aufbewahrt werden für deren „Nachweisbarkeit“ etc. Denn was nützt einem im papierlosen Büro ein Dokument, dass zehn Jahre aufbewahrt wird, aber von dem man nicht mehr beweisen kann, wer dieses Dokument signiert hat [20]?

7.14 Revoked Certificate

Was geschieht, wenn nun mal ein Certificate vor Ablauffrist gesperrt oder gelöscht (revoke) werden soll? Was kann der Aussteller CA des Certificates unternehmen? Sicher ist ja, dass keine Möglichkeit besteht, von irgendeinem Server aus, das entsprechende Certificate im Client Browser zu löschen! Darum bleibt nichts anderes, als alle gelöschten Certificates in einem öffentlichen Verzeichnis zu publizieren. Und es ist nun Sache jedes einzelnen Web Servers, die ein gesichertes Web anbieten, vor jedem Zugriff auf eine entsprechenden Revoked Certificate List zuzugreifen und zu kontrollieren, ob dieses Certificate evtl. schon gelöscht wurde. Der Web Server hat hierfür spezielle Funktionen.

Certificate- und Certificatewiderrufslisten werden mit dem Web-Verzeichnisprotokoll Lightweight Directory Access Protocol (LDAP) auf Verzeichnisservern veröffentlicht [21].

[20] Literatur [PAY]
[21] Literatur [NCS]

8 Sicherheits Verfahren

8.1 Verschlüsselungstheorie

Als erstes möchte ich das Schlüsselaustauschprotokoll erklären, das den Grundstein für die eigentliche Public- und Private Verschlüsselung (RSA-Verfahren) legte.

8.1.1 Schlüsselaustauschprotokoll

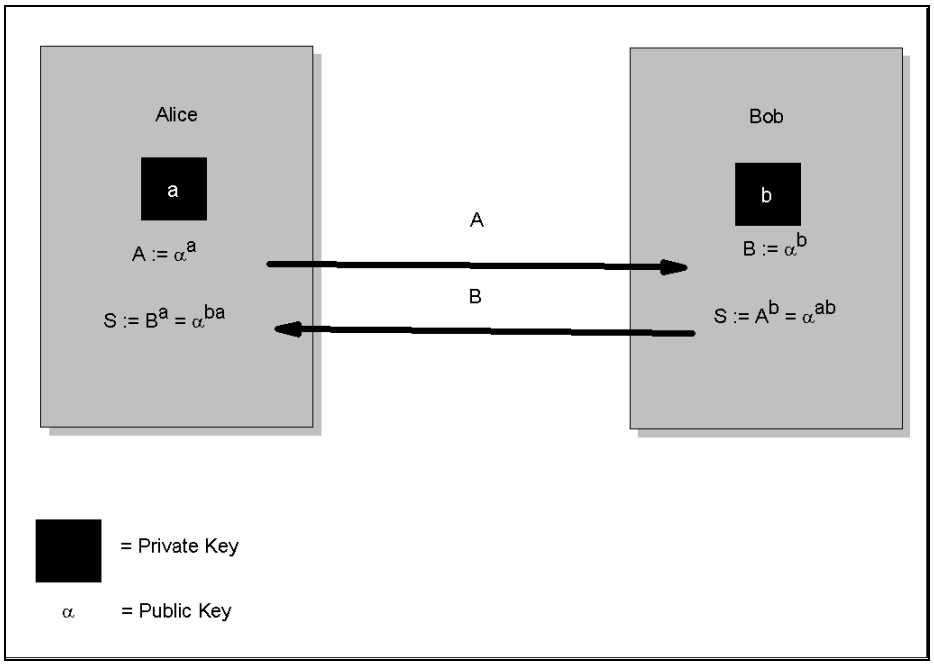

Abbildung 14: Schlüsselaustauschprotokoll

Das Schlüsselaustauschprotokoll (siehe Abb. 14) nach Whitfield Diffie und Martin E. Hellmann wurde 1976 theoretisch in einem Konzept festgehalten. Eine Einwegfunktion dient Alice und Bob dazu, sich einen Schlüssel S für die nachfolgende Verständigung zu verschaffen, der beider gemeinsames Geheimnis ist. Alice wählt sich eine Zufallszahl a und hält sie geheim. Aus a berechnet sie mit der Einwegfunktion die Zahl A = α^a und schickt sie an Bob. Der verfährt ebenso, indem er eine geheime Zufallszahl b wählt, daraus B = α^b berechnet und an Alice schickt. Die Zahl α (Public Key) ist beliebig und darf öffentlich bekannt sein. Alice wendet die Einwegfunktion mit ihrer Geheimzahl a auf B an, Bob tut gleiches mit seiner Geheimzahl b und der empfangenen Zahl A. Das Ergebnis S ist in beiden Fällen dasselbe, weil die Einwegfunktion kommutativ ist: $\alpha^{ab} = \alpha^{ba}$. Aber selbst Bob kann Alices Geheimnis a nicht aus den ihm vorliegenden Daten rekonstruieren, Alice wiederum Bobs Geheimnis b nicht ermitteln, und ein Lauscher, der α kennt und sowohl A als auch B mitgelesen hat, vermag daraus weder a noch b noch S zu berechnen. (Anmerkung Bosshard Stefan: Als Lauscher ist es möglich gewisse Keys zu rekonstruieren. Beim Bilden des gemeinsamen Schlüssels S wird das Potenzieren angewendet und ein allfälliger Lauscher müsste das Logarithmieren anwenden. Der Trick liegt nun darin, dass es eine sogenannte

Falltürfunktion ist, d.h. das Logarithmieren dauert viel zu lange, um in anständiger Zeit die gewünschten Schlüssel zu berechnen) [22].

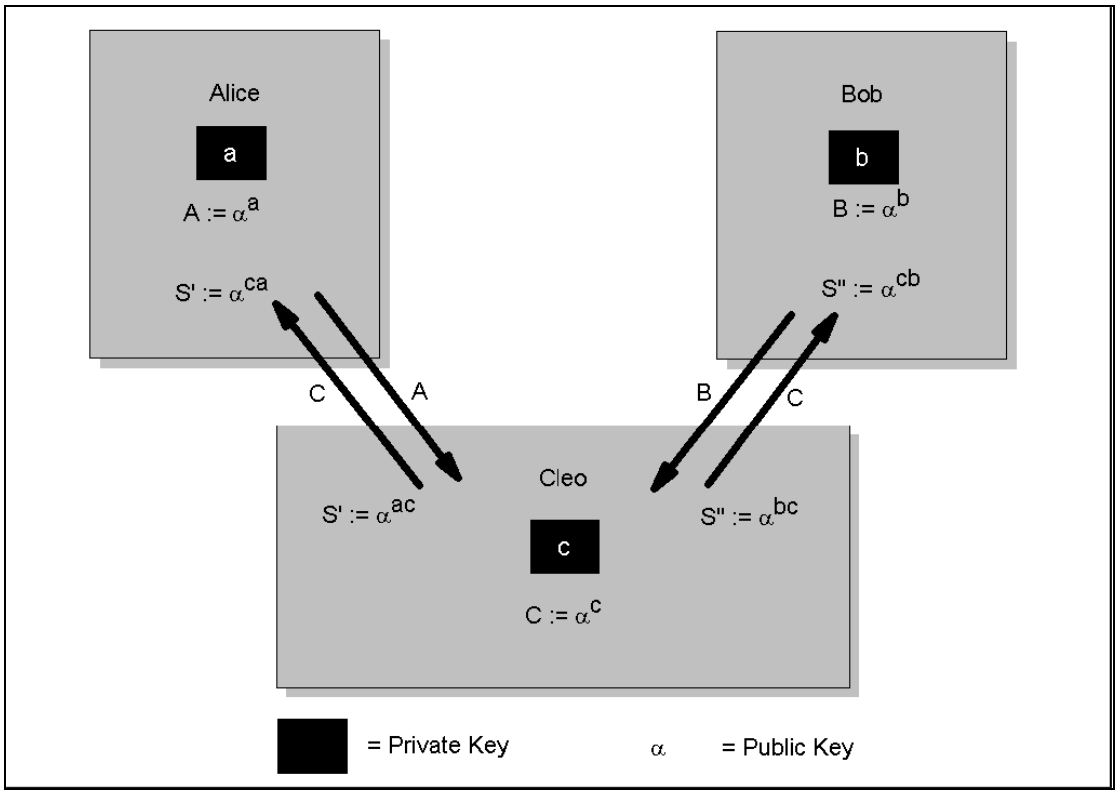

Abbildung 15: Schlüsselaustauschprotokoll mit Lauscher

In Abbildung 15 wird aufgezeigt, wie es aussehen würde, wenn es einem Eindringling doch gelingen würde, sich dazwischen zu hängen und die Nachricht „zu belauschen"!

Ein böswilliger Netzteilnehmer (üblicherweise Cleo genannt) manipuliert das Vermittlungssystem so, dass Alice und Bob statt miteinander mit ihm verbunden sind. Cleo macht Alice glauben, er sei Bob, und führt mit ihr das Schlüsselaustauschprotkoll nach Diffie und Hellman durch. Entsprechend verfährt er mit Bob. Wenn Alice nun eine mit S' verschlüsselte Nachricht dem vermeintlichen Bob sendet, entschlüsselt Cleo sie mit S', verändert sie nach Belieben und schickt sie mit S'' verschlüsselt an Bob weiter. Sowohl Alice als auch Bob senden und empfangen Nachrichten, die mit dem verschlüsselt sind, was sie gutgläubig für ihren gemeinsam gefundenen Schlüssel halten, und wiegen sich deshalb in falscher Sicherheit [23].

[22] Literatur [SOD]
[23] Literatur [SOD]

8.1.2 Public- und Private Key Verfahren

Ronald Rivest, Adi Shamir und Leonard Aldemann entwickelten 1978 das Rivest Shamir Aldemann RSA-Verfahren (Public Key), das selbstverständlich auf dem Schlüsselprotokoll Modell aufbaut. Trick dabei ist, „Einweg-Funktion“ bzw. Falltür-Funktion, da bei der Entschlüsselung die Logarithmus-Funktion angewendet werden muss [24].

Technologie

Sie benötigen beide Schlüsselarten, sowohl den Private Key als auch den Public Key. Der Private Key bleibt in Ihrem Computer und wird nie ausgegeben. Ein Public Key kann mehrmals kopiert und an alle ausgegeben werden.

Diese beiden Schlüssel stehen in einer algorithmischen Beziehung. Die Public Key Crypthography Algorithmen gewährleisten, dass eine Nachricht die mit einem Schlüssel verschlüsselt ist, nur mit dem anderen Schlüssel aus dem Schlüsselpaar entschlüsselbar ist. Zwar kann der Public Key an beliebige Internet-Teilnehmer weitergegeben werden. Damit ein Schlüssel aber entschlüsseln kann, muss er auf den verknüpften Schlüsselpartner passen.

Sicherheitslücken

Die eingesetzten Verschlüsselungs-Algorithmen sind nach heutigen Erkenntnissen nur mit sog. „Brute Force Attack“ (Rohgewalt) knackbar. Dies bedeutet, ein Dritter, der in Besitz einer verschlüsselten Meldung kommt, muss alle möglichen Schlüssel ausprobieren, um die Meldung zu entschlüsseln (z.B. bei einem 1024-Bit breitem Schlüssel sind dies 2^1024 Kombinationen).

Daraus ergeben sich folgende Zeiten, die benötigt würden, um mit 100 Millionen parallel geschalteten Rechnern (so viele wurden im Jahr 1995 weltweit verkauft) mit einem Pentium 100 MHZ Prozessor, einen Pretty Good Privacy (PGP) RSA Schlüssel zu knacken. Siehe Tab. 5 [25].

Tabelle 5: Schlüssel „Knackbarkeit“ [26]

Schlüsselbreite	Zeit für das Knacken des Schlüssels
429-Bit Schlüssel	15 Sekunden
512-Bit Schlüssel	22 Minuten
700-Bit Schlüssel	153 Tage
1024-Bit Schlüssel	280 000 Jahre

Diverse Informationen

Sicherheitsprobleme treten vor allem deswegen auf, weil über das Internet übertragene Informationen normalerweise einen Umweg über mehrere übermittelnde Computer nehmen, bevor sie den Zielcomputer erreichen. Die tatsächliche Route, die die Informationen bis zu ihrem Ziel zurücklegen, kann nicht kontrolliert werden.

[24] Literatur [MAK]
[25] Literatur [VIS]
[26] Literatur [VIS]

Während die Informationen über Internet-Computer übertragen werden, kann potentiell jeder übermittelnde Computer die Informationen anzapfen und Kopien machen. Ein übermittelnder Computer könnte Sie sogar hintergehen, Informationen mit Ihrem Computer austauschen und sich selbst als der von Ihnen beabsichtigte Zielcomputer ausgeben. Diese Möglichkeit macht die Übertragung vertraulicher Informationen, wie z.B. Kennwörter und Kreditkartennummern, missbrauchanfällig [27].

Kritischer Moment: Gelingt es einem Betrüger, einen dieser Schlüssel durch seinen eigenen öffentlichen Schlüssel zu ersetzen, so kann er anschliessend die Nachrichten abfangen und mit dem zugehörigen privaten Schlüssel entziffern. Halten wir also fest: Die Achillesferse der Public Key-Verschlüsselung ist die Uebergabe der öffentlichen Schlüssel! Die Public-key-Verschlüsselung bietet folgende Vorteile:

- Mit Hilfe des privaten Schlüssels kann einem Text eine elektronische Unterschrift hinzugefügt werden, welche mit dem zugehörigen öffentlichen Schlüssel überprüft werden kann. Eine solche Unterschrift kann somit nur der Inhaber des privaten Schlüssels erstellen. Sie kann aber von jedem, der im Besitz des zugehörigen öffentlichen Schlüssels ist, verifiziert werden.

- Zusätzlich zu dieser digitalen Unterschrift kann einem Text auch ein sogenannter „message digest“ hinzugefügt werden. Dies bedeutet, dass der Inhalt des Textes in reduzierte Form in die Unterschrift eingeht. Wird der Text nach digitaler Unterzeichnung modifiziert, so kann dies beim Ueberprüfen der Unterschrift mit dem „message digest“ erkannt werden.

Die benötigten Schlüsselpaare lassen sich von geeigneter Software relativ rasch erzeugen. Darüber hinaus bleibt dem Benutzer die Schlüsselverwaltung in den meisten kommerziellen Softwareprodukten verborgen. Es sei darauf hingewiesen, dass dieses Konzept unabhängig vom benutzten Medium ist: Ob die Schlüssel nun in einer Datei auf der Festplatte, in eine Applikation integriert oder auf einer Chipkarte untergebracht sind, spielt dabei keine Rolle [28].

Ihr Computer und der beabsichtigte Zielcomputer können Ihre Informationen ver- und entschlüsseln. Während der Übertragung sind die verschlüsselten Informationen durcheinander; ein Übermittler kann die Informationen weiterleiten und sie zwar kopieren, verfügt aber nicht über die Möglichkeiten, die Informationen zu entschlüsseln.

Navigator und Netscape-Server verwenden die patentierte RSA Public Key-Verschlüsselung-Technologie und massgeschneiderte Software, mit der Sie Informationen durch eingebaute Verschlüsselungsvorrichtungen versenden und empfangen können. Die Protokolle verwenden einen öffentlichen Standard.

Navigator und Netscape-Server bieten als Teil der Verschlüsselung-Technologie einen Mechanismus zur Server-Identifizierung im Internet. Dadurch wird es übermittelnden Computern erschwert, sich als Ihr Zielcomputer auszugeben [29]

[27] Literatur [NCSH]
[28] Literatur [HAN], Seite 453
[29] Literatur [NCSH]

8.2 Secure Socket Layer (SSL)

SSL 3.0 Protokoll implementiert eine erhöhte Version von gesicherten Sockets (Anschluss) auf dem Transport Level. SSL unterstützt einen gesicherten Kommunikations-Link ohne Involvierung der Applikation welche ihn aufgerufen hat. HTTP Server, welche SSL implementiert haben, benutzen die Socket Adresse 443 anstelle von Standard 80. Das SSL Protokoll unterstützt:

- Private Client/Server Interaktion mittels Verschlüsselung
- Server Authentifizierung

Das SSL „handshake" Protokoll muss zuerst vollständig sein, bevor eine Applikation Daten senden oder empfangen kann. SSL benötigt Public Key Authentifizierung und Verschlüsselungstechnologie, entwickelt von RSA Data Securitiy, Inc.

Nachteil von SSL ist einzig, dass sich die Uebertragungszeit ein wenig erhöht.

8.2.1 Die Schutzintensität

SSL verwendet eine von RSA Data Security entwickelte Authentizitäts- und Verschlüsselungstechnologie. Für die Exportanwendung von SSL (mit Genehmigung der US-Regierung. In der USA und Canada werden 56-bit, 128-bit, oder 168-bit Schlüssel verwendet) benutzt Netscape Navigator als Beispiel einen mittelmässigen 40 Bit grossen Schlüssel für den RC4-Datenstrom-Verschlüsselungsalgorithmus. Die zwischen Ihrem Computer und dem Server aufgebaute Verschlüsselung gilt für mehrere Verbindungen, doch die zum Knacken einer Nachricht aufgewendeten Anstrengungen sind bei dem Versuch, die nächste Nachricht zu knacken, nutzlos.

Um eine mit 40 Bit RC4 verschlüsselte Nachricht zu knacken, werden durchschnittlich 64-MIPS-Jahre benötigt (d.h. der Prozessor eines 64-MIPS-Computer muss ein Jahr lang ausschliesslich an der Entschlüsselung der Nachricht arbeiten). Die hochwertige, für die Verwendung in den USA bestimmte Version von 128 Bit bietet einen exponentiell höheren Schutz. Der Aufwand, der betrieben werden müsste, um einen Informationsaustausch zu knacken, sollte abschreckend genug sein. Die Server-Authentizität verwendet die Public-Key-Verschlüsselung von RSA in Verbindung mit den digitalen Certificates nach ISO X.509v3.

Die Browser und SSL-Server bieten Server-Authentizität durch unterzeichnete digitale Certificates, die von vertrauenswürdigen Dritten ausgestellt werden, die als Zertifizierungsbehörden bekannt sind. Ein digitales Certificate bestätigt die Verbindung zwischen dem Public Key und der Identifizierung des Servers (genauso wie ein Personalausweis die Verbindung zwischen Ihrem Photo und Ihrer Person bestätigt).

Kryptographische Prüfmittel wie die Verwendung digitaler Unterschriften stärken das Vertrauen, das Sie den Informationen in einem Certificate schenken [30].

[30] Literatur [NCSH]

8.2.2 Verbindung zu S-HTTP

S-HTTP und SSL unterstützen das Sicherheitsproblem von zwei verschiedenen Perspektiven aus. S-HTTP bezeichnet bzw. filtert individuelle Dokumente als privat oder signiert. SSL im Gegensatz ermöglicht, dass der Kommunikationskanal zwischen zwei Parteien privat und authentifizierbar ist. SSL-Layer Sicherheit befindet sich unterhalb Applikations Protokollen, während S-HTTP „message-based security“ sich oberhalb HTTP befindet.

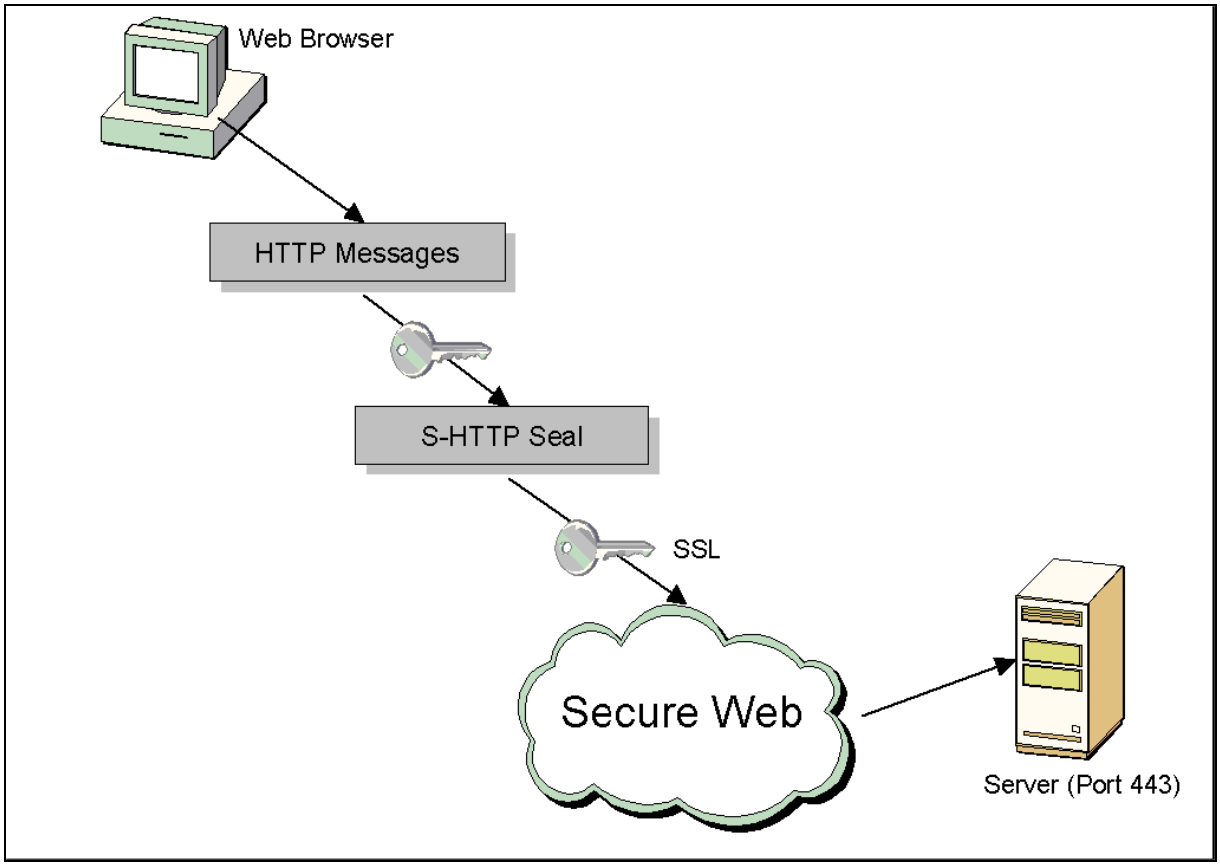

Abbildung 16: S-HTTP on Top of SSL

Gut ist, dass S-HTTP und SSL sich nicht ausschliessen. Diese können einfach nebeneinander bestehen bei Anordnung von S-HTTP zuoberst auf SSL (siehe Abb. 16). Beim gemeinsamen Nutzen dieser Komponenten ist die Sicherheit besser gewährleistet als beim Nutzen von SSL oder S-HTTP alleine. Ist diese Sicherheit übertrieben? Mit Sicherheit nicht. Man kann nie genug Sicherheit im Internet einbinden [31].

[31] Literatur [ECS] Seite 506, „Are SSL and S-HTTP Mutually Exclusive?

8.2.3 SSL Connection (Verbindung) Aufbau

Der Messageinhalt von allen Kommunikationen zwischen Web Server und Kunde ist geschützt vor einer Routenänderung. Die in diese Transaktion involvierten Parteien wissen, dass was sie sehen exakt das ist, was von der anderen Seite gesendet wurde.

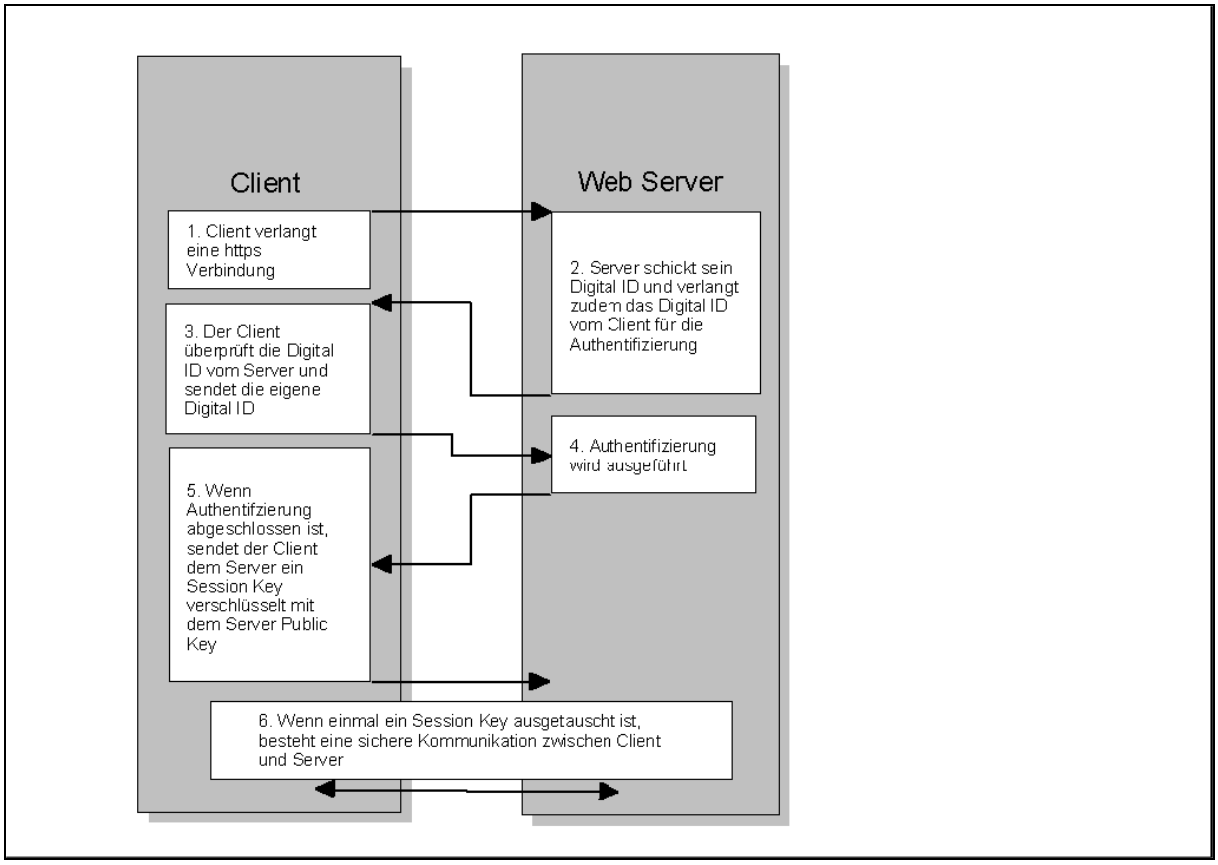

Abbildung 17: Connection Aufbau „https://“ zwischen Browser und Web Server

Die Grafik in der Abb. 17 illustriert den Aufbau einer sicheren Kommunikation über SSL, die eine geschützte Kommunikation zwischen Web Server und Client garantiert. Alle Austausche von Digital-IDs geschehen innert weniger Sekunden.

9 Prototyp

9.1 Architektur

Nach dem Dokumentenstudium setzte ich mich an die Planung der Architektur des Prototypes und versuchte eine geeignete Lösung aufzuzeichnen, wie der Ablauf in etwa aussehen sollte. Danach sammelte ich die nützlichen Informationen der verschiedenen Software Komponenten, bevor ich mit der eigentlichen Entwicklung begann.

Folgende Punkte mussten enthalten sein:

- Ausdruck der Bestätigung des Certificate Antrages
- Einfache Verwaltung für Administrator (übers Internet)
- Doppeltes Abholen von Certificates verhindern
- Certificate Raubkopien verhindern
- Automatische Mail Generierung
- Automatisches Mapping von Client Certificate zu Benutzer Account
- Verwaltung von Logfile und Inifile im Administrationstool

9.1.1 Dienste

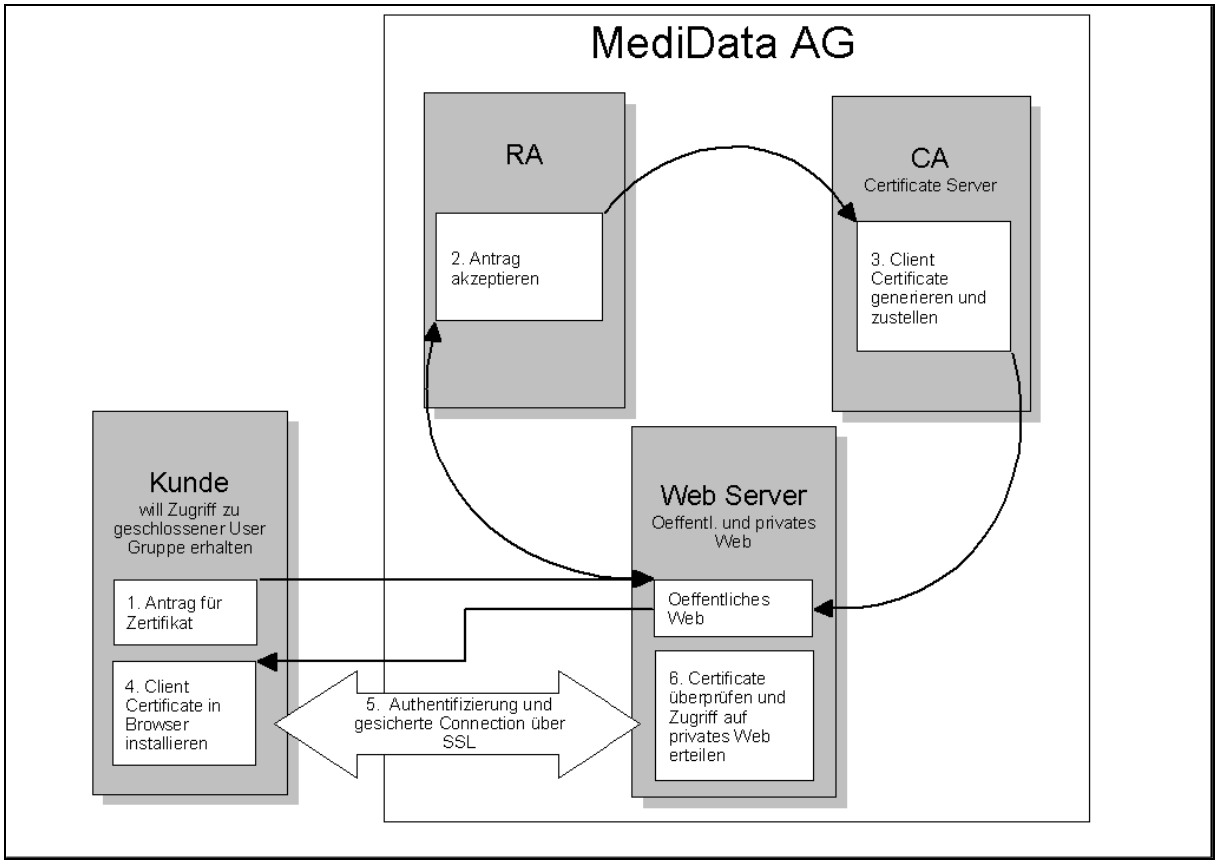

Abbildung 18: Prototyp MediData AG

Abb. 18 illustriert den Grobablauf der Funktionsweise des Prototyps. Im Gegensatz zur Theorie im Kapitel „Certificate-Technologie“ bietet die MediData AG gleich den Webdienst, Registrationsstelle und Certificate Aussteller in einem an.

9.1.2 Online Registrierung

Wie soll die Online Registrierung im Detail ablaufen? Ich wählte eine Variante, die beinahe lückenlos übers Internet abspielt. Einzige Ausnahme bildet der zusätzliche Ausdruck der Bestätigung des Kundenantrages, um wirklich sicher gehen zu können, wer den Antrag stellt.

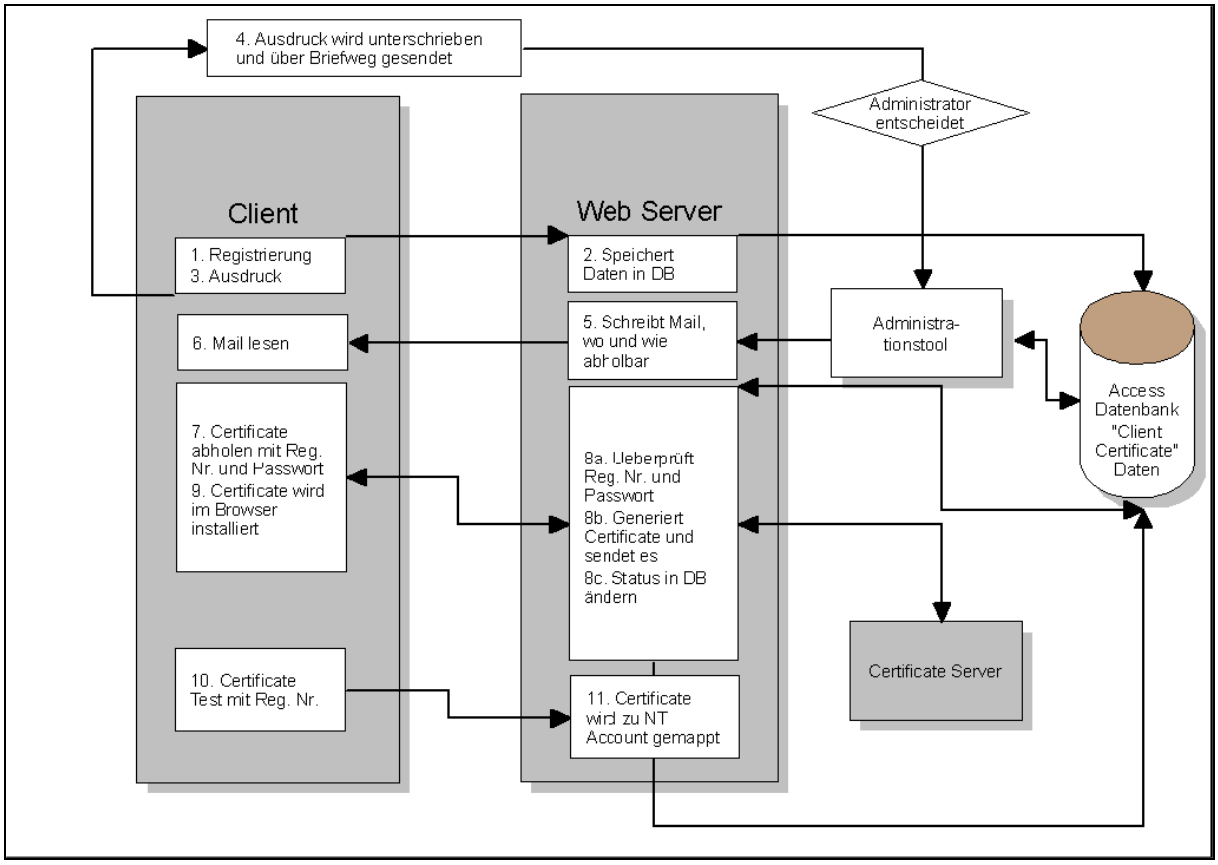

Abbildung 19: Online Registrierung

In der Abb. 19 ist der Ablauf der Registrierung zu sehen, mit der korrekten Reihenfolge und des Zusammenspiels zwischen den einzelnen Komponenten. Die Pfeile symbolisieren den „Datenaustausch“ zwischen den einzelnen Komponenten.

Der Kunde möchte unseren „geschlossenen“ Dienst beanspruchen und hat sich entschieden, ein Certificate anzufordern.

1. User füllt verlangte Kundendaten aus und schickt die Daten übers Netz an den Webserver der MediData AG. Für die später folgende Abholung des Certificates muss ein Passwort erfasst werden.

2. Webserver kontrolliert das ausgefüllte HTML Formular nach Vollständigkeit und speichert die Daten in die Access Datenbank.

3. Im Browser erscheint ein Bestätigungsformular mit den eben erfassten Kundendaten. Der User wird aufgefordert, die Seite auszudrucken.

4. Der Kunde unterschreibt die Bestätigung und schickt diese zusammen mit einer Ausweiskopie an die MediData AG.
5. Der Certificate Administrator prüft den schriftlichen Antrag und entscheidet, ob der Kunde ein Zertifikat erhält oder nicht. Je nachdem startet der Administrator einen Batch, der den Status in der Datenbank ändert und dem User ein Mail zustellt mit der URL und der dazugehörigen Request ID um das Certificate abzuholen.

6. Kunde liest Mail.

7. Der User kopiert die angegebene URL Adresse in den Browser und öffnet entsprechende Seite, wo er nach dem Passwort gefragt wird.

8a. Web Server kontrolliert anhand der Request ID und des Passwortes, ob es wirklich der richtige User ist.

8b. Der Web Server liest die entsprechenden Daten aus der Datenbank und löst damit ein Certificate Request an den Certificate Server aus. Nach dem Generieren wird der Bytestrom an den User übermittelt. Gleichzeit wird sicherheitshalber eine Kopie des Certificates auf dem Server abgespeichert (für manuelles Mapping).

8c. Der Status wird geändert, damit der Kunde das Zertifikat nicht zweimal abholen kann.

9. Der Browser generiert Public und Private Key. Das Certificate wird in den Browser installiert (vorhandene Funktion jedes aktuellen Browsers). Diese Daten werden in einer speziellen Zertifikats Datenbank Client-seitig abgespeichert. Während Installierung wird ein Cookie geschrieben, das verhindern soll, dass das Zertifikat nicht weiterkopiert wird. Denn der Server überprüft jedesmal das Cookie.

10. Der Kunde muss als nächsten Schritt die zweite URL Adresse eintippen und das erhaltene Zertifikat testen, indem er aufgefordert wird, dieses vorzuweisen.

11. Der Webserver prüft das jetzt übermittelte Certificate und liest die Daten aus dem Bytestrom und mappt das Client Certificate zu einem vordefinierten Windows NT Account. Selbstverständlich wird auch bei diesem Vorgang ein Eintrag in der Server-seitigen Datenbank vorgenommen. Ab diesem Zeitpunkt ist der private Web Bereich für den Kunden, gegen Vorweisung seines Certificates, zugänglich.

Aufgrund der Cookie „Nachprüfung“ verzichtete ich auf die Variante, dass Client Certificate auf einer Diskette zuzustellen.

9.1.3 Web

Das Web teilte ich in drei Komponenten ein um wirklich vorzuführen, wie einzelne Bereiche „geschlossen“ werden können:

- Oeffentliches Web (Bestandteil von Default Web Site)
- Privates Web (Bestandteil von Default Web Site)
- Administrations Web

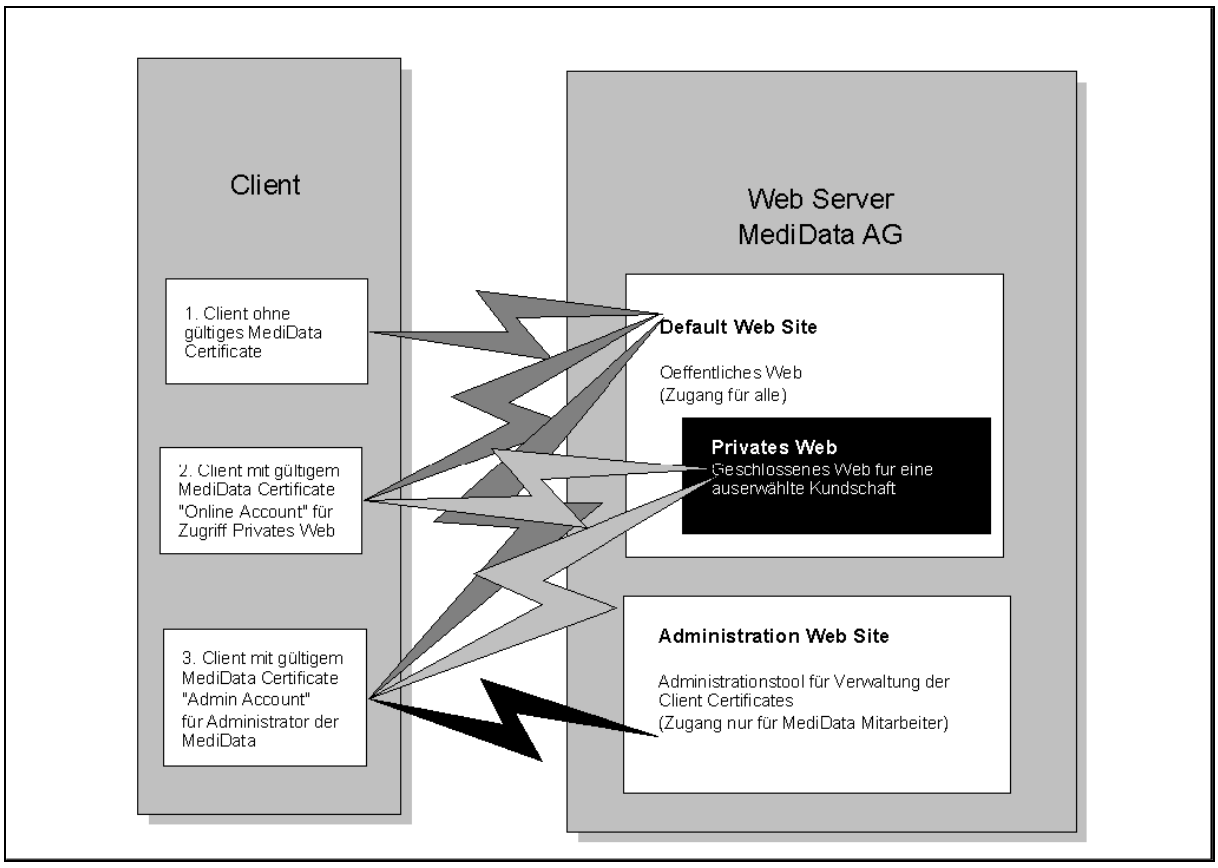

Abbildung 20: Web Sites

Abb. 20 beschreibt die Zugangsberechtigung zu den einzelnen Webbereichen für die drei geplanten Berechtigungstypen, wobei für das „Private“ und „Administration“ Web ein Client Certificate der MediData AG benötigt wird:

1. **Anonymer Web User**
   ⇨ Hat Zugriff auf öffentlichen Web Teil

2. **Kunde des Online Produktes**
   ⇨ Hat Zugriff auf öffentlichen und privaten Web Teil

3. **Certificate Administrator**
   ⇨ Hat Zugriff auf alle drei Webs

Die Certificates für den Kunden des Online Produktes und den Certificate Administrator sind ansich gleich. Nun, wie kann ich diese zwei verschiedenen Usergruppen auseinanderhalten? Dies geschieht durch das Eröffnen von zwei unterschiedlichen Benutzer Accounts auf NT Server (Details folgen). Die Certificates werden nun je nachdem zu einem dieser Accounts gemappt.

Linkstruktur

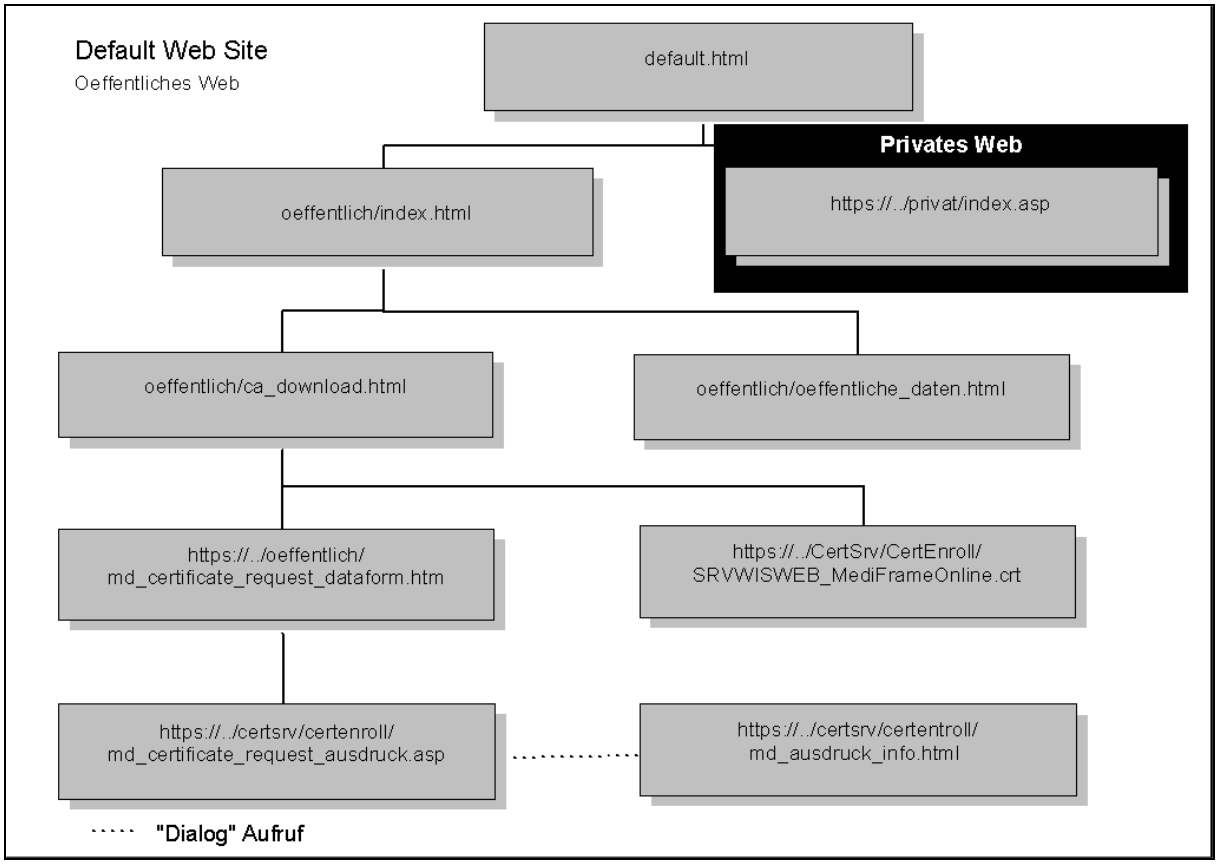

Abbildung 21: Linkhierarchie Default Web Site

Abb. 21 zeigt die Linkstruktur im Bereich Default Web Site und Privates Web, die der Kunde anwählen kann. Zu beachten ist, dass einige Web Seiten nur mit https:// (SSL Connection) aufgerufen werden können, aufgrund der oben beschriebenen Konfiguration.

Beispielsweise kann ein Kunde ohne gültiges Certificate von der default.html Seite nicht zur https://../privat/index.asp gelangen, da diese Seite ein Certificate verlangt und ein gültiges „one-to-one“ Mapping sucht! Dagegen kann von oeffentlich/ca_download.html problemlos auf die Seite https://../oeffentlich/md_certificate_request_dataform.html zugegriffen werden, da diese zwar eine sichere Verbindung herstellt aber kein Client Certificate dazu benötigt bzw. auffordert.

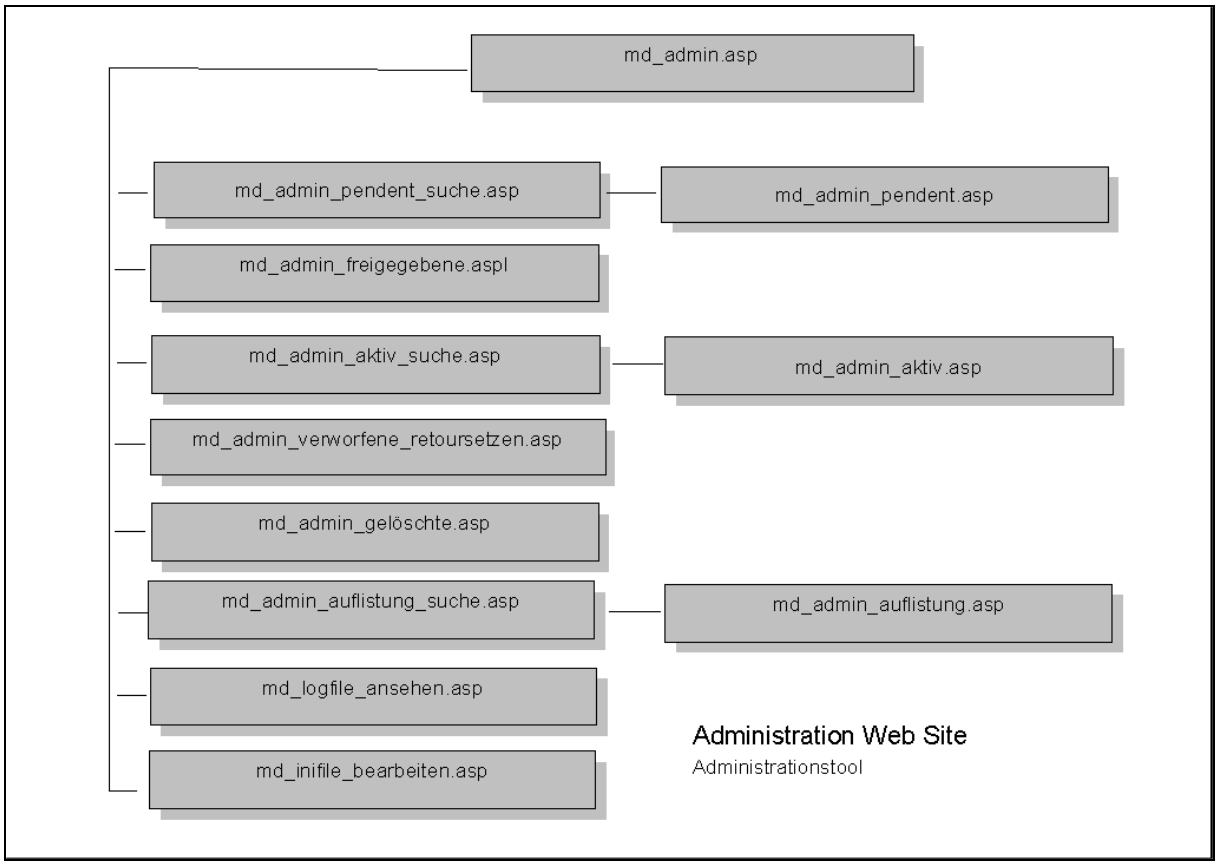

Abbildung 22: Linkhierarchie Administrations-Web Site

Die Struktur im Administrations Web ist in Abb. 22 abgebildet. Für alle Seiten ist ein Client Certificate notwendig, dass zu einem Administrator NT Account gemappt ist.

9.2 Funktionsweise Software Komponenten

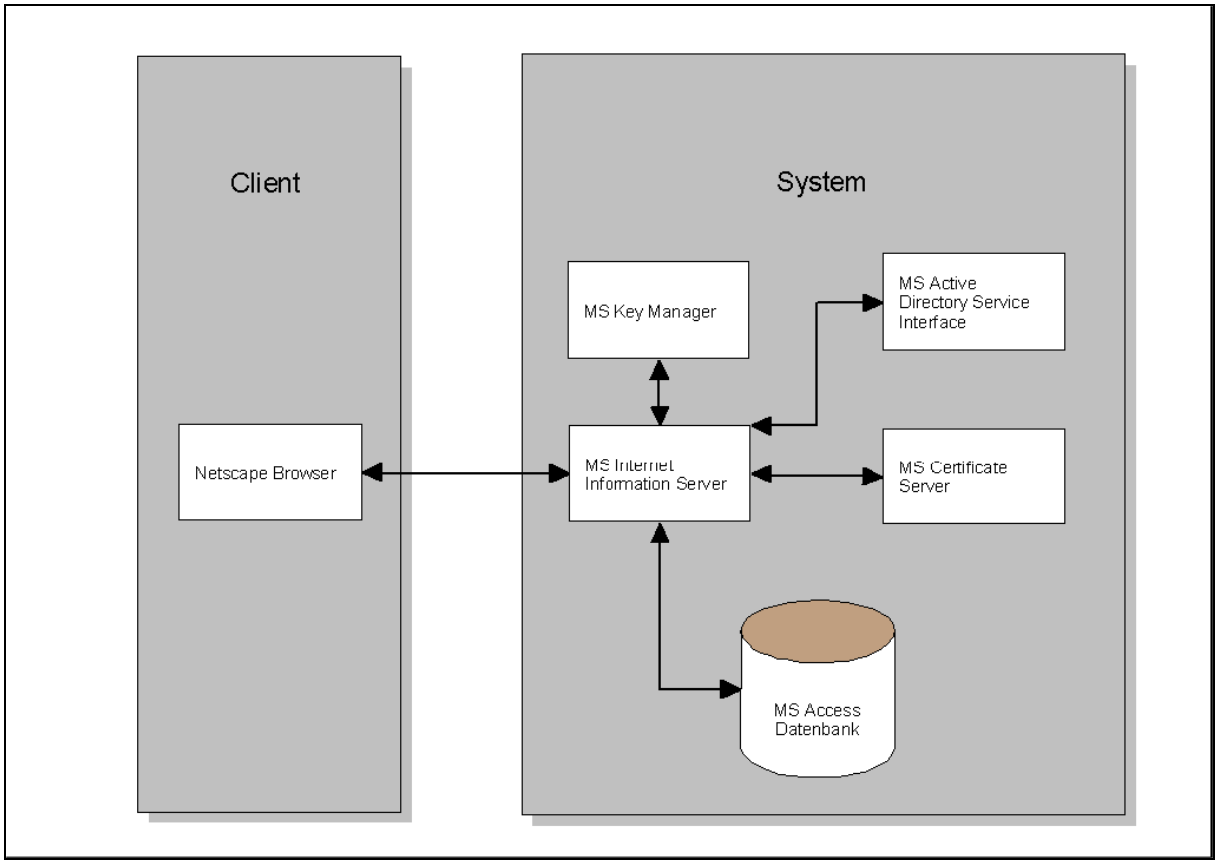

Abbildung 23: Software Komponenten

Die Funktionsweise der einzelnen Software Komponenten werden nachfolgend in den Subkapiteln näher beschrieben. Im Prototyp enthalten sind alle in Abb. 23 aufgeführten Software Komponenten. Die Pfeile bilden den Datenfluss zwischen den einzelnen Komponenten ab.

9.2.1 Internet Information Server (IIS)

Der Webserver Dienst ermöglicht eigentlich das Verwalten und Bereitstellen der Internetseiten. Es können diverse Webs und Directories verwaltet werden, zudem kann auf Web, Directory oder File Ebenen die Berechtigung konfiguriert werden.

Certificate Enrollment (Certificate Einschreibung)

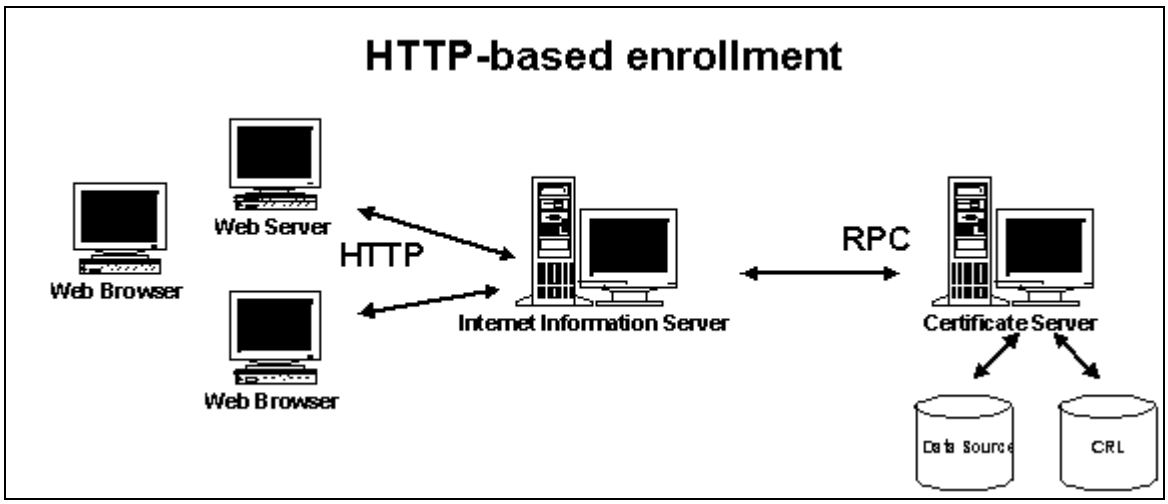

Abbildung 24: HTTP-based enrollment

Abb. 24 zeigt den Standpunkt einer HTTP-basierenden Einschreibung des Microsoft Internet Information Server (IIS). Der Client sendet anhand seines Eintrages im Browser ein Client Request an den Web Server IIS. Dieser leitet die Anfrage weiter an den Certificate Server. IIS benötigt Remote Procedure Call (RPC) um mit dem Certificate Server zu kommunizieren. Der IIS Server kann lokal auf der gleichen Maschine administriert werden wie der Certificate Server (oder auf Remote Server).

9.2.2 Certificate Server

Systemvoraussetzung für die Installation von Certificate Server 1.0 sind:

- Windows NT Server 4.0 mit Service Pack 3
- Internet Information Server IIS 4.0

Das eigentliche Herz der Zertifizierung ist der Certificate Server, der das ganze Handling der Certificates übernimmt. Dieser besteht aus der Server Engine, welche die Certificate-Anforderungen durchführt und anderen Modulen, welche mit der Server Engine kommunizieren.

Um nun ein Certificate anzufordern müssen die Daten für den Request in einem speziellen Format an den Certificate Server gesandt werden. Dieser generiert das Client Certificate und schickt via Web Server das Certificate an den Client weiter.

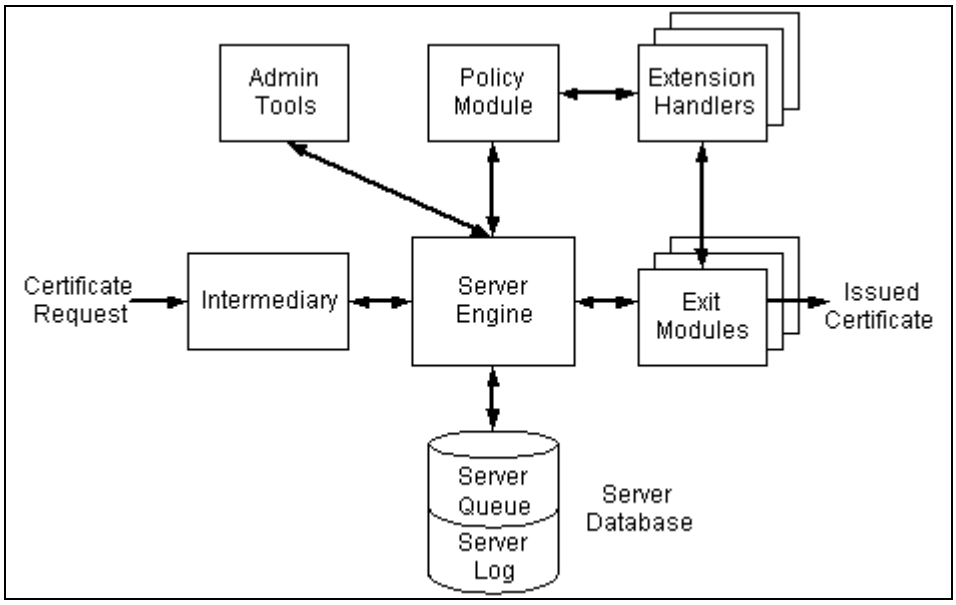

Abb. 25: Microsoft Certificate Server Architektur

Abb. 25 zeigt den Informationsfluss zwischen Certificate Server (in Abbildung „Server Engine“ genannt) und den anderen Komponenten auf. Diese Komponenten sind Server Engine, Server Database, eine oder mehrere Exit Modules, ein policy Module, Administration Tools etc.

Der Microsoft Certificate Server verrichtet folgende Schritte bei einer Certificate Request Verarbeitung:

1. **Request-Empfang**
   Der Certificate Request wird vom Client an eine vermittelnde Applikation gesandt, welche diese in ein PKCS #10 Format (siehe Tab. 6) umwandelt und an die Server Engine weiterleitet.

Tabelle 6: PKCS Standards

Standard	Kurzbeschreibung
PKCS#1	RSA Encryption Standard (512, 1024, 2048 bit)
PKCS#5	Password Based Encryption Standard
PKCS#7	Cryptographic Message Syntax Standard
PKCS#8	Private Key Information Syntax Standard
PKCS#9	Selected Attribute Types
PKCS#10	RSA Certification Request

2. **Request-Genehmigung**
   Die Server Engine ruft die Policy Module auf, welche die Request Eigenschaften durchsuchen und entscheiden, ob der Request authorisiert ist oder nicht und setzen zusätzliche Certificate-Eigenschaften.

3. **Certificate-Gestaltung**
   Wenn der Request angenommen ist, die Server Engine (Certificate Server) nimmt die Informationen aus dem Request und bildet das komplette Certificate.

4. **Certificate-Veröffentlichung**

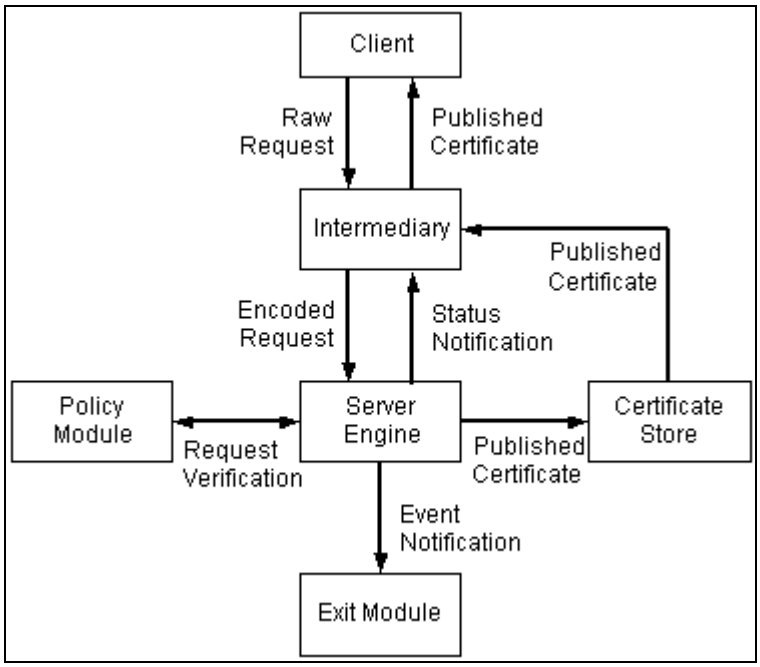

Abbildung 26: Verarbeitung eines Certificate Request durch einen Certificate Server

Die Server Engine stellt das komplette Certificate in den Certificate-Bestand und benachrichtigt die vermittelnde Applikation über den Request Status. Falls das Exit Module den Request angefragt hat, die Server Engine wird das Module benachrichtigen sobald die „Certificate-Abholung“ bereit ist. Dies erlaubt dem Exit Module weitere Operationen zu verrichten, wie das Certificate in einem Directory Service zu veröffentlichen. Der Vermittler bekommt das publizierte Certificate vom Certificate Bestand und reicht es dem Client weiter. Abb. 26 illustriert, wie die Zusammenhänge zwischen den Komponenten bestehen.

Der Certificate Server generiert Certificates in X.509 Format-Standard (Schlüsselbreite Prototyp 512-bit). Certificates in X.509 Format werden für Authentifizierungen zwischen Client und Server - bezüglich einer gesicherten Kommunikation mit Verwendung des Secure Sockets Layer Protokoll (SSL, Schlüsselbreite Prototyp nur 40-bit!) - eingesetzt.

Aufdeckung von „Revoked Client Certificates“

Alle gängigen CA unterhalten eine Certificate Revocation List (CRL). Dies ist eine Liste, die widerrufene Certificates öffentlich auflistet, die vor dem Ablaufdatum rückgängig gemacht wurden. Zum Beispiel: Wenn eine CA ein Client-Certificate ausstellt und im nachhinein merkt, dass die Identitätsinformationen falsch sind, kann die CA dieses Certificate widerrufen! Da dieses Certificate physisch nicht gelöscht werden kann, wird dieses Certificate in der CRL aufgelistet.

Das Windows NT Betriebssystem benützt Microsoft CryptoAPI 2.0, ein Dienst, der entscheidet, ob ein Client-Certificate widerrufen wurde oder nicht.

Interfaces

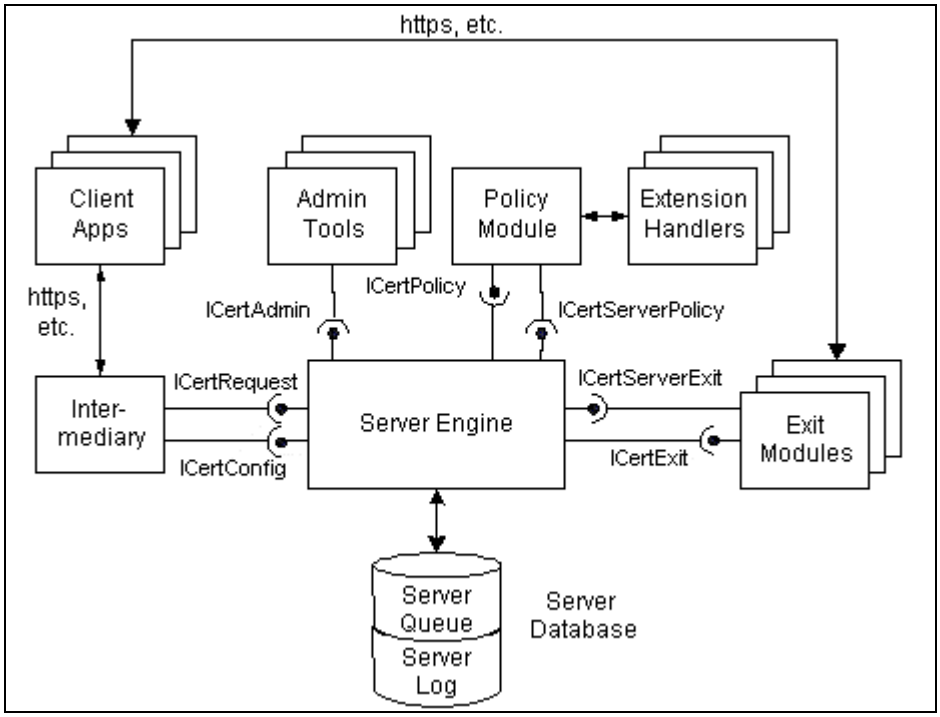

Abbildung 27: Certificate Server Microsoft – Interfaces für Programmierung

Abb. 27 bildet die Interfaces für die Programmiersprachen Visual Basic, C/C++ und Java ab.

9.2.3 Active Directory Services Interface (ADSI)

ADSI ist ein Set von Programm Interfaces für Zugriff auf verschiedenste „Directory- und Administrationsdienste“. Microsoft Produkte, die ADSI unterstützen sind Windows NT 4.0 Server, Exchange, IIS und Site Server. ADSI wird in NT 5.0 das Interface zum Betriebssystem sein. Damit erhält man Zugriff auf alle Konfigurationsinformationen eines Betriebssystemes. Zugriff zu ADSI haben alle Sprachen die COM unterstützen.

ADSI benötige ich um die Certificates zu NT User Accounts zu mappen, d.h. ich vergebe somit gewisse Zugriffsberechtigungen auf gewisse Directories oder Files im Web. Man muss sich das so vorstellen, wie wenn man Beispielsweise nur dem Administrator die Berechtigung für das Directory c:\privat erteilt.

Hinweis: Um von Active Server Pages aus auf ADSI Objekte zuzugreifen, muss man als Administrator eingeloggt sein oder wenn man „anonymous access“ (IUSR_MACHINENAME) konfiguriert hat, muss man Administrator Privilegien haben!

IIsCertMapper

Das für mich wichtigste Object von ADSI für diese Arbeit ist sicherlich IisCertMapper, welches das „Mapping“ von Client Certificates und Windows NT User Accounts erlaubt. Das Client Certificate wird zu einem NT Account gemappt.

Nachfolgend werden die Objekte detailiert beschrieben, wie wir es aus den Handbüchern von Programmiersprachen gewohnt sind:

AdsPath:
IIS://MachineName/W3SVC/n/IIsCertMapper
(Wobei MachineName irgendein Name oder "LocalHost" sein kann)

Syntax:
varReturn = IIsCertMapper.{Method}

Methoden siehe Tab. 7.

Tabelle 7: Methoden des Objektes IIsCertMapper

Methode	Beschreibung
CreateMapping	Kreiert Mapping von Certificate zu NT User Account
DeleteMapping	Löscht bestehendes Mapping
GetMapping	Erhält ein bestehendes Mapping
SetAcct	Aendert ein neues NT Account für ein bestehendes Certificate Mapping
SetEnabled	Aktiviert oder deaktiviert ein bestehendes Mapping
SetName	Aendert Mapping Name
SetPwd	Aendert NT Account Passwort des bestehenden Mappings

CreateMapping

Syntax:
IIsCertMapper.CreateMapping vCert, NtAcct, NtPwd, strName, IEnabled

Parameters:
vCert
Client Certificate. Das Certificate ist entweder ein String oder ein Array von Bytes. Normalerweise wird dieser Parameter von der ClientCertificate Kollektion der ASP Request Objekte abgefüllt.

NtAcct
Windows NT Account Bezeichnung (String).

NtPwd
Beinhaltet Passwort des Windows NT Accounts (String).

strName
Mapping Bezeichnung (String).

IEnabled

True aktiviert Mapping, False deaktiviert Mapping.

Code Beispiel für Active Server Page:

```
<%
Dim CertObj, vCert
vCert = Request.ClientCertificate("CERTIFICATE")
Set CertObj = GetObject(„IIS://srvwisweb/w3svc/1/IisCertMapper“)
'n = 1, 2, 3 usw. – 1 für Default Web Site, 2 für Administration Web Site...
CertObj.CreateMapping vCert, "dummyAccount", "admin", "TestMapping", True
%>
```

DeleteMapping

Syntax:
IIsCertMapper.DeleteMapping IMethod, vKey

Parameters:
IMethod
Bezeichnet „seek method“ für die Mapping Suche. Gültige „seek method“ sind 1, 2, 3 oder 4.
1 = Suche nach Certificate
2 = Suche nach Name bzw. Mapping Beschreibung
3 = Suche Windows NT Account
4 = Suche nach 1-based numeric String Index (Bsp. „1“, „2“ usw.)

vKey
Bezeichnet den Inhalt der bei IMethod spezifizierten Codes.
1 = Certificate
2 = Name bzw. Mapping Beschreibung (String)
3 = Windows NT Account (String)
4 = Zahl (String)

Code Beispiel:

```
<%
Dim CertObj
Set CertObj = GetObject(„IIS://srvwisweb/w3svc/1/IisCertMapper“)
' Sucht hier nach Windows NT Account
CertObj.DeleteMapping 3, " dummyAccount "
' Löscht alle Mappings mit diesem NT Account
%>
```

9.2.4 Zugriffskontrolle

Windows NT File System - Zugriffskontrolle

Mit der Windows NT File System (NTFS) Zugriffskontrolle, der Grundlage der Web Server Sicherheit, kann der Zugriff auf Files und Directories definiert werden (mit entsprechender Zuweisung von Windows NT User und Gruppen Accounts).

Nach dem Zuweisen von Berechtigungen auf Files und Directories, muss ein User sich „ausweisen“, sei dies mittels Passwort oder neu mit Client Certificate, um Zugriff zu erlangen. Man kann die Web Server-Authentifikation so konfigurieren, dass der User sich zuerst ausweisen muss, bevor eine entsprechende Internet Verbindung erstellt wird (Hierfür bestehen verschiedene Varianten) [32].

[32] Literatur [MSS]

Webserver

IIS unterstützt drei verschiedene Zugriffssicherheiten, die natürlich auch kombiniert eingesetzt werden können:

Anonymous Access und Authentication Control
- Allow Anonymous Access
  ⇨ Jeder User hat Zugriff auf die Ressourcen.
- Basic Authentication
  ⇨ User ID und Passwort werden vom Benutzer verlangt und in Textform übermittelt.
- Windows NT Challenge/Response
  ⇨ Dies benutzt eine verschlüsselte Technik für die Authentifizierung des Users und benötigt keine Uebermittlung von aktuellen Passwörtern etc. übers Netzwerk.

Secure Communication
- Require Secure Channel, when accessing this resource
  ⇨ Falls eine SSL Connection gewünscht wird, kann diese Eigenschaft aktiviert werden.

- Client Certificate Authentication
  - Do not accept Client Certificate
    ⇨ Es werden keine Certificates akzeptiert für SSL.
  - Accept Certificate
    ⇨ Certificates werden akzeptiert aber nicht benötigt.
  - Require Certificate
    ⇨ Es muss ein Certificate vorgewiesen werden.

    - Enable Client Certificate Mapping
      ⇨ Hier können nun Gruppen von Certificates mittels Wildcard Regeln den Windows NT User Accounts zugewiesen werden. Es besteht auch die Möglichkeit, spezielle Client-Certificates zu einem Account zu mappen!

IP Adresses und Domain Name Restrictions
- Hier kann mittels IP Adresse und Domain Name der Zugriff auf nur diese User beschränkt oder aber auch gesperrt werden.

Berechtigung Client Certificate

Man kann verlangen, dass der User nur über einen „secure link“ auf unser Web Zugriff hat und dies benötigt ein Client Certificate. Wie auch immer, dies schützt noch keinesfalls vor unauthorisiertem Zugriff. Denn jeder User mit irgendeinem Client Certificate kann eine gesicherte Connection errichten und somit Zugriff zu den Resourcen erlangen. Um das Web vor unauthorisierten Zugriff zu schützen muss folgendes konfiguriert werden:

Ich benutze Basic und Windows NT Challenge/Response authentication, zusätzlich zum Verlangen eines Client Certificate und erstelle ein Windows NT Account Mapping für Client-Certificate.

Der Web Server kann keine Client-Certificates generieren, solange kein Server Certificate installiert ist. Falls auf eine spezielle Web Seite eine Security eingerichtet wird, werden automatisch alle Directories und Files, welche zu dieser Seite gehören, mit dem gleichen Security eingerichtet.

Freigabe Client Certificate-Zugriff

Im Internet Service Manager, eine Web Seite, ein Verzeichnis oder eine Datei auswählen und Eigenschaften öffnen. Nun muss die Verzeichnis oder Datei Security Eigenschaft unter „Secure Communication“ angeklickt werden.

Im „Secure Communications“ Fenster, „Require Secure Channel when accessing this resource“ auswählen. Benötige einen gesicherten Channel bedeutet, dass ein User keine Verbindung aufbauen kann ohne einen gesicherten Link herzustellen, d.h. die URL des Linkes muss beginnen mit https://.
Unter Client Certificate Authentication eines der folgenden Varianten für die Freigabe eines Client-Certificates auswählen:

- Akzeptiere Certificate
  ⇨ Der User kann problemlos auf Resourcen zugreifen mit keinem oder irgendeinem Client-Certificate.

- Benötige Certificate
  ⇨ Der Server verlangt ein Client-Certificate bevor eine Verbindung zugelassen wird. User ohne ein gültiges Client-Certificate wird abgewiesen.

Mapping Client Certificates zu User Accounts

Man kann User, welche mit einem Client Certificate einloggen, mittels einem kreierten Windows NT Mapping, authentifizieren. Dazu kann im Web Server der Teil „Certificate Mapping“ gebraucht werden. Dort kann man zum einen ein spezielles Certificate zu einem Account mappen (ein one-to-one Mapping) oder man mappt mehrere Certificates zu einem Account. Um mehrere Certificate zu mappen, benötigt man eine Wildcard Definition. Beispiel: Wenn ich unter allen Usern, welche mit einem Client Certificate einloggen, nur denjenigen Zugriff geben möchte, die das Certificate vom Aussteller „MediData AG“ haben, erstelle ich ein Wildcard Mapping (Beispiel: Certificate Aussteller = medi*. D.h., alle Certificates, die von der MediData ausgestellt wurden, erhalten Berechtigung) auf ein Windows NT Account.

Im Fenster „Secure Communication“, kann entschieden werden, ob ein „Secure channel“ benötigt wird oder nicht, falls ja, ob ein Client-Certificate benötigt wird oder nicht. Nun muss „Enable Client Certificate Mapping“ editiert werden. Hier können neue dazugefügt werden. Dann muss das gewünschte Certificate ausgewählt werden (Certificate muss auf dem Server zwischengespeichert sein in einem TextFile Format).
Das Problem liegt nun darin, dass natürlich nicht automatisch diese Informationen zu uns gelangen. Da wir aber selber Client Certificates generieren, können wir im gleichen Zug diese bei uns separat speichern! Möchte man nun aber auch andere Client Certificate zulassen, die zum Beispiel von Swisskey ausgestellt wurden, muss in einem Zwischenschritt das Certificate vom Client gesendet und „bewusst abgefangen“ werden, um es Server-seitig zu speichern (mittels ASP File) und es zu einem Account zu mappen.

9.3 Installation Server Komponenten

Die Installation Server Komponenten beschreibt detailliert alle Einstellungen, die vorgenommen wurden beim Einrichten des Servers. Anhand dieser Informationen soll es möglich sein, alles zu rekonstruieren um einen Certificate Dienst einzurichten.

Folgende Komponenten von Option Pack (Zusatz Win NT 4.0) wurden installiert:
- Internet Information Server IIS
- Certificate Server
- Frontpage 1998 Server Extension
- MS Data Access Server
- Management Console
- NT Option Pack Common Files

9.3.1 Internet Information Server (IIS)

Tabelle 8: Einstellung IIS 4.0

Bezeichnung	Einstellung
WWW Service: (default Web)	C:\internet\medidataweb
FTP Service: (default FTP)	C:\internet\medidataftp
Application Installation Point	C:\apps

Folgende physische Struktur wurde standardmässig erstellt:
C:\internet\Catalog.wci
\iissamples
\mail
\mailroot
\medidataftp
\medidataweb
_vti_txt
_vti_log
_vti_bin
_private
\images
\cgi-bin
_vti_pvt
_vti_cnf
\postinfo.html
_vti_inf.html
\default.asp
\scripts

9.3.2 Certificate Server

Tabelle 9: Einstellung Certificate Server

Bezeichnung	Einstellung
Certificate Authority Certificates	⦿ Shared Folder: C:\zertifikat (Directory muss auf gleichem Server sein wie Certificate Server. Ausserdem soll dieses Verzeichnis öffentlich sein, damit alle User ein Zertifikat installieren können)
Database Location:	C:\winnt\system32\certlog (default Vorschlag, hier ist eine MS Access Datenbank hinterlegt, die alle Einträge entgegennimmt)
Log Location:	C:\winnt\system32\certlog (default Vorschlag)
	☑ Show Advanced
CSP	MS Base Cryptografic Provider v1
Hash: (Algorithmus)	MD5 (Standard Vorschlag)
	☑ Make this Certificate Server the default
	⦿ Root CA (Standard Vorschlag)
CA Name (WICHTIG: keine Leerzeichen!)	MediFrameOnline
Organisation	MediData AG
Organisation Unit	MediFrame
Localy	Luzern
State	Kt. Luzern
Country	CH
CA-Description	Usergruppe fuer MediFrame Online

Hinweis: Automatischer Start dieses Dienstes beim Aufstarten von Win NT 4.0. Selbstverständlich kann dieser Dienst manuell gestoppt werden.

Es wurde automatisch folgendes Verzeichnis mit dem Server Certificate für MediData AG erstellt (momentan ohne Key, muss später im Key Manager generiert werden):

C:\zertifikat\SRVWISWEB_MediFrameOnline_Exchange.crt
\SRVWISWEB_MediFrameOnline.crt
\Certsrv.txt
\cacerts.htm

Inhalt von Certsrv.txt:
MediFrameOnline, MediFrame, "MediData AG", Luzern, "Kt. Luzern", CH,
SRVWISWEB\MediFrameOnline, SRVWISWEB_MediFrameOnline_Exchange.crt,
SRVWISWEB_MediFrameOnline.crt, "Usergruppe fuer MediFrame Online"

Inhalt von cacerts.htm:
Schlägt dem „Kunden“ vor, das Certificate des CA (akzeptiert, dass MediData eine sichere CA ist) in seinem Browser zu installieren, damit ab dann eine sichere Verbindung stattfindet.

Die MS Access DB für die Verwaltung der Certificate wurde in folgendes Directory gestellt:

C:\winnt\system32\certlog\certsrv.mdb

Weitere Dateien:
C:\winnt\system32\certsrv\certenroll
\srvwisweb_mediframeonline.crt (CA Schlüssel)
\nsrev_mediframeonline.asp

9.3.3 Active Directory Service Interface (ADSI)

Für das automatische Mapping von Client Certificates auf NT User Accounts fand ich bei der Microsoft unter http://backoffice.microsoft.com/downtrial/moreinfo/adsi2.asp den Zusatz Active Directory Services Interfaces ADSI 2.0 den ich herunterlud und auf dem Server installierte mittels Befehlsaufruf adsx86.exe.

ADSI ist lauffähig unter Microsoft Windows NT Server und Windows NT Workstation Version 4.0 Service Pack 3 Release (Build 1381: Service Pack 3) und Microsoft Windows 95 Betriebssystem.

9.3.4 Mail

Um von den Scriptfiles aus automatisch Emails zu generieren, benötigte ich ein Mailmodul für VisualBasic Scripts. Fündig wurde ich auf folgender URL Adresse http://www.persits.com/aspemail.html der Persits Software Inc., die das AspEmail 1.0 als Freeware zur Verfügung stellen.

Nach dem Download der Files kopierte ich das DLL auf den Server C:\apps\ASP_DLL\aspemail.dll und registrierte dieses durch die Ausführung des Befehls
C:\ regsrv32 C:\apps\ASP_DLL\aspemail.dll.

Ab diesem Zeitpunkt war das Mail fürs Senden von Emails aktiv. Selbstverständlich ist es nicht möglich, mit diesem einfachen Modul Emails zu empfangen.

9.4 Konfiguration Server Komponenten

Massgebend für folgende Konfiguration der Server Komponenten war die Architekturplanung die zu Beginn erstellt wurde.

Um einen Certificate Dienst anzubieten ist es selbstverständlich notwendig, dass unser Web Server selber ein Server Certificate besitzt, damit wir uns gegenüber dem Client authentifizieren können. Ausserdem sind andere wichtige Einstellungen vorgenommen worden.

9.4.1 Generation Key Pair

Für unser Server Certificate benötigen wir zuerst einen Private und Public Key den wir im Key Manager generieren lassen und Server-seitig in der Schlüsseldatenbank abspeichern.

Schritt für Schritt Beschreibung:

1. Key Manager öffnen
2. Local Computer auswählen
3. New Key Pair
4. ⦿ Automatically send the request to an online authority: Beispielsweise Microsoft Certificate Server

Formularinhalt (siehe Tab. 10).

Tabelle 10: Formular Key Manager

Bezeichnung	Einstellung
Key Name	MediFrameOnlineProdukt
Passwort	Administrator
Bit Length	512
Organisation	MediData AG
Organisation Unit	MediFrame
Common Name (Beispiel http://www.medidata.ch – wichtig, da diese durch DNS überprüft wird, falls Aenderung – neuen Key generieren)	SRVWISWEB
Country	Switzerland
State	Kt Luzern
City	Luzern
IP Adresse Web Server	192.168.1.9
Port	80

9.4.2 Server Root Certificate in Web Server

Damit wir unserem eigenen Server Root Certificate vertrauen, musste eine Einstellung im „Internet Service Manager (HTML) von Microsoft Internet Information Server erledigt werden.

9.4.3 Certificate Authority Certificates im Web Server

Um User bzw. deren Client-Certificate zu authentifizieren, welches von irgendeinem Certificate Authority ausgestellt wurde, benötigt der Server einen Eintrag, dass wir diesem Certificate Authority vertrauen. Um ein neuen Certificate Authority in unserer vertrauten Authorities hinzuzufügen, muss explizit ein Authority Certificate, genannt Root Certificate, in den Web Server hinzugefügt werden (Bemerke, dass bereits alle gängigen Certificate Authority Root Certificates Standardmässig im Web Server integriert sind). Dieser Vorgang kann mit dem Microsoft Internet Explorer (auf Server) oder mit dem Befehl „lisca.exe“ erfolgen, um das Root Certificate in den Web Server zu implementieren.

9.4.4 Einfügung Certificate Authority Certificate

Logon Web Server als Windows NT User Administrator (Wichtig: Alle anderen User sind nicht berechtigt).

Fordere ein Root Certificate eines Certificate Authority an. Hinweis, beim Benutzen des Microsoft Certificate Server 1.0 für Ausstellung von eigens generierten Certificates, muss ebenfalls ein Root Certificate installiert werden, um diesen Dienst überhaupt zu beanspruchen. Da wir in der Testumgebung selbstverständlich diese Variante benutzen, haben wir ein MediData AG Root Certificate beim Installieren des Certificate Server generiert.

Im Internet Explorer (IE) 4.0 einfach den Pfad des Server Certificates „c:/zertifikat/srvwisweb_mediframeonline.crt“ (die Dateierweiterung für solche Root Certificates ist *.crt. Selbstverständlich könnte dies auch von einer Diskette so mittels Browser installiert werden) erfassen und bestätigen.

Um diese Root Certificate Informationen vom Web Browser zum Web Server zu spiegeln bzw. zu kopieren muss folgendes erledigt werden:

1. Oeffnen einer „command-prompt“ Fensters und starten dieses Befehls
   ⇨ %windir%\system32\inetsrv\lisca.exe

2. Falls alles korrekt kopiert wurde erfolgt Meldung „List of certifying authorities <CA> successfully transferred to IIS“

3. Um die Aenderung zu aktivieren muss der Web Server Service gestoppt und neu gestartet werden (Hinweis, anstelle des unten aufgeführten Prozedere kann auch der Server Computer neu gestartet werden).

4. Oeffnen einer „command-prompt“ Fensters und starten dieses Befehls
   ⇨ net stop iisadmin /y

5. Nun müssen wieder alle Services gestartet werden

6. Oeffnen einer „command-prompt“ Fensters und starten dieses Befehls
   ⇨ net start %service name%
   (Jeder einzelne Dienst muss separat gestartet werden Web, FTP, SMTP oder NNTP)

9.4.5 Benutzer Account

Für den Prototyp erstellte ich zwei neue NT User Accounts.

Internet Kunde

Für die Zulassung ins Private Web wurde Account gem. Tab. 11 eröffnet.

Tabelle 11: NT Account für Internet Kunde

Was	Einstellung
Username	MEDIDATA\SRVWISWEB_Test
Name	Client Certificate
Beschreibung	Testkunde MediFrameOnline
Passwort	Admin
Member of	Domänen-Benutzer SRVWISWEB_MFO_Test

Internet Administrator

Für die Zulassung ins Administration Web wurde Account gem. Tab. 12 eröffnet, der nur für bestimmte Mitarbeiter der MediData AG bestimmt ist.

Tabelle 12: NT Account für Internet Administrator

Was	Einstellung
Username	MEDIDATA\SRVWISWEB_Test_admin
Name	Administrator für Freigabe Certificate
Beschreibung	Administrator für Freigabe Certificate
Passwort	Admin
Member of	Administratoren Domänen-Admins Domänen-Benutzer MDAdmin MDGAdmin MDGsrvwiswebadmin Mdlocaladmin SRVWISWEB_MFO_Test

9.4.6 ODBC Treiber

Um von den Active Server Pages auf die MS Access Datenbank zugreifen zu können, erstellte ich einen ODBC Eintrag (siehe Tab. 13). Diese Datenbank beinhaltet alle Certificate Daten.

Tabelle 13: ODBC Treiber Einstellung

Was	Einstellung
System DSN: Name Treiber	 CertSrv MS Access Treiber (*.mdb)
Data Source Name	CertSrv
Datenbankpfad	C:\winnt\system32\certlog\certificate.mdb
Login Name	Admin
Passwort	--

9.4.7 Internet Information Server (IIS)

Wie in Abb. 20 geplant wurden die drei Webs erstellt.

Einrichtung Default Website

Die einzelnen Einstellungen sind in den Tab. 14 – 19 zu sehen.

Tabelle 14: Erstellung Default Web

Was	Einstellung
IP Adresse	All Unassigned
TCP Port	80 (Internetstandard)
SSL Port	443 (Internetstandard)
Physischer Name	C:/internet/medidataweb
Virtueller Web Name	Srvwisweb
Files	Default.html
Defaultseitenverweis	Default.html
Dialog Authentication Dialog	☑ Allow Anonymous Access: IUSR_SRVWISWEB ☐ Basic Authentication ☑ Windows NT Challenge/Response
Dialog Secure Communications	☐ Require Secure Channel when accessing this resource ◉ Do not accept Client Certificate ○ Accept Certificates ○ Require Client Certificates ☐ Enable Client Certificate Mapping

Tabelle 15: Erstellung öffentliches Unterverzeichnis

Was	Einstellung
Physischer Name	C:/internet/oeffentlichweb
Virtueller Directory Name	Srvwisweb/oeffentlich
Files	Ca_download.html Index.html Md_certificate_request_dataform.html Oeffentliche_daten.html
Dialog Authentication Dialog	☑ Allow Anonymous Access: IUSR_SRVWISWEB ☐ Basic Authentication ☑ Windows NT Challenge/Response
Dialog Secure Communications	☐ Require Secure Channel when accessing this resource ◉ Do not accept Client Certificate ○ Accept Certificates ○ Require Client Certificates ☐ Enable Client Certificate Mapping

Tabelle 16: Erstellung privates Unterverzeichnis

Was	Einstellung
Physischer Name	C:/internet/privatweb
Virtueller Directory Name	Srvwisweb/privat
Files	Index.asp
Dialog Authentication Dialog	☐ Allow Anonymous Access ☐ Basic Authentication ☑ Windows NT Challenge/Response
Dialog Secure Communications	☑ Require Secure Channel when accessing this resource ○ Do not accept Client Certificate ○ Accept Certificates ◉ Require Client Certificates ☑ Enable Client Certificate Mapping
Dialog Account Mappings „one-to-one mapping“	MEDIDATA\SRVWISWEB_Test (Nur User mit diesem speziellen Account haben Zugriff auf dieses Directory bzw. Subdirectory und Files)

Tabelle 17: Bestehendes CertEnroll Unterverzeichnis

Was	Einstellung
Physischer Name	C:/winnt/system32/certsrv/certenroll
Virtueller Directory Name	Srvwisweb/certsrv/certenroll
Files	Blank.asp Cacerts.htm Ceaccept.asp Ceadv.asp Ceenroll.asp Certdef.htm Default.htm Index.htm Kgaccept.asp Krenroll.asp Md_ausdruck_info.html Md_certificate.asp Md_certificate_install.asp Md_certificate_request_ausdruck.asp Md_certificate_request_dataform_insert.asp Md_test.asp Md_test1.asp Newcert.cer Nsrev_mediframeonline.asp SRVWISWEB_mediframeonline.crt
Dialog Authentication Dialog	☑ Allow Anonymous Access: IUSR_SRVWISWEB ☐ Basic Authentication ☑ Windows NT Challenge/Response
Dialog Secure Communications	☑ Require Secure Channel when accessing this resource ◉ Do not accept Client Certificate ○ Accept Certificates ○ Require Client Certificates ☐ Enable Client Certificate Mapping (Für online Anmeldung bzw. Uebermittlung der Daten und der späteren Generierung des Certificates wird eine SSL Verbindung hergestellt. Selbstverständlich ist in diesem Falle noch kein Client Certificate nötig)

Tabelle 18: Erstellung Java Unterverzeichnis

Was	Einstellung
Physischer Name	C:/internet/java /classes
Virtueller Directory Name	Srvwisweb/java /classes
Files	Classes/HollywoodText/HollywoodText.cab Classes/HollywoodText/Hollywoodtext.class
Dialog Authentication Dialog	☑ Allow Anonymous Access: IUSR_SRVWISWEB ☐ Basic Authentication ☑ Windows NT Challenge/Response
Dialog Secure Communications	☐ Require Secure Channel when accessing this resource ◉ Do not accept Client Certificate ○ Accept Certificates ○ Require Client Certificates ☐ Enable Client Certificate Mapping

Tabelle 19: Erstellung Admin Unterverzeichnis

Was	Einstellung
Physischer Name	C:/internet/includes
Virtueller Directory Name	Srvwisweb/admin
Files	Certificate.ini Md_func_sub.asp Md_warten.asp
Dialog Authentication Dialog	☑ Allow Anonymous Access: IUSR_SRVWISWEB ☐ Basic Authentication ☑ Windows NT Challenge/Response
Dialog Secure Communications	☐ Require Secure Channel when accessing this resource ◉ Do not accept Client Certificate ○ Accept Certificates ○ Require Client Certificates ☐ Enable Client Certificate Mapping

Einrichtung Administration Website

Die einzelnen Einstellung sind in den Tab. 20 und 21 zu sehen.

Tabelle 20: Erstellung Administrations Web

Was	Einstellung
IP Adresse	All Unassigned
TCP Port	7351 (kann irgendeine Zahl zwischen 2000 und 9999 sein, jedoch muss Port frei sein. Nur Administratoren haben Zugriff)
SSL Port	2443 (kann irgendeine Zahl zwischen 2000 und 9999 sein, jedoch muss Port frei sein. Nur Administratoren haben Zugriff)
Physischer Name	C:/internet/adminweb
Virtueller Web Name	Srvwisweb:7351 bzw. Srvwisweb:2443
Files	Md_admin.asp Md_admin_aktiv.asp Md_admin_aktiv_suche.asp Md_admin_auflistung.asp Md_admin_auflistung_suche.asp Md_admin_freigegebene.asp Md_admin_gelöschte.asp Md_admin_pendent.asp Md_admin_pendent_suche.asp Md_admin_verarbeitung.asp Md_admin_verworfene_retoursetzen.asp Md_ini_speichern.asp Md_inifile_bearbeiten.asp Md_logfile_ansehen.asp Md_logfile_archivieren.asp
Defaultseitenverweis	Md_admin.asp
Dialog Authentication Dialog	☐ Allow Anonymous Access ☐ Basic Authentication ☑ Windows NT Challenge/Response
Dialog Secure Communications	☑ Require Secure Channel when accessing this resource ○ Do not accept Client Certificate ○ Accept Certificates ◉ Require Client Certificates ☑ Enable Client Certificate Mapping
Dialog Account Mappings „one-to-one mapping“	MEDIDATA\SRVWISWEB_Test_Admin (Nur Administratoren mit entsprechenden Client Certificates der MediData AG haben Zugriff auf dieses Directory bzw. Files)

Tabelle 21: Erstellung Admin Unterverzeichnis (Diese Files werden von Default Web Site und Administrations Web Site benötigt)

Was	Einstellung
Physischer Name	C:/internet/includes
Virtueller Directory Name	Srvwisweb/admin
Files	Certificate.ini Md_func_sub.asp Md_warten.asp
Dialog Authentication Dialog	☑ Allow Anonymous Access: IUSR_SRVWISWEB ☐ Basic Authentication ☑ Windows NT Challenge/Response
Dialog Secure Communications	☐ Require Secure Channel when accessing this resource ◉ Do not accept Client Certificate ○ Accept Certificates ○ Require Client Certificates ☐ Enable Client Certificate Mapping

9.5 Entwicklung

9.5.1 Datenbank

Alle Daten betreffend Zertifizierung werden in einer MS Access Datenbank 97 abgespeichert und verwaltet. Bei der Installation des Certificate Servers ist Standardmässig folgende, in Tab. 22 näher beschrieben, Datenbank integriert. Der Name der Datenbank wurde geändert zu „certificate.mdb“.

Tabelle 22: Bestehende Tabellen in Access Datenbank

Tabellennamen	Beschreibung
CertificateExtension	Certificate Zusatzinformationen
Certificates	Diverse Infos
CRLs	Revocation Lists, die veröffentlicht wurden
Miscellaneous	Verschiedene fortlaufende Daten
NameExtensions	Namen Zusatzinformationen
Names	Certificate Inhalt
RequestAttributes	Request Zusatzinformationen
Requests	Request Informationen und Status

Um mit verschiedenen Certificate Status arbeiten zu können erstellte ich eine Tabelle (Tab. 23, Beschreibung Datenstruktur in Tab. 24), die alles integriert hat, um die benötigten Kundendaten zu verwalten.

Tabelle 23: Neue Tabelle in Access Datenbank

Tabellennamen	Beschreibung
md_certificate_anfrage	Zwischenspeicherung der Request Daten, welche dann von Administrationstool weiterverwendet wird!

Tabelle 24: Beschreibung Datenstruktur „md_certificate_anfrage“

Feldname	Feldtyp	Beschreibung	Beispiel
RegistrierungsNr	AutoWert Zahl	Primary Key	43
RequestIDCertificate	Zahl	Foreign Key	105
DateRequest	Datum	Antragsdatum	01.08.1998
Org	Char	Unternehmensname	Krankenkasse XY
OrgUnit	Char	Abteilung	Tarifabteilung
CommonName	Char	Name und Vorname	Muster Anna
Street	Char	Strasse	Postfach 80
Locality	Char	PLZ und Wohnort	6000 Luzern
Email	Char	Email Adresse	Anna.muster@xy.ch
Phone	Char	Telefon	041/777 77 77
Fax	Char	Fax	041/777 77 66
Country	Char 2	Landabkürzung	CH
Zustellung	Char	Wie Zustellung des Certificates gewünscht: - Internet https - Mail - Diskette	Internet https
Passwort	Char	Passworteingabe, nur bestimmt für Certificate Abholung	Admin
RequestID	Char	Server vergibt eine ID, welche zum Abholen des Certificates bestimmt ist	01_Te3156Mu
Status	Zahl	Status des Certificates: 1 Request 10 Freigabe für Abholung 11 Certificate wurde abgeholt (aktiv) 12 Certificate wurde zu NT Account gemappt 20 Certificate verwerfen (nicht generieren) 99 Certificate revoked	12
StatusDatum	Datum	Datum der Statusänderung	05.08.1998
MailzustellungErfolgreichAm	Text	Bei Freigabe, Verwerfen, Rücksetzung oder Löschung erhält Kunde ein Mail zugestellt. Dieses Feld speichert wann gemailt und ob Error oder nicht.	01.08.98: OK
Cert_ausstell_datum	Datum	Datum, wann das Certificate abgeholt wurde	04.08.1998
NT_Account	Text	Windows NT Account	MEDIDATA\SRVWISWEB_Test
Mapping_Name	Text	Der Name, der für die Beschreibung des Mappings verwendet wird	98-08-31-Krankenkasse_XY-Muster_Anna-01_Te4356Mu
Mapping_erfolgt_am	Datum	Wann das Mapping erstellt wurde	05.08.1998

9.5.2 Web

Um all die Datenbankeinträge sowie Certificate Methoden aufzurufen benutzte ich Active Server Pages (ASP) von Internet Information Server. ASP unterstützt die Scriptsprachen „Visual Basic Script“ und „Java Script“, welche Server-seitig ausgeführt werden. Als Client-seitige Script Sprache wurde Java Script eingesetzt, die von allen gängigen Browsertypen unterstützt wird. Die restlichen Inhalte der diversen Web Pages wurden in HTML realisiert.

Betreffend Entwicklung der einzelnen Seiten möchte ich nur einige mir wichtig erscheinenden Punkte speziel aufführen. Einen Auszug des Sourcecodes finden sie im Anhang unter Sourcecode, der dort direkt dokumentiert ist.

Inifile

Um diverse Einstellungen und Aenderungen als Administrator direkt Online vorzunehmen erstellte ich ein Inifile „certificate.ini“ das nichts anderes als ein ASP File ist. In diesem File sind alle Konstanten aufgeführt die in den restlichen ASP Files benötigt werden. Das Administrationstool hat nun Zugriff auf diese File und lässt eine einfache Verwaltung zu.

Da diese Script Files ja zur Laufzeit interpretiert werden kann so problemlos mit diesem einfachen Trick gearbeitet werden.

```
<%
Const cACCOUNT01_NR             = "01"
Const cACCOUNT01_NAME           = "MEDIDATA\SRVWISWEB_Test"
Const cHTTPS_ZERTIFIKAT_ABHOLUNG_URL    =
       "https://srvwisweb/certsrv/certenroll/md_certificate.asp?cert="
Const cMAIL_ABSENDER_EMAIL      = "webmaster@medidata.ch"
Const cMAIL_ABSENDER_BESCHREIBUNG       = "MediData AG, webmaster"
Const cMAIL_ABSENDER_NAME       = "Webmaster"
Const cMAIL_SUBJECT             = "Client Certificate MediFrame"
Const cMAIL_GRUSSFORMEL         = "Mit freundlichen Gruessen"
%>
```

Abbildung 28: Listing-Inifile Auszug

In Abb. 28 ist ein Auszug aus dem Inifile zu sehen. Die dort aufgeführten Konstanten „Const“ können nun von allen beliebigen ASP Seiten wie folgt implementiert werden:

```
<!--#include virtual="admin/certificate.ini"-->
```

Funktionen

Funktionen können auch mit Visual Basic Script realisiert werden. Ich fügte alle benötigten Funktionen ins ASP File *„md_func_sub.asp“.*

```
...
'*****************************************************************************
' Function func_logfileArchivierenUndLeeren()
' Beschreibung:     Archiviert aktuelles Logfile, d.h. es wird kopiert und
'                   Aktuelles wird gelöscht. Es wird automatisch ein neues File
'                   generiert.
' Input:            --
' Output:           string Archivefilename
' Aktivität:        Siehe Beschreibung
'*****************************************************************************
public Function func_logfileArchivierenUndLeeren()
       Set FileObject = Server.CreateObject("Scripting.FileSystemObject")
       logfile = ini_logFile_pfad()

       dummyText = logfile + "-old-" + ini_datum("yy-mm-dd")

       'kopiert
       FileObject.CopyFile logfile, dummyText, true
       'löschen
       Set FileObject = Server.CreateObject("Scripting.FileSystemObject")
       FileObject.DeleteFile logfile
       'neues Logfile schreiben
       Set FileObject = Server.CreateObject("Scripting.FileSystemObject")
       Set OutStream = FileObject.OpenTextFile (logfile, 8, True)
       OutputString = "Aufzeichnungsbeginn: " + ini_datum("dd.mm.yy")
       OutStream.WriteLine OutputString
       Set OutStream = Nothing
       'Rückgabewert FileName
       func_logfileArchivierenUndLeeren = dummyText
End Function
...
```

Abbildung 29: Listing-Funktionsfile Auszug

In Abb. 29 ist als Beispiel eine Funktion abgebildet, die nun ebenfalls von jeder beliebigen ASP Seite aufgerufen werden kann. File kann wie folgt in die ASP Seiten implementiert werden:

```
<!--#include virtual="admin/md_func_sub.asp"-->
```

Mail

Nachdem der Administrator das Certificate freigibt wird automatisch ein Mail an den Kunden gesandt. Der Sourcecode für den Aufruf dieses Mailobjektes ASPMail 1.0 wird in Abb. 30 aufgezeigt.

```
...
'*****************************************************************************
' Public Function ini_mail_senden(string mailAdresse, string mailEmpfaengerName,
'                   string mailAnrede, string text)
' Beschreibung:     Server sendet automatisch Mail an entsprechende Adresse.
'                   Keinerlei Anpassungen an Mailprogramm etc. Empfang nicht
'                   möglich.
' Input:            mailAdresse = Email Bsp. stefan.bosshard@medidata.ch
'                   mailEmpfaengerName  = Name Bsp. Bosshard Stefan
'                   mailAnrede = Anrede Bsp. Sehr geehrter Herr Bosshard
'                   text = Mailtext
' Output:           string dummytext = Datum plus OK oder Error Status
' Admin:            Benötigt folgendes dll Serverseitig:
'                   C:\apps\ASP_DLL\aspemail.dll
'*****************************************************************************
public   Function   ini_mail_senden(mailAdresse,   mailEmpfaengerName,   mailAnrede,
textMail)

      nZ = Chr(13) + Chr(10)    'Zeilenumbruch
      Dim tempText

      Set Mail = Server.CreateObject("Persits.MailSender")
      Mail.Host = cMAIL_HOST
      Mail.From = cMAIL_ABSENDER_EMAIL
      Mail.FromName = cMAIL_ABSENDER_BESCHREIBUNG
      Mail.AddAddress mailAdresse, mailEmpfaengerName
      Mail.Subject = cMAIL_SUBJECT
      'Anrede plus Grussformel anhängen an Text
            tempText = mailAnrede + nZ + textMail + nZ + cMAIL_GRUSSFORMEL + nZ +
            cMAIL_ABSENDER_FIRMA + nZ + cMAIL_ABSENDER_NAME
      Mail.Body = tempText
      On Error Resume Next
      Mail.Send
      dummytext = ini_datum("dd.mm.yy") + ": OK"
      If Err <> 0 Then
            'Mail konnte nicht zugestellt werden aufgrund folgendem Error: " &
            Err.Description
            'Achtung: dieser Aufruf checkt aber nicht, ob die Mailadresse gültig
            ist oder nicht!
            dummytext = ini_datum("dd.mm.yy") + " Error: " + Err.Description

      End If
      ini_mail_senden = dummytext
End function
...
```

Abbildung 30: Listing-Mail senden

Cookie für eindeutige Erkennung

Damit ein Kunde sein Certificate nicht einfach verteilen kann, wird ein Cookie auf die Client Maschine geschrieben! Mir ist bewusst, dass viele Personen und Unternehmungen die Cookies entweder nicht erlauben oder periodisch löschen. Dennoch wollte ich diese Variante im Prototyp aufzeigen.

Beim Installieren des Client Certificates wird auf dem Client ein Cookie gesetzt (Abb. 31).

```
<%
'*****************************************************************************
' ALLGEMEIN FUER GANZE SEITE GUELTIGKEIT
' Sprache:          VBScript plus JavaScript
' WebSeite:         certenroll\md_certificate_install.asp
' Author:           MediData AG, Bosshard Stefan
' Datum:            31.08.1998, V1.0
' Aenderungen:      --
' Beschreibung:     Diese Seite wird von md_certificate.asp aufgerufen und checkt
'                   diverse Sachen wie Schlüsselgrösse, Browsertyp etc. und startet
'                   dann kgaccept.asp.
' Bemerkung:        Der Code, der nicht beschrieben wird, ist eine Standardlösung
'                   von Microsoft, d.h. der Code wurde nicht von MediData AG ge-
'                   geschrieben.
'*****************************************************************************
%>
<%Response.Expires=0%>
<%
dummy = Request.Form("Org")
dummy = dummy & Request.Form("CommonName")
dummy = dummy & Request.Form("Email")
commonName = Request.Form("CommonName")
'Cookie schreiben auf Client Maschine, für eineindeutige Erkennung des Zertifikates
Response.Cookies("Eineindeutige_Erkennung") (commonName) = dummy
'Cookie benötigt ein "gültig bis" Datum, damit es überhaupt auf Client Maschine
'geschrieben wird. Ohne Datum wäre es nur temporär.
Response.Cookies("Eineindeutige_Erkennung").Expires = "31/12/2010"
%>
...
```

Abbildung 31: Listing-Cookie schreiben

Sobald der Kunde die private Webseite „privatweb\index.asp“ (Abb. 32) aufruft, verlangt der Web Server das zuständige Client Certificate. Gleichzeit sucht diese Internetseite das „gesetzte“ Cookie und vergleicht den Inhalt mit einem Teil des Certificates. Falls identisch wird Zutritt gewährt. Wird kein entsprechendes Cookie gefunden, wird zwar Zulass zur Seite gewährt, aber es erscheint eine Meldung, dass gewünschte Daten nicht ersichtlich, da Certificate nicht von dieser Maschine installiert wurden.

```
<--
'*****************************************************************************
' ALLGEMEIN FUER GANZE SEITE GUELTIGKEIT
' Script:        VBScript
' WebSeite:      privatweb\index.asp
' Author:        MediData AG, Bosshard Stefan
' Datum:         31.08.1998, V1.0
' Aenderungen:   Checkt Cookie und Zertifikat, falls beides übereinstimmt, wird
'                Zulass gewährt in öffentliches Web.
'*****************************************************************************
-->
<%
'Client-Zertifikat - Inhalte abfragen
dummyZertifikat = Request.ClientCertificate("SubjectO")
dummyZertifikat = dummyZertifikat & Request.ClientCertificate("SubjectCN")
dummyZertifikat = dummyZertifikat & Request.ClientCertificate("SubjectEmail")
%>
<%
commonName = Request.ClientCertificate("SubjectCN")
'Holt Inhalt aus dem Client Cookie. Bsp.  "..\netscape\user\%username%\cookies.txt"
'Eineindeutige_Erkennung   = Cookie Name
'commonName                = Cookie Subteil
dummyCookie = Request.Cookies("Eineindeutige_Erkennung")(commonName)
%>
<html>

<head>
<title>Privat - Index</title>
</head>

<body>
<%
'Cookie Inhalt mit Client-Zertifikat Inhalt vergleichen:
'Falls identisch, hat User Zugriff, andernfalls nicht, da warscheinlich Zertifikat
'weiterkopiert!
'ACHTUNG: Falls Unternehmung oder User Cookies deaktiviert hat, funktioniert dies
'nicht und der User wird nie Zugriff erhalten!!!
if dummyCookie = dummyZertifikat then %>
...
<% else %>
...
<% End if %>
</body>
</html>
```

Abbildung 32: Listing-Cookie lesen

Mapping

Um all die ausgestellten Client Certificates manuell zu den zwei konfigurierten Windwos NT User Accounts zu mappen, benutzte ich das Object IIsCertMapper von ADSI. In Abb. 33 wird ein Mapping erstellt.

```
...
<%
'*************************************************************************
' Beschreibung:
' Ueberprüfung plus Mapping des Certificates
'
' IIsCertMapper ist ein Objekt der ADSI (Active Directory Services Interface)
' und musste speziell auf Server installiert werden!
' CertObj.CreateMapping Zertifikat, AccountName, PasswortAccount, EnableMapping
'*************************************************************************
'Zertifikat in Variable vCert zwischenspeichern
vCert = Request.ClientCertificate("Certificate")
'vorgewiesenes Zertifikat Organisation plus CommonName abfragen
certOrg = Request.ClientCertificate("SubjectO")
certName = Request.ClientCertificate("SubjectCN")

'Wenn DB Eintrag mit vorgewiesenem Zertifikat übereinstimmt, dann Mapping ausführen
if certOrg = dbOrg AND certName = dbName then
    'IisCertMapper verlangt hier einen Standardpfad für Webserver
    'Wie kann aber nun auf diverse versch. Webs gezeigt werden?
    'MappingObjectZuteilung:
    '1 = Default Web Site
    '2 = Administrations Web Site
    '3 = ?
    objectString = "IIS://localhost/W3SVC/" +  MappingObjectZuteilung
     objectString = objectString + "/IIsCertMapper"
    Set CertObj = GetObject(objectString)
    'Mit diesem Befehl wird das Mapping auf entsprechenden NT Account durchgeführt
    CertObj.CreateMapping vCert, accountNameMapping,
        - ini_account_passwort(mid(Request("cert"),1,2)), beschriftungMapping, True
    'Mapping Eintrag in DB, dass per heute ok

    temp = "UPDATE md_Certificate_Anfrage SET Status = '12' , Mapping_erfolgt_am =
      - '" + ini_datum("dd/mm/yy") + "' WHERE RequestID = '" + Request("cert") + "'"
    call sub_sqlExecute(temp)
    response.write "Test war erfolgreich <a href='http://srvwisweb'><font
face='Arial'>Home Seite</font></a>"
else
    response.write "Test failed. Bitte nehmen Sie Kontakt mit MediData AG auf! <a
href='http://srvwisweb'><font face='Arial'>Home Seite</font></a>"
end if
%>
...
```

Abbildung 33: Listing-Mapping erstellen

Abb. 34 zeigt den Sourcecode, wie ein bestehendes Mapping gelöscht werden kann.

```
...
'*****************************************************************************
' Sub sub_mappingLoeschen(string mappName)
' Beschreibung:     Löscht Certificate NT Account Mapping auf Server
' Input:            mappName = NT Mapping Name
'                   MappingObjectZuteilung    = Welches Web es betrifft, Bsp. 1 =
'                   Default Web
' Output:           --
' Aktivität:        Löscht das Mapping
'*****************************************************************************
Sub sub_mappingLoeschen(mappName, MappingObjectZuteilung)
      Dim CertObj
      'IIsCertMapper verlangt hier einen Standardpfad für Webserver
      'Wie kann aber nun auf diverse versch. Webs gezeigt werden?
      'MappingObjectZuteilung:
      '1 = Default Web Site
      '2 = Administrations Web Site
      '3 = ?
      objectString = "IIS://localhost/W3SVC/" + MappingObjectZuteilung
       objectString = objectString +  "/IIsCertMapper"
      Set CertObj = GetObject(objectString)
      'Search by MappingName
      CertObj.DeleteMapping 2, mappName
End Sub
...
```

Abbildung 34: Listing-Mapping löschen

Hinweis: Passwort für Mapping muss mit dem erfassten NT User Account übereinstimmen. Ausserdem muss die Seite, die das Mapping ausführt von einem „Administrator“ Account ausgeführt werden.

9.6 Certificate Request Ablauf

9.6.1 Sicht User

Certificate Aussteller MediData AG akzeptieren

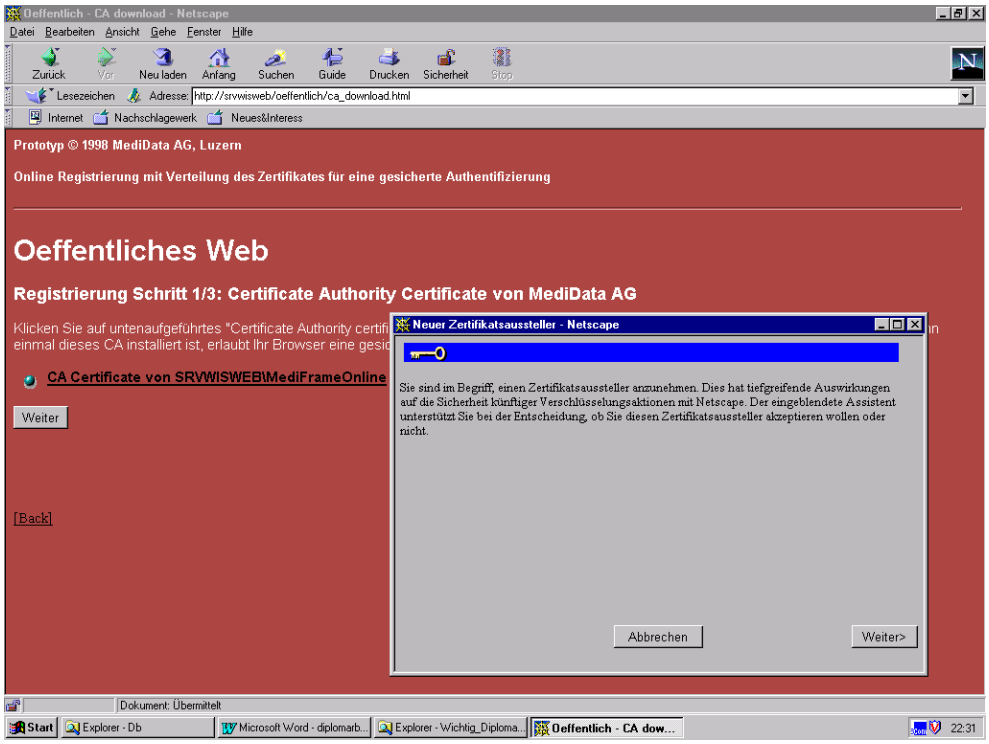

Abbildung 35: Annahme Certificate Aussteller MediData AG

Zu allererst muss der User der MediData AG das Vertrauen schenken, um eine gesicherte Verbindung zuzulassen (SSL Verbindung). Dies erreicht er durch das Herunterladen und automatische Installieren des CA Certificates (Abb. 35 - 38) der MediData AG. D.h., ab diesem Zeitpunkt vertraut er dem Certificate Aussteller MediData AG und somit auch allen von uns ausgestellten Client Certificates.

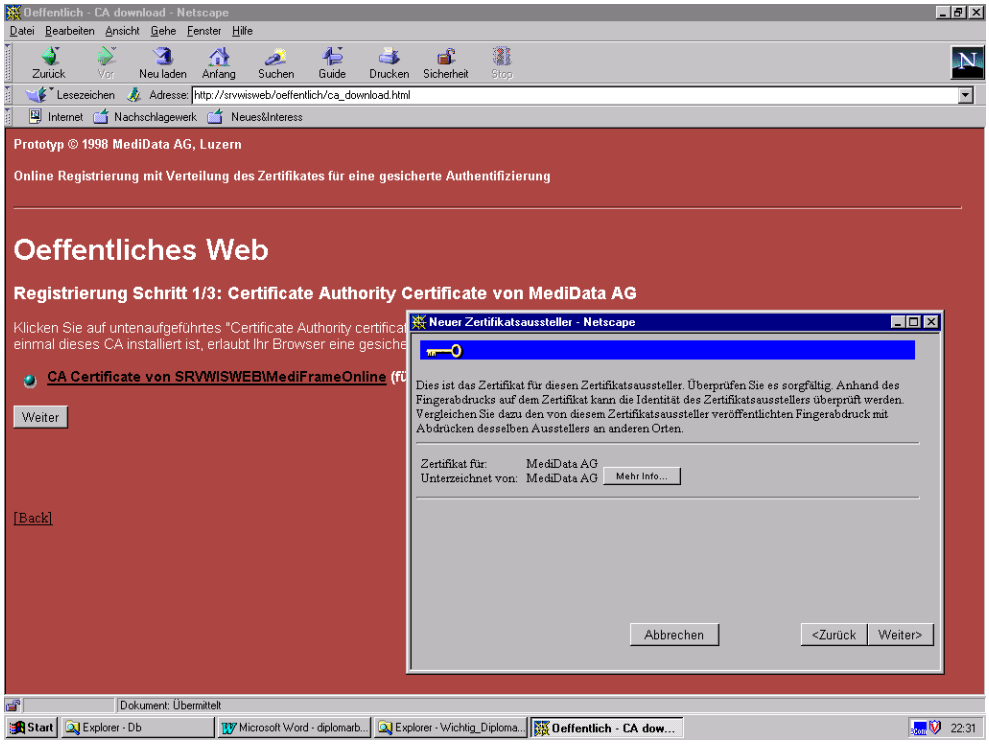

Abbildung 36: Certificate von Certificate Aussteller anzeigen

Der User hat hier die Möglichkeit das CA Certificate des Zertifikatsausstellers MediData AG anzusehen.

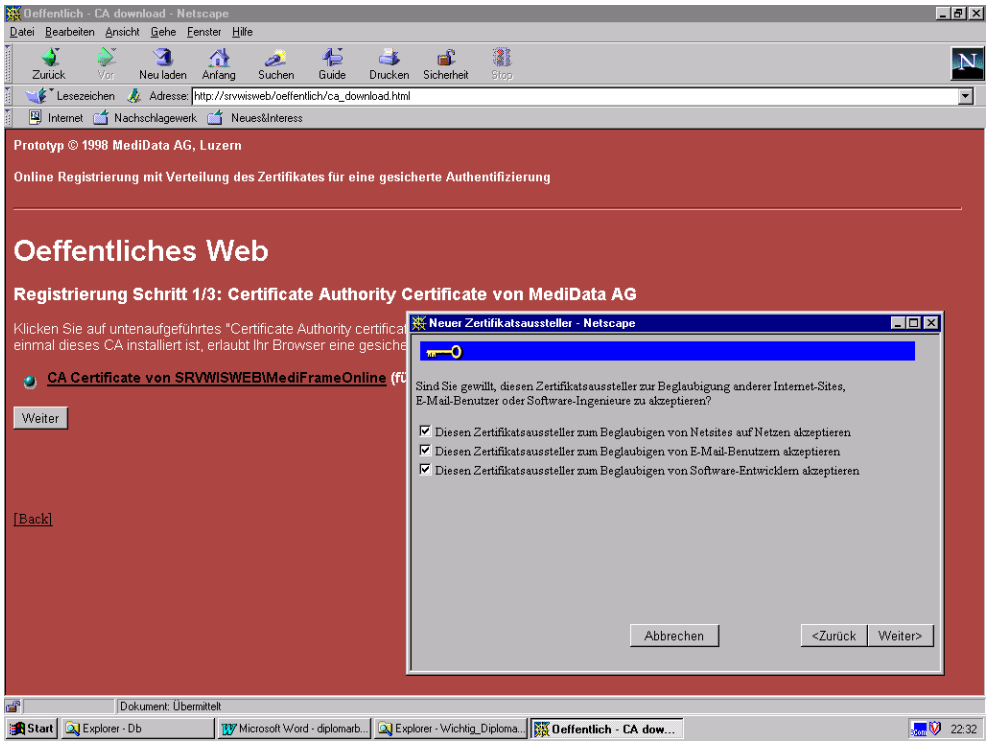

Abbildung 37: Auswahl der Beglaubigung des Certificate Ausstellers

Dieses CA Certificate kann nun für Internetseiten, Email oder Software akzeptiert werden (Abb. 37).

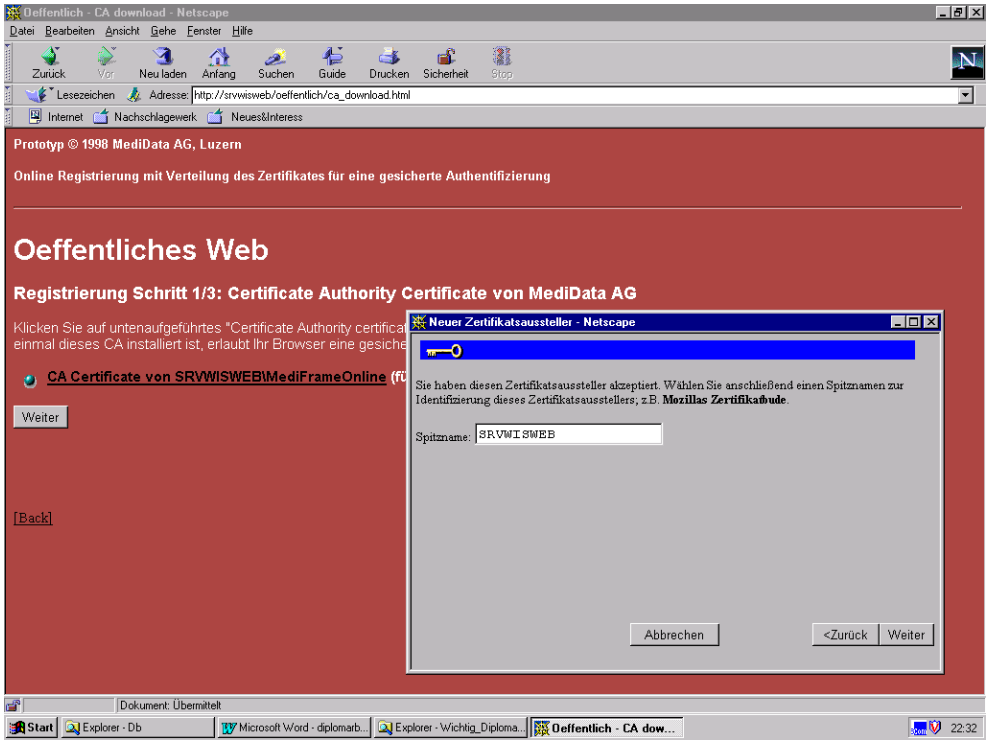

Abbildung 38: Beschreibung des CA's

Sobald in Abb. 38 der [Weiter] Button gedrückt wird, ist die MediData AG als Certificate Aussteller beglaubigt und der User vertraut dieser.

Anmeldung für Certificate bei MediData AG

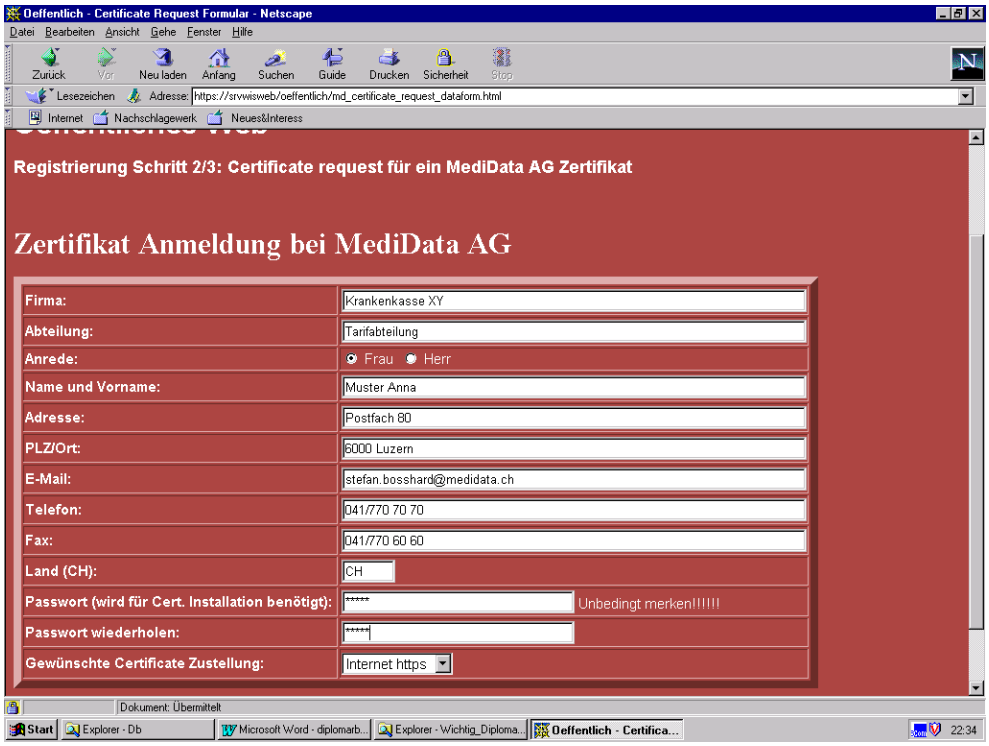

Abbildung 39: Registrierungsformular

Der Kunde füllt Formular gem. Abb. 39 aus und sendet die Daten an den Web Server. Das erfasste Passwort wird später für die Abholung des Client Certificates wiederverwendet.

MediData AG
MediFrame
Rösslimattstr. 39
6002 Luzern

6000 Luzern, 04.09.98

Bestätigung Online Registrierung für Client Certificate

Ich bestätige, ein Client Certificate bei der MediData AG angefordert zu haben und bezeuge die Richtigkeit der unten aufgeführten Angaben:

Firma	Krankenkasse XY
Abteilung	Tarifabteilung
Name und Vorname	Muster Anna
Email	stefan.bosshard@medidata.ch
Strasse	Postfach
Phone	041/770 70 70
PLZ/Ort	6000 Luzern
Fax	041/770 60 60
Certificate angefordert am	04.09.98
Certifcate Zustellungswunsch	Internet https

Ich verpflichte mich, dieses Client Certificate vor unbefugtem Zugriff zu schützen und nicht an Drittpersonen weiterzugeben. Zusammen mit diesem Schreiben benötigen wir eine Kopie Ihrer Identitäskarte ID oder Ihres Passes. Denn nur so können wir eine gesicherte Authentifizierung gewährleisten.

Bei selbstpraktizierenden Aerzten, benötigen wir eine Kopie der Arztzulassung.

Selbstverständlich werden diese Dokumente bzw. Daten geschützt aufbewahrt und nicht an Drittpersonen weitergegeben.

Datum/Unterschrift: Stempel:

_____

Allgemein: Ohne Einreichung dieser Bestätigung und der oben erwähnten Dokumente können wir Ihnen leider kein Client Certificate zustellen und sind somit auch nicht zu unserem gewünschten Produkt zugelassen.

Wir bitten Sie, dieses Schreiben unterzeichnet auf dem Postweg oder per Fax 041/368 21 39 zu retournieren. Bei Fragen oder Unklarheiten, bitten wir Sie, sich mit uns in Verbindung zu setzen Tel. 041/368 21 27.

Besten Dank für Ihr Vertrauen.

Abbildung 40: Ausdruck Bestätigungsformular

Es erscheint nun eine HTML Seite (Abb. 40) mit den eben erfassten Kundendaten. Der Kunde wird aufgefordert, diese Bestätigung auszudrucken und unterschrieben an die MediData AG weiterzuleiten.

Abholung Client Certificate

Administrator erhält Bestätigung und löst aufgrund dieses Nachweises die Freigabe aus. Das Administrationstool sendet automatisch ein Mail an den Kunden (Abb. 41) mit den nötigen Informationen, wo das Certficate abgeholt werden.

Date sent: Fri, 4 Sep 1998 22:47:25 +0200 (MET DST)
From: "MediData AG, webmaster" <webmaster@medidata.ch>
To: "Muster Anna" <stefan.bosshard@medidata.ch>
Subject: Client Certificate MediFrame

Sehr geeehrte Frau Muster Anna
Herzlichen Dank fuer Ihre Anfrage eines Client-Certificates
vom 04.09.98 fuer die Benuetzung unseres MediFrameOnline Produktes.

1. Schritt - Das Zertifikat koennen Sie ab sofort mit folgender URL:
https://srvwisweb/certsrv/certenroll/md_certificate.asp?cert=01_Mu704Kr
uebers Internet beziehen. Das Zertifikat wird automatisch in Ihren
Browser installiert (Navigator oder Internet Explorer 3.0 oder aktueller.

2. Schritt Testen des Certificates:
https://srvwisweb/certsrv/certenroll/md_test.asp?cert=01_Mu704Kr

Zur Information: Das Zertifikat kann nur einmal abgeholt werden!
Ausserdem kann das Zertifikat nur auf dem PC benutzt werden,
auf welchem es auch online installiert wird!

Bei Fragen oder Unklarheiten stehen wir Ihnen gerne zur Verfuegung

Mit freundlichen Gruessen
MediData AG
Webmaster

Abbildung 41: Mailzustellung mit URL für Certificate Abholung

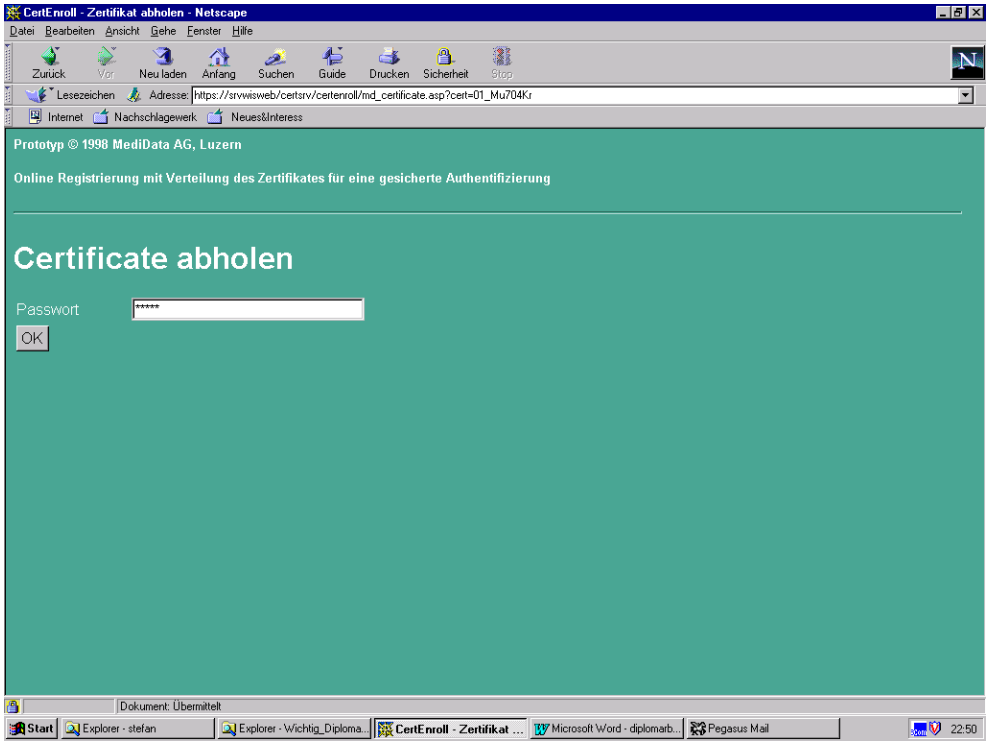

Abbildung 42: Certificate Abholung

Beim Oeffnen dieser URL Adresse (Abb. 42) wird User aufgefordert, das bei der Anmeldung erfasste Passwort zu erfassen. Web Server kontrolliert ob User ID und Passwort übereinstimmen.

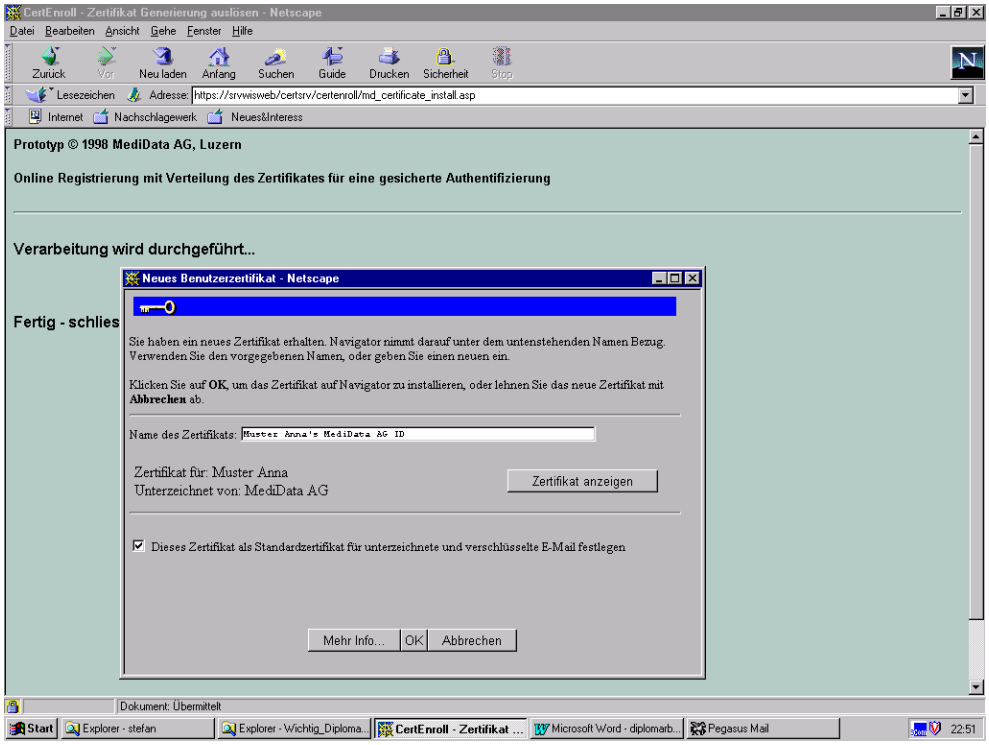

Abbildung 43: Certificate Installation

Das Client Certificate wird auf dem Server generiert und dem Kunden zugestellt. Es erscheint automatisch Dialog gem. Abb. 43. Der Kunde hat die Möglichkeit, das Certificate zuerst anzuschauen um es eventuell zu verwerfen.

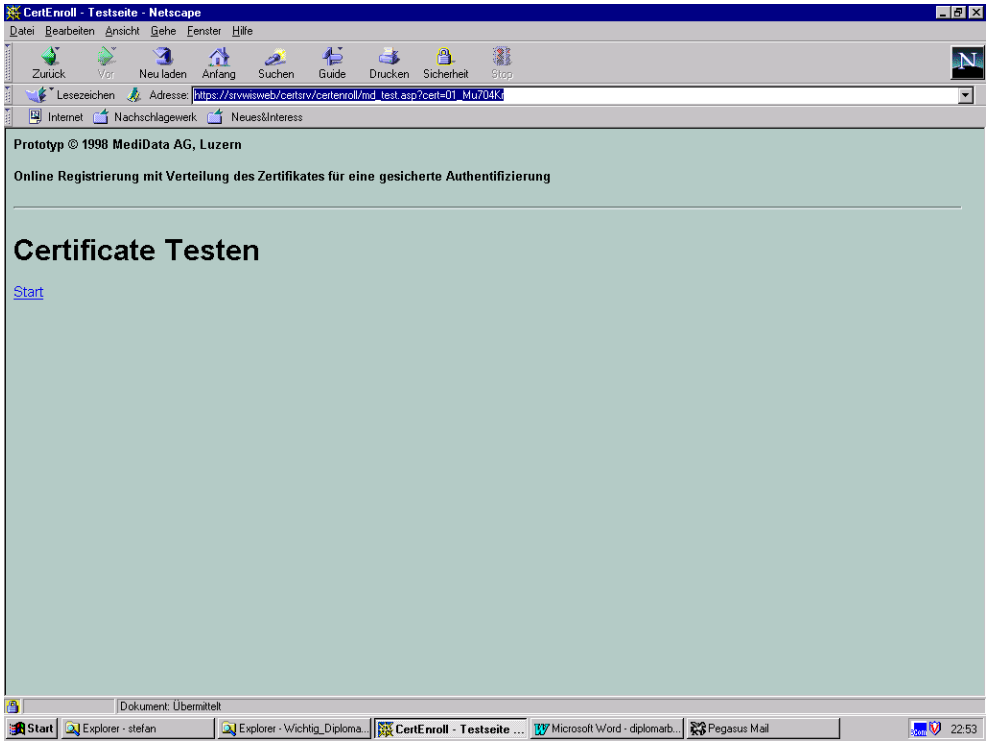

Abbildung 44: Test des neu erhaltenen Certificates

User wird im Mail aufgefordert, mittels einer zweiten URL Eingabe (Abb. 44), das Certificate zu testen. Sobald er diesen Test startet, wird mitgeschicktes Certificate zum entsprechenden NT Benutzer Account (auf Web Server) gemappt. Ab diesem Zeitpunkt hat User Zugriff auf privates Web.

9.6.2 Sicht Administrator

Das Administrationstool bietet einige verschiedene Listenauszüge an, die auf der Startseite auswählbar sind (Abb. 45).

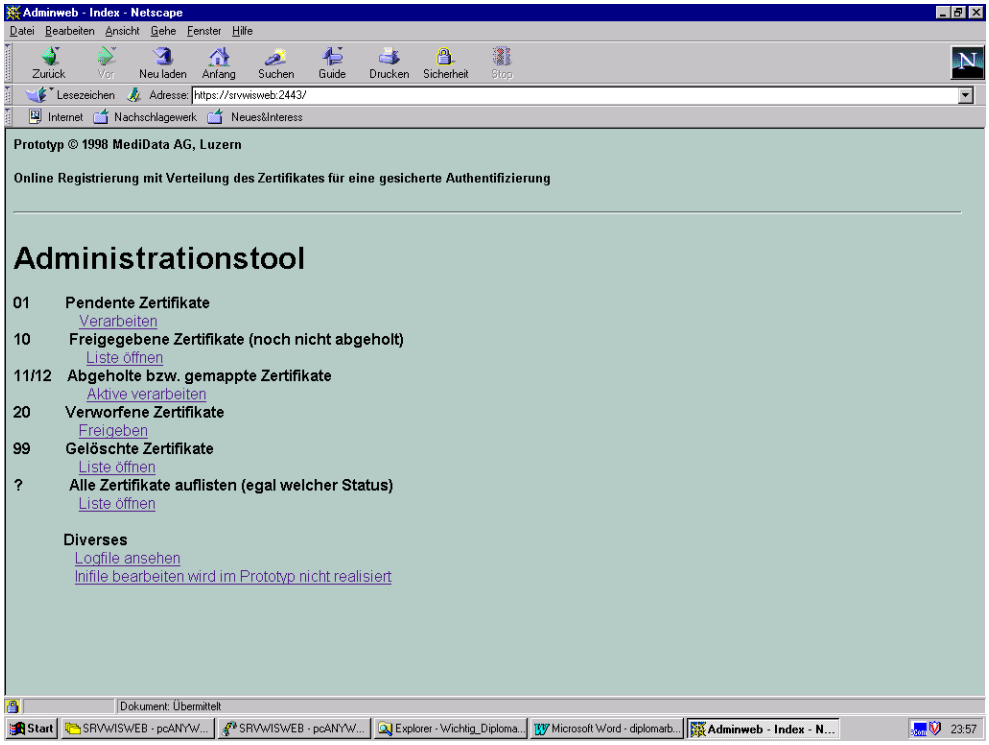

Abbildung 45: Index Administrationstool

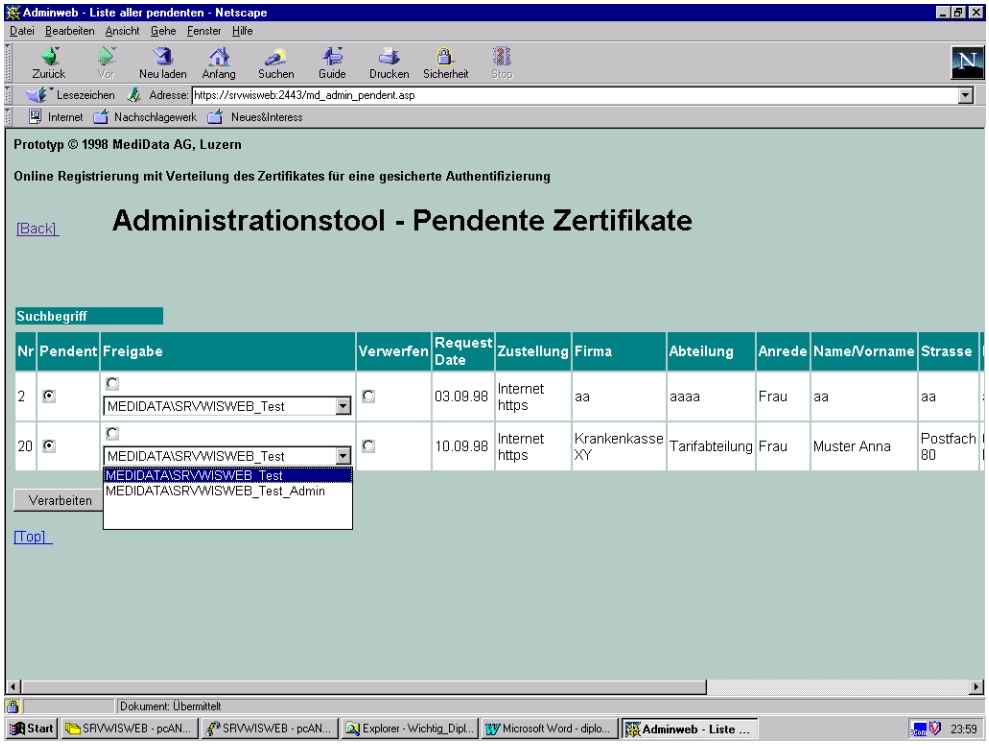

Abbildung 46: Freigabe von Client Certificates

Der Administrator öffnet die „Pendente Zertifikate“ Liste (Abb. 46), wenn er einzelne bzw. mehrere freigeben oder verwerfen will. Er hat hier die Möglichkeit, einen entsprechenden NT User Account dem Kunden zuzuweisen.

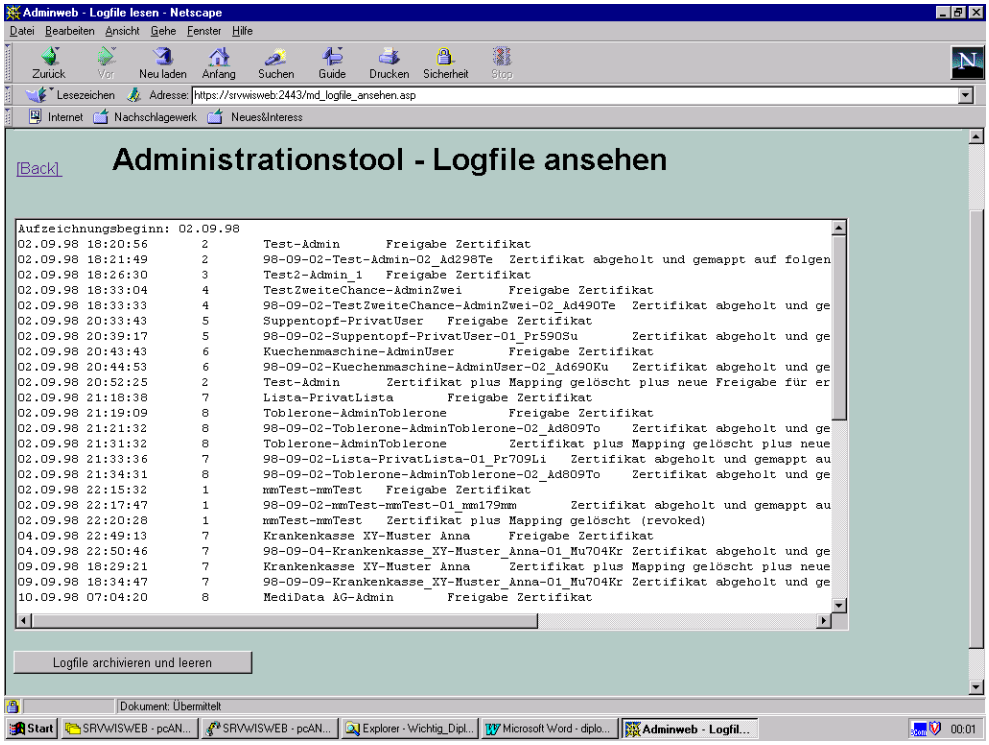

Abbildung 47: Logfile

In einem Logfile werden alle Vorgänge rapportiert (Abb. 47).

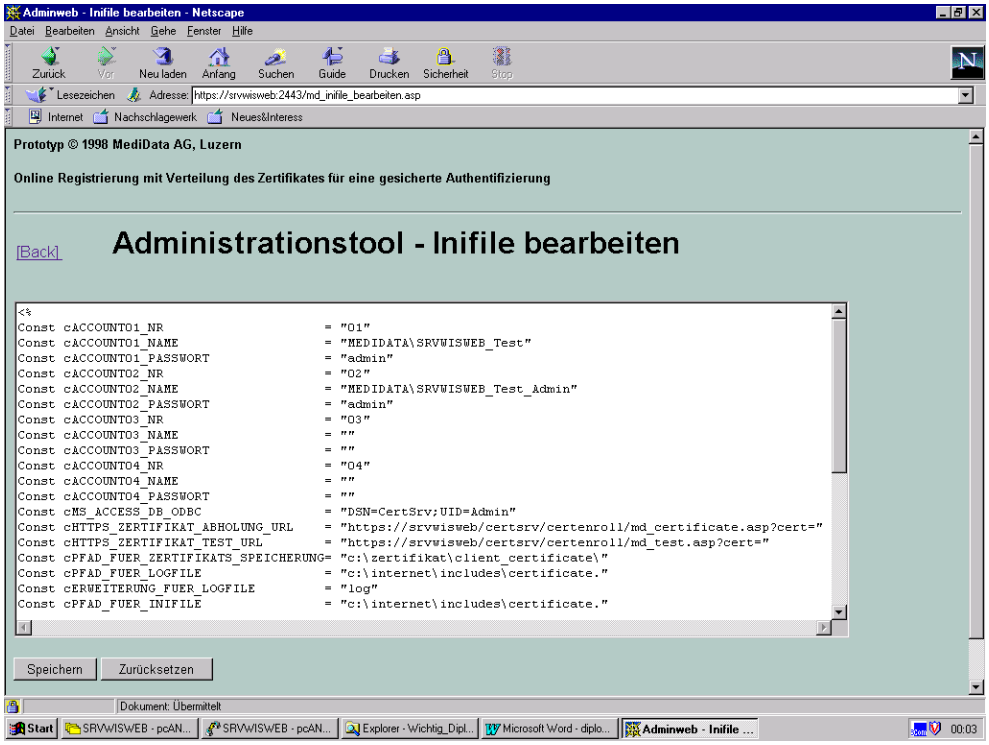

Abbildung 48: Inifile

Der Administrator hat ebenfalls die Möglichkeit, online gewisse Daten im Inifile zu mutieren (Abb. 48).

10 Ausblick

10.1 MediData AG

Nach der Auseinandersetzung mit dem ganzen Verfahren sehe ich keine Hindernisse, den Prototyp auszubauen und in den MediFrame Online Dienst zu integrieren und somit die jetztige Passwortlösung abzulösen. Zudem müsste ein zusätzlicher Punkt realisiert werden, nämlich der direkte Zugriff vom Administrationstool auf unsere Kundenverwaltungsdatenbank ACT! 3.0. Damit könnte ein wichtiger Zwischenritt – das manuelle Erfassen – wegfallen, da die Daten mittels Mausklick von der Certificate Datenbank in die Kundendatenbank kopiert werden könnten.

Zum einen ist es für die MediData AG eine Vereinfachung der Administration und zum anderen ist es ein Aushängeschild für die Kundschaft und die Geschäftspartner, dass wir gewissermassen „Up to date“ in der Technik sind. Zum zweiten Punkt möchte ich noch hinzufügen, dass es marketingmässig sicherlich interessant wäre, das ganze Know How betreffend dieser Arbeit im Internet zu veröffentlichen und zu präsentieren!

10.1.1 Mögliche Certificate Aussteller

Mögliche Anbieter von CA und Client-Certificates für die MediData AG sind in der Tab. 25 aufgelistet.

Tabelle 25: Certificate Aussteller (Stand: Mitte August 1998)

Unternehmung	Weitere Infos unter URL	Preis Server Certificate für Web Server	Preis Client Certificate (für Identitätssicherheit, hohe Sicherheit, d.h. keine Email Certificates etc.)
Verisign (USA)	http://www.versign.com	Mind. $ 349.00	$ 19.95/Jahr (USA und Canada)
Thawte (USA)	http://www.thawte.com	$ 125.00 Erstanforderung $ 100.00 Erneuerung	$ 100.00 Erstanforderung $ 50.00 Erneuerung
Swisskey (CH)	http://www.swisskey.ch (ab Oktober 1998 produktiv) [33]	CHF 650.00/Jahr ohne MWSt	CHF 35.00/Jahr ohne MWSt CHF 150.00/Jahr für Client-Certificate für Firmen
BelSign (BE)	http://www.belsign.be	$ 189.00/Jahr	$ 65.00/Jahr
Weitere CAs: GTE (USA), Entrust (USA), Interauth (Japan), COST (Schweden) und EuroSign (UK) Hinweis: Bei Thawte, BelSign und Swisskey können Test Client-Certificates angefordert werden.			

Momentan ist in der Schweiz nur gerade die Swisskey vertreten, deshalb möchte ich nachfolgend auch näher auf dieses Unternehmen eingehen und erklären:

[33] Literatur [SWI]

Swisskey

Swisskey ist ein Gemeinschaftsunternehmen der Firmen Swisscom, Telekurs und DigiSigna. EUROPAY vertritt in diesem Joint Venture die Interessen der Telekurs. Swisskey wird ab Oktober 1998 in der Schweiz die Rolle einer Zertifizierungsstelle für elektronische Ausweise (Certificates) übernehmen. Es werden sowohl X.509-Certificates für SSL oder S/MIME-Mails als auch EDIFACT-Certificates angeboten.

Um jederzeit eine hohe Qualität bei den Zertifizierungsrichtlinien zu gewährleisten, werden sie durch DigiSigna (Verein der Handelskammern der Schweiz und des Fürstentum Liechtenstein) erstellt und überwacht. DigiSigna ist in Zusammenarbeit mit Swisskey auch für die Akzeptanz der Swisskey-Certificates im Ausland (Cross-Zertifizierung) zuständig, damit die Swisskey-Certificates auch International anerkannt werden.

In Tab. 26 finden Sie einige wichtige Fragen und Antworten zu diesem ersten Schweizerischen Certificate Aussteller.

Tabelle 26: Einige wichtige Hinweise zu Swisskey [34]

FAQs	Antwort
Wo sind die Registrierungsstellen?	Handelskammern der einzelnen Kantone. Bis Ende Jahr wird das Registrierstellennetz erweitert werden mit Bankfilialen und Poststellen.
Welche Ausweispapiere sind anerkannt?	Die Swisskey-Registrierstellen anerkennen als amtliches Ausweisdokument einen Pass, eine Schweizer Identitätskarte, einen Schweizer Fahrausweis und bei Firmen den Handelsregisterauszug (nicht älter als einen Monat).
Wieviel kostet ein Certificate pro Jahr?	Swisskey Personal ID: CHF 35.00 Swisskey Corporate ID: CHF 150.00 Swisskey Server ID: CHF 650.00
Wie lange sind diese Certificates gültig?	2 Jahre
Welche Schlüssellängen werden von Swisskey AG unterstützt?	Zur Zeit unterstützt Swisskey AG die Zertifizierung von 512-bit und 1024-bit Schlüsseln. Die aktuellen Exportversionen der Browser unterstützen nur die Generierung von 512-bit Schlüsseln. Swisskey AG wird eine Anwendung zur Verfügung stellen, um auch lange d.h. 1024-bit Schlüssel erzeugen zu können. Diese können in die aktuelle Version des Netscape Communicator auch in der Exportversion importiert werden.
Zusatzdienste	Swisskey sorgt dafür, dass alle erstellten Certificates in mindestens einem Verzeichnis jederzeit verfügbar und abrufbar sind. Weitere Dienste werden eine Sperr-Hotline und die Verteilung von Sperrlisten an Abonnenten sein.

[34] Literatur [SWI]

10.1.2 Certificate Authority im Gesundheitswesen

Erstrebenswert wäre sicherlich, dass die MediData AG im Bereich Gesundheitswesen der Schweiz **der** Certificate Aussteller wird und dies gleich von Beginn weg, bevor jede einzelne Krankenkasse und Versicherung seinen eigenen Certificate Server eingerichtet hat! In Abbildung 49 ist eine mögliche Certificate Hierarchie abgebildet.

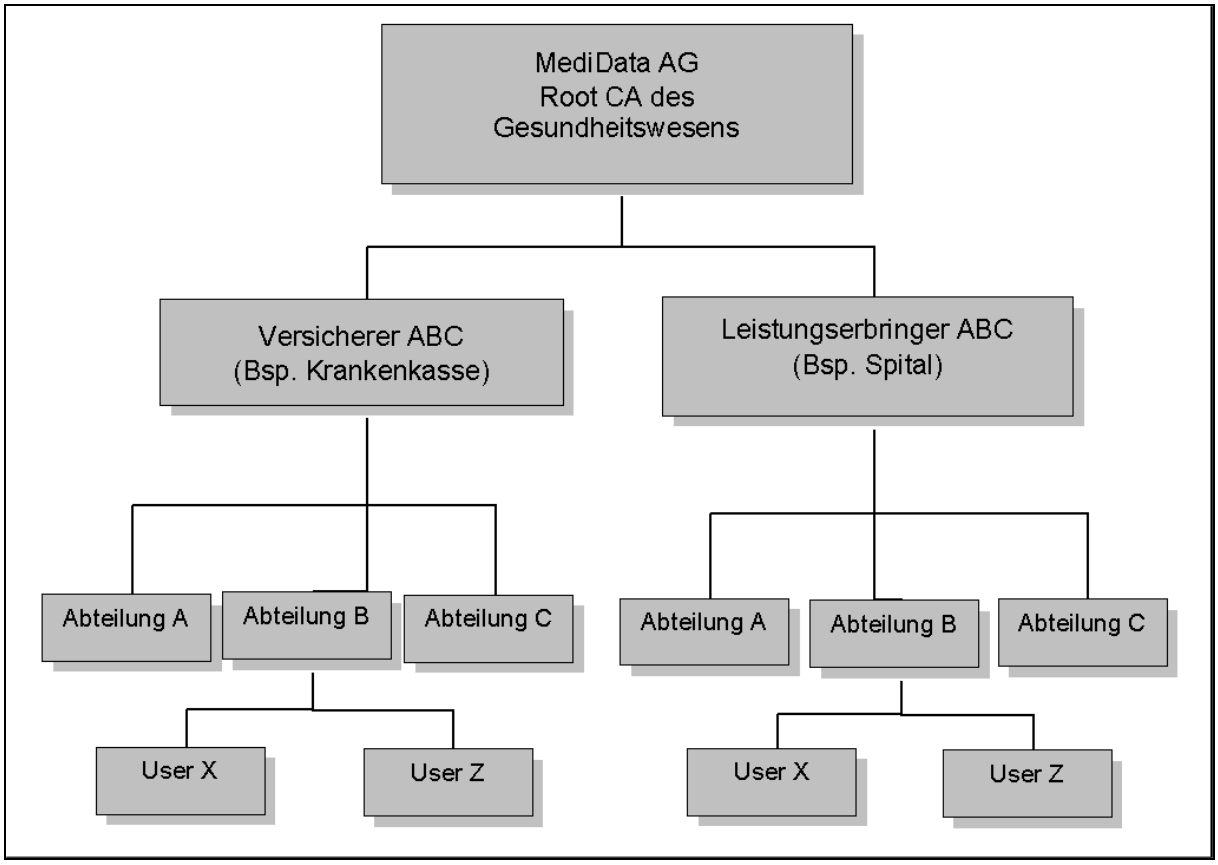

Abbildung 49: Vorstellbare CA Hierarchie Struktur im Gesundheitswesen

10.2 Technologie

Für mich ist es ganz klar, das dieses Verfahren in nächster Zeit boomen wird, weil der Drang nach Sicherheit und privaten Netzen enorm hoch ist. Sei es für den Gebrauch von Internet Banking, Bestellungen, Usenets und vieles mehr. Oder gar neuartigen Anwendungen wie

- Mobile Telefonie
- Zeitstempel-Dienste
- „Ein-Mal-Certificates“
- Elektronische Abstimmungen

Ein weiterer ausgezeichneter Aspekt ist, dass es heutzutage soviele verschiedene Passworte gibt, die man sich merken muss. Warum soll man es nicht einfacher haben? Man kauft sich ein günstiges Certificate und benutzt dieses für alle Zugriffe auf geschlossene Web Bereiche etc.?

Doch sehe ich auch hier ein kleiner Nachteil. Wie bei den Kreditkarten, in der „Anfangszeit“ wird jeder User Dutzende von verschiedenen Certificates verwalten müssen, anstelle von nur einem einzigen, welches von allen Dienstleistungen akzeptiert wird!

10.2.1 Datenträger für Client-Certificates

Als Speicherträger für ein Certificate kommt sicherlich eine Chipkarte (engl.: chip card; IC card; smart card) in Frage. Dies ist eine Plastikkarte in der Standardgrösse (85,6 x 54 x 0,76 mm) und mit den gleichen physikalischen Eigenschaften wie eine Magnetstreifenkarte. Ein in die Karte implantierter Chip enthält einen Mikroprozessor und Speicher (ROM, RAM und EEPROM) [35].

Diese Smart Card wird wohl in Zukunft die Verteilung der Client-Certificate übernehmen. Was heutzutage leider noch standardmässig fehlt, ist das Lesegerät, das die Smard Card einliest. D.h., der Vorgang wird folgendermassen aussehen: Wenn ich mich am Computer anmelde, stecke ich die Karte ein und tippe zur Sicherheit noch ein Passwort ein (wie beim Bankomaten). Das Passwort ist selbstverständlich nur auf der Karte selber abgespeichert, d.h., die Karte „kontrolliert“ sich selbst und startet keine Anfrage übers Netz, ob das Passwort gültig ist oder nicht. Nun, dies alleine bringt noch keine Vorteile. Wenn ich nun aber im Internet einen geschlossenen Dienst nutzen möchte, stecke ich zur gegebenen Zeit die Karte wieder ein und gebe dem Lesegerät die Möglichkeit, mein Client-Certificate einzulesen und an den Dienst weiterzuleiten. Funktioniert also wie ein physischer Schlüssel.

[35] Literatur [HAN], Seite 651

10.2.2 Transport Layer Security (TLS)

Das neue Transport Layer Security (TLS) Protokoll ist nicht radikal anders als die bekannte SSL Technologie und wird zu SSL kompatibel sein. Vorteil ist, dass es einfacher für die Handhabung sein wird und es mehr zusätzliche Authentication Optionen aufweisen wird.

Die neue TLS Spezifikation, welche auf SSL aufbaut, wurde von der Task Force bereits genehmigt und wird demnächst in den verschiedensten Komponenten implementiert sein.

Zusätze gegenüber SSL sind ein erweitertes Certificate Management, verbesserte Authentication und zusätzliches Error Handling.

Details über diese Technologie möchte ich in dieser Arbeit nicht weiter behandeln [36].

10.2.3 Verschlüsselung

Geheimcodes in Gefahr?

Die derzeit gängigsten Geheimcodes zur sicheren Datenübermittlung im Internet sind im Prinzip ohne Probleme zu knacken. Für diese bittere Erkenntnis ist der US-Mathematiker Peter Shor auf dem Internationalen Mathematiker-Kongress in Berlin mit dem arrivierten Nevanlinna-Preis ausgezeichnet worden. Beruhigend für Versicherungen, Banken und Internet-Surfer: Einen Rechner, auf dem Shors preisgekrönter Algorithmus mit genügender Effizienz laufen könnte, gibt es heute noch nicht.

Das Prinzip vieler derzeitiger Verschlüsselungssyteme, der sogenannten Verschlüsselungs-Verfahren: Ein herkömmlicher Rechner hat erhebliche Schwierigkeiten, eine grosse Zahl in ihrer Primfaktoren, ihre verschiedenen Teiler, aufzusplitten. Die Folge: Je mehr Stellen eine als Geheimschlüssel dienende Zahl aufweist, desto mühsamer wird es für den Rechner eines Computerhackers, den Schlüssel zu decodieren. Genau an dieser Stelle könnte Shors genialer Algorithmus seine Stärken ausspielen. Er würde eine ausserordentlich schnelle Faktorisierung von Zahlen ermöglichen und wäre somit die Grundlage für einen perfekten Code-Knacker Maschine.

Der Algorithmus kennt die Faktorisierung deswegen in ungeahnter Geschwindigkeit abwickeln, da er das Problem parallel angehen würde – unter Umständen in vielen Millionen Zweigen gleichzeitig. Aber: Dazu wäre ein völlig neuer Rechnertypus notwendig, ein sogenannter Quantencomputer. Seine Grundbauteile bildeten Beispielsweise einzelene Atome, eingefangen in speziellen Magnetkäfigen (Ionenfallen) und angesteuert von hochpräzisen Lasern.

Bislang existiert von diesen High-Tech-Rechner nur eine Handvoll primitiver Prototypen. Viele Fachleute schätzen, dass mindestens zwei Jahrzente vergehen werden, bevor ein Quantenrechner zuverlässig funktioniert. „Praktisch besteht für die nächsten Jahre keine Gefahr“, resümiert Kryptographie-Experte Albrecht Beutelspacher von der Universität Giessen. „Aber sollten sich Quantencomputer eines Tages tatsächlich realisieren lassen, dann wäre die ganze heutige Kryptographie neu zu überdenken [37].

[36] Literatur [TLS]
[37] Literatur [TAG]

Ein Schweizer Duo hält Hacker vom Web fern

„Absolute Sicherheit bietet heute kein Kryptosystem“, so Philippe Janson, Leiter des Departements IT-Solutions am IBM Forschungslabor in Rüschlikon. Kopfzerbrechen bereiten Sicherheitsexperten vor allem sogenannte aktive Hackerangriffe. Den Beweis dafür erbrachte zuletzt Ende Juni Daniel Bleichenbacher. Dem Mitarbeiter der Bell Labs, der sein Handwerk an der ETH Zürich gelernt hat, war es damals gelungen, das WebSicherheitsverfahren Secure Sockets Layer (SSL) mit dieser Methode zu überlisten. Er deckte dabei einen mit SSL gesicherten Server millionenfach mit Scheintexten ein. Aus den Informationen in den Rückmeldungen konnte er den Schlüssel rekonstruieren und das Sicherheitssystem knacken.

Ronald Cramer vom Institut für theoretische Informatik der ETH Zürich und Victor Shoup vom IBM Forschungslabor haben nun eine Verschlüsselungsmethode entwickelt, die Kryptosysteme gegen aktive Attacken immun macht, womit die seit Jahren bekannte Lücke wohl geschlossen sein dürfte. Wie dies im Detail funktioniert, darüber hüllt man sich in Rüschlikon in Schweigen.

Die Wirksamkeit des Verfahrens lasse sich – ganz im Gegensatz zum Softwareflicken, mit dem RSA dem Loch zu Leibe rückt – mathematisch beweisen. Das dem tatsächlich so ist, bestätigt auch der erfolgreiche Hacker Bleichenbacher: „Das Ganze funktioniert und kann bewiesen werden“, lässt er sich im „Wall Street Journal“ zitieren.

Das Cramer-Shoup-Verfahren beruht auf einem mathematischen Beweis, den Victor Shoup bereits vor mehr als einem Jahr niedergeschrieben hat. Dessen tiefere Bedeutung hat der IBM-Forscher allerdings erst vor kurzem und nach längeren Diskussionen mit dem ETH-Mann Cramer erkannt. „Ich ging aus, trank ein paar Flaschen Bier – und da wurde es mir schlagartig klar“, erklärt er.

IBM-Forscher versuchen der Lücke schon seit Anfang der neunziger Jahre beizukommen. Die Lösungsansätze scheiterten allerdings samt und sind am zu hohen Berechnungsaufwand gescheitert. Die Cramer-Shoup-Methode beansprucht nur noch doppelt bis dreimal soviel Rechenleistung wie des heutigen Verfahren, bietet im Gegenzug aber mehr Sicherheit.

IBM beabsichtigt, das Kryptosystem in ihre Sicherheitslösungen zu integrieren. Als erstes soll die Zertifizierungs-Infrastruktur Vault Registry damit ausgerüstet werden.

Mit dem Stopfen von Sicherheitslücken beschäftigt sich seit längerem auch das U.S. National Institute of Standards and Technology (Nist). Nachdem Meldungen, die mit 56-Bit-DES-Schlüssel (Data Encryption Standard) chiffriert waren, kürzlich erfolgreich entziffert wurden, besteht Handlungsbedarf. Das Nist hat bereits vorgespurt und die Branche vor Monaten aufgefordert, Vorschläge für einen Nachfolge-Algorithmus einzureichen. 15 Entwürfe liegen inzwischen vor. Das Standardisierungsgremium wird die Eingaben in den kommenden Monate zusammen mit Experten begutachten. Das Ergebnis soll im Jahr 2001 bekanntgegeben werden. Der künftige Standard – Advanced Encryption Standard (AES) genannt – soll für 30 Jahre Bestand haben und über drei Schlüssellängen (128-, 192- und 256-Bit) verfügen [38].

[38] Literatur [COM]

10.2.4 Ein Certificate pro Person

Ein grosser Vorteil wäre, als Person nur ein einziges Certificate (Smart Card) erwerben zu müssen, nutzbar für Geschäftszwecke oder als Privatperson, egal wo ich mich gerade aufhalte. D.h., ich muss auf diese Karte wie auf meine Kredit Karte acht geben und bei Verlust sofort Meldung an die zuständige CA machen, dass das Certificate gesperrt (revoked) werden soll! Eine Aenderung gäbe es somit auch im Geschäftsleben. Anstelle dass ich vom Systemadministrator ein Passwort erhalte, melde ich mich bei Stellenantritt bei der betreffenden Stelle und lasse mein persönliches Client Zertfikat auf ein User Account mappen... Doch dieser Ansatz wird wohl eine Vision bleiben.

10.2.5 Digitale Unterschrift

Vielleicht nimmt die Papierflut doch mal ein Ende?

Viel Korrespondenz wird heute mit dem Personalcomputer geschrieben, ausgedruckt und unterzeichnet und verschickt, obschon der Empfänger eine Email Adresse hätte. Ist ja klar, wie sollte mir der Empfänger auch glauben, dass **ich** die Bestellung geschrieben habe. Nun, mit der digitalen Unterschrift sollte der Durchbruch nun endlich erreicht werden!

Werden digitale Unterschriften in der Schweiz rechtlich anerkannt? Zur Zeit gibt es in der Schweiz noch keine Regelung. Das BAKOM sammelt im Auftrag des Bundesrates und verschiedener Departemente Meinungen bez. einer zukünftigen Regelung. Zur Zeit wird eher mit der Selbstregulierung des Marktes als mit staatlichen Regelungen gerechnet. Länder die schon rechtliche Regelungen für digitale Unterschriften haben sind Deutschland und Italien. Die EU hat im Mai 1998 den Entwurf einer entsprechenden Richtlinie vorgestellt. (siehe http://www.ispo.cec.be/eif/policy/) [39]

[39] Literatur [SWI]

10.2.6 Mögliches Einsatzgebiet

Zukunftsvision des Autors:

Ich sehe da persönlich ein grosses Potential, wie man die Software Piraterie stoppen könnte.

Beim Installieren verlangt das Setup Programm eine Digitale Bestätigung, die „just-in-time“ durch eine Internetanfrage (Automatisches Mitliefern des Lizenz-Keys, der eindeutig ist) angefordert wird. Der Web Server des Softwareherstellers überprüft, ob diese ID bereits angefragt wurde, falls nein wird ein „Ein-Mal-Certificate“ generiert, dass den Lizenz-Key mit implementiert. Natürlich wird in diesem Zeitpunkt beim Softwarehersteller der Status für eine nochmalige Anfrage gesperrt.

Das Setup Programm erhält nun das benötigte Bestätigungs Certificate und prüft dieses nach den verschiedenen Komponenten. Falls Ueberprüfung erfolgreich wird Setup ausgeführt.

Der Clou dabei ist, dass dieses „Ein-Mal-Certificate“ auf dem Client nirgends zwischengespeichert wird und somit nicht kopierbar ist.

Nun, dass Programm an sich kann zwar problemlos „schwarz“ kopiert werden, aber bei der Installation verlangt das Programm wiederum ein Installations Certificate, d.h. der User muss sich wiederum anmelden. Doch in diesem Zeitpunkt merkt der Web Server, dass der Status gesperrt ist und sendet ein Error Certificate zurück, welches das Setup Programm frühzeitig beendet.

11 Abbildungsverzeichnis

12 Tabellenverzeichnis

13 Literaturverzeichnis

13.1 Zitierte Literatur

[SOD] Beth, T. Sichere offene Datennetze. (1995, Mai). Spektrum der Wissenschaft, S. 48 – 55.

[TAG] Chaos, Zahlen und Algorithmen. Geheimcodes in Gefahr? (1998, 28. August). Tages-Anzeiger, S. 46.

[TLS] CMP Media Inc. (1998). Web Security Technology Gains. USA: CMP Media Inc. (Online abrufbar http://www.techweb.com/wire/story/TWB19980703S0010)

[COM] Ein Schweizer Duo hält Hacker vom Web fern. IBM und die ETH Zürich entwicklen ein Sicherheitsverfahren. (1998, 31. August). Computerworld, S. 1.

[HAN] Hansen, H.R. (1996). Wirtschaftsinformatik I. 7. Auflage. Stuttgart: Lucius & Lucius.

[MAK] Marfurt, K. (1996/97). Kryptographie. Luzern: Höhere Fachschule für Wirtschaftsinformatik Luzern.

[MSS] Microsoft. (1998). Microsoft Securitiy Advisor. USA: Microsoft. (Online abrufbar http://www.microsoft.com/security).

[NCS] Netscape. (1998). Certificate Server 1.0. USA: Netscape. (Online abrufbar http://home.netscape/de/comprod/se...ral/product/certificate/datasheet.html).

[NCSH] Netscape. (1998). NetHelp – Netscape. Sicherheit. Certificates und digitale Unterschriften. USA: Netscape. (Helpfile Netscape Navigator).

[ECS] Orfali, R., Harkey, D. & Edwards, J. (1996). The Essential Client/Server Survival Guide. Second Edition. Canada: John Wiley & Sons, Inc.

[PAY] Payserv AG. (1996). TBSS (Telematic Base Security Services). Approved procedures and mechanisms for the protection of electronic data communications. Zürich: Payserv AG.

[SWI] Swisskey AG. (1998). Swisskey Certificate Aussteller. Zürich: Swisskey AG. (Online abrufbar http://www.swisskey.ch).

[SWK] Swisskey AG. (1998). Wie werden Certificates ausgestellt?. Zürich: Swisskey AG. (Online abrufbar http://www.europay.ch/ecommerce/set/set-de.html).

[UBS] Union Bank of Switzerland [UBS]. (1998). Public-Key Infrastructure. Vouche Replacement, Draft 0.1. Zürich: UBS.

[UPE] Union Bank of Switzerland [UBS]. (1998). UBS PKI Evaluation. Request for Proposal v1.1. Zürich: UBS.

[VER] Verisign. (1998). Introduction to Cryptography. USA: Verisign. (Online abrufbar http://www.verisign.com/docs/pk_intro.html).

[VIS] VISANA. (1997). Arbeitspapier: EDI über Internet. V 0.1. Zürich: VISANA.

13.2 Weiterführende Literatur

Fischer, P. (1997). Smart Books. Computer-LEXIKON für Studium & Praxis. Kilchberg: SmartBooks Publishing AG.

Microsoft. (1998). Windows NT 4.0 Option Pack. Product Documentation. USA: Microsoft. (Helpfile Option Pack).

14 Anhang

Sourcecodeauszug HTML und ASP Seiten

Nachfolgend sind alle benötigten HTML und ASP Seiten aufgeführt, wobei bei einigen normalen HTML Seiten nur die Beschreibung abgebildet ist. Bemerkungen sind in Visual Basic Script mit einem Hochkomma bezeichnet. Und die Scriptsprache wird zwischen folgende Tags eingeführt „<% Code %>“.

Includes

File Certificate.ini

```
<%
Const cACCOUNT01_NR             = "01"
Const cACCOUNT01_NAME           = "MEDIDATA\SRVWISWEB_Test"
Const cACCOUNT01_PASSWORT       = "admin"
Const cACCOUNT02_NR             = "02"
Const cACCOUNT02_NAME           = "MEDIDATA\SRVWISWEB_Test_Admin"
Const cACCOUNT02_PASSWORT       = "admin"
Const cACCOUNT03_NR             = "03"
Const cACCOUNT03_NAME           = ""
Const cACCOUNT03_PASSWORT       = ""
Const cACCOUNT04_NR             = "04"
Const cACCOUNT04_NAME           = ""
Const cACCOUNT04_PASSWORT       = ""
Const cMS_ACCESS_DB_ODBC        = "DSN=CertSrv;UID=Admin"
Const cHTTPS_ZERTIFIKAT_ABHOLUNG_URL    =
      "https://srvwisweb/certsrv/certenroll/md_certificate.asp?cert="
Const cHTTPS_ZERTIFIKAT_TEST_URL =
      "https://srvwisweb/certsrv/certenroll/md_test.asp?cert="
Const cPFAD_FUER_ZERTIFIKATS_SPEICHERUNG= "c:\zertifikat\client_certificate\"
Const cPFAD_FUER_LOGFILE        = "c:\internet\adminweb\certificate."
Const cERWEITERUNG_FUER_LOGFILE  = "log"
Const cPFAD_FUER_INIFILE        = "c:\internet\includes\certificate."
Const cERWEITERUNG_FUER_INIFILE  = "ini"
Const cMAIL_HOST                        = "mail.medidata.ch"
Const cMAIL_ABSENDER_FIRMA      = "MediData AG"
Const cMAIL_ABSENDER_EMAIL      = "webmaster@medidata.ch"
Const cMAIL_ABSENDER_BESCHREIBUNG       = "MediData AG, webmaster"
Const cMAIL_ABSENDER_NAME       = "Webmaster"
Const cMAIL_SUBJECT             = "Client Certificate MediFrame"
Const cMAIL_GRUSSFORMEL         = "Mit freundlichen Gruessen"
%>
```

File md_warten.asp

```
<%
'*****************************************************************************
' ALLGEMEIN FUER GANZE SEITE GUELTIGKEIT
' Script:                 keine
' WebSeite:               includes\md_warten.asp
' Author:                 MediData AG, Bosshard Stefan
' Datum:                  31.08.1998, V1.0
' Aenderungen:            --
' Beschreibung:     Zeigt Text, dass etwas in Verarbeitung ist.
'*****************************************************************************
%>
<html>
...
</html>
```

File md_func_sub.asp

```
<!--#include virtual="admin/certificate.ini"-->
<%

'*****************************************************************************
' #include virtual="admin/certificate.ini"     //Ini Daten ini_*
' Beschreibung:     Inifile mit all seinen Constanten (Abänderbar im Web)
'*****************************************************************************

'*****************************************************************************
' ALLGEMEIN FUER GANZE SEITE GUELTIGKEIT
' Script:           VBScript
' WebSeite:         includes\md_func_sub.asp
' Author:           MediData AG, Bosshard Stefan
' Datum:            31.08.1998, V1.0
' Aenderungen:      --
' Beschreibung:     Allgemein Funktionen für den ganzen Prototyp sind
'                   in diesem Script File abgebildet.
' Hinweis:          Diese Sub und Functionen können von jeder ASP Seite einge-
'                   bunden werden:
'                   !--#include virtual="admin/md_func_sub.asp"--
' Admin:            WICHTIG: Für Client Certificate Mapping muss dieses
'                   ASP File von einem "Administrator" ausgeführt werden!
'*****************************************************************************

'*****************************************************************************
' Public Function ini_account_combobox()
' Beschreibung:     Liefert Combobox in HTML Beschreibung für Admin Tool
' Input:                  --
' Output:           HTML mässige Beschreibung plus Inhalt der Combobox
' Aktivität:        --
'*****************************************************************************
public Function ini_account_combobox()
       temp = temp + "<select name='" + boxName + "account' size='1'>"
       temp = temp + "<option selected value='" + cACCOUNT01_NR + "'>" +
                      cACCOUNT01_NAME + "</option>"
       temp = temp + "<option value='" + cACCOUNT02_NR + "'>" +  cACCOUNT02_NAME +
                      "</option>"
       temp = temp + "<option value='" + cACCOUNT03_NR + "'>" +  cACCOUNT03_NAME +
                      "</option>"
       temp = temp + "<option value='" + cACCOUNT04_NR + "'>" +  cACCOUNT04_NAME +
                     "</option>"
       temp = temp + "</select>"
       ini_account_combobox = temp
End function

'*****************************************************************************
' Public Function ini_acccountname(string accountauswahl)
' Beschreibung:           Function liefert den entsprechenden Account für diesen
' Code
' Input:                  accountauswahl = Zweistellige Ziffer Bsp. 01
' Output:                 string Account=NT Accountnamen Bsp. '
                          MEDIDATA\SRVWISWEB_Test
' Aktivität:              --
' Admin:                  Diese Liste muss manuell angepasst werden

'*****************************************************************************
public Function ini_accountname(accountauswahl)

       Select Case accountauswahl
          Case cACCOUNT01_NR
               ini_accountname = cACCOUNT01_NAME
        Case cACCOUNT02_NR
               ini_accountname = cACCOUNT02_NAME
```

```
        Case cACCOUNT03_NR
              ini_accountname = cACCOUNT03_NAME
        Case cACCOUNT04_NR
              ini_accountname = cACCOUNT04_NAME
       End Select
End function

'*****************************************************************************
' Public Function ini_account_passwort(string accountauswahl)
' Beschreibung:           Function liefert zum entsprechenden Account das Passwort
' Input:                  accountauswahl = Zweistellige Ziffer Bsp. 01
' Output:                 string Passwort = NT Accountpasswort Bsp. admin
' Aktivität:              --
' Admin:                  Diese Liste muss manuell angepasst werden
'*****************************************************************************
public Function ini_account_passwort(accountauswahl)
       Select Case accountauswahl
         Case cACCOUNT01_NR
              ini_account_passwort = cACCOUNT01_PASSWORT
        Case cACCOUNT02_NR
              ini_account_passwort = cACCOUNT02_PASSWORT
        Case cACCOUNT03_NR
              ini_account_passwort = cACCOUNT03_PASSWORT
        Case cACCOUNT04_NR
              ini_account_passwort = cACCOUNT04_PASSWORT
       End Select

End function

'*****************************************************************************
' Public Function ini_datenbank()
' Beschreibung:     Liefert DSN Datenbankname plus UID und falls gewünscht auch
'                   Passwort für MS Access DB
' Input:                  --
' Output:           String DBInfo= DSN; UID; Passwort Bsp. "DSN=CertSrv;UID=Admin"
' Aktivität:
'*****************************************************************************
public Function ini_datenbank()
       ini_datenbank = cMS_ACCESS_DB_ODBC
End function

'*****************************************************************************
' Public Function ini_https_cert_abholung()
' Beschreibung:     Liefert 1. Teil der URL für Client-Zertifikat Abholung.
' Input:            --
' Output:           String DBInfo = https://....
' Aktivität:        --
' Admin:            Achtung! ?cert= ist fest codiert, dies darf nicht geändert
' werden.
'                   md_certificate.asp ist spezial File.
'*****************************************************************************
public Function ini_https_cert_abholung()
       ini_https_cert_abholung = cHTTPS_ZERTIFIKAT_ABHOLUNG_URL
End function

'*****************************************************************************
' Public Function ini_cert_test()
' Beschreibung:     Liefert 1. Teil der URL für Client-Zertifikat Test bzw. Mapping
' Input:            --
' Output:           String DBInfo= https://....
' Aktivität:
' Admin:            Achtung! ?cert= ist fest codiert, dies darf nicht geändert
'                   werden.
'                   md_test.asp ist spezial File.
'*****************************************************************************
public Function ini_cert_test()
```

```
       ini_cert_test = cHTTPS_ZERTIFIKAT_TEST_URL
End function

'*****************************************************************************
' Public Function ini_datum(string format)
' Beschreibung:     Liefert ein selber definiertes Datumsformat
' Input:            format = Je nach vorhandensein Bsp. "yy-mm-dd"
' Output:           string date  = Datum Bsp. "1998-08-31"
' Aktivität:        Heutiges Server Systemdatum
'*****************************************************************************
public Function ini_datum(format)
       tempDatum = Formatdatetime(now(),vbshortdate)
       Select Case format
        Case "yy-mm-dd"
               tempDatum = mid(tempDatum,7,2) & "-" & mid(tempDatum,4,2) & "-" &
                           mid(tempDatum,1,2)
              'speziell für MS Access DB
        Case "dd/mm/yy"
              tempDatum = mid(tempDatum,4,2) & "/" & mid(tempDatum,1,2) & "/" &
                           mid(tempDatum,7,2)
        Case "dd.mm.yy"
              tempDatum = mid(tempDatum,1,2) & "." & mid(tempDatum,4,2) & "." &
                           mid(tempDatum,7,2)
       End Select
       ini_datum = tempDatum
End function

'*****************************************************************************
' Public Function ini_pfad_zertifikat_speicherung_server()
' Beschreibung:     Pfad für Zertifikats Speicherung auf Server
' Input:            --
' Output:           string pfad = Pfad Bsp. "c:\temp"
' Aktivität:
'*****************************************************************************
public Function ini_pfad_zertifikat_speicherung_server()
       ini_pfad_zertifikat_speicherung_server = cPFAD_FUER_ZERTIFIKATS_SPEICHERUNG
End function

'*****************************************************************************
' Public Function ini_logFile_pfad()
' Beschreibung:     Pfad für Logfile
' Input:            --
' Output:           string pfad = Pfad Bsp. "c:\temp\logfile.log"
' Aktivität:
'*****************************************************************************
public Function ini_logFile_pfad()
       ini_logFile_pfad = cPFAD_FUER_LOGFILE + ini_logFile_Erweiterung()
End function

'*****************************************************************************
' Public Function ini_logFile_Erweiterung()
' Beschreibung:     Erweiterung für Logfile
' Input:            --
' Output:           string pfad = Pfad Bsp. "log"
' Aktivität:
'*****************************************************************************
public Function ini_logFile_Erweiterung()
       ini_logFile_Erweiterung = cERWEITERUNG_FUER_LOGFILE
End function

'*****************************************************************************
' Public Function ini_IniFile_pfad()
' Beschreibung:     Pfad für Inifile
' Input:            --
' Output:           string pfad = Pfad Bsp. "c:\temp\certificate.ini"
' Aktivität:
```

```
'*****************************************************************************
public Function ini_IniFile_pfad()
	ini_IniFile_pfad = cPFAD_FUER_INIFILE + ini_IniFile_Erweiterung()
End function

'*****************************************************************************
' Public Function ini_IniFile_Erweiterung()
' Beschreibung:     Erweiterung für Inifile
' Input:            --
' Output:           string pfad = Pfad Bsp. "ini"
' Aktivität:
'*****************************************************************************
public Function ini_IniFile_Erweiterung()
	ini_IniFile_Erweiterung = cERWEITERUNG_FUER_INIFILE
End function

'*****************************************************************************
' Public Function ini_mail_senden(string mailAdresse, string mailEmpfaengerName,
'                   string mailAnrede, string text)
' Beschreibung:     Server sendet automatisch Mail an entsprechende Adresse.
'                   Keinerlei Anpassungen an Mailprogramm etc. Empfang nicht
'                   möglich.
' Input:            mailAdresse = Email Bsp. stefan.bosshard@medidata.ch
'                   mailEmpfaengerName  = Name Bsp. Bosshard Stefan
'                   mailAnrede = Anrede Bsp. Sehr geehrter Herr Bosshard
'                   text = Mailtext
' Output:           string dummytext = Datum plus OK oder Error Status
' Admin:            Benötigt folgendes dll Serverseitig:
'                   C:\apps\ASP_DLL\aspemail.dll
'*****************************************************************************
public   Function   ini_mail_senden(mailAdresse,   mailEmpfaengerName,   mailAnrede,
textMail)

	nZ = Chr(13) + Chr(10)    'Zeilenumbruch
	Dim tempText

	Set Mail = Server.CreateObject("Persits.MailSender")
	Mail.Host = cMAIL_HOST
	Mail.From = cMAIL_ABSENDER_EMAIL
	Mail.FromName = cMAIL_ABSENDER_BESCHREIBUNG
	Mail.AddAddress mailAdresse, mailEmpfaengerName
	'Mail.AddAddress "paul@company.com", "Paul L.Johnson, Esq."
	'Mail.AddReplyTo "receptions@Kremlin.gov.ru"
	'Mail.AddAttachment "c:\images\map.gif"
	Mail.Subject = cMAIL_SUBJECT
	'Anrede plus Grussformel anhängen an Text
		tempText = mailAnrede + nZ + textMail + nZ + cMAIL_GRUSSFORMEL + nZ +
		cMAIL_ABSENDER_FIRMA + nZ + cMAIL_ABSENDER_NAME
	Mail.Body = tempText
	On Error Resume Next
	Mail.Send
	dummytext = ini_datum("dd.mm.yy") + ": OK"
	If Err <> 0 Then
		'Mail konnte nicht zugestellt werden aufgrund folgendem Error: " &
		Err.Description
		'Achtung: dieser Aufruf checkt aber nicht, ob die Mailadresse gültig
		ist oder nicht!
		dummytext = ini_datum("dd.mm.yy") + " Error: " + Err.Description

	End If
	ini_mail_senden = dummytext
End function


'*****************************************************************************
' Sub sub_logfileSchreiben(string text)
```

```
' Beschreibung:     Schreibt ein Record in ein Text Logfile (Serverseitig)
' Input:            text = Record, was in Logfile geschrieben werden soll
' Output:           --
' Aktivität:        Schreibt im "Append modus" Zeile für Zeile in Logfile.
'                   31.08.1998   44     Zertifikat gelöscht
'*****************************************************************************
Sub sub_logfileSchreiben(text)
       Set FileObject = Server.CreateObject("Scripting.FileSystemObject")
       logfile = ini_logFile_pfad()
       Set OutStream = FileObject.OpenTextFile (logfile, 8, True)
       OutputString = now() & chr(09) & text
       OutStream.WriteLine OutputString
       Set OutStream = Nothing
End Sub


'*****************************************************************************
' Sub sub_logfileLesen()
' Beschreibung:     Liest von Logfile alle Records ein.
' Input:            --
' Output:           --
' Aktivität:        Schreibt alle Records ins gewünschte Objekt.
'*****************************************************************************
Sub sub_logfileLesen()
       Set FileObject = Server.CreateObject("Scripting.FileSystemObject")
       logfile = ini_logFile_pfad()
       Set InStream= FileObject.OpenTextFile (logfile, 1, False, False)
       temp = InStream.ReadAll
       response.write temp
       Set Instream=Nothing
End Sub

'*****************************************************************************
' Function func_logfileArchivierenUndLeeren()
' Beschreibung:     Archiviert aktuelles Logfile, d.h. es wird kopiert und
'                   Aktuelles wird gelöscht. Es wird automatisch ein neues File
'                   generiert.
' Input:            --
' Output:           string Archivefilename
' Aktivität:        Siehe Beschreibung
'*****************************************************************************
public Function func_logfileArchivierenUndLeeren()
       Set FileObject = Server.CreateObject("Scripting.FileSystemObject")
       logfile = ini_logFile_pfad()

       dummyText = logfile + "-old-" + ini_datum("yy-mm-dd")

       'kopiert
       FileObject.CopyFile logfile, dummyText, true
       'löschen
       Set FileObject = Server.CreateObject("Scripting.FileSystemObject")
       FileObject.DeleteFile logfile
       'neues Logfile schreiben
       Set FileObject = Server.CreateObject("Scripting.FileSystemObject")
       Set OutStream = FileObject.OpenTextFile (logfile, 8, True)
       OutputString = "Aufzeichnungsbeginn: " + ini_datum("dd.mm.yy")
       OutStream.WriteLine OutputString
       Set OutStream = Nothing
       'Rückgabewert FileName
       func_logfileArchivierenUndLeeren = dummyText
End Function

'*****************************************************************************
' Sub sub_inifilelesen()
' Beschreibung:     Liest von Inifile alle Records ein.
' Input:            --
```

```
' Output:           --
' Aktivität:        Schreibt alle Records ins gewünschte Objekt.
'*******************************************************************************
Sub sub_inifilelesen()
       Set FileObject = Server.CreateObject("Scripting.FileSystemObject")
       Inifile = ini_IniFile_pfad()
       Set InStream= FileObject.OpenTextFile (Inifile, 1, False, False)
       temp = InStream.ReadAll
       response.write temp
       Set Instream=Nothing
End Sub

'*******************************************************************************
' Function func_inifilespeichern()
' Beschreibung:     Speichert Inifile und überschreibt alten Inhalt.
' Input:            --
' Output:           Filename der alten Datei *.old
' Aktivität:        Schreibt alle Records ins gewünschte Objekt.
'*******************************************************************************
Public Function func_inifilespeichern(text)
       Set FileObject = Server.CreateObject("Scripting.FileSystemObject")
       Inifile = ini_IniFile_pfad()
       dummyText = Inifile + "-old"
       'kopiert
       FileObject.CopyFile Inifile, dummyText, true
       'löscht
       Set FileObject = Server.CreateObject("Scripting.FileSystemObject")
       FileObject.DeleteFile Inifile
       'schreibt geänderten Inhalt in Object
       Set FileObject = Server.CreateObject("Scripting.FileSystemObject")
       Set OutStream= FileObject.OpenTextFile (Inifile, 8, True)
       OutputString = text
       OutStream.WriteLine OutputString
       Set OutStream = Nothing
       func_inifilespeichern = dummyText
End Function

'*******************************************************************************
' Sub sub_certificateSpeichern(string inhaltCertificate, string neuerPfad)
' Beschreibung:     Speichert generiertes Zertifikat Serverseitig ab (für Notfall)
' Input:            inhaltCertificate = Client-Zertifikat in PKCS#7 Format
'                   neuerPfad  = Pfad plus Filename für Zertifikat
' Output:           --
' Aktivität:        Speichert die Datei ab (allenfalls bei gleichem
'                   Zertifikatinhaber
'                   wird im Append Modus
'*******************************************************************************
Sub sub_certificateSpeichern(inhaltCertificate, neuerPfad)
       Set fs = Server.CreateObject("Scripting.FileSystemObject")
         Set OutStream = fs.OpenTextFile( neuerPfad + ".txt", 8, True )
       OutStream.WriteLine(inhaltCertificate)
       Set OutStream = Nothing
End Sub

'*******************************************************************************
' Function func_replace_Umlaute(string text)
' Beschreibung:     endert Umlaute ö, ä und ü ind ASCII Zeichen um.
' Input:            text = Stringtext
' Output:           string = geänderter Stringtext
' Aktivität:        --
'*******************************************************************************
Function func_replace_Umlaute(text)
       text = Replace(text, "ä", "ae")
       text = Replace(text, "ö", "oe")
       text = Replace(text, "ü", "ue")
       func_replace_Umlaute = text
```

```
End function

'*****************************************************************************
' Sub sub_text_link(string textausgabe, string URLadresse)
' Beschreibung:     Wenn irgendetwas zu Ende ist, kann diese Routine aufgerufen
'                   werden
'                   und es kommt eine Nachricht plus ein Link.
' Input:            textausgabe = Eine Meldung
'                   URLadresse   = Eine Url Adresse Bsp. http://srvwisweb
' Output:           --
' Aktivität:        Schreibt diese Meldungen in gewünschtes Objekt
'*****************************************************************************
sub sub_text_link(textausgabe, URLadresse)
      response.write textausgabe + "<img src='happyface.gif' width='' height=''>"
      response.write "<BR><BR>"
      response.write     "<p><a     href='"     +     URLadresse     +     "'><font
face='Arial'>[Weiter]</font></a></p>"
End Sub

'*****************************************************************************
' Sub sub_mappingLoeschen(string mappName)
' Beschreibung:     Löscht Certificate NT Account Mapping auf Server
' Input:            mappName = NT Mapping Name
'                   MappingObjectZuteilung    = Welches Web es betrifft, Bsp. 1 =
'                   Default Web
' Output:           --
' Aktivität:        Löscht das Mapping
'*****************************************************************************
Sub sub_mappingLoeschen(mappName, MappingObjectZuteilung)
      Dim CertObj
      'IIsCertMapper verlangt hier einen Standardpfad für Webserver
      'Wie kann aber nun auf diverse versch. Webs gezeigt werden?
      'MappingObjectZuteilung:
      '1 = Default Web Site
      '2 = Administrations Web Site
      '3 = ?
      objectString = "IIS://localhost/W3SVC/" + MappingObjectZuteilung +
                     "/IIsCertMapper"
      Set CertObj = GetObject(objectString)
      'Search by MappingName
      CertObj.DeleteMapping 2, mappName
End Sub

'*****************************************************************************
' Sub sub_sqlExecute(string sqlString)
' Beschreibung:     Führt Update, Delete oder Insert SQL Statements aus.
' Input:            sqlString = SQL Statement für DB
' Output:           --
' Aktivität:        Führt SQL Befehl aus.
'*****************************************************************************
Sub sub_sqlExecute(sqlString)
      'für Update, Insert und delete
      fp_sQry = sqlString
      fp_iMaxRecords = 0
      fp_iTimeout = 0
      fp_iCurrent = 1
      fp_fError = False
      fp_bBlankField = False
      If fp_iTimeout <> 0 Then Server.ScriptTimeout = fp_iTimeout
      Do While (Not fp_fError) And (InStr(fp_iCurrent, fp_sQry, "%%") <> 0)
            ' öffnendes Anführungszeichen gefunden, schließendes Anführungszeichen
            ' wird gesucht.
            fp_iStart = InStr(fp_iCurrent, fp_sQry, "%%")
            fp_iEnd = InStr(fp_iStart + 2, fp_sQry, "%%")
            If fp_iEnd = 0 Then
                  fp_fError = True
```

```
                Response.Write "<B>Datenbankbereichsfehler: Parameterbegrenzer
                passen nicht zusammen</B>"
        Else
                fp_sField = Mid(fp_sQry, fp_iStart + 2, fp_iEnd - fp_iStart - 2)
                If Mid(fp_sField,1,1) = "%" Then
                        fp_sWildcard = "%"
                        fp_sField = Mid(fp_sField, 2)
                Else
                        fp_sWildCard = ""
                End If
                fp_sValue = Request.Form(fp_sField)

                ' Prüfen, ob das angegebene Formularfeld tatsächlich existiert.
                If (len(fp_sValue) = 0) Then
                        fp_iCurrentField = 1
                        fp_bFoundField = False
                        Do While (InStr(fp_iCurrentField, fp_sDefault, fp_sField)
        <> 0) _ And Not fp_bFoundField
        fp_iCurrentField = InStr(fp_iCurrentField, fp_sDefault, fp_sField)
        fp_iStartField = InStr(fp_iCurrentField, fp_sDefault, "=")
        If fp_iStartField = fp_iCurrentField + len(fp_sField) Then
        fp_iEndField = InStr(fp_iCurrentField, fp_sDefault, "&")
        If (fp_iEndField = 0) Then fp_iEndField = len(fp_sDefault) + 1
        fp_sValue = Mid(fp_sDefault, fp_iStartField+1, fp_iEndField-1)
        fp_bFoundField = True
        Else
                fp_iCurrentField = fp_iCurrentField + len(fp_sField) - 1
                End If
        Loop
        End If

                ' Das folgende Codestück sucht nach dem angegebenen
                ' Formularfeldnamen,
                ' und verdoppelt darin alle einfachen Anführungszeichen,
                ' damit SQL nicht durch das Auftreten von nicht-paarweisen
                ' einfachen Anführungszeichen verwirrt wird.
                If (Mid(fp_sQry, fp_iStart - 1, 1) = """") Then
                        fp_sValue = Replace(fp_sValue, """", """""")
                ElseIf (Mid(fp_sQry, fp_iStart - 1, 1) = "'") Then
                        fp_sValue = Replace(fp_sValue, "'", "''")
                ElseIf Not IsNumeric(fp_sValue) Then
                        fp_sValue = ""
                End If

                If (len(fp_sValue) = 0) Then fp_bBlankField = True

                fp_sQry = Left(fp_sQry, fp_iStart - 1) + fp_sWildCard +
                fp_sValue + _
                        Right(fp_sQry, Len(fp_sQry) - fp_iEnd - 1)

                ' Die aktuelle Position so korrigieren, daß sie nach dem
                ' ersetzten Wert liegt.
                fp_iCurrent = fp_iStart + Len(fp_sValue) + Len(fp_sWildCard)
        End If
Loop
On Error Resume Next
set fp_rs = CreateObject("ADODB.Recordset")
fp_rs.Open fp_sQry, ini_datenbank()
If Err.Description <> "" Then
        Response.Write "<B>Datenbankfehler: " + Err.Description + "</B>"
        if fp_bBlankField Then
                Response.Write "  Eines oder mehrere Formularfelder waren leer."
        End If
        fp_rs.Close
' If-Anweisung schließen und auf Verbindungsfehler prüfen.
End If
```

```
      set fp_rs = Nothing
End Sub
%>
```

Medidataweb

File default.html

```
'*************************************************************************************
' ALLGEMEIN FUER GANZE SEITE GUELTIGKEIT
' Script:        Java Applet
' WebSeite:      medidataweb\default.html
' Author:        MediData AG, Bosshard Stefan
' Datum:         31.08.1998, V1.0
' Aenderungen:   --
' Beschreibung:  Link für öffentliches und privates Web.
'                JavaApplet HollywoodText.class wurde von folgender Seite
'                heruntergeladen:
'                http://www.Free-Applets.com
'                Einzige Aenderung wurden in "demo.txt" vorgenommen, darin kann
'                der Text mutiert werden, welcher im Browser erscheinen soll!
'*************************************************************************************
...
```

OeffentlichesWeb

File ca_download.html

```
'*******************************************************************************
' ALLGEMEIN FUER GANZE SEITE GUELTIGKEIT
' Script:        keine Scripts
' WebSeite:      oeffentlichesweb\ca_download.html
' Author:        MediData AG, Bosshard Stefan
' Datum:         31.08.1998, V1.0
' Aenderungen:   --
' Beschreibung:  Link für das Download des CA Certificates der MediData AG.
'                Client benötigt dies, um überhaupt eine SSL Verbindung etc.
'                aufbauen zu können.
'*******************************************************************************
...
```

File index.html

```
'*******************************************************************************
' ALLGEMEIN FUER GANZE SEITE GUELTIGKEIT
' Script:        keine Scripts
' WebSeite:      oeffentlichesweb\index.html
' Author:        MediData AG, Bosshard Stefan
' Datum:         31.08.1998, V1.0
' Aenderungen:   --
' Beschreibung:  Link zur öffentlichen Seite und Link zur Online Registrierung.
'*******************************************************************************
...
```

File md_certificate_request_dataform.html

```
'*******************************************************************************
' ALLGEMEIN FUER GANZE SEITE GUELTIGKEIT
' Script:        JavaScript
' WebSeite:      oeffentlichesweb\md_certificate_request_dataform.html
' Author:        MediData AG, Bosshard Stefan
' Datum:         31.08.1998, V1.0
' Aenderungen:   --
' Beschreibung:  Der Kunde kann sich mittels Eintragung online für ein Client
'                Zertifikat anmelden.
'*******************************************************************************
...
```

File oeffentliche_daten.html

```
'*******************************************************************************
' ALLGEMEIN FUER GANZE SEITE GUELTIGKEIT
' Script:        keine Scripts
' WebSeite:      oeffentlichesweb\oeffentliche_daten.html
' Author:        MediData AG, Bosshard Stefan
' Datum:         31.08.1998, V1.0
' Aenderungen:      --
' Beschreibung:  In dieser HTML Seite können öffentliche Daten für jederman stehen.
'*******************************************************************************
...
```

PrivatWeb

File index.asp

```
<--
'*****************************************************************************
' ALLGEMEIN FUER GANZE SEITE GUELTIGKEIT
' Script:        VBScript
' WebSeite:      privatweb\index.asp
' Author:        MediData AG, Bosshard Stefan
' Datum:         31.08.1998, V1.0
' Aenderungen:   Checkt Cookie und Zertifikat, falls beides übereinstimmt, wird
'                Zulass gewährt in öffentliches Web.
'*****************************************************************************
-->
<%
'Client-Zertifikat - Inhalte abfragen
dummyZertifikat = Request.ClientCertificate("SubjectO")
dummyZertifikat = dummyZertifikat & Request.ClientCertificate("SubjectCN")
dummyZertifikat = dummyZertifikat & Request.ClientCertificate("SubjectEmail")
%>
<%
commonName = Request.ClientCertificate("SubjectCN")
'Holt Inhalt aus dem Client Cookie. Bsp.  "..\netscape\user\%username%\cookies.txt"
'Eineindeutige_Erkennung   = Cookie Name
'commonName                = Cookie Subteil
dummyCookie = Request.Cookies("Eineindeutige_Erkennung")(commonName)
%>
<html>

<head>
<title>Privat - Index</title>
</head>

<body>
<%
'Cookie Inhalt mit Client-Zertifikat Inhalt vergleichen:
'Falls identisch, hat User Zugriff, andernfalls nicht, da warscheinlich Zertifikat
'weiterkopiert!
'ACHTUNG: Falls Unternehmung oder User Cookies deaktiviert hat, funktioniert dies
'nicht und der User wird nie Zugriff erhalten!!!
if dummyCookie = dummyZertifikat then %>
...
<% else %>
...
<% End if %>
</body>
</html>
```

Certenroll

File md_ausdruck_info.html

```
'*******************************************************************************
' ALLGEMEIN FUER GANZE SEITE GUELTIGKEIT
' Script:        keine Scripts
' WebSeite:      certenroll\md_ausdruck_info.html
' Author:        MediData AG, Bosshard Stefan
' Datum:         31.08.1998, V1.0
' Aenderungen:   --
' Beschreibung:  Infofenster, das Anmeldung ausgedruckt werden soll und wie der
'                Kunde informiert wird.
'*******************************************************************************
```

File kgaccept.asp

```
'*******************************************************************************
' ALLGEMEIN FUER GANZE SEITE GUELTIGKEIT
' Script:        VBScript
' WebSeite:      certenroll\kgaccept.asp
' Author:        MediData AG, Bosshard Stefan
' Datum:         31.08.1998, V1.0
' Aenderungen:   --
' Beschreibung:  Generierung des Certificates auslösen beim Certificate Server
'                plus Installation im Browser. Unter anderem wird noch ins
'                Logfile geschrieben und der Status in der DB geändert.
'*******************************************************************************
<%
Response.ContentType = "application/x-x509-user-cert"
Dim CertReq, SubmitFlag, GetCertFlag, Disp, Cert

'Abfüllen der Feldinhalte von md_certificate_install.asp
atrbs = "challenge: test" + vbCrLf
atrbs = atrbs + "commonname: " + Request.Form("commonname") + vbCrLf
atrbs = atrbs + "orgunit: " + Request.Form("orgunit") + vbCrLf
atrbs = atrbs + "org: " + Request.Form("Org") + vbCrLf
atrbs = atrbs + "locality: " + Request.Form("Locality") + vbCrLf
trbs = atrbs + "state: " + Request.Form("State") + vbCrLf
atrbs = atrbs + "country: " + Request.Form("Country") + vbCrLf
atrbs = atrbs + "email: " + Request.Form("email") + vbCrLf
'Abfrage von md_certificate_install.asp - welches Zertifikat Format
CertReq = Request.Form("CertRequest")
%>
<%
set ICertRequest = Server.CreateObject("CertificateAuthority.Request")
set ICertConfig = Server.CreateObject("CertificateAuthority.Config")
'Abfrage nach Bezeichnung des WebServers
ConfigString = ICertConfig.GetConfig(0)
'Client-Zertifikat request wird gestartet und an Server gesendet
Disp = ICertRequest.Submit(513, CertReq, atrbs, ConfigString)
'Zertifikat wird von Server abgeholt.
'Returnwert ist das Zertifikat in PKCS#7 Format - Bsp. "------BEGIN CERTIF ......"
'Folgende Input Werte sind möglich:
'CR_OUT_BINARY
'CR_OUT_BASE64
'CR_OUT_BASE64HEADER  = 0 (Standard)
Certificate = ICertRequest.GetCertificate(0)
'Folgendes Statement übergibt dem Client das Zertifikat
%>
<%=Certificate%>
<!--#include virtual="admin/md_func_sub.asp"-->
<%
```

```
'*****************************************************************************
' #include virtual="admin/md_func_sub.asp"
' Beschreibung:         siehe File selber (alles beschrieben betr. Input/Output)
' Functionen und Subs:  func_*; sub_*; ini_* (ini beinhaltet Konstanten)
'*****************************************************************************
'Datum Format yy-mm-dd
dummydate = ini_datum("yy-mm-dd")
'Nimmt ersten beiden Zahlen für Bestimmung, welcher Account
NT_Accountname = ini_accountname(mid(Request.Form("ReqDummy"),1,2))
'Pfad, wo Zertifikat auf Server abgespeichert werden soll
pfad = ini_pfad_zertifikat_speicherung_server()
tempOrg = Request.Form("Org")
'Alle Leerzeichen ersetzen
tempOrg = Replace(tempOrg, " ", "_", 1, 1)
tempCommonName = Request.Form("CommonName")
'Alle Leerzeichen ersetzen
tempCommonName = Replace(tempCommonName, " ", "_", 1, 1)
beschriftung = dummydate + "-" + tempOrg + "-" + tempCommonName + "-" +
Request.Form("ReqDummy")
neuerPfad = pfad + beschriftung

'Certificate auf Server speichern
call sub_certificateSpeichern(Certificate, neuerPfad)
'Logfile schreiben
call sub_logfileSchreiben(Request.Form("RegistrierungsNr") & chr(09) & beschriftung
& chr(09) & "Zertifikat abgeholt und gemappt auf folgender Account: " &
NT_Accountname)
'Hier wird Status (10 Freigegeben auf 11 Certificate abgeholt) geändert, damit nur
einmal request
call sub_sqlExecute("UPDATE md_Certificate_Anfrage SET Status = '11' , NT_Account =
'" + NT_Accountname + "' , Mapping_Name = '" + beschriftung + "', StatusDatum = '"
+ ini_datum("dd/mm/yy") + "' , Cert_ausstell_datum = '" + ini_datum("dd/mm/yy") +
"' WHERE RequestID = '" + Request.Form("ReqDummy") + "'")
%>
```

File md_certificate.asp

```
'*****************************************************************************
' ALLGEMEIN FUER GANZE SEITE GUELTIGKEIT
' Script:        VBScript plus JavaScript
' WebSeite:      certenroll\md_certificate.asp
' Author:        MediData AG, Bosshard Stefan
' Datum:         31.08.1998, V1.0
' Aenderungen:   --
' Beschreibung:  Der Kunde erhält unsere automatisch generierte URL und kann
'                mittels dieser und dem Passwort das Zertifikat abholen. Zuvor
'                wird aber noch überprüft, ob Zertifikat nicht schon einmal
'                bezogen wurde.
'*****************************************************************************
<%Response.Expires=0%>
<%
%>
<html>
...
<script language="javascript">
//*****************************************************************************
// function start()
// Beschreibung: Checked Passwort, ob wirklich Abholungsberechtigt. Passwort wurde
//               bei Request in DB gespeichert. Ausserdem wird nachgeprüft, ob
//               Zertifikat wirklich zum Erstenmal abgeholt wird, ansonsten wird
//               gesperrt.
// Input:        --
// Output:       --
// Aktivität:    Falls alles i.O. wird Zertifikat installiert
//*****************************************************************************
```

```
function start()
{
    dummy = 1
    //Passwortüberprüfung
    if (document.test.passwort_abholen.value == document.test.check.value)
    {
        //Wenn Status 10, darf Zertifikat abgeholt werden
        if (document.test.Status.value == 10)
        {
            document.test.submit()
            //Seite öffnen, welche das Zertifikat installiert
            document.test.action="md_certificate_install.asp"
        }
        else
        {
            //Falls bereits abgeholt
            if (document.test.Status.value == 11)
            {
                msg     =     "Certificate     wurde     bereits     am     "     +
document.test.Cert_ausstell_datum.value + " abgeholt. Bitte nehmen Sie Kontakt mit
MediData AG auf. Danke"
                alert(msg)
            }
            //Bei anderem Status
            else
            {
                alert("Certificate wurde entweder noch nicht freigegeben oder wurde
gesperrt!")
            }
        }
    }
    else
    {
        document.test.check.value = ""
        alert("Keine Berechtigung. Versuchen Sie nochmals. Bei Problemen nehmen Sie
bitte mit MediData AG Kontakt auf.")
        dummy = dummy + 1
            //Wenn User mehr als zweimal versuchte sich einzuloggen, wird Browser
geschlossen
            if (dummy == 3)
            {
                alert("Sorry, Sie versuchten bereits zweimal einzuloggen")
                window.close();
            }
    }
}
</script>
<!--#include virtual="admin/md_func_sub.asp"-->
<%
'*****************************************************************************
' #include virtual="admin/md_func_sub.asp"
' Beschreibung:         siehe File selber (alles beschrieben betr. Input/Output)
' Functionen und Subs:  func_*; sub_*; ini_* (ini beinhaltet Konstanten)
'*****************************************************************************

'*****************************************************************************
' Beschreibung:
' Setzt folgende URL voraus: ../certsrv/certenroll/md_certificate.asp?cert=nummer
' Nummer ist die URLID, die automatisch generiert wurde, um das Zertifikat abzu-
' holen. Diese zusammen mit dem Passwortcheck erlauben das Abholen!
' Diese SQL Abfrage holt alle benötigten Daten aus der DB und stellt diese bereit
' für die Certificate Generierung.
'*****************************************************************************
reqdummy = Request("cert")
fp_sQry = "Select * from md_certificate_anfrage WHERE requestID = '" + reqdummy +
...
```

```
            If Not IsEmpty(fp_rs) And Not (fp_rs Is Nothing) Then
                dummy0 = CStr(fp_rs("RegistrierungsNr"))
                dummy1 = CStr(fp_rs("Org"))
                dummy2 = CStr(fp_rs("OrgUnit"))
                dummy3 = CStr(fp_rs("CommonName"))
                dummy4 =  "_"
                dummy5 = CStr(fp_rs("Locality"))
                dummy6 = CStr(fp_rs("Email"))
                dummy7 = CStr(fp_rs("Country"))
                dummy8 = CStr(fp_rs("Passwort"))
                dummy9 = CStr(fp_rs("Status"))
                dummy10 = CStr(fp_rs("Cert_ausstell_datum"))
            End if
...
%>
<body bgcolor="#48A492" text="#FFFFFF" link="#FFFFFF" vlink="#FF0000">
...
<form name="test" method="POST">
...
<input type="button" onclick="javascript:start()" value="OK" name="B1">
</form>
</body>
</html>
```

File md_certificate_install.asp

```
'*************************************************************************************
' ALLGEMEIN FUER GANZE SEITE GUELTIGKEIT
' Script:        VBScript plus JavaScript
' WebSeite:      certenroll\md_certificate_install.asp
' Author:        MediData AG, Bosshard Stefan
' Datum:         31.08.1998, V1.0
' Aenderungen:   --
' Beschreibung:  Diese Seite wird von md_certificate.asp aufgerufen und checkt
'                diverse Sachen wie Schlüsselgrösse, Browsertyp etc. und startet
'                dann kgaccept.asp.
' Bemerkung:     Der Code, der nicht beschrieben wird, ist eine Standardlösung
'                von Microsoft, d.h. der Code wurde nicht von MediData AG ge-
'                geschrieben.
'*************************************************************************************
```

File md_certificate_request_ausdruck.asp

```
'*************************************************************************************
' ALLGEMEIN FUER GANZE SEITE GUELTIGKEIT
' Script:        VBScript plus JavaScript
' WebSeite:      certenroll\md_certificate_request_ausdruck.asp
' Author:        MediData AG, Bosshard Stefan
' Datum:         31.08.1998, V1.0
' Aenderungen:   --
' Beschreibung:  Diese Seite stellt das Anmeldeformular bzw. die Bestätigung dar,
'                welches vom Kunden ausgedruckt wird und unterzeichnet an uns
'                gesendet wird. Aufgrund dieses Schreibens wird das Zertifikat
'                freigegeben.
'*************************************************************************************
```

File md_certificate_request_dataform_insert.asp

```
'*************************************************************************************
' ALLGEMEIN FUER GANZE SEITE GUELTIGKEIT
' Script:        VBScript
' WebSeite:      certenroll\md_certificate_request_dataform_insert.asp
' Author:        MediData AG, Bosshard Stefan
' Datum:         31.08.1998, V1.0
```

```
' Aenderungen:   --
' Beschreibung:  Schreibt den Inhalt der online Anmeldung in die DB.
'*******************************************************************************
```

File md_test.asp

```
'*******************************************************************************
' ALLGEMEIN FUER GANZE SEITE GUELTIGKEIT
' Script:        VBScript
' WebSeite:      certenroll\md_test.asp
' Author:        MediData AG, Bosshard Stefan
' Datum:         31.08.1998, V1.0
' Aenderungen:   --
' Beschreibung:  Der Kunde erhält eine automatisch generierte URL. Sobald er diese
'                öffnen will, verlangt diese Seite das Zertifikat. Durch anklicken
'                des Linkes, öffnet es die Seite md_test1.asp.
'                Setzt folgende URL voraus: ../certenroll/md_test.asp?cert=nummer
'*******************************************************************************
```

File md_test1.asp

```
<%
'*******************************************************************************
' ALLGEMEIN FUER GANZE SEITE GUELTIGKEIT
' Script:        VBScript
' WebSeite:      certenroll\md_test1.asp
' Author:        MediData AG, Bosshard Stefan
' Datum:         31.08.1998, V1.0
' Aenderungen:   --
' Beschreibung:  Der Browser (gegen Vorweisung des gerade installierten
'                Certificates)
'                bzw. der Server checkt, ob alles in Ordnung ist. Falls ja,
'                wird dieses Zertifikat automatisch zum entsprechenden NT Account
'                gemappt. Ab diesem Zeitpunkt hat der Kunde Zugriff auf das
'                private Web.
'                Setzt folgende URL voraus: ../certenroll/md_test1.asp?cert=nummer
' Admin:         Für Client Certificate Mapping muss dieses ASP File von einem
'                "Administrator" Account ausgeführt werden!
'*******************************************************************************
%>
<%Response.Expires=0%>
<html>
<body bgcolor="#B7C8C2">
...
<!--#include virtual="admin/md_func_sub.asp"-->
<%
'*******************************************************************************
' #include virtual="admin/md_func_sub.asp"
' Beschreibung:         siehe File selber (alles beschrieben betr. Input/Output)
' Functionen und Subs:  func_*; sub_*; ini_* (ini beinhaltet Konstanten)
'*******************************************************************************

'*******************************************************************************
' Beschreibung:
' Diese SQL Abfrage holt alle benötigten Daten aus der DB und stellt diese bereit
' für die Certificate Generierung. Status 11 = Zertifikat ausgestellt.
' Rückgaebwerte DB: CommonName, Org, Mappingname und Accountname
'*******************************************************************************
Dim CertObj, vCert
fp_sQry = "Select * from md_Certificate_Anfrage WHERE Status = 11 AND RequestID =
...
If Not IsEmpty(fp_rs) And Not (fp_rs Is Nothing) Then
```

```
    'Werte werden in Variable abgefüllt
    dbName = CStr(fp_rs("CommonName"))
    dbOrg = CStr(fp_rs("Org"))
    beschriftungMapping = CStr(fp_rs("Mapping_Name"))
    accountNameMapping = CStr(fp_rs("NT_Account"))
    'Bsp. 01 = Default Web Site
    '     02 = Administration Web Site
    '     03 = ?
    MappingObjectZuteilung = mid(CStr(fp_rs("RequestID")), 2, 1)
End if
...
'*****************************************************************************
' Beschreibung:
' Ueberprüfung plus Mapping des Certificates
'
' IIsCertMapper ist ein Objekt der ADSI (Active Directory Services Interface)
' und musste speziell auf Server installiert werden!
' CertObj.CreateMapping Zertifikat, AccountName, PasswortAccount, EnableMapping
'*****************************************************************************
'Zertifikat in Variable vCert zwischenspeichern
vCert = Request.ClientCertificate("Certificate")
'vorgewiesenes Zertifikat Organisation plus CommonName abfragen
certOrg = Request.ClientCertificate("SubjectO")
certName = Request.ClientCertificate("SubjectCN")

'Wenn DB Eintrag mit vorgewiesenem Zertifikat übereinstimmt, dann Mapping ausführen
if certOrg = dbOrg AND certName = dbName then
    'IIsCertMapper verlangt hier einen Standardpfad für Webserver
    'Wie kann aber nun auf diverse versch. Webs gezeigt werden?
    'MappingObjectZuteilung:
    '1 = Default Web Site
    '2 = Administrations Web Site
    '3 = ?
    objectString    =    "IIS://localhost/W3SVC/"    +       MappingObjectZuteilung    +
"/IIsCertMapper"
    Set CertObj = GetObject(objectString)
    'Mit diesem Befehl wird das Mapping auf entsprechenden NT Account durchgeführt
    CertObj.CreateMapping                  vCert,                  accountNameMapping,
ini_account_passwort(mid(Request("cert"),1,2)), beschriftungMapping, True
    'Mapping Eintrag in DB, dass per heute ok

    temp = "UPDATE md_Certificate_Anfrage SET Status = '12' , Mapping_erfolgt_am =
'" + ini_datum("dd/mm/yy") + "' WHERE RequestID = '" + Request("cert") + "'"
    call sub_sqlExecute(temp)
    response.write    "Test    war    erfolgreich    <a    href='http://srvwisweb'><font
face='Arial'>Home Seite</font></a>"
else
    response.write "Test failed. Bitte nehmen Sie Kontakt mit MediData AG auf! <a
href='http://srvwisweb'><font face='Arial'>Home Seite</font></a>"
end if
%>
</body>
</html>
```

AdminWeb

File ca_admin.asp

```
'*************************************************************************************
' ALLGEMEIN FUER GANZE SEITE GUELTIGKEIT
' Script:        keine Scripts
' WebSeite:      adminweb\md_admin.asp
' Author:        MediData AG, Bosshard Stefan
' Datum:         31.08.1998, V1.0
' Aenderungen:   --
' Beschreibung:  Index für alle Verwaltungszwecke
' Admin:         Das ganze Administrationsdirectory darf nur von "Administratoren"
'                verwaltet werden, d.h. Administ. Zertifikate ausstellen und mappen.
'*************************************************************************************
```

File md_admin_aktiv_retoursetzen.asp

```
'*************************************************************************************
' ALLGEMEIN FUER GANZE SEITE GUELTIGKEIT
' Script:        VBScript
' WebSeite:      adminweb\md_admin_aktiv_retoursetzen.asp
' Author:        MediData AG, Bosshard Stefan
' Datum:         31.08.1998, V1.0
' Aenderungen:   --
' Beschreibung:  Aktive bzw. gemappte Zertifikate nochmals zurücksetzen für
'                Freigabe, da etwas schief lief bei Installation
'                (Status 11 bzw. 12 zurück auf 10) plus Mail an Kunde.
'*************************************************************************************
```

File md_admin_loeschen_suche.asp

```
'*************************************************************************************
' ALLGEMEIN FUER GANZE SEITE GUELTIGKEIT
' Script:        JavaScript
' WebSeite:      adminweb\md_admin_loeschen_suche.asp
' Author:        MediData AG, Bosshard Stefan
' Datum:         31.08.1998, V1.0
' Aenderungen:       --
' Beschreibung:  Suchen der aktiven bzw. gemappten Zertifikate um diese zu löschen.
'                Starten der Seite md_admin_loeschen.asp.
'*************************************************************************************
```

File md_admin_auflistung.asp

```
'*************************************************************************************
' ALLGEMEIN FUER GANZE SEITE GUELTIGKEIT
' Script:        VBScript
' WebSeite:      adminweb\md_admin_auflistung.asp
' Author:        MediData AG, Bosshard Stefan
' Datum:         31.08.1998, V1.0
' Aenderungen:   --
' Beschreibung:  Auflistung aller Statuse ?
'                Verarbeitung von folgenden Aufgaben:
'                - Zertifikate pendent Status 1
'                - Zertifikate freigeben Status 10
'                - Zertifikate abgeholt Status 11
'                - Zertifikate gemappt Status 12
'                - Zertifikate verwerfen Status 20
'                - Zertifikate "wieder" freigeben
'                - Zertifikate löschen Status 99
'*************************************************************************************
```

File md_admin_alle_suche.asp

```
'*****************************************************************************
' ALLGEMEIN FUER GANZE SEITE GUELTIGKEIT
' Script:        JavaScript
' WebSeite:      adminweb\md_admin_alle_suche.asp
' Author:        MediData AG, Bosshard Stefan
' Datum:         31.08.1998, V1.0
' Aenderungen:      --
' Beschreibung:  Suche von allen Zertifikaten (Status neutral)
'                Oeffnet md_admin_auflistung.asp
'*****************************************************************************
```

File md_admin_freigegebene.asp

```
'*****************************************************************************
' ALLGEMEIN FUER GANZE SEITE GUELTIGKEIT
' Script:        VBScript
' WebSeite:      adminweb\md_admin_freigegebene.asp
' Author:        MediData AG, Bosshard Stefan
' Datum:         31.08.1998, V1.0
' Aenderungen:   --
' Beschreibung:  Anzeige von allen Freigegebenen Zertifikaten (Status 10)
'*****************************************************************************
```

File md_admin_gelöschte.asp

```
'*****************************************************************************
' ALLGEMEIN FUER GANZE SEITE GUELTIGKEIT
' Script:        VBScript
' WebSeite:      adminweb\md_admin_gelöschte.asp
' Author:        MediData AG, Bosshard Stefan
' Datum:         31.08.1998, V1.0
' Aenderungen:   --
' Beschreibung:  Anzeige von allen gelöschten Zertifikaten (Status 99)
'*****************************************************************************
```

File md_admin_pendent.asp

```
'*****************************************************************************
' ALLGEMEIN FUER GANZE SEITE GUELTIGKEIT
' Script:        VBScript
' WebSeite:      adminweb\md_admin_pendent.asp
' Author:        MediData AG, Bosshard Stefan
' Datum:         31.08.1998, V1.0
' Aenderungen:   --
' Beschreibung:  Anzeige von allen pendenten Zertifikaten (Status 1)
'*****************************************************************************
<html>
<body bgcolor="#B7C8C2">
...
<%
if request.form ("suchbegriff") = "" then
   tempFeld = ""
else
   tempFeld = "[" + request.form ("feldauswahl") + "] = "
end if
tempSuch = request.form ("suchbegriff")

response.write tempFeld + tempSuch

%>
```

```
...
<!--#include virtual="admin/md_func_sub.asp"-->
<%

'*************************************************************************
' #include virtual="admin/md_func_sub.asp"
' Beschreibung:         siehe File selber (alles beschrieben betr. Input/Output)
' Functionen und Subs:  func_*; sub_*; ini_* (ini beinhaltet Konstanten)
'*************************************************************************
if request.form("suchbegriff") = "" then
   tempSQL = "Select * from md_certificate_anfrage where status = 1 order by
registrierungsnr"
else
   tempSQL = "Select * from md_certificate_anfrage where status = 1 AND " +
request.form ("feldauswahl")
   tempSQL = tempSQL + " like '%%suchwort%%' ORDER BY registrierungsnr"
End if

'Variable für Checkbox
Dim boxName
boxName = ""
'Variable für höchsten Wert der Registr. Nummer
maxWert = 0
dummyMaxWert = 0
' Formularparameter in der Abfrage ersetzen.
' Nur die Pendenten anzeigen Status = 1
fp_sQry = tempSQL
fp_sDefault = ""
fp_sNoRecords = "Keine Datensätze erhalten"
fp_iMaxRecords = 0
fp_iTimeout = 0
fp_iCurrent = 1
fp_fError = False
fp_bBlankField = False
If fp_iTimeout <> 0 Then Server.ScriptTimeout = fp_iTimeout
Do While (Not fp_fError) And (InStr(fp_iCurrent, fp_sQry, "%%") <> 0)
   '  öffnendes Anführungszeichen gefunden, schließendes Anführungszeichen wird
gesucht.
   fp_iStart = InStr(fp_iCurrent, fp_sQry, "%%")
   fp_iEnd = InStr(fp_iStart + 2, fp_sQry, "%%")
   If fp_iEnd = 0 Then
      fp_fError = True
      Response.Write "<B>Datenbankbereichsfehler: Parameterbegrenzer passen nicht
zusammen</B>"
   Else
      fp_sField = Mid(fp_sQry, fp_iStart + 2, fp_iEnd - fp_iStart - 2)
      If Mid(fp_sField,1,1) = "%" Then
         fp_sWildcard = "%"
         fp_sField = Mid(fp_sField, 2)
      Else
         fp_sWildCard = ""
      End If
      fp_sValue = Request.Form(fp_sField)

      ' Prüfen, ob das angegebene Formularfeld tatsächlich existiert.
      If (len(fp_sValue) = 0) Then
         fp_iCurrentField = 1
         fp_bFoundField = False
         Do While (InStr(fp_iCurrentField, fp_sDefault, fp_sField) <> 0) _
            And Not fp_bFoundField
            fp_iCurrentField = InStr(fp_iCurrentField, fp_sDefault, fp_sField)
            fp_iStartField = InStr(fp_iCurrentField, fp_sDefault, "=")
            If fp_iStartField = fp_iCurrentField + len(fp_sField) Then
               fp_iEndField = InStr(fp_iCurrentField, fp_sDefault, "&")
               If (fp_iEndField = 0) Then fp_iEndField = len(fp_sDefault) + 1
               fp_sValue = Mid(fp_sDefault, fp_iStartField+1, fp_iEndField-1)
```

```
                fp_bFoundField = True
            Else
                fp_iCurrentField = fp_iCurrentField + len(fp_sField) - 1
            End If
         Loop
      End If

      ' Das folgende Codestück sucht nach dem angegebenen Formularfeldnamen,
      ' und verdoppelt darin alle einfachen Anführungszeichen,
      '  damit  SQL  nicht  durch  das  Auftreten  von  nicht-paarweisen  einfachen
Anführungszeichen verwirrt wird.
      If (Mid(fp_sQry, fp_iStart - 1, 1) = """) Then
         fp_sValue = Replace(fp_sValue, """", """""")
      ElseIf (Mid(fp_sQry, fp_iStart - 1, 1) = "'") Then
         fp_sValue = Replace(fp_sValue, "'", "''")
      ElseIf Not IsNumeric(fp_sValue) Then
         fp_sValue = ""
      End If

      If (len(fp_sValue) = 0) Then fp_bBlankField = True

      fp_sQry = Left(fp_sQry, fp_iStart - 1) + fp_sWildCard + fp_sValue + _
         Right(fp_sQry, Len(fp_sQry) - fp_iEnd - 1)

      '  Die  aktuelle  Position  so  korrigieren,  daß  sie  nach  dem  ersetzten  Wert
liegt.
      fp_iCurrent = fp_iStart + Len(fp_sValue) + Len(fp_sWildCard)
   End If
Loop

If Not fp_fError Then
   ' Den Verbindungstext so verwenden, wie er im Assistenten eingegeben wurde.
   On Error Resume Next
   set fp_rs = CreateObject("ADODB.Recordset")
   If fp_iMaxRecords <> 0 Then fp_rs.MaxRecords = fp_iMaxRecords
   fp_rs.Open fp_sQry, ini_datenbank()
   If Err.Description <> "" Then
      Response.Write "<B>Datenbankfehler: " + Err.Description + "</B>"
      if fp_bBlankField Then
         Response.Write "  Eines oder mehrere Formularfelder waren leer."
      End If
   Else
      ' Den Fall prüfen, daß kein Datensatz vorhanden ist.
      If fp_rs.EOF And fp_rs.BOF Then
         Response.Write fp_sNoRecords
      Else
         ' While-Schleife zum Einlesen jedes Datensatzes aus der Ergebnismenge.
         Do Until fp_rs.EOF
%>
    <tr>
      <td width="24"><font face="Arial"><%
If   Not   IsEmpty(fp_rs)   And   Not   (fp_rs   Is   Nothing)   Then   Response.Write
CStr(fp_rs("RegistrierungsNr"))
dummyMaxWert = CStr(fp_rs("RegistrierungsNr"))
if dummyMaxWert > maxWert then
   maxWert = dummyMaxWert
end if
%> <!--webbot bot="DatabaseResultColumn" startspan
      s-column local_preview="Datenbank: " preview="Datenbank: " --><%
If   Not   IsEmpty(fp_rs)   And   Not   (fp_rs   Is   Nothing)   Then   Response.Write
CStr(fp_rs(""))
%>
<!--webbot
      bot="DatabaseResultColumn" endspan --></font></td>
      <td width="20"><%
boxName = CStr(fp_rs("RegistrierungsNr"))
```

```
Response.Write ("<input type='radio' value='pendent' checked name='" + boxName +
"'>")%>
</td>
      <td width="20"><%
Response.Write ("<input type='radio' value='freigabe' name='" + boxName + "'>")%>
<% Response.Write ini_account_combobox()%>
</td>
      <td width="20"><%
Response.Write ("<input type='radio' value='verwerfen' name='" + boxName + "'>")%>
</td>
      <td   width="83"><font   face="Arial"><!--webbot   bot="DatabaseResultColumn"
startspan
      s-column="DateRequest" local_preview="Datenbank: DateRequest"
      preview="Datenbank: DateRequest" s-ColumnNames --><%
If   Not   IsEmpty(fp_rs)   And   Not   (fp_rs   Is   Nothing)   Then   Response.Write
CStr(fp_rs("DateRequest"))
%>
<!--webbot bot="DatabaseResultColumn"
      endspan --></font></td>
      <td   width="83"><font   face="Arial"><!--webbot   bot="DatabaseResultColumn"
startspan
      s-column="Zustellung" local_preview="Datenbank: Zustellung"
      preview="Datenbank: Zustellung" s-ColumnNames --><%
If   Not   IsEmpty(fp_rs)   And   Not   (fp_rs   Is   Nothing)   Then   Response.Write
CStr(fp_rs("Zustellung"))
%>
<!--webbot bot="DatabaseResultColumn"
      endspan --></font></td>
      <td   width="83"><font   face="Arial"><!--webbot   bot="DatabaseResultColumn"
startspan
      s-
columnnames="Org,CommonName,Zustellung,RequestId,Status,StatusDatum,Cert_ausstell_d
atum"
      s-column="Org" b-tableformat="TRUE" clientside local_preview="Datenbank: Org"
      preview="Datenbank: Org" --><%
If   Not   IsEmpty(fp_rs)   And   Not   (fp_rs   Is   Nothing)   Then   Response.Write
CStr(fp_rs("Org"))
%>
<!--webbot bot="DatabaseResultColumn" i-checksum="25783"
      endspan -->  </font></td>
      <td   width="156"><font   face="Arial"><!--webbot   bot="DatabaseResultColumn"
startspan
      s-
columnnames="Org,CommonName,Zustellung,RequestId,Status,StatusDatum,Cert_ausstell_d
atum,OrgUnit"
      s-column="OrgUnit"  b-tableformat="TRUE"  clientside  local_preview="Datenbank:
OrgUnit"
      preview="Datenbank: OrgUnit" --><%
If   Not   IsEmpty(fp_rs)   And   Not   (fp_rs   Is   Nothing)   Then   Response.Write
CStr(fp_rs("OrgUnit"))
%>
<!--webbot bot="DatabaseResultColumn" i-checksum="26326"
      endspan --> </font></td>
      <td   width="122"><font   face="Arial"><!--webbot   bot="DatabaseResultColumn"
startspan
      s-
columnnames="Org,CommonName,Zustellung,RequestId,Status,StatusDatum,Cert_ausstell_d
atum,Anrede"
      s-column="Anrede"  b-tableformat="TRUE"  clientside  local_preview="Datenbank:
Anrede"
      preview="Datenbank: Anrede" --><%
If   Not   IsEmpty(fp_rs)   And   Not   (fp_rs   Is   Nothing)   Then   Response.Write
CStr(fp_rs("Anrede"))
%>
<!--webbot bot="DatabaseResultColumn" i-checksum="25569"
      endspan --></font></td>
```

```
      <td   width="122"><font   face="Arial"><!--webbot   bot="DatabaseResultColumn"
startspan
      s-
columnnames="Org,CommonName,Zustellung,RequestId,Status,StatusDatum,Cert_ausstell_d
atum"
      s-column="CommonName" b-tableformat="TRUE" clientside
      local_preview="Datenbank: CommonName" preview="Datenbank: CommonName" --><%
If   Not   IsEmpty(fp_rs)   And   Not   (fp_rs   Is   Nothing)   Then   Response.Write
CStr(fp_rs("CommonName"))
%>
<!--webbot
      bot="DatabaseResultColumn" i-checksum="38061" endspan --> </font></td>
      <td   width="124"><font   face="Arial"><!--webbot   bot="DatabaseResultColumn"
startspan
      s-
columnnames="Org,CommonName,Zustellung,RequestId,Status,StatusDatum,Cert_ausstell_d
atum,Street"
      s-column="Street"  b-tableformat="TRUE"  clientside  local_preview="Datenbank:
Street"
      preview="Datenbank: Street" --><%
If   Not   IsEmpty(fp_rs)   And   Not   (fp_rs   Is   Nothing)   Then   Response.Write
CStr(fp_rs("Street"))
%>
<!--webbot bot="DatabaseResultColumn" i-checksum="26591"
      endspan --> </font></td>
      <td   width="107"><font   face="Arial"><!--webbot   bot="DatabaseResultColumn"
startspan
      s-
columnnames="Org,CommonName,Zustellung,RequestId,Status,StatusDatum,Cert_ausstell_d
atum,Locality"
      s-column="Locality" b-tableformat="TRUE" clientside local_preview="Datenbank:
Locality"
      preview="Datenbank: Locality" --><%
If   Not   IsEmpty(fp_rs)   And   Not   (fp_rs   Is   Nothing)   Then   Response.Write
CStr(fp_rs("Locality"))
%>
<!--webbot bot="DatabaseResultColumn"
      i-checksum="28496" endspan --> </font></td>
      <td   width="140"><font   face="Arial"><!--webbot   bot="DatabaseResultColumn"
startspan
      s-
columnnames="Org,CommonName,Zustellung,RequestId,Status,StatusDatum,Cert_ausstell_d
atum,Email"
      s-column="Email"  b-tableformat="TRUE"  clientside  local_preview="Datenbank:
Email"
      preview="Datenbank: Email" --><%
If   Not   IsEmpty(fp_rs)   And   Not   (fp_rs   Is   Nothing)   Then   Response.Write
CStr(fp_rs("Email"))
%>
<!--webbot bot="DatabaseResultColumn" i-checksum="23454"
      endspan --> </font></td>
      <td   width="189"><font   face="Arial"><!--webbot   bot="DatabaseResultColumn"
startspan
      s-
columnnames="Org,CommonName,Zustellung,RequestId,Status,StatusDatum,Cert_ausstell_d
atum,MailzustellungErfolgreichAm,Phone"
      s-column="Phone"  b-tableformat="TRUE"  clientside  local_preview="Datenbank:
Phone"
      preview="Datenbank: Phone" --><%
If   Not   IsEmpty(fp_rs)   And   Not   (fp_rs   Is   Nothing)   Then   Response.Write
CStr(fp_rs("Phone"))
%>
<!--webbot bot="DatabaseResultColumn" i-checksum="25269"
      endspan --></font></td>
      <td   width="189"><font   face="Arial"><!--webbot   bot="DatabaseResultColumn"
startspan
```

```
      s-
columnnames="Org,CommonName,Zustellung,RequestId,Status,StatusDatum,Cert_ausstell_d
atum,Fax"
      s-column="Fax" b-tableformat="TRUE" clientside local_preview="Datenbank: Fax"
      preview="Datenbank: Fax" --><%
If  Not  IsEmpty(fp_rs)  And  Not  (fp_rs  Is  Nothing)  Then  Response.Write
CStr(fp_rs("Fax"))
%>
<!--webbot bot="DatabaseResultColumn" i-checksum="26583"
      endspan --> </font></td>
    </tr>
    <!--webbot bot="DatabaseRegionEnd" startspan b-tableformat="TRUE" local_preview
preview
    clientside tag="BODY" --><%
              ' Schleife zum Abarbeiten der Datensätze schließen.
              fp_rs.MoveNext
           Loop
       End If
       fp_rs.Close
   ' If-Anweisung schließen und auf Verbindungsfehler prüfen.
   End If
'  Die  If-Anweisung  schließen,  die  den  Code  zum  Überprüfen  von  Parserfehlern
enthält.
End If
set fp_rs = Nothing
%>
<!--webbot bot="DatabaseRegionEnd" i-checksum="40971" endspan
    -->

  </table>
<%
'Hier wird höchste Registrierungszahl übergeben, wird verwendet für Schlaufe
response.write ("<input type='hidden' name='höchstWert' value='" + maxWert + "'>")
%>
  <p><input type="submit" value="Verarbeiten" name="B1"> <input type="reset"
  value="Zurücksetzen" name="B2"></p>
</form>

<p><a href="#top"><font face="Arial">[Top]  </font></a></p>
</body>
</html>
```

File md_admin_pendent_suche.asp

```
'*************************************************************************
' ALLGEMEIN FUER GANZE SEITE GUELTIGKEIT
' Script:        JavaScript
' WebSeite:      adminweb\md_admin_pendent_suche.asp
' Author:        MediData AG, Bosshard Stefan
' Datum:         31.08.1998, V1.0
' Aenderungen:   --
' Beschreibung:  Suche von pendenten Zertifikaten (Status 1) und öffnen von
'                md_admin_pendent.asp
'*************************************************************************
```

File md_admin_verarbeitung.asp

```
<%
'*************************************************************************
' ALLGEMEIN FUER GANZE SEITE GUELTIGKEIT
' Script:        VBScript
' WebSeite:      adminweb\md_admin_verarbeitung.asp
' Author:        MediData AG, Bosshard Stefan
' Datum:         31.08.1998, V1.0
```

```
' Aenderungen:   --
' Beschreibung:  Verarbeitung von folgenden Aufgaben:
'                - Zertifikate freigeben Status 10
'                - Zertifikate verwerfen Status 20
'                - Zertifikate "wieder" freigeben
'                - Zertifikate löschen Status 99
'*****************************************************************************
%>
<html>

<head>
<title>Adminweb - Verarbeitung</title>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<meta name="GENERATOR" content="Microsoft FrontPage 3.0">
</head>

<body>
<!--#include virtual="admin/md_func_sub.asp"-->
<!--#include virtual="admin/md_warten.asp"-->
<%
'*****************************************************************************
' #include virtual="admin/md_func_sub.asp"
' #include virtual="admin/md_warten.asp"    //Zeigt GIF Animation - bitte warten...
' Beschreibung:  siehe File selber (alles beschrieben betr. Input/Output)
' Functionen und Subs:  func_*; sub_*; ini_* (ini beinhaltet Konstanten)
'*****************************************************************************

'*****************************************************************************
' Beschreibung:  Abfrage, welche Variante ausgeführt werden soll
'*****************************************************************************
dummydate = ini_datum("dd/mm/yy")
'Maximal Wert der höchsten Nummer (ID)
maxWert = CInt(request.form("höchstWert")  )
Dim    zaehler,    stringZaehler,    welcheZertifikate,    URLrequestID,    tempEmail,
tempCommonName
Dim tempAnrede, tempOrg, datum, tempMapping_Name, tempStatus
zaehler = 1
nZ = Chr(13) + Chr(10)  'Zeilenumbruch

while zaehler <= maxWert
    stringZaehler = CStr(zaehler)
    nummerRec = request.form(stringZaehler)
    Select Case nummerRec
        Case "pendent"
         'nichts verarbeitet, weiterhin pendent
        Case "freigabe"
         'Freigabe Function starten
         call freigabe(stringZaehler)
        Case "verwerfen"
         'Verwerfen Function starten
         call verwerfen(stringZaehler)
        Case "löschen"
         'löschen Function starten (revoken)
         call loeschen(stringZaehler)
        Case "aktiv_retour_abholen"
         'aktiv_retour_abholen Function starten
         call aktiv_retour_abholen(stringZaehler)
        Case Else
         'nichts
    End Select
    zaehler = zaehler + 1
wend
'Wenn Verarbeitung abgeschlossen soll Schlusstext erscheinen
call   sub_text_link("Verarbeitung   erfolgreich   abgeschlossen!   Details   siehe
Logfile.", "md_admin.asp")
```

```
'*****************************************************************************
' Sub freigabe(string nummer)
' Beschreibung:  Gibt Zertifikat frei und benachrichtigt Kunde.
' Input:         string nummer: RegistrierungsNr MS Access DB
' Output:        --
' Aktivität:     Status 1 auf 10 ändern. URL ID generieren. Mail an Kunde senden.
'                Logfile Eintrag.
'*****************************************************************************
Sub freigabe(nummer)
   'SQL Aufrufen und diverse Sachen abfüllen in temporäre Variablen für Mail etc.
   call sql_select("Select * from md_Certificate_Anfrage WHERE RegistrierungsNr = "
+ nummer, nummer)

   text = "Herzlichen Dank fuer Ihre Anfrage eines Client-Certificates" + nZ
   text = text + "vom " + datum + " fuer die Benuetzung unseres MediFrameOnline
Produktes." + nZ + nZ
   text = text + "1. Schritt - Das Zertifikat koennen Sie ab sofort mit folgender
URL:" + nZ
   text = text + ini_https_cert_abholung() + URLrequestID + nZ
   text = text + "uebers Internet beziehen. Das Zertifikat wird automatisch in
Ihren" + nZ
   text = text + "Browser installiert (Navigator oder Internet Explorer 3.0 oder
aktueller." + nZ + nZ
   text = text + "2. Schritt Testen des Certificates:" + nZ
   text = text + ini_cert_test() + URLrequestID + nZ + nZ
   text = text + "Zur Information: Das Zertifikat kann nur einmal abgeholt werden!"
+ nZ
   text = text + "Ausserdem kann das Zertifikat nur auf dem PC benutzt werden, " +
nZ
   text = text + "auf welchem es auch online installiert wird!" + nZ + nZ
   text = text + "Bei Fragen oder Unklarheiten stehen wir Ihnen gerne zur
Verfuegung" + nZ

   dummySendText = ini_mail_senden(tempEmail, tempCommonName, tempAnrede , text)
   'Hier wird Status 1 Request auf 10 Freigabe geändert, damit frei für Abholung
   call sub_sqlExecute("UPDATE md_Certificate_Anfrage SET Status = '10',
MailzustellungErfolgreichAm = '" + dummySendText + "', RequestID = '" +
URLrequestID + "' , StatusDatum = '" + dummydate + "' WHERE RegistrierungsNr = " +
nummer)
   call sub_logfileSchreiben(nummer & chr(09) & tempOrg & "-" & tempCommonName &
chr(09) & "Freigabe Zertifikat")
End Sub

'*****************************************************************************
' Sub verwerfen(string nummer)
' Beschreibung:  Verwirft Zertifikats Anfrage und benachrichtigt Kunde.
' Input:         string nummer: RegistrierungsNr MS Access DB
' Output:        --
' Aktivität:     Status 1 auf 20 ändern. Mail an Kunde senden.
'                Logfile Eintrag.
'*****************************************************************************
Sub verwerfen(nummer)
   'SQL Aufrufen und diverse Sachen abfüllen in temporäre Variablen für Mail etc.
   call sql_select("Select * from md_Certificate_Anfrage WHERE RegistrierungsNr = "
+ nummer, nummer)

   text = "Herzlichen Dank fuer Ihre Anfrage eines Client-Certificates" + nZ
   text = text + "vom " + datum + " fuer die Benuetzung unseres MediFrameOnline
Produktes." + nZ+ nZ
   text = text + "Leider koennen wir Ihnen kein Zertifikat ausstellen!" + nZ + nZ
   text = text + "Bei Fragen oder Unklarheiten stehen wir Ihnen gerne zur
Verfuegung" + nZ + nZ

   dummySendText = ini_mail_senden(tempEmail, tempCommonName, tempAnrede , text)

   'wenn "verwerfen"
```

```
   call sub_sqlExecute("UPDATE md_Certificate_Anfrage SET Status = '20',
MailzustellungErfolgreichAm = '" + dummySendText + "', StatusDatum = '" + dummydate
+ "' WHERE RegistrierungsNr = "  + nummer)
   call sub_logfileSchreiben(nummer & chr(09) & tempOrg & "-" & tempCommonName &
chr(09) & "Zertifikat verworfen")
End Sub

'*****************************************************************************
' Sub loeschen(string nummer)
' Beschreibung:  Löscht ausgestelltes bzw. bereits gemapptes Zertifikat und
'                benachrichtigt Kunde.
' Input:         string nummer: RegistrierungsNr MS Access DB
' Output:        --
' Aktivität:     Status 12 auf 99 ändern. Mail an Kunde senden.
'                Logfile Eintrag. Mapping löschen.
'*****************************************************************************
Sub loeschen(nummer)
   'SQL Aufrufen und diverse Sachen abfüllen in temporäre Variablen für Mail etc.
   call sql_select("Select * from md_Certificate_Anfrage WHERE RegistrierungsNr = "
+ nummer, nummer)

   text = "Ihr Client-Zertifikate muss leider aus gewissen Gruenden" + nZ
   text = text + "per sofort geloescht werden!" + nZ + nZ
   text = text + "Leider koennen wir Ihnen kein neues Zertifikat mehr ausstellen!"
+ nZ  + nZ
   text = text + "Bei Fragen oder Unklarheiten stehen wir Ihnen gerne zur
Verfuegung" + nZ

   dummySendText = ini_mail_senden(tempEmail, tempCommonName, tempAnrede , text)

   'wenn "löschen"
   call sub_sqlExecute("UPDATE md_Certificate_Anfrage SET Status = '99',
MailzustellungErfolgreichAm = '" + dummySendText + "', StatusDatum = '" + dummydate
+ "' WHERE RegistrierungsNr = "  + nummer)
   call sub_logfileSchreiben(nummer & chr(09) & tempOrg & "-" & tempCommonName &
chr(09) & "Zertifikat plus Mapping gelöscht (revoked)")
   'Dieser Aufruf löscht Mapping
   call sub_mappingLoeschen(tempMapping_Name, mid(URLrequestID, 2, 1))
End Sub

'*****************************************************************************
' Sub aktiv_retour_abholen(string nummer)
' Beschreibung:  Setzt ein bereits aktives oder gemapptes Zertifikat zurück
'                und benachrichtigt Kunde. Grund für diesen Vorfall wären Bsp.
'                falls Probleme auftraten beim Installieren beim Kunden oder falls
'                Zertifikat gelöscht etc.
' Input:         string nummer: RegistrierungsNr MS Access DB
' Output:        --
' Aktivität:     Status 11 bzw. 12 auf 10 ändern. URL ID generieren.
'                Mail an Kunde senden. Logfile Eintrag. Altes Mapping löschen.
'                ! Variablen müssen Global sein.
'*****************************************************************************
Sub aktiv_retour_abholen(nummer)
   'SQL Aufrufen und diverse Sachen abfüllen in temporäre Variablen für Mail etc.
   call sql_select("Select * from md_Certificate_Anfrage WHERE RegistrierungsNr = "
+ nummer, nummer)

   text = "Herzlichen Dank fuer Ihre Anfrage eines Client-Certificates" + nZ
   text = text + "vom " + datum + " fuer die Benuetzung unseres MediFrameOnline
Produktes." + nZ + nZ
   text = text + "1. Schritt - Das Zertifikat koennen Sie ab sofort mit folgender
URL:" + nZ
   text = text + ini_https_cert_abholung() + URLrequestID + nZ
   text = text + "uebers Internet beziehen. Das Zertifikat wird automatisch in
Ihren" + nZ
```

```
   text = text + "Browser installiert (Navigator oder Internet Explorer 3.0 oder
aktueller." + nZ + nZ
   text = text + "2. Schritt Testen des Certificates:" + nZ
   text = text + ini_cert_test() + URLrequestID + nZ + nZ
   text = text + "Zur Information: Das Zertifikat kann nur einmal abgeholt werden!"
+ nZ
   text = text + "Ausserdem kann das Zertifikat nur auf dem PC benutzt werden, " +
nZ
   text = text + "auf welchem es auch online installiert wird!" + nZ + nZ
   text = text + "Bei Fragen oder Unklarheiten stehen wir Ihnen gerne zur
Verfuegung" + nZ

   dummySendText = ini_mail_senden(tempEmail, tempCommonName, tempAnrede , text)

   'wenn "aktiv_retour_abholen"
   call sub_sqlExecute("UPDATE md_Certificate_Anfrage SET Status = '10',
MailzustellungErfolgreichAm = '" + dummySendText + "', RequestID = '" +
URLrequestID + "' , StatusDatum = '" + dummydate + "' WHERE RegistrierungsNr = " +
nummer)
   call sub_logfileSchreiben(nummer & chr(09) & tempOrg & "-" & tempCommonName &
chr(09) & "Zertifikat plus Mapping gelöscht plus neue Freigabe für erneute
Abholung.")

   if tempStatus = 12 then
      'Dieser Aufruf löscht Mapping. Nur bei 12, da 11 noch nicht gemappt!
      call sub_mappingLoeschen(tempMapping_Name, mid(URLrequestID, 2, 1))
   end if
End Sub

'*****************************************************************************
' Sub sql_select(string sqlString, integer dummyNummer)
' Beschreibung:  Datensatzinformationen zwischenspeichern.
' Input:         string sqlString:   SQL Statement für MS Access DB
'                integer dummyNummer:   für ComboBox Account Auswahl
' Output:        --
' Aktivität:     Liest bestimmte Feldinhalte dieses Datensatzes und speichert die
'                Daten in temp Variablen
'*****************************************************************************
Sub sql_select(sqlString, dummyNummer)
   fp_sQry = sqlString
   fp_sDefault = ""
   fp_sNoRecords = ""
   fp_iMaxRecords = 0
   fp_iTimeout = 0
   fp_iCurrent = 1
   fp_fError = False
   fp_bBlankField = False
   If fp_iTimeout <> 0 Then Server.ScriptTimeout = fp_iTimeout
   Do While (Not fp_fError) And (InStr(fp_iCurrent, fp_sQry, "%%") <> 0)
      ' öffnendes Anführungszeichen gefunden, schließendes Anführungszeichen wird
gesucht.
      fp_iStart = InStr(fp_iCurrent, fp_sQry, "%%")
      fp_iEnd = InStr(fp_iStart + 2, fp_sQry, "%%")
      If fp_iEnd = 0 Then
         fp_fError = True
         Response.Write  "<B>Datenbankbereichsfehler:  Parameterbegrenzer  passen
nicht zusammen</B>"
      Else
         fp_sField = Mid(fp_sQry, fp_iStart + 2, fp_iEnd - fp_iStart - 2)
         If Mid(fp_sField,1,1) = "%" Then
            fp_sWildcard = "%"
            fp_sField = Mid(fp_sField, 2)
         Else
            fp_sWildCard = ""
         End If
         fp_sValue = Request.Form(fp_sField)
```

```
          ' Prüfen, ob das angegebene Formularfeld tatsächlich existiert.
          If (len(fp_sValue) = 0) Then
             fp_iCurrentField = 1
             fp_bFoundField = False
             Do While (InStr(fp_iCurrentField, fp_sDefault, fp_sField) <> 0) _
                And Not fp_bFoundField
                fp_iCurrentField = InStr(fp_iCurrentField, fp_sDefault, fp_sField)
                fp_iStartField = InStr(fp_iCurrentField, fp_sDefault, "=")
                If fp_iStartField = fp_iCurrentField + len(fp_sField) Then
                   fp_iEndField = InStr(fp_iCurrentField, fp_sDefault, "&")
                   If (fp_iEndField = 0) Then fp_iEndField = len(fp_sDefault) + 1
                   fp_sValue = Mid(fp_sDefault, fp_iStartField+1, fp_iEndField-1)
                   fp_bFoundField = True
                Else
                   fp_iCurrentField = fp_iCurrentField + len(fp_sField) - 1
                End If
             Loop
          End If
          ' Das folgende Codestück sucht nach dem angegebenen Formularfeldnamen,
          ' und verdoppelt darin alle einfachen Anführungszeichen,
          '  damit  SQL  nicht  durch  das  Auftreten  von  nicht-paarweisen  einfachen
Anführungszeichen verwirrt wird.
          If (Mid(fp_sQry, fp_iStart - 1, 1) = """") Then
             fp_sValue = Replace(fp_sValue, """", """""")
          ElseIf (Mid(fp_sQry, fp_iStart - 1, 1) = "'") Then
             fp_sValue = Replace(fp_sValue, "'", "''")
          ElseIf Not IsNumeric(fp_sValue) Then
             fp_sValue = ""
          End If

          If (len(fp_sValue) = 0) Then fp_bBlankField = True

          fp_sQry = Left(fp_sQry, fp_iStart - 1) + fp_sWildCard + fp_sValue + _
             Right(fp_sQry, Len(fp_sQry) - fp_iEnd - 1)

          ' Die aktuelle Position so korrigieren, daß sie nach dem ersetzten Wert
liegt.
          fp_iCurrent = fp_iStart + Len(fp_sValue) + Len(fp_sWildCard)
       End If
    Loop
    If Not fp_fError Then
       ' Den Verbindungstext so verwenden, wie er im Assistenten eingegeben wurde.
       On Error Resume Next
       set fp_rs = CreateObject("ADODB.Recordset")
       If fp_iMaxRecords <> 0 Then fp_rs.MaxRecords = fp_iMaxRecords
       fp_rs.Open fp_sQry, ini_datenbank()
       If Err.Description <> "" Then
          Response.Write "<B>Datenbankfehler: " + Err.Description + "</B>"
          if fp_bBlankField Then
             Response.Write "  Eines oder mehrere Formularfelder waren leer."
          End If
       Else
          ' Den Fall prüfen, daß kein Datensatz vorhanden ist.
          If fp_rs.EOF And fp_rs.BOF Then
             Response.Write fp_sNoRecords
          Else
             ' While-Schleife zum Einlesen jedes Datensatzes aus der Ergebnismenge.
             Do Until fp_rs.EOF

                'Text abfüllen in Variablen für Weiterverwendung wie Mail etc.
                If Not IsEmpty(fp_rs) And Not (fp_rs Is Nothing) Then
                       tempStatus = fp_rs("Status")
                       tempEmail =  CStr(fp_rs("EMail"))
                       tempCommonName = CStr(fp_rs("CommonName"))
                       tempCommonName = func_replace_Umlaute(tempCommonName)
```

```
                        tempAnrede = fp_rs("Anrede")
                        Select Case tempAnrede
                           Case "Herr"
                              tempAnrede = "Sehr geehrter Herr " + tempCommonName
                           Case "Frau"
                              tempAnrede = "Sehr geeehrte Frau " + tempCommonName
                           Case Else
                              tempAnrede = "Sehr geeehrter Kunde " + tempCommonName
                        end select
                        tempOrg = fp_rs("Org")
                        tempOrg = func_replace_Umlaute(tempOrg)
                        datum =   CStr(fp_rs("DateRequest"))
                        tempMapping_Name = fp_rs("Mapping_Name")
                        'Hole hier den Ausgewählten Account Wert von DB Liste plus
restliches aus DB selber

   'https://srvwisweb/certsrv/certenroll/md_certificate.asp?cert=      //hier  wird
irgendeine Zahl, Alphanumerisch etc. ergänzt
                        'Format für cert=
                        '2 stellen = Accountzuordnung 01 ist MEDIDATA\SRVWISWEB_Test
                        '_
                        '2 stellen = Die ersten beiden Buchstaben des CommonName's
                        'x stellen = RegistrieungsNr
                        '2 stellen = Die ersten beiden Zahlen der TelefonNummer
                        '2 stellen = Die ersten beiden Buchstaben der Org
                        'Z = Zahl
                        'C = Buchstabe
                        '            ZZ_CCZZZZCCCC
                        'Beispiel    01_Bo1305Me
                        URLrequestID = fp_rs("RequestID")
                        If request.form(dummyNummer + "Account") then
                           URLrequestID  =  request.form(dummyNummer  +  "Account")  +
"_"    +    mid(tempCommonName,    1,    2)    +    CStr(fp_rs("RegistrierungsNr"))    +
mid(fp_rs("Phone"), 1, 2) + mid(tempOrg, 1,2)
                        End if
                End if
                ' Schleife zum Abarbeiten der Datensätze schließen.
                fp_rs.MoveNext
             Loop
          End If
          fp_rs.Close
       ' If-Anweisung schließen und auf Verbindungsfehler prüfen.
       End If
    '  Die  If-Anweisung  schließen,  die  den  Code  zum  Überprüfen  von  Parserfehlern
enthält.
    End If
    set fp_rs = Nothing
End Sub
%>
</body>
</html>
```

File md_admin_verworfene_retoursetzen.asp

```
'*************************************************************************
' ALLGEMEIN FUER GANZE SEITE GUELTIGKEIT
' Script:        VBScript
' WebSeite:      adminweb\md_admin_verworfene_retoursetzen.asp
' Author:        MediData AG, Bosshard Stefan
' Datum:         31.08.1998, V1.0
' Aenderungen:      --
' Beschreibung:  Verworfene Zertifikate freigeben.
'                (Status 20 auf 10 setzen) plus Mail an Kunde.
'*************************************************************************
```

File md_ini_speichern.asp

```
'*************************************************************************************
' ALLGEMEIN FUER GANZE SEITE GUELTIGKEIT
' Script:        VBScript
' WebSeite:      adminweb\md_ini_speichern.asp
' Author:        MediData AG, Bosshard Stefan
' Datum:         31.08.1998, V1.0
' Aenderungen:   --
' Beschreibung:  Request des Textareas des Formulares. Starten Sub Inifile
'                speichern und meldet das gespeichert!
'*************************************************************************************
```

File md_inifile_bearbeiten.asp

```
'*************************************************************************************
' ALLGEMEIN FUER GANZE SEITE GUELTIGKEIT
' Script:        VBScript
' WebSeite:      adminweb\md_inifile_bearbeiten.asp
' Author:        MediData AG, Bosshard Stefan
' Datum:         31.08.1998, V1.0
' Aenderungen:   --
' Beschreibung:  Liest zuerst aktuelle Daten aus Inifile.
'                In diesem Formular kann Text nach belieben geändert werden.
'                Speichern öffnet diese Seite: md_ini_speichern.asp und führt
'                Kopieren des bestehenden Files in *.old um und erzeugt neues
'                Files mit gleichem Namen!
'                Zurück macht Aenderungen rückgängig!
'*************************************************************************************
```

File md_logfile_ansehen.asp

```
'*************************************************************************************
' ALLGEMEIN FUER GANZE SEITE GUELTIGKEIT
' Script:        VBScript
' WebSeite:      adminweb\md_logfile_ansehen.asp
' Author:        MediData AG, Bosshard Stefan
' Datum:         31.08.1998, V1.0
' Aenderungen:   --
' Beschreibung:  Oeffnet Logfile von Server und listet dieses auf.
'*************************************************************************************
```

File md_logfile_archivieren.asp

```
'*************************************************************************************
' ALLGEMEIN FUER GANZE SEITE GUELTIGKEIT
' Script:        VBScript
' WebSeite:      adminweb\md_logfile_archivieren.asp
' Author:        MediData AG, Bosshard Stefan
' Datum:         31.08.1998, V1.0
' Aenderungen:   --
' Beschreibung:  Archiviert Logfile
'*************************************************************************************
```

15 Eidesstattliche Erklärung

Ich erkläre hiermit, dass ich die vorliegende Arbeit selbständig, ohne Mithilfe Dritter und nur unter Benützung der angegebenen Quellen verfasst habe, und dass ich ohne schriftliche Zustimmung der Schulleitung keine Kopien dieser Arbeit an Dritte aushändigen werde, ausgenommen an Personen, die mir wesentliche Informationen für die Diplomarbeit zur Verfügung gestellt haben.

Ort/Datum Unterschrift