Sicherung des Zugangs zu Netzwerkdiensten der Fachhochschule Düsseldorf

Peter Ophey

19.10.2005

Diplomarbeit

Fachhochschule Düsseldorf, Fachbereich Medien

Düsseldorf University of Applied Sciences, Department of Media

Betreuer: Prof. Dr. Günther Franke

Koprüfer: Dipl.-Ing. Ernst Schawohl

Zusammenfassung

Der Autor untersucht im Rahmen dieser Diplomarbeit die Sicherheit des Zugangs zu Netzwerkdiensten in der Fachhochschule Düsseldorf. Dabei werden sicherheitsrelevante Fragen wie Klartext Kommunikation bei der Authentifizierung und beim Transport der Daten über Netzwerkdienste beschrieben und Alternativen hierzu aufgezeigt.

Im Rahmen der Diplomarbeit wurde die Teilnahme der Fachhochschule Düsseldorf am Pilotprojekt DFN-PKI-2 für Public Key Infrastrukturen des Deutschen Forschungsnetzes initiiert und organisiert.

Weiterhin stellt der Autor die PKI-Technologie und deren praktische Einsatz anhand eines VPN Dienstes zur Sicherung der bestehenden Netzwerkdienste vor.

Danksagung

Meinen Dank möchte ich dem Team der DFN-PCA des Deutschen Forschungsnetzes aussprechen, das mich bei der Organisation zur Teilnahme der Fachhochschule Düsseldorf am Pilotprojekt DFN-PKI-2 unterstützt hat. Dies gilt insbesondere für Herrn Dipl.-Inform. Reimer Karlsen-Masur für die Hilfe bei der Erstellung der Policies der FH-D-CA, Herrn Dipl.-Inform. Jürgen Brauckmann für die Beantwortung technischer Fragen, sowie Herrn Dr. Marcus Pattloch.

Weiterhin bedanke ich mich beim Leiter der Datenverarbeitungszentrale der Fachhochschule Düsseldorf, Herrn Dipl.-Ing. Ernst Schawohl, der mir während der gesamten Zeit mit Informationen und Rat geholfen hat und dafür, dass er mir die Bearbeitung dieses Themas ermöglicht hat.

Letztlich bedanke ich mich bei meiner Familie und meiner Lebensgefährtin für die unablässige Unterstützung während des gesamten Studiums.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

1. Einleitung

In den Anfängen von Computerdatennetzen wurde dem Thema Sicherheit nur eine geringe Bedeutung beigemessen, denn sie wurden für die gemeinsame Nutzung von Ressourcen (Rechnerkapazitäten) entwickelt. Der Vorläufer des Internets, das ARPAnet, diente der Vernetzung von Universitäten und Forschungseinrichtungen in den USA, und wurde 1969 von der *Advanced Research Project Agency* in Betrieb genommen. Zu dieser Zeit ahnte niemand, dass 30 Jahre später Millionen von Menschen das Internet für den Zugang zu Informationen über das *World Wide Web*, online Banking und elektronisches Einkaufen benutzen würden. Die elementaren Protokolle für den Transport der Datenpakete im Internet, die TCP/IP Protokolle, wurden schon ab 1982 im ARPAnet eingesetzt. Mit der Zeit wurden Techniken von Firmen und Gremien entwickelt, um die Sicherheit von sensiblen Daten beim Transport über Datennetze zu gewährleisten. Beispiele hierfür sind Protokollfamilien wie IPsec und SSL. Die Sicherheit von Daten beim Transport über Datennetze kann allerdings nicht alleine durch Sicherheitsprotokolle auf einer Schicht im OSI-Referenzmodell erreicht werden. Alle Schichten, die beim Transport involviert sind, müssen berücksichtigt werden, angefangen mit der untersten, der physischen Bitübertragungsschicht, bis hin zur obersten, der Anwendungsschicht.[Ta03] Eine "angezapfte" Leitung beeinträchtigt die Sicherheit ebenso, wie ein durch einen Programmierfehler möglichen *Buffer-Overflow* in einer Anwendung. Diesem Umstand kann die vorliegenden Arbeit nicht in allen Punkten Rechnung tragen. Es wird also nicht die Sicherheit der eingesetzten Anwendungen und Betriebssysteme betrachtet. Vielmehr konzentriert sich die Arbeit des Autors auf kritische Bereiche wie Klartext Kommunikation bei den eingesetzten Netzwerkdiensten und Protokollen, sowie auf Möglichkeiten und Vorschläge, Probleme zu beheben.

Sicherheit in Datennetzen ist ein komplexes Thema und berührt nicht nur die technische Seite, sondern auch den Benutzer und dessen Interaktion mit Computernetzwerken. Die beste Technologie bringt keinen Gewinn an Sicherheit, wenn deren Benutzung für den Anwender als zu kompliziert empfunden und aufgrund dessen womöglich nicht eingesetzt wird. Auf der anderen Seite ist mit dem Einsatz von Technologien zur Verbesserung der Sicherheit auch immer ein Lernprozess seitens der Anwender verbunden. Damit die Anwender bereit sind sich auf diesen Prozess einzulassen, ist es nötig, sie von der Notwendigkeit des Einsatzes der Technik zu überzeugen.

1.1. Motivation

Die ersten Erfahrungen mit Computerdatennetzen machte der Autor in den 90er Jahren mit Homecomputern und Modemverbindungen zu lokalen Mailboxbetreibern, also lange vor der explosiven Verbreitung des Internets in der öffentlichen Wahrnehmung. In den ersten Monaten des Studiums machte der Autor auch Erfahrungen mit dem Internet, hauptsächlich mit dem *World-Wide-Web* zu Recherchezwecken für Vorlesungen und Projekte. Diese Projekte und der damit verbundenen Einsatz von Betriebssystemen wie Linux und IRIX förderte das Verständnis und das Interesse an den zugrunde liegenden Technologien von Computernetzwerken. So wurde in einem Projekt eine Videostreaminglösung entwickelt, die es ermöglichte MPEG4 Live-Videostreaming von der Düsseldorfer Messe "Boot" im Internet verfügbar zu machen. Dazu wurden neue Netzwerktechnologien wie *Wireless-LAN* eingesetzt, um den drahtlosen Transport der aufbereiteten AV-Datenströme zu ermöglichen, die wiederum an einen Streamingserver für die weitere Verbreitung weitergeleitet wurden. Eine parallel absolvierte Ausbildung zum *Cisco Certified Network Associate* intensivierte das Verständnis und das Interesse an Netzwerktechnologien.

Durch dieses Wissen erkannte der Autor jedoch auch die Sicherheitsproblematik beim Einsatz von Datennetzen. Dieses Thema beschäftigte ihn in der folgenden Zeit seines Studiums, zum Beispiel im Praxissemester, und es entstand der Wunsch die abschließende Diplomarbeit diesem Thema zu widmen. Der Besuch eines Workshops im Jahr 2004 an der Technischen Universität München, zum Thema *Public Key Infrastrukturen* von den Entwicklern der *Open Source* PKI Software OpenCA, gab dem endgültigen Thema weitere Konturen.

1.2. Gliederung der Folgekapitel

Die Arbeit umfasst im Wesentlichen drei Punkte:

- Das Kapitel 2 behandelt die Analyse der Dienste im Netzwerk der Fachhochschule Düsseldorf. Außerdem wird ein Ausblick auf *Selfservice* Funktionen über das Internet im Bereich der Hochschulverwaltungssoftware gegeben.
- Kapitel 3 bietet einen Einblick in die Technologie von Public Key Infrastrukturen, der Notwendigkeit von PKI bei der Nutzung von Protokollen wie SSL/TLS und IPsec, die Teilnahme der FH Düsseldorf am DFN-PKI-2 Pilotprojekt sowie die praktische Nutzung der DFN-PKI Software.

- Kapitel 4 behandelt das Thema *Virtual Private Networks*, um Netzwerkdienste zu sichern bzw. sie außerhalb der IP-Netze der Fachhochschule Düsseldorf nutzbar zu machen.

2. Analyse der Dienste

Um das Sicherheitsniveau der Netzwerkdienste im Netzwerk der Fachhochschule Düsseldorf zu erhöhen, ist zuvor eine Analyse der bestehenden IT Infrastruktur vorzunehmen. Die Analyse konzentriert sich auf den Betrieb der Dienste des FH Netzwerks und deren Nutzung von Studenten, Professoren und Angestellten der Fachhochschule. Hierbei liegt der Augenmerk auf sicherheitsrelevanten Fragen. Es geht hierbei allerdings nicht um eine Betrachtung der zugrunde liegenden Softwareprodukte und Betriebssysteme. Vielmehr sollen kritische Punkte wie Klartext Kommunikation bei der Authentifizierung angesprochen und Alternativen bzw. Maßnahmen zur Sicherung der betreffenden Dienste aufgezeigt werden.

Mit Diensten sind hier nicht nur Dienste im Kontext der Netzwerktechnik (z.B. Webserver) gemeint, sondern zum Beispiel auch die Authentifizierung für den Netzzugang (z.B. WLAN). Die Nutzung des Netzwerks ist auch als ein Dienst zu sehen und dementsprechend zu sichern. Auch die (zukünftigen) Dienste der Hochschulverwaltung sollen angesprochen werden, denn dort ist eine Entwicklung hin zu mehr "Selbstverwaltung für die Anwender" zu beobachten. Hier werden besonders sensible Daten von Netzwerkdiensten bearbeitet.

Ziel der Analyse ist es, -

1. - ein möglichst vollständiges Bild des derzeitigen "Dienstangebotes" des FH Netzwerks und dessen Sicherheit bei der Benutzung zu zeigen.
2. - Möglichkeiten zu zeigen, wie das Sicherheitsniveau verbessert werden kann und muss.
3. - einen Ausblick auf Veränderungen durch den Einsatz einer Public Key Infrastruktur (PKI) zu geben.

2.1. Zugang zum Netzwerk

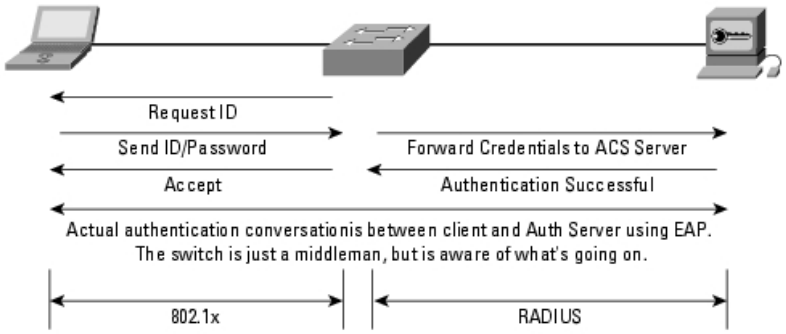
2.1.1. Kabelgebunden (LAN)

Beim Zugang zum kabelgebundenen Netz findet zur Zeit keine Authentifizierung über ein standardisiertes Protokoll wie das *Extensible Authentication Proto-*

col (EAP) statt. Die notwendigen Informationen für den Betrieb eines Rechners im Netzwerk der Fachhochschule wie IP-Adresse, Netzmaske, Gateway und DNS Server werden bei der schriftlichen Anmeldung durch die Datenverarbeitungszentrale (DVZ) vergeben und mitgeteilt. Es erfolgt keine automatische Konfiguration durch spezielle Dienste wie einen DHCP-Server. Es ist jedoch auch ohne Anmeldung möglich, das Netz zu nutzen, indem zum Beispiel mit Tools wie *ARPing*[1] auf der Netzzugangsschicht des TCP/IP Modells (hier also *Ethernet*) nach unbenutzten IP-Adressen geschaut wird. Der Zugang zu freien Netzwerkdosen und gute Netzwerkkenntnisse sind jedoch Voraussetzung, sodass dieser Fall eher zu vernachlässigen ist. Dem Autor sind bisher auch keine Werkzeuge/Tools bekannt, die das automatisieren.

Wünschenswert ist eine Authentifizierung der Person oder des Rechners über ein Protokoll der EAP Familie wie PEAP, EAP-TLS oder EAP-TTLS, wie sie in Abbildung 1 schematisch dargestellt ist. Diese Protokolle erfordern jedoch

Abbildung 1: Authentifizierung nach IEEE802.1x/EAP *(Quelle: Cisco)*



den Einsatz von X.509 Zertifikaten; im Fall von PEAP & EAP-TTLS "nur" ein Serverzertifikat auf dem Authentifizierungssystem, bei EAP-TLS zusätzlich auch Clientzertifikate auf jedem Rechner (höherer Aufwand). Der Einsatz von EAP erfordert in der Infrastruktur der Netzwerks den Einsatz von Hardware (Switches), die EAP unterstützen, und dies trifft nur auf den Kern des Netzes der Fachhochschule zu. Somit ist ein durchgehender Einsatz von EAP bis zum Einzelarbeitsplatz nicht möglich.

2.1.2. Drahtlos (WLAN)

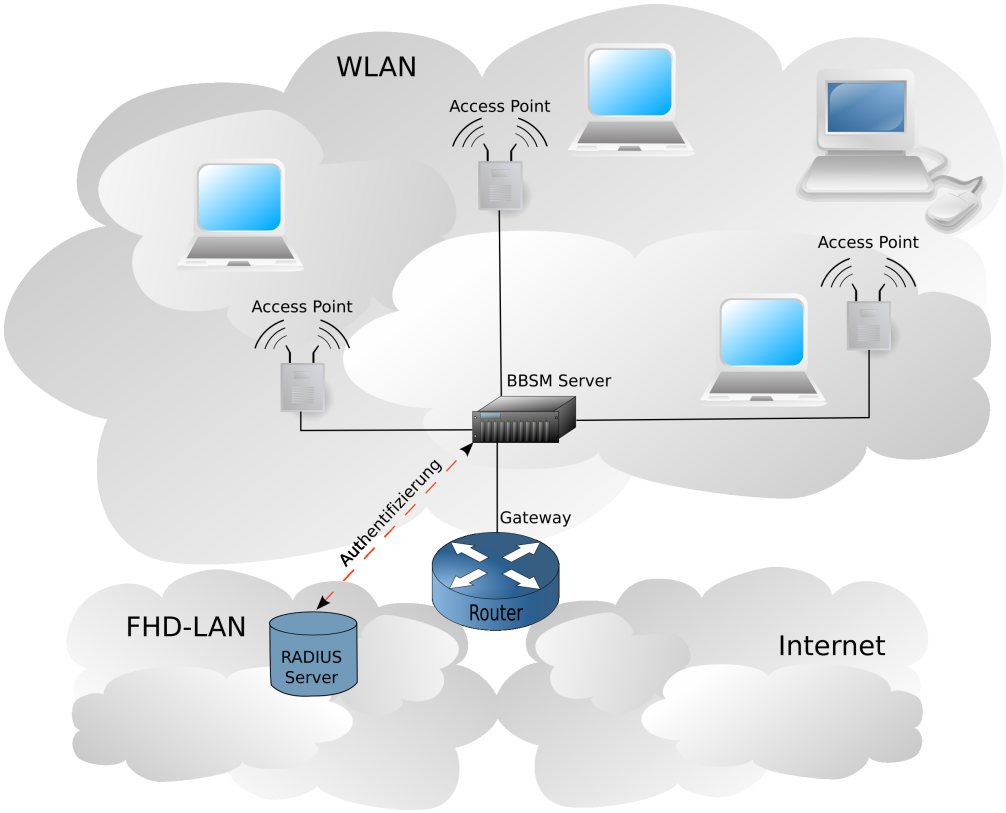
Eine besondere Bedeutung kommt der Sicherheit des drahtlosen Teils des Netzwerks der FH Düsseldorf zu, wenn man bedenkt, dass die Reichweite von IEEE-

[1] http://www.habets.pp.se/synscan/programs.php?prog=arping (Stand 15.09.05)

802.11-Netzen mehrere hundert Metern betragen kann. Ein Angreifer muss sich nicht in den Räumlichkeiten der FH aufhalten, um potenzielle Schwachstellen ausnutzen zu können.

Der Zugang zum FH Netzwerk über das WLAN ist über den proprietären *Building Broadband Service Manager Server (BBSM)* der Firma Cisco Systems gelöst und in der Abbildung 2 dargestellt. Auf der OSI-Schicht 2 (Abbildung 7)

Abbildung 2: WLAN mit BBSM-Server



wird keine Authentifizierung und keine Verschlüsselung (WEP/WPA/WPA2) nach IEEE 802.11 vorgenommen *(Open System Authentication)*. Der Rechner bekommt vom BBSM über das *Dynamic Host Configuration Protocol (DHCP)* eine IP Adresse aus dem Subnetz 195.37.237.0/25, Netzmaske, standard Gateway und DNS Server zugewiesen, kann das Gateway und damit den Zugang zum kabelgebundenen Teil des Netzwerks der Fachhochschule und zum Internet aber erst nach erfolgreicher Authentifizierung nutzen.

Im Folgenden wird die Sicherheit der Authentifizierung für den Zugang zum WLAN der Fachhochschule beschrieben und wie sich die Nutzung des WLAN durch den Betrieb des BBSM Servers für den Anwender darstellt. Daran schließen

sich Empfehlungen des Autors zu möglichen Alternativen für den Betrieb des drahtlosen Netzwerks der Fachhochschule Düsseldorf an.

Authentifizierung

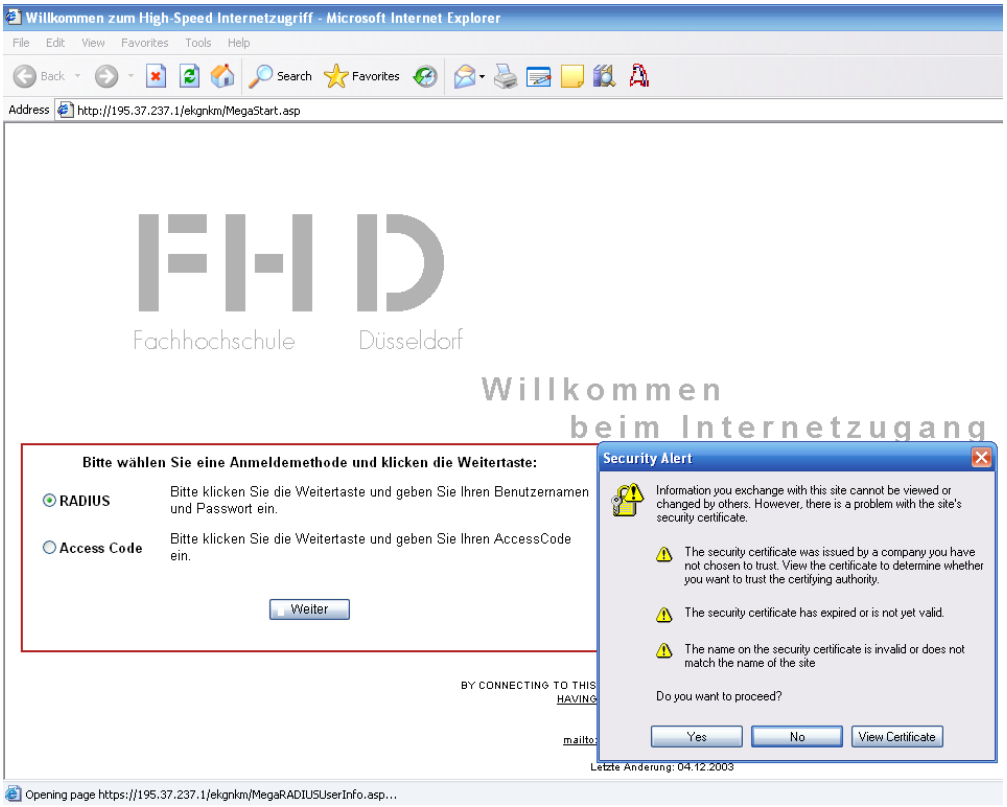
Nach Eingabe einer beliebigen URL im Webbrowser wird der User zur Anmeldung zum BBSM Webserver geleitet (*http-redirect)*, wo er Username und Passwort über ein HTML-Formular an den BBSM Server sendet. (Abbildung 3) Dieser überprüft die Eingaben über den DVZ internen RADIUS Server, und

Abbildung 3: WLAN Authentifizierung

nach erfolgreicher Authentifizierung erfolgt der Zugriff auf FH LAN und Internet. Die Übertragung von Username und Passwort zum BBSM Server erfolgen dabei SSL/TLS verschlüsselt. Der Webserver ist hierzu mit einem SSL-Server-Zertifikat ausgestattet. Dies erlaubt dem Anwender zum einen, den Server anhand seines Zertifikats zu identifizieren und zum anderen, dass seine Benutzereingaben mit dem *public key* des Servers verschlüsselt werden und nur der zum Zertifikat passende *private key* diese Daten entschlüsseln kann.

Zur Zeit wird ein abgelaufenes "Test-Zertifikat" *(selfsigned)* benutzt, welches der User beim Betreten der Webseite über einen Warnhinweis des Browsers prüfen und dann importieren sollte, es jedoch nicht ("manuell") überprüfen kann, weil kein *Fingerprint* (Fingerabdruck) veröffentlicht wurde. (Abbildung 4) Wird

Abbildung 4: Warnhinweise des Webbrowsers



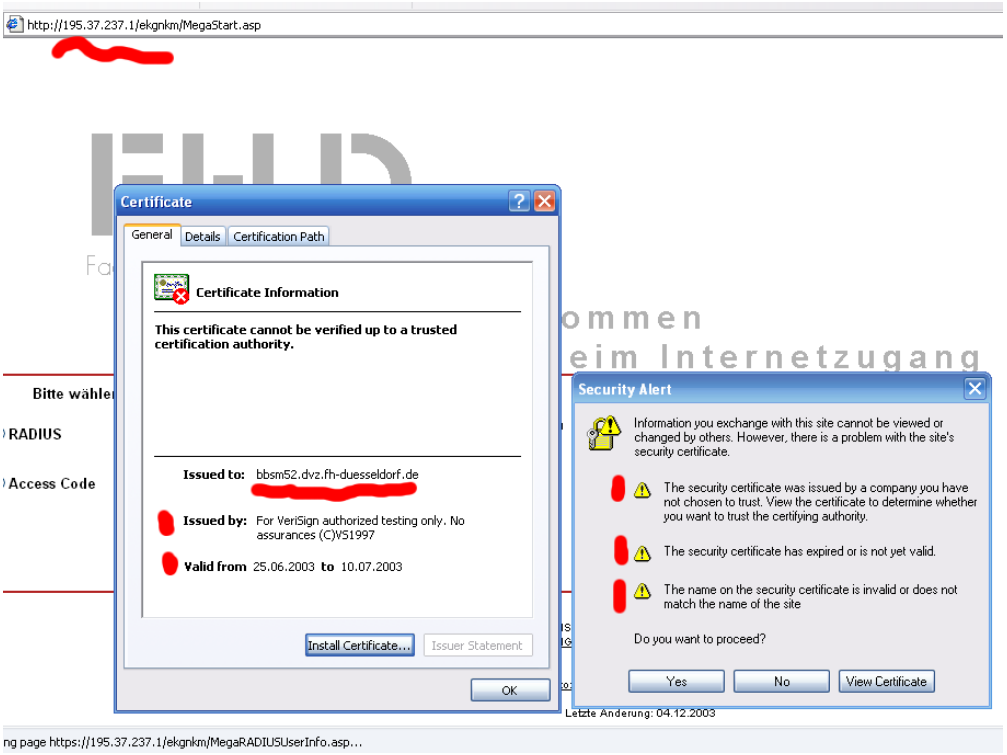
das Zertifikat nicht importiert, sondern nur temporär akzeptiert, dann erscheint dieser Dialog bei jeder erneuten Verbindung zum BBSM-Server.

Eine solche Warnung wird dem Anwender vom Browser angezeigt, wenn eine der folgenden Bedingungen NICHT zutrifft:

1. Das Zertifikat wurde von einer vertrauenswürdigen Zertifizierungsstelle ausgestellt.
2. Das Zertifikat ist zur Zeit gültig und ist nicht abgelaufen.
3. Der *Common Name (CN)* stimmt mit der HTTP Adresse überein, auf die der Browser zugreifen will.

Diese Bedingungen treffen ALLE nicht auf das zur Zeit verwendete Zertifikat des BBSM-Servers zu, was im Folgenden beschrieben wird und in Abbildung 5 zu sehen ist.

Abbildung 5: Details des SSL-Zertifikats des BBSM-Servers



1. Das Testzertifikat ist von einer Zertifizierungsstelle (CA) ausgestellt, welche vom Browser als nicht vertrauenswürdig eingestuft wird, weil das Wurzelzertifikat der CA nicht im Browser vorhanden ist. Das Zertifikat der CA ist auch nicht veröffentlicht, sodass es nicht vom Anwender in den Browser aufgenommen werden kann.

2. Das Zertifikat ist seit dem 10.07.2003 abgelaufen und damit nicht mehr gültig.

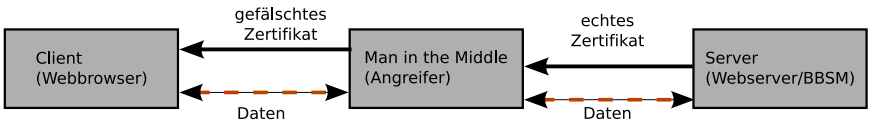
3. Das Zertifikat enthält im Attribut *Common Name* (CN) den DNS Namen des BBSM-Servers (bbsm52.dvz.fh-duesseldorf.de). Hier sollte jedoch die IP-Adresse (195.37.237.1) stehen, da der Webbrowser direkt über die IP-Adresse auf den BBSM-Server zugreift (siehe URL in den Abbildungen 4 & 5).

Die Sicherheit durch SSL/TLS wird durch den Einsatz von Testzertifikaten bzw. selbtsignierten Zertifikaten "verwässert", weil er bei den Anwendern für Verwirrung sorgt. Ein großer Teil der Anwender vertraut solch einem Zertifikat ohne dem Warnhinweis nachzugehen und zum Beispiel den Fingerabdruck zu prüfen und wiegt sich so in falscher Sicherheit. Diese Warnhinweise sind für den Endanwender schwer zu verstehen und nach Meinung des Autors müssten die Hersteller der Webbrowser, allen voran Microsoft und die *Mozilla Foundation*, die Implementierung solcher Fehlermeldungen überarbeiten. Es existieren aber auch bereits *Workarounds* auf der Seite des Servers, zum Beispiel für den weit verbreiteten Web-Server *Apache*, um individuelle HTML-Seiten in Abhängigkeit der SSL-Fehlermeldung anzeigen zu lassen und damit den Endanwender besser zu informieren.[Sm05]

Das oben beschriebene Fehlverhalten der Anwender lässt sich ausnutzen. Ein Angreifer kann eine entsprechende Website mit gefälschtem Zertifikat präparieren, um mit einem *Man-in-the-Middle (MitM)* Angriff (ARP, DNS & SSL) die eigentlich verschlüsselte Sitzung zwischen Client und Server zu belauschen, um an die User-Accounts zu kommen. Der Angriff ist recht aufwendig (*ARP-spoofing, DNS-spoofing,* usw.); es existieren aber Softwaretools wie *dsniff* oder *ettercap*, die einen Angriff extrem vereinfachen, wie er zum Beispiel in [Bp02] und [Dd01] beschrieben ist.

Der Angreifer ist dann im Besitz der DVZ internen RADIUS Accounts und hat damit gleichzeitig auch die Authentifizierungs-Informationen für andere wichtige Dienste wie Email (POP3/SMTP).

Abbildung 6: Schema eines SSL-Man-in-the-Middle Angriffs



Ein gültiges Zertifikat muss bei einem Dienstleister erworben werden oder bedarf des Betriebs einer eigenen *Public Key Infrastruktur* (PKI). Mit einem gültigen Zertifikat ist keine "manuelle" Prüfung des Zertifikats über einen Fingerprint nötig, und es erscheint kein Warnhinweis beim Besuch der Webseiten sofern das Zertifikat der CA installiert ist. Der Versuch ein Zertifikat zu fälschen, um einen *Man-in-the-Middle (MitM)* Angriff durchzuführen, fällt sofort auf, denn

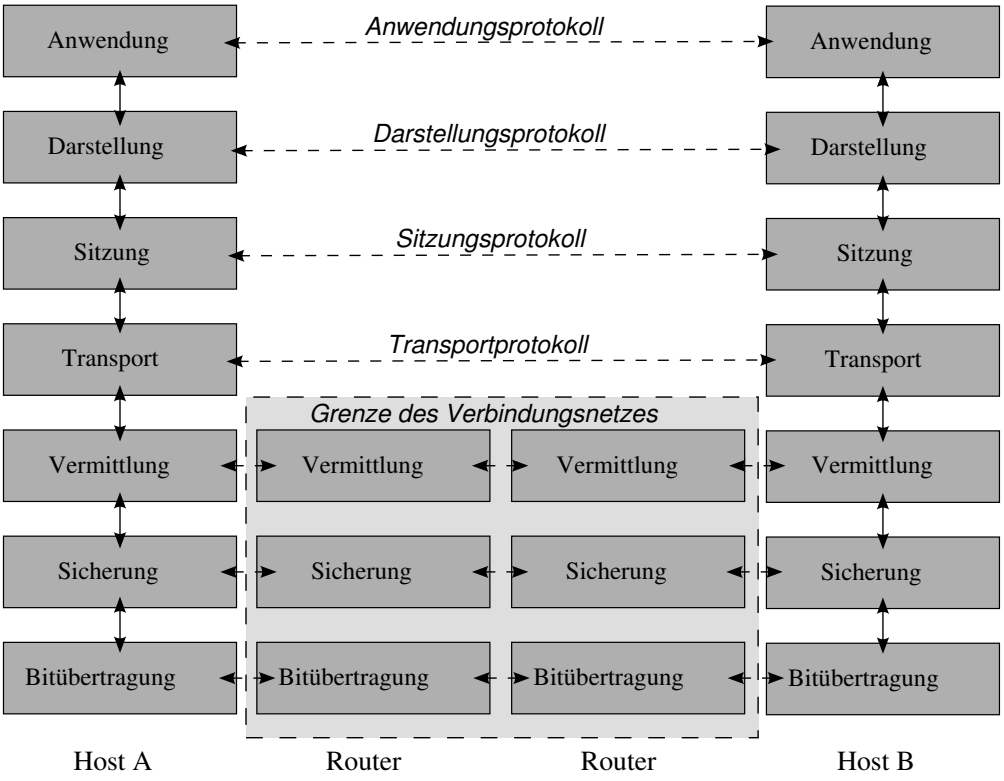
der User wird darauf hingewiesen, dass das Zertifikat nicht vertrauenswürdig ist bzw. gefälscht wurde.

Der BBSM-Server muss mit einem Zertifikat ausgestattet werden, welches den oben genannten Bedingungen entspricht, um von SSL/TLS zu profitieren.

Verschlüsselung

Wie eingangs erwähnt wurde, wird im WLAN keine Verschlüsselung auf der OSI-Schicht 2 (z.B. WEP/WPA/WPA2) genutzt. Dies bedeutet, dass sämtlicher

Abbildung 7: OSI-Schichtenmodell, Grafik nach[Ta03]



Datenverkehr zwischen Client und *Accesspoint* unverschlüsselt über das Medium Luft stattfindet und mitgehört werden kann. Ausgenommen davon sind natürlich Netzwerkverbindungen, bei denen auf eine verschlüsselte Verbindung in einer höheren Schicht im OSI-Modell zwischen Client und Server gesetzt wird, also zum Beispiel bei einem SSL-Webserver.

Die Verschlüsselung des Datenverkehrs auf der OSI-Schicht 2 mittels WEP bzw. WPA ist sehr problematisch. Die offensichtlichen Schwachstellen von WEP und dessen Implementierung des Verschlüsselungsalgorithmus RC4 haben diesen

Standard praktisch unbrauchbar gemacht. Dessen Nachfolger IEEE802.11i wurde im Jahr 2004 verabschiedet und setzt auf den *Advanced Encryption Standard* (AES) als Verschlüsselungsalgorithmus. IEEE802.11i gilt bisher als sicher, ist allerdings aufgrund von AES inkompatibel mit der bisherigen Hardware (Accesspoints, Netzwerkkarten). Aufgrund der späten Verabschiedung dieses Standards und der beschriebenen Inkompatibilitäten wurde von der Industrie *(Wi-Fi Alliance)* ein Pseudostandard namens *Wi-Fi Protected Access* (WPA) eingeführt, der eine Untermenge des damals kommenden Standards IEEE802.11i darstellt. Nach der Verabschiedung von IEEE802.11i im Juni 2004 wurde von der *Wi-Fi Alliance* ein Label namens WPA2 für standardkonforme Hardware herausgebracht. Die Verwirrung des Anwenders war perfekt! Die in der FH-D genutzten Accesspoints der Firma *Cisco* (Modell *Aironet 1200*) sind nicht kompatibel zu IEEE802.11i. Des Weiteren schließt man die Anwender aus, deren Hardware und/oder Software nur WEP oder WPA unterstützt.

Alternativen zum momentanen Betrieb

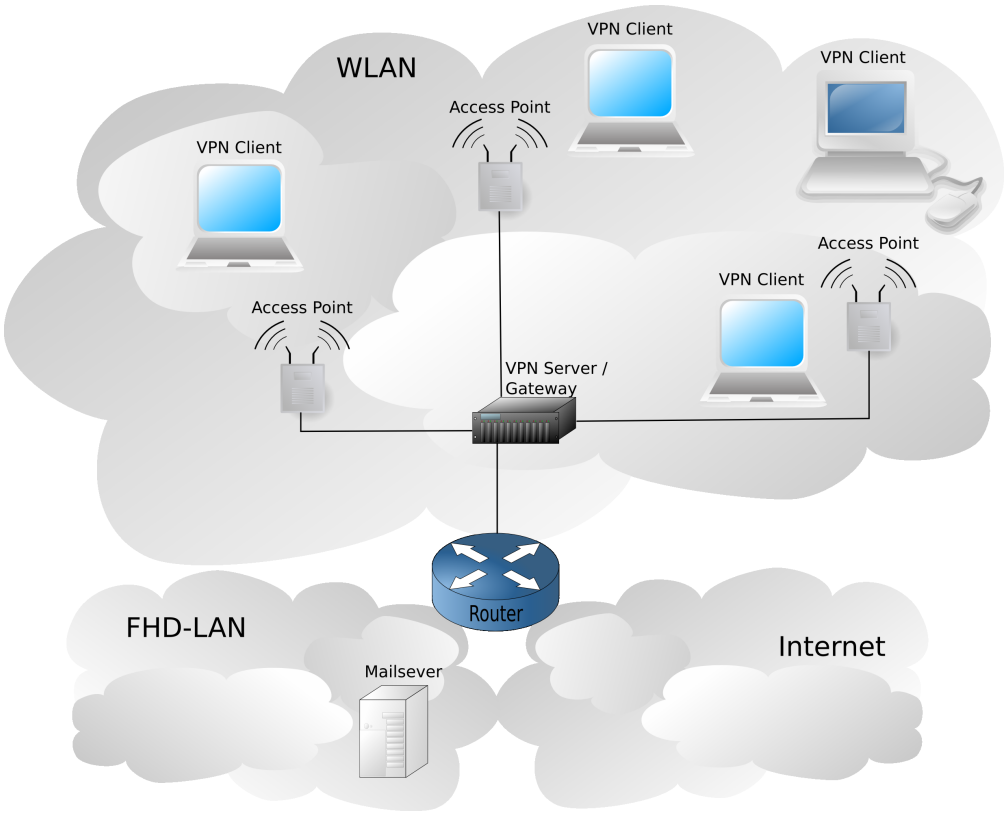
Die Verschlüsselung des Datenverkehrs und die Authentifizierung der Teilnehmer kann auch in einer höheren Schicht des OSI-Modells vorgenommen werden. Es kommen Protokolle wie *IPsec* auf der Netzwerkschicht oder *SSL/TLS* auf der Sitzungsschicht in Frage, die für den Einsatz in *Virtual Private Networks (VPN)* entworfen worden sind und den Einsatz von *X.509* Zertifikaten empfehlen. Vom Einsatz von *Pre-Shared Secrets* bei *IPsec* ist aufgrund von Problemen bei deren Wechsel und der Skalierung abzuraten.

Eine sicherere Alternative wäre es also, das WLAN weiterhin im 802.11 Modus *Open System* zu betreiben, jedoch als Gateway hinter den Accesspoints einen VPN-Server zu benutzen, der die Clients auf Basis von Benutzer-Zertifikaten authentifiziert und den Datenverkehr mit SSL bzw. IPsec verschlüsselt. (siehe Abbildung 8) Ein solches Szenario ist mit der im Abschnitt 4.2 verwendeten Software OpenVPN denkbar.

Weitere Aspekte

Eine zusätzliche Gefahr stellen Accesspoints dar, die ohne Wissen und Genehmigung der Datenverarbeitungszentrale (DVZ) betrieben werden. Diese sind unter Umständen falsch konfiguriert, sodass sie eine große Gefahr für die Netzwerksicherheit der Fachhochschule darstellen. Hier hilft nur die gezielte Analyse des Datenverkehrs zu "verdächtigen" IP Adressen, da man allein aufgrund der großen

Abbildung 8: WLAN mit VPN-Server



Fläche der Fachhochschule nicht periodisch den Äther nach den so genannten "wilden" Accesspoints absuchen kann. Ein Beispiel für einen solchen "wilden" Accesspoint ist ein im Fachbereich Design betriebener Accesspoint, der WEP-Verschlüsselung einsetzt, und dem Autor bei Recherchen zu dieser Arbeit in der Bibliothek auffiel.

Des Weiteren ist es möglich, normale WLAN-Netzwerkkarten im *Master-Mode* zu betreiben, welche dann einen Accesspoint darstellen, bei dem sich wiederum unerfahrene Anwender anmelden und dies unter Umständen nicht bemerken. Diese so genannten *rogue Accesspoints* stellen letztlich wiederum eine Variante des *Man-in-the-Middle* Angriffs dar, indem sie einen DHCP-Server betreiben, um dem "Client" eine (private) IP-Adresse zu geben und den Datenverkehr dann mittels *Network Address Translation (NAT)* in das Netzwerk der FH bzw. das Internet zu routen. Hier kann wiederum der gesamte Datenverkehr des Clients mitgehört werden. Dieser Angriff funktioniert allerdings nur dann, wenn die Feldstärke der Netzwerkkarte des Angreifers größer ist als die eines der fest installierten Accesspoints in der Fachhochschule.

2.2. Netzwerkdienste

Bei den einzelnen Netzwerkdiensten muss zwischen öffentlichen, nichtöffentlichen und internen Diensten unterschieden werden. Es kann allerdings nicht immer ganz zwischen öffentlich und nicht öffentlich differenziert werden, da Teilbereiche eines Servers vor öffentlichem Zugriff geschützt sein können. Daher sind einige Dienste in beiden Bereichen, öffentlich und nichtöffentlich, aufgeführt. Die Netzwerkdienste wurden unter anderem auf Klartext Kommunikation bei der Authentifizierung und Transport der Daten untersucht und in Tabellen zusammengefasst.

Der Hochschulverwaltung und deren (mögliche) Netzwerkdienste ist ein eigener Abschnitt (Abschnitt 2.3) gewidmet, da ein separates Netzwerk der Hochschulverwaltung am Standort betrieben wird. Dieses Netzwerk wird durch eine Firewall vom öffentlichen Netz der Fachhochschule und dem Internet getrennt und geschützt, und es werden derzeit (Mai 2005) keine Netzwerkdienste außerhalb dieses Netzwerks angeboten.

2.2.1. öffentliche Dienste

Öffentliche Dienste können ohne Authentifizierung von jedem User bzw. Rechner mit einer beliebigen IP Adresse in Anspruch genommen werden. Beispiele

sind die zentralen Webserver und der DNS Server der FH Düsseldorf. Da diese Dienste öffentlich sind und keine privaten oder schützenswerte Daten publizieren oder verwenden, bedurften sie auch keiner genaueren Betrachtung hinsichtlich Schwachstellen wie Klartext Kommunikation. Die öffentlichen Dienste sind in der Tabelle 1 aufgeführt.

Tabelle 1: öffentliche Dienste

DNS Name(n)	Typ/Dienst	Beschreibung	Status/Sicherheit
www.fh-duesseldorf.de & <xyz>.mki.fh-duesseldorf.de	Webserver	zentraler Webserver der FH-D	
www.et.fh-duesseldorf.de & ents01.et.fh-duesseldorf.de	Webserver	zentraler Webserver des FB Elektrotechnik	
www.medien.fh-duesseldorf.de & nux50.medien.fh-duesseldorf.de.	Webserver	zentraler Webserver des FB Medien	
vpn01.dvz.fh-duesseldorf.de	VPN-Server	VPN-Server der DVZ	nicht in Betrieb
www.bibl.fh-duesseldorf.de & opus.fh-duesseldorf.de & tww.fh-duesseldorf.de	Webserver	Webangebot der Bibliothek & Digitale Hochschulschriften & "alte" Webseiten der FH-D	nur teilweise öffentlich
svn.mki.fh-duesseldorf.de	SVN/ WebDAV/ Webserver	MKI Subversion Server (Versionsveraltung für Softwareprojekte)	teilweise öffentlich (Lesezugriff)

2.2.2. nichtöffentliche Dienste

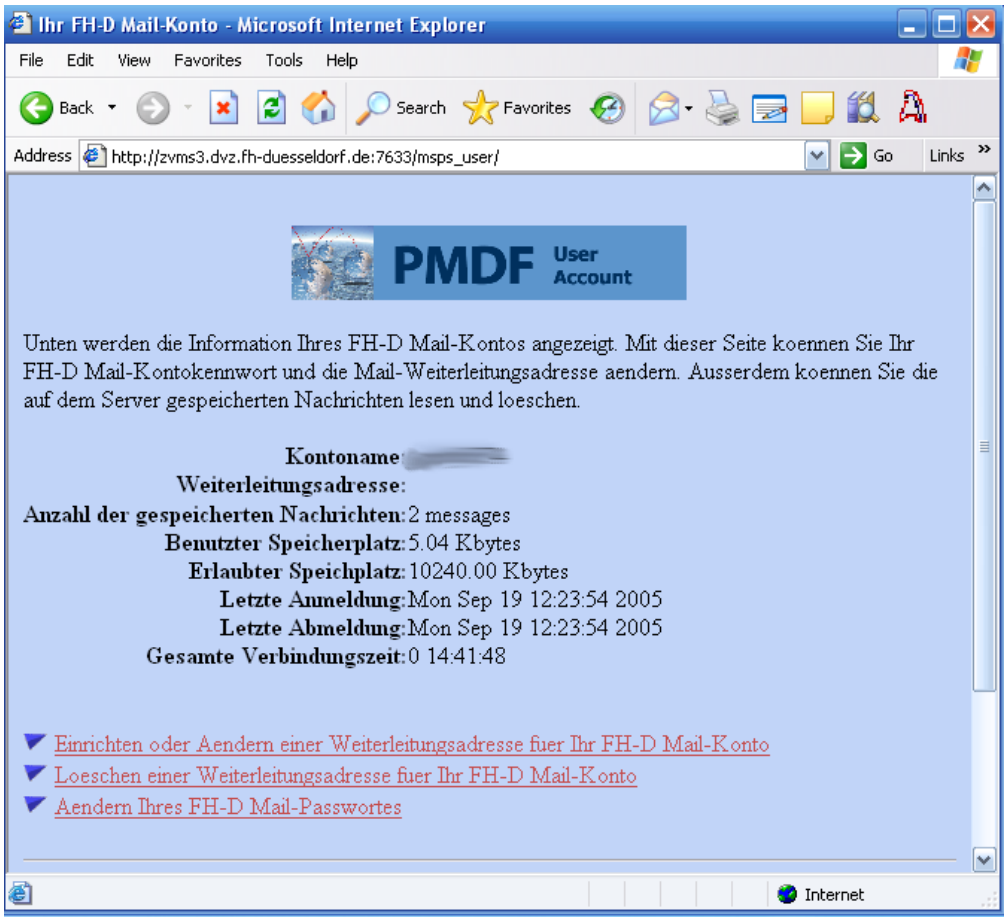
Nichtöffentliche Dienste sind nur begrenzten Gruppen zugänglich und können von jedem User bzw. Rechner mit einer beliebigen IP Adresse genutzt werden.

Emaildienste sind zum Beispiel nur FH Angehörigen (Studenten, Professoren und Mitarbeitern) zugänglich, die im DVZ internen RADIUS Server verzeichnet sind. Diese Gruppe von Diensten ist in der Tabelle 2 aufgeführt. Die Untersuchung hinsichtlich der Sicherheit wird für die wichtigsten Dienste im Folgenden beschrieben.

Emaildienste

Die Datenverarbeitungszentrale (DVZ) betreibt ihre Emaildienste auf einer Servergruppe *(Cluster)*. Die Angehörigen der Fachhochschule können diese Mailserver von jedem *Internet Service Provider* (ISP) nutzen, d.h. Emails vom POP3 Server abholen, per SMTP Server verschicken oder alternativ das Webinterface des Mailservers benutzen. Die Authentifizierungsdaten zur Anmeldung an den

Abbildung 9: Webinterface des Mailservers



Servern werden unverschlüsselt übertragen. Dies stellt eine große Gefahr dar, wenn ein Benutzer sich aus einem entfernten Netzwerk (z.B. WLAN Hotspot,

Firmennetzwerk, usw.) an diesen Mailservern anmeldet. Insbesondere die Nutzung aus einem *Wireless Network* (WLAN), zum Beispiel auch aus dem FH-WLAN *(DVZ-Hotpot)* ist gefährlich, da es hier ein Angreifer besonders einfach hat, sich mittels *ARP-Spoofing* als *Default Gateway* auszugeben und so sämtlichen Datenverkehr belauschen kann, inklusive den Zugangsdaten des Benutzers und dessen privaten Emails. Die Basis der Zugangsdaten wird auf dem zentralen RADIUS Server der DVZ gepflegt und dient nicht nur der Authentifizierung für den Emailservice, sondern zum Beispiel auch dem Zugang zum Wireless LAN der Fachhochschule Düsseldorf.

Wie bereits angedeutet wurde, bietet der Mailserver der Fachhochschule Düsseldorf alternativ die Benutzung des Emailpostfachs via Webinterface an. Die Problematik der Authentifizierung an webbasierten Diensten wird im folgenden Abschnitt 2.2.2 näher erläutert.

Die beste Alternative wäre es, neben Standard Protokollen wie POP3 und SMTP parallel auch deren verschlüsselnde Pendants, POPs und SMTPs anzubieten, die per *Secure Sockets Layer* (SSL) die Authentifizierung (Zugangsdaten) und die Daten (Emails) *End-to-End* verschlüsseln. Die Unterstützung für POPs und SMTPs ist in nahezu allen Emailclients (MUA) vorhanden und einfach umzustellen. Hierzu sind die Mailserver mit passenden X.509 Zertifikaten auszustatten. Zertifikate erfordern den Einsatz einer PKI oder müssen bei externen Dienstleistern erworben werden.

Eine andere Alternative wäre es, einen DVZ internen VPN-Server zu nutzen, der eine transparente Verbindung (Tunnel) zu den Mailservern herstellt. Somit könnten die Einstellungen in den Emailclients der Anwender beibehalten werden. Der sichere Tunnel für die sensiblen Daten wird nur bei Verbindung zum VPN-Server genutzt und ist für den Anwender nicht wahrnehmbar. In diesem Fall wären die Daten vom Client bis zum VPN-Server geschützt, nicht jedoch zwischen diesem und dem Mailserver. Diese Alternative wurde vom Autor zur Demonstration auf Basis der VPN-Software *OpenVPN* auf einem Server der DVZ implementiert und wird im Abschnitt 4.2 erläutert.

Authentifizierung an webbasierten Diensten (Webserver)

Im Netz der Fachhochschule wird bei einigen Diensten auf die Authentifizierung der Anwender durch den Webserver gesetzt. Hier gibt es neben der Authentifizierung auf Basis von X.509 Benutzerzertifikaten nach RFC2617 [RF99] zwei Methoden zur Authentifizierung auf Basis von Usernamen und Passwörtern:

- *HTTP Basic Authentication*
- *HTTP Digest Authentication*

Wie in Tabelle 2 bereits erwähnt, werden die Informationen zur Authentifizierung, also in der Regel Username und Passwort, bei der *HTTP Basic Authentication* unverschlüsselt (*Base64* kodiert) übertragen. Bei der Methode *HTTP Digest Authentication* werden diese Informationen mittels kryptographischer Hashfunktionen (MD5) übertragen. Aus dem mitgelesenen *Hashwert*, kann nicht auf die Ursprungsinformation geschlossen werden. *HTTP Digest Authentication* stellt also die sicherere Variante dar und wird von allen modernen Servern wie zum Beispiel *Apache* und Browsern wie dem Internet Explorer (ab Version 5) und Firefox unterstützt. Sie ist daher der *HTTP Basic Authentication* vorzuziehen und erfordert nur minmale Änderungen an der Konfiguration von Webservern.[AF05, Hm00]

Für besonders sensible bzw. schützenswerte Informationen, die auf Webservern zugänglich gemacht werden sollen (Verwaltung, Prüfungsleistungen, usw.) empfiehlt sich der Einsatz von zertifikatsbasierter Authentifizierung der Anwender. Der Vorteil hierbei ist, dass der Anwender im Besitz einer Datei (das Zertifikat) sein muss, um sich zu authentisieren. Eine Kombination aus Benutzername und Passwort kann dagegen leicht über das Netzwerk mitgelesen oder weitergegeben werden, zum Beispiel aus Bequemlichkeit der Anwender. Authentifizierung auf Basis von Zertifikaten wird zur Zeit zum Beispiel für den Zugriff auf die Webschnittstelle der Registrierungsstelle der PKI der FH Düsseldorf implementiert. Der Autor hat dieses Verfahren auch auf einem Test-Webserver in der Fachhochschule erfolgreich getestet und zeigt im Anhang A ein Beispiel für die Konfiguration des Webservers Apache mit *SSL-Client-Authentication*.

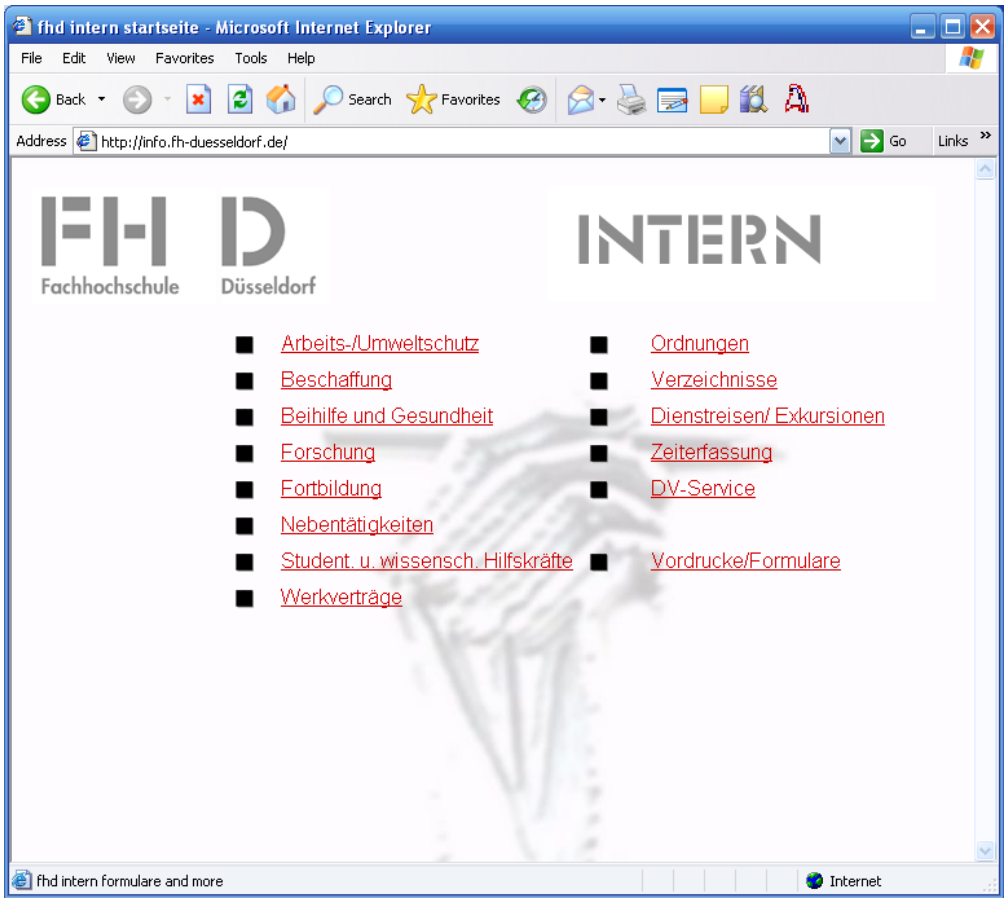
Tabelle 2: nichtöffentliche Dienste

DNS Name(:Port)	Typ/Dienst	Beschreibung	Status/Sicherheit
smtp.dvz.fh-duesseldorf.de	SMTP-Mailserver	zentraler Mailserver	Login&Daten unverschlüsselt
pop3.dvz.fh-duesseldorf.de	POP3-Mailserver	zentraler Mailserver	Login&Daten unverschlüsselt
zvms3.dvz.fh-duesseldorf.de:7633	Webserver	Webinterface des Mailservers	Login&Daten unverschlüsselt *(HTTP-Basic-Auth)*
bscw.dvz.fh-duesseldorf.de	BSCW Server	webbasiertes Wissensmanage-mentsystem	Login+Daten unverschlüsselt *(HTTP-Basic-Auth)*
alex.fh-duesseldorf.de	SSL-Webserver	E-Learning Plattform	Zertifikat der "alten" ServerCA der DFN-PCA. gültig bis 05/2006
bnts6.bibl.fh-duesseldorf.de	Webserver	Authentifizierung Benutzer (Bücher verlängern, vormerken)	Login&Daten werden unverschlüsselt übertragen (im Klartext über ein HTML-Formular!)
svn.mki.fh-duesseldorf.de	svn WebDAV Webserver	MKI Subversion Server	Login&Daten werden unverschlüsselt übertragen *(HTTP-Basic-Auth)*
weblogs.mki.fh-duesseldorf.de	CMS/-Webserver	Weblogs	Login wird unverschlüsselt übertragen (im Klartext über ein CGI-Script)
video.mki.fh-duesseldorf.de	Streaming-/Webserver	On-Demand/Live AV-Streaming	Login für Videoupload unklar (RTSP/FTP?)

2.2.3. interne Dienste

Hierzu zählen Dienste, die nur aus dem FH internen Netzwerk genutzt werden können und/oder einer begrenzten Gruppe zugänglich sind. Der Infoserver hält beispielsweise ein umfangreiches Angebot an Verwaltungsdokumenten und Antivirensoftware (Sophos Campuslizenz) für Angestellte der FH bereit (Abbildung 10).

Abbildung 10: Übersicht des Infoservers



Ein anderes Beispiel sind Managementsysteme für das Netzwerk der Fachhochschule, welche von DVZ internen Rechnern über eine SSL gesicherte Webschnittstelle administriert werden. Diese Webschnittstellen nutzen wiederum bisher keine oder selbstsignierte Zertifikate, und daher würden diese Systeme von Zertifikaten aus der FH eigenen Zertifizierungstelle profitieren.

Des Weiteren sind einige Dienste der Bibliothek aus lizensrechtlichen Gründen nur aus dem internen IP-Netz der Fachhochschule nutzbar (IP beschränkte Nutzung). Dies sind unter anderem:

- digitale Hochschulbibliothek (DigiBib): parallele Suche in Bibliothekskatalogen und Literaturdatenbanken, zum Beispiel Brockhaus Enzyklopädie und viele Duden Standardwerke (insgesamt ca. 120 Datenbanken)
- Elektronische Zeitschriftenbibliothek (EZB): ca. zwei- bis dreitausend(!) Volltextzeitschriften, zum Beispiel aus dem Springer Verlag, *Association for Computing Machinery* (zum Beispiel ACM SIGGRAPH Computer Graphics und ACM SIGCOMM Computer Communication Review) usw.

Dieses extrem umfangreiche Angebot der Bibliothek kann nur aus den FH internen IP-Netzen genutzt werden. Der Autor stellt in Kapitel 4.2 eine Möglichkeit vor, dieses Angebot auch für beliebige IP-Netzen verfügbar zu machen.

Die internen Dienste sind in Tabelle 3 aufgeführt.

Tabelle 3: interne Dienste

DNS Name/ WINS- Name	Typ/Dienst	Beschreibung	Status/Sicherheit
info.fh-duesseldorf.de	Webserver	Infoserver mit umfangreichem Angebot, z.B.Finanz- & Sachmittelverwaltung	Zugriff IP beschränkt
digilink.digibib.net bzw. cdroms.digibib.net	Webserver	Webserver mit Literaturdatenbanken	Zugriff teilweise IP beschränkt
rzblx1.uni-regensburg.de	Webserver	Elektronische Zeitschriftenbibliothek (EZB)	Zugriff teilweise IP beschränkt
helpdesk.dvz.fh-duesseldorf.de	?	?	?
193.23.168.181	WINS Server	WINS-Server der DVZ zur Namensauflösung im Windows Netzwerkprotokoll	
193.23.168.37 / \\ZNTS10	*Protokoll unbekannt*	Sophos Viren-Update-Server	SMB Protokoll notwendig, daher nur FH intern nutzbar
193.23.168.202 / \\ZNTS07	SUS Server (HTTP)	Windows-Update-Service (Service Packs und Patches für Microsoft Windows)	SMB Protokoll notwendig, daher nur FH intern nutzbar
	Control-SA	"Single-Point-Control-User-Management"	Management-system DVZ
	Patrol-Management	Serversysteme	Management-system DVZ
	Visualis	Netzwerk-management	Management-system DVZ
	Cisco Works	Netzwerk-management	Management-system DVZ

2.3. Hochschulverwaltung

2.3.1. Software der HIS GmbH

In der Hochschulverwaltung werden alle verwaltungsspezifischen Daten aufbereitet und verarbeitet. Hierzu dient unter anderem die modulare Software der HIS GmbH *(Hochschul-Informations-System GmbH)*. Derzeit werden folgende HIS Module in der Hochschulverwaltung der FH Düsseldorf eingesetzt:

- HIS-SOS Studentenverwaltung
- HIS-POS Prüfungsverwaltung
- HIS-ZUL Zulassungsverwaltung

Als Datenbasis dient eine *Informix* Datenbank. Die Verwaltung ist durch ein *Borderware* Firewallsystem vom Rest des FH Netzwerks und des Internets getrennt, um ausreichenden Schutz für die sensiblen Daten zu bieten.

Zur Zeit (Mai 2005) werden keine Verwaltungsdienste online bereitgestellt. Die HIS GmbH hat für so genannte *Self-Service-Funktionen-im-WWW* unter dem Namen HIS-QIS eine Reihe von Softwaremodulen entwickelt, die es ermöglichen ausgewählte Funktionen der vorhandenen HIS-Verwaltungsoftware über das Internet zu nutzen.

Zur Zeit gibt es folgende QIS Module:

- QIS Modul FSV: Finanz- und Sachmittelverwaltung
- QIS Modul POS: Prüfungsverwaltung (Anwendungen für Prüfer und Studenten)
- QIS Modul SOS: Studentenverwaltung
- QIS Modul ZUL: Internet-Einschreibung

Diese Module ermöglichen laut HIS Webseite zur Zeit folgende Funktionen:

- Semesterrückmeldung
- Adressänderung
- Ausdruck von Bescheinigungen

- Prüfungsanmeldung
- Notenverbuchung und Prüfungsstatistik
- Materialanforderungen und Anzeige von Kontoständen können sowohl von dezidierten Service-Terminals, die auf dem Unicampus aufgestellt und in das Verwaltungsnetzwerk eingebunden sind, als auch über das Internet mit einem der gängigen Webbrowser aufgerufen werden.

Abbildung 11: HISQIS Modul POS Prüfer *(Quelle: HIS)*

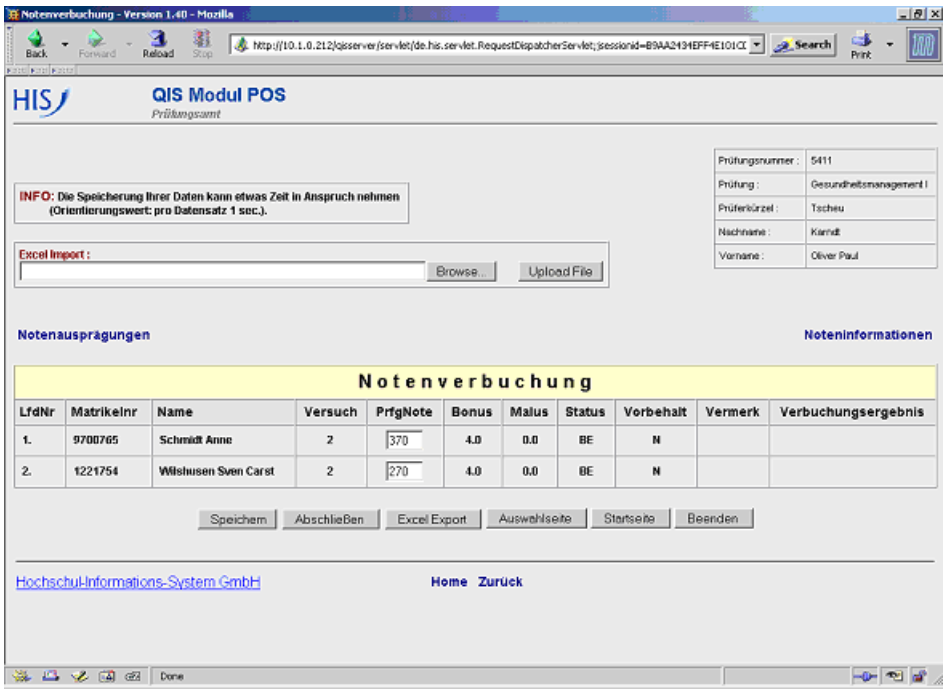
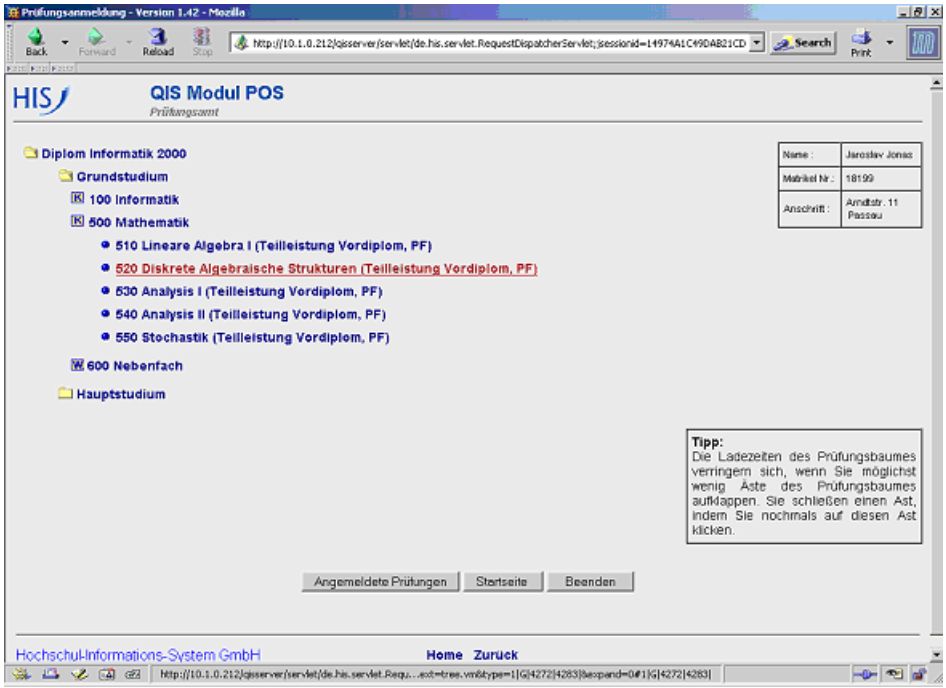


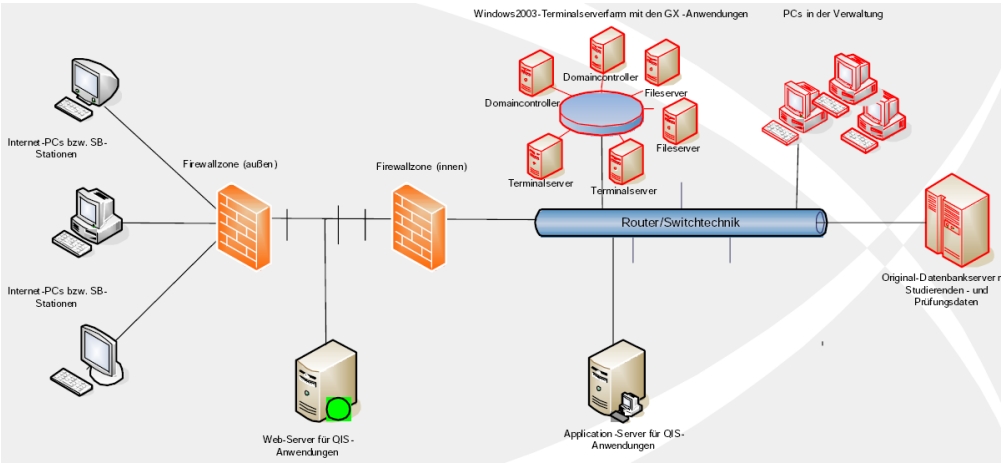
Abbildung 12: HISQIS Modul POS Student *(Quelle: HIS)*



2.3.2. Architektur von HIS-QIS

Die Basis für die QIS Module bilden Middlewareserver in der Verwaltung. Diese vermitteln die Anfragen eines Clients an den Datenbankserver. Diese Middlewareserver sind zum einen ein Webserver (z.B. Apache+modssl) und zum anderen ein Applikationsserver (Apache Tomcat + Java J2SE). Die Darstellung ist rein HTML basiert und kann somit ohne spezielle Software in fast jedem Browser genutzt werden. Der "vorgelagerte" Webserver übernimmt die SSL Verschlüsselung der Daten zum Client und muss zwingend mit einem SSL-Server Zertifikat ausgestattet werden. (Abbildung 13)

Abbildung 13: HIS-QIS Aufbau, hier mit *HISPRO (Quelle: DFN [Pm05])*



Derzeit nutzen 56 Hochschulen das QIS-POS Modul und 40 das QIS-SOS Modul. Der aktuelle Stand der Einführung von QIS Modulen in der Hochschullandschaft kann auf den Webseiten[2] [3] der HIS GmbH verfolgt werden.

2.3.3. Authentifizierung

Das QIS-Framework bietet Schnittstellen zur Nutzerauthentifizierung via LDAP, IMAP, RADIUS und Datenbanken an. Bisher wird zur Authentifizierung und für Transaktionen ein PIN/TAN Verfahren eingesetzt. Nach Aussagen von Mitarbeitern der HIS GmbH und Informationen aus dem Internet[4] hat die TU Ilmenau bereits ein Authentifizierungsverfahren auf Basis von X.509 Zertifikaten implementiert. Diese Funktionalität wird mit der neuen Version 8 der QIS-Module ausgeliefert werden können.

2.3.4. Open Source Alternativen

An dieser Stelle möchte ich auf die *CampusSource* Inititative hinweisen.

> "Ziel dieser vom Ministerium für Wissenschaft und Forschung des Landes Nordrhein-Westfalen unterstützten Initiative ist es, kooperative Prozesse für den Aufbau eines virtuellen Hochschulraums in Gang zu setzen."[5]

[2] http://www.his.de/Abt1/HISQIS/qisPOS/einfuehrungen (Stand 15.09.05)
[3] http://www.his.de/Abt1/HISQIS/qisSOS/einfuehrungen (Stand 15.09.05)
[4] http://www.tu-ilmenau.de/ca (Stand 15.09.05)
[5] [http://www.campussource.de/ (Stand 15.09.05)

Im Kontext der Hochschulverwaltung sind insbesondere die Projekte zu den Schnittstellen der HIS Systeme zu erwähnen, zum Beispiel das *Virtuelle Prüfungsamt - VirPa*[6] und das Projekt *SuperX*[7], sowie die große Auswahl an *Open Source* Software in weiteren Bereichen der Hochschulsoftware.

3. DFN-PKI

3.1. Public Key Infrastrukturen

Die Bemühungen der verantwortlichen Gremien, wie der *Internet Engineering Task Force* (IETF), die Internetsicherheit zu erhöhen, resultierten in Sicherheitsprotokollen wie *Secure Multipurpose Internet Mail Extensions* (S/MIME), *Transport Layer Security* (TLS) und *Internet Protocol Security* (IPSec). Diese Protokolle vertrauen alle auf *asymmetrischer Kryptographie* um Vertraulichkeit, Authentizität und Integrität zu ermöglichen. Eine *Public Key Infrastruktur* (PKI) bietet ein vertrauenswürdiges und effizientes Schlüssel- und Zertifikatsmanagement und ermöglicht erst dadurch eine effektive Nutzung dieser Protokolle. Eine PKI bildet also die Grundlage für die Anwendung dieser Protokolle in großen Firmen und Institutionen, indem sie Zertifikate für Mitarbeiter und Computersysteme erstellt und verwaltet. Diese Zertifikate können dann in unterschiedlichsten Anwendungen für Authentifizierung *(Authentication)*, Verschlüsselung *(Encryption)*, Datenintegrität *(Data Integrity)* und Verbindlichkeit *(Non-Repudiation)* genutzt werden.

Der bekannteste und gleichzeitig auch erfolgreichste Anwendungsbereich für den Einsatz von PKI ist *E-Business*, bei dem von der PKI serverseitige Zertifikate ausgestellt werden, um sichere (Geld-)Transaktionen zwischen *Client* (Käufer) und *Server* (Anbieter) über eine SSL-Sitzung zu ermöglichen.

Eine der wichtigsten Anwendungen, bei der eine PKI unverzichtbar ist, ist die Abwicklung von "Behördengängen" über das Internet. Stichwort ist hier die *elektronische Signatur* von Dokumenten wie der Einkommensteuererklärung. Die *elektronische Steuererklärung (Elster)* ist bereits heute in den Bundesländern Nordrhein-Westfalen, Bayern, Hessen und Sachsen im Rahmen eines Pilotprojekts möglich. Umfangreiche Informationen und die Registrierung zu *Elster* werden auf dem zugehörigen Internetportal[8] bereitgestellt.

[6] http://www.campussource.de/software/virpa/ (Stand 15.09.05)
[7] http://www.campussource.de/software/superx/ (Stand 15.09.05)
[8] https://www.elster.de/portal (Stand 15.09.05)

3.1.1. digitale Signaturen

Das Signaturgesetz der Bundesrepublik Deutschland behandelt die rechtliche Gleichstellung von elektronischen Signaturen mit der handschriftlichen Unterschrift. Vereinfacht wird zwischen drei Signaturformen unterschieden, die unterschiedliche Anforderungen erfüllen müssen:

1. einfache elektronische Signatur (niedrige Anforderungen)
2. fortgeschrittene elektronische Signatur
3. qualifizierte elektronische Signatur (höchste Anforderungen)

Die unterschiedlichen Sicherheitsniveaus bestimmen den Anwendungsbereich bzw. letztlich die Beweiskraft in einem juristischem Streitfall. Nur die qualifizierte Signatur eines elektronischen Dokuments (z.B. ein Kaufvertrag) entspricht weit gehend der handschriftlichen Signatur (Unterschrift) eines Dokuments in Papierform (Gleichstellung). Um Zertifikate für qualifizierte elektronische Signaturen ausstellen zu können, muss sich der *Zertifizierungsdienstanbieter* von der *Regulierungsbehörde für Telekommunikation und Post (RegTP)*, kürzlich umbenannt in *Bundesnetzagentur*, akkreditieren lassen. Für diese Akkreditierung wird das Sicherheitskonzept des Dienstanbieters durch die Bundesnetzagentur überprüft. Das Sicherheitskonzept muss die enorm hohen Anforderungen aus dem Signaturgesetz erfüllen. Weitere Informationen zum Signaturgesetz liefern die Webseiten[9] und Dokumente der Bundesnetzagentur.

> "Die fortgeschrittene Signatur unterscheidet sich formell kaum von der einfachen Signatur und rangiert mit dieser auf der untersten Stufe als "sonstiges" Signaturverfahren i. S. d. § 1 II SigG. Sie erfüllt - anders als u.a. die qualifizierten Signaturen- nicht die Anforderungen der gesetzlichen Schriftform gem. § 126a BGB oder § 3a VwVfG. Wohl aber erfüllt sie die Voraussetzungen gewillkürter Schriftform im Rahmen der §§ 127 II, III BGB (wobei jedoch der Empfänger der elektronischen Willenserklärung zur Sicherung der Beweiskraft gem. § 127 III BGB nachträglich eine qualifizierte Signatur oder die Ausstellung einer Urkunde verlangen kann). Es bleibt dennoch für die fortgeschrittene Signatur ein beachtlicher Anwendungsbereich

[9] http://www.bundesnetzagentur.de/enid/46409476d90b3069bd55961b7fde6d37,0/Technische_Regulierung_Telekommunikation/Elektronische_Signatur_gz.html (Stand 15.09.05)

> im Rahmen des Zivilrechts, wo vergleichsweise wenige Willenserklärungen dem gesetzlichen Schriftformerfordernis genügen müssen. Lediglich im öffentlichen Recht müssen für viele Anträge und Verwaltungsakte die Voraussetzungen des § 3a VwVfG erfüllt sein."[Ht04]

Der DFN verfolgt in seiner DFN-PKI Strategie mit seinen Angeboten DFN-PKI (1&2), "die klare Ausrichtung auf fortgeschrittene Zertifikate & Signaturen"[Pm04].

3.1.2. Grundfunktion von PKI

Die Grundfunktionen einer PKI auf Basis des X.509 Standards wurden durch die Arbeitsgruppe *Public Key Infrastructure X.509 (PKIX)* der *Internet Engineering Task Force (IETF)* im *PKIX-Modell* definiert. Im Einzelnen sind die wichtigsten Grundfunktionen einer PKI nach [ND01]:

- Registrierung
- Initialisierung
- Zertifizierung
- Wiederherstellung von Schlüsselpaaren
- Erzeugung von Schlüsseln
- Aktualisierung von Schlüsseln
- Gegenseitige Zertifizierung
- Sperrung von Zertifikaten
- Verteilung und Veröffentlichung von Zertifikaten & Zertifikatssperrlisten

In der Praxis wird oftmals nur eine Teilmenge der Grundfunktionen implementiert. Schlüsselerzeugung und die Wiederherstellung von Schlüsselpaaren werden beispielsweise benötigt, wenn die Schlüssel nicht "nur" für Signaturen, sondern auch zur Verschlüsselung von unternehmenseigenen/vertraulichen Daten benutzt werden. Im Fall einer Archivierung dieser verschlüsselten Daten muss das evtl. bereits zeitlich abgelaufene Schlüsselpaar wiederherstellbar sein, um die Daten zu einem späteren Zeitpunkt entschlüsseln und nutzen zu können. Ein anderes Beispiel ist das Ausscheiden eines Mitarbeiters aus dem Dienst oder Arbeitsverhältnis. Die verschlüsselten (unternehmenseigenen) Daten dieses

Mitarbeiters müssen natürlich auch weiterhin dem Unternehmen zu Verfügung stehen. Ebenso muss aber gewährleistet sein, dass der Mitarbeiter nach dessen Ausscheiden keinen Zugriff mehr auf die Daten erhält. An dieser Stelle wird deutlich, welcher Stellenwert dem Management von Schlüsseln und Zertifikaten in einer PKI beigemessen werden muss. Der Aufwand für Schlüsselerzeugung und Wiederherstellung von Schlüsselpaaren ist erheblich. Wird das Schlüsselpaar durch die PKI generiert, sei es zwecks Möglichkeit der Wiederherstellung des Schlüsselpaars oder mangels der Möglichkeit des Anwenders, dies selbst zu tun, so ergeben sich weitere Probleme:

- Wie kann der sichere Transport des Schlüsselpaars zum Anwender gewährleistet werden?
- Die PKI muss das Schlüsselpaar sicher vor unautorisiertem Zugriff archivieren.
- Der Anwender muss der PKI in den obigen beiden Punkten vertrauen.

3.1.3. Architektur von PKI und Interaktion der Komponenten

Laut [ND01] sind die Kernkomponenten einer PKI und deren Hauptfunktion:

- *Certificate Authority CA:* Bereitstellung der Zertifikate und der CRL

  Die CA (Zertifizierungsstelle) bildet den Kern einer jeden PKI. Sie signiert mit dem CA-Zertifikat die auszustellenden Zertifikate für Benutzer und Systeme.

- *Registration Authority RA:* Beantragung von Zertifikaten; Prüfung von Anträgen

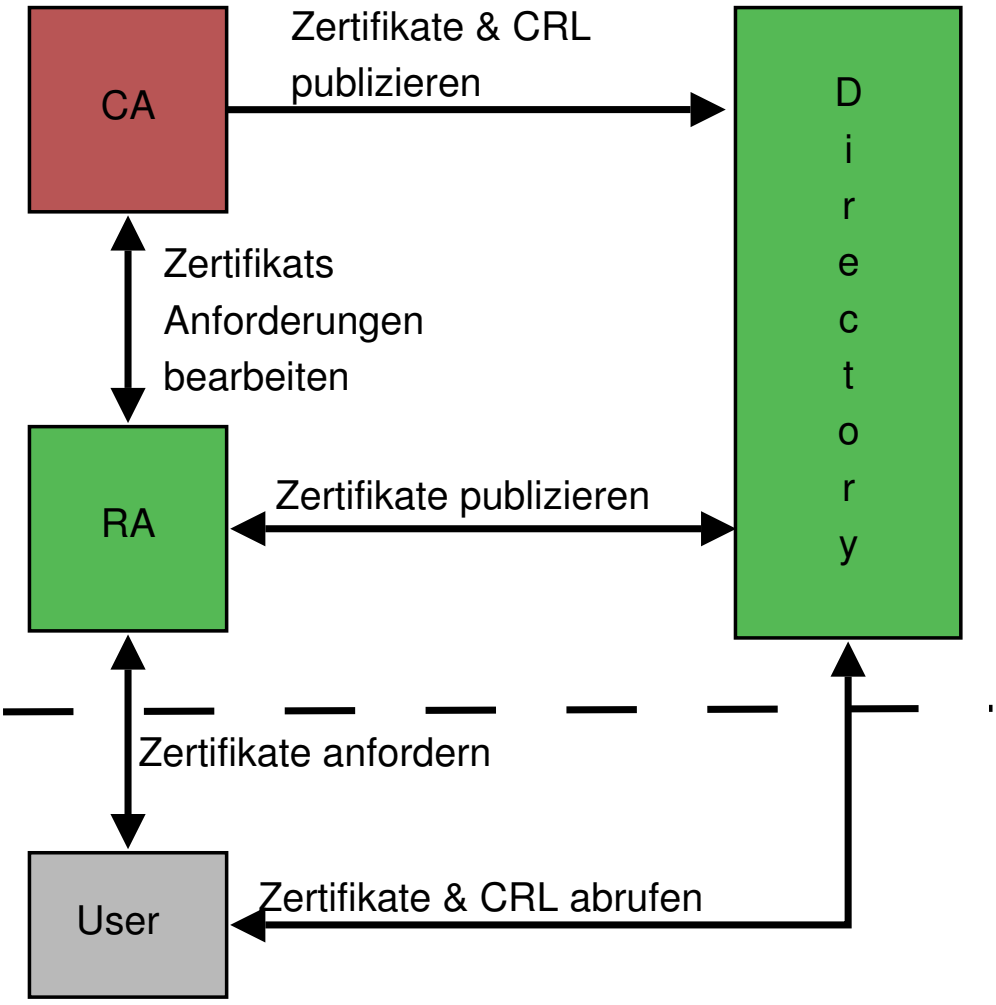
  Die RA (Registrierungsstelle) interagiert mit der CA und dem Benutzer. Sie nimmt Zertifizierungsanträge (*Certificate Signing Requests - CSR)* entgegen, prüft die Identität des Benutzers und reicht den Antrag an die CA weiter. Damit stellt die RA eine beglaubigte Verbindung der Identität des Benutzers und dessen *Public Key* her, die nach der Signatur durch die CA in Form eines Zertifikats dem Benutzer zugestellt wird.

- *Directory Service:* Ein öffentliches Verzeichnis mit ausgestellten, gesperrten bzw. ungültigen Zertifikaten, meist durch das *Leightweight Directory Access Protocol* (LDAP) implementiert.

- *Certificate Revocation List CRL:* Eine von der CA signierte Liste mit widerrufenen Zertifikaten, die in festen Intervallen von der CA veröffentlicht wird.
- *User/Client:* Benutzer oder Systeme

Die Interaktion dieser Komponenten ist in der Abbildung 14 dargestellt. Die Abbildung zeigt jedoch nur EINE Möglichkeit der Interaktion der Komponenten einer PKI. Vereinfacht kann man die Interaktion wie folgt beschreiben:

Abbildung 14: PKI Komponenten



Die Hauptaufgabe einer PKI ist die Erzeugung von Zertifikaten. Diesen Vorgang kann man in vier Stufen einteilen:

1. Antrag

2. Identifizierung
3. Zertifizierung
4. Publizierung

Die wichtigste Komponente einer PKI ist die CA. Diese wird meist auf einem stark gesicherten System betrieben, das zum Schutz der Integrität nicht mit einem öffentlichen Netz verbunden ist *(offline CA)*. Die ausgestellten Zertifikate und andere wichtige Informationen müssen jedoch auf einen öffentlich erreichbaren Rechner mit dem *Directory Service* übertragen werden, um sie den Anwendern zur Verfügung zu stellen. Die Schnittstelle zwischen CA und Benutzer repräsentieren die RA und der *Directory Service*. Die RA nimmt die Zertifizierungsanträge der Benutzer entgegen, prüft deren Identität anhand eines offiziellen Dokuments (Personalausweis, Studentenausweis,..) und reicht den Antrag an die CA weiter, die daraufhin die persönlichen Daten und den öffentlichen Schlüssel des Benutzers in einem X.509 Zertifikat mit dem öffentlichen Schlüssel der CA signiert. Für die Überprüfung der Identität muss der Anwender je nach *Policy* meist persönlich bei der Registrierungsstelle erscheinen. Das ausgestellte Zertifikat wird im Verzeichnis veröffentlicht und kann dem Anwender per Email zugestellt werden.

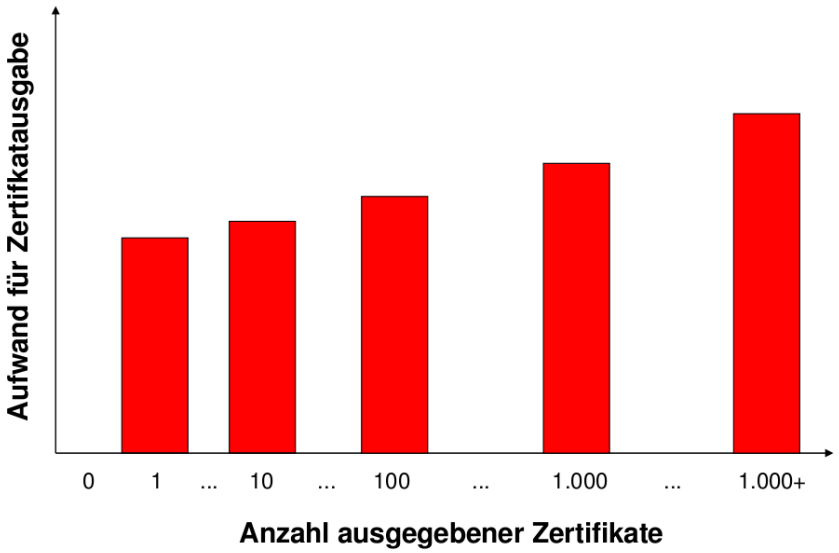
Bei der RA kann der Anwender auch sein Zertifikat zurückrufen (*Revocation*), beispielsweise wenn der private Schlüssel kompromittiert worden ist. Der Rückrufantrag wird wiederum durch die RA geprüft, und daraufhin wird das Zertifikat von der CA gesperrt und in die *Certificate Revocation List* aufgenommen.

3.1.4. Betrieb einer PKI

Der Betrieb einer PKI erfordert einigen administrativen und damit verbundenen finanziellen Aufwand. Hardware und Software müssen bereitgestellt, administriert und geschützt werden. Insbesondere sind die Einstiegshürden sehr hoch (siehe Abbildung 15). Für kleine und mittelgroße Einrichtungen/Unternehmen ist dieser Aufwand oftmals zu groß, und so werden benötigte (Server-)Zertifikate meist bei externen Dienstleistern eingekauft.

Da Zertifikate meistens nur eine begrenzte Gültigkeitsdauer besitzen (für SSL-Webserver typischerweise 2 bis 3 Jahre) ist der finanzielle Aufwand in regelmäßigen Intervallen an den Dienstleister zu entrichten. Für ein einzelnes, 2 Jahre gültiges SSL-Server-Zertifikat für einen Webserver sind bei führenden Dienst-

Abbildung 15: typischer Aufwand für den Betrieb einer PKI *(Quelle: DFN [Pm05])*



leistern, wie *Thawte*[10] oder *Verisign*[11] Beträge zwischen 350 Euro und 1600 Euro zu entrichten. Die Kosten für eine größere Anzahl von Zertifikate als auch die Erneuerung eines Zertifikates bei Ablauf des Gültigkeitszeitraums sind geringer. Häufig werden die Zertifikate von *Resellern,* im Rahmen eines "kompletten" Dienstpaketes, neben den "reinen Serverkapazitäten" eingekauft. Damit werden die Zertifikate für kleine und mittelständische Unternehmen erst bezahlbar.

Kleine und mittelständische Unternehmen haben aber auch aufgrund einer anderen Tatsache keine andere Wahl, als Zertifikate bei einem der "führenden" Dienstleister zu erwerben. Der Grund hierfür ist die Integration der Wurzelzertifikate der Zertifizierungstellen in die ausgelieferten Webbrowser der Kunden, zumeist der *Mircrosoft Internet Explorer*. Die Wurzelzertifikate sind nötig, um das SSL-Serverzertifikat eines Webservers automatisch auf dessen Gültigkeit überprüfen zu können. Vereinfacht läuft eine solche SSL-Sitzung wie folgt ab:

1. Der Webbrowser fordert ein Dokument von einem SSL-Webserver an.
2. Der Webserver sendet sein SSL-Server Zertifikat zum Browser.
3. Der Browser überprüft, ob das Zertifikat von einer ihm bekannten/installierten Zertifizierungsstelle (CA) ausgestellt wurde.

[10] http://www.thawte.com/ssl-digital-certificates/ssl/index.html (Stand 15.09.05)
[11] http://www.verisign.de/products-services/security-services/ssl/buy-ssl-certificates/compare/index.htm (Stand 15.09.05)

4. Der Browser prüft die im Zertifikat enthaltenen Informationen (Signatur des öffentlichen Schlüssels des Servers durch die Wurzel CA, den Zeitraum der Gültigkeit mit aktuellem Datum, Domainnamen des Servers mit *Common Name CN* und evtl. die CRL der CA).

Trifft der Punkt 3 oder eine der Bedingungen aus Punkt 4 nicht zu, so wird der Anwender durch den Webbrowser gewarnt. Dies schreckt potenzielle Kunden eines Angebots ab und sollte daher vom Anbieter des Angebots verhindert werden. Er kann dies aber nur vermeiden, indem er ein Zertifikat von einem der Dienstleister mit vorinstalliertem CA-Zertifikat erwirbt und damit seinen Webserver betreibt.

Die vorinstallierten Wurzelzertifikate begründen also die monopolartige Stellung einiger Dienstleister wie *Verisign* oder *Geotrust* im PKI-Business, und die Hersteller von Webbrowsern verdienen mit deren Integration in ihr Produkt Geld.

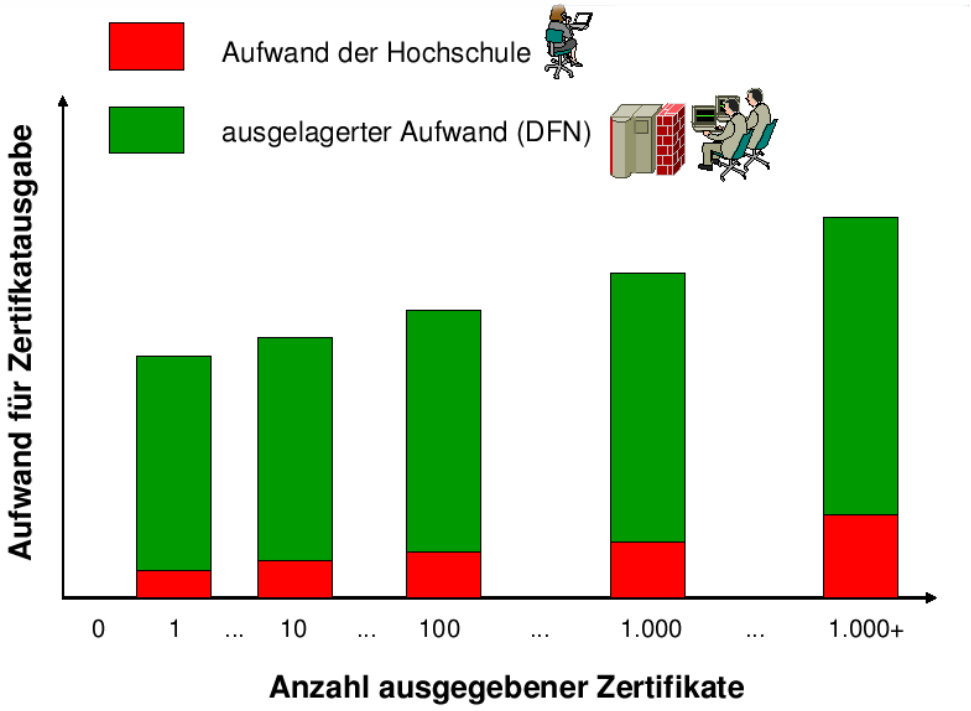
3.1.5. Auslagerung von Komponenten

Eine andere Möglichkeit ist die Auslagerung von Komponenten der PKI, die einen hohen administrativen oder finanziellen Aufwand erfordern. So kann beispielsweise die *Certificate Authority (CA)* und der Verzeichnisdienst ausgelagert werden. Auch die *Registration Authority (RA)* kann physisch (Hardware) ausgelagert werden. Dies ermöglicht eine Trennung von technischen, administrativen Aufgaben der CA und organisatorischen Aufgaben der RA. Die Zertifikatsanträge und die Identität der Antragsteller werden lokal im Unternehmen geprüft und bearbeitet; die Zertifikate werden von der CA am Ort des Dienstleisters erstellt und verwaltet. Damit wird der administrative und finanzielle Aufwand für die Einrichtung bzw. das Unternehmen erheblich vermindert. (Abbildung 16)

Durch die Auslagerung wird die PKI Technologie erst für kleine und mittelständische Unternehmen/Einrichtungen interessant. So bietet sich jetzt auch für die Fachhochschule Düsseldorf, im Rahmen des Dienstangebots *DFN-PKI-2* des *Deutschen Forschungsnetzes (DFN)*, die Möglichkeit PKI Technologie kostengünstig, effizient und ohne großen administrativen Aufwand zu nutzen. Dies wurde im Rahmen dieser Diplomarbeit angestrebt und wird in den nachfolgenden Abschnitten beschrieben.

Der Kostenfaktor gewinnt aufgrund des zunehmenden Wettbewerbs für die Hochschullandschaft zunehmend an Bedeutung, und so ist die Entscheidung des DFN zu begrüßen, den Dienst DFN-PKI-2 ohne zusätzliches Entgelt für die DFN-Anwender bzw. Nutzer des Dienstes DFN-Internet anzubieten.

Abbildung 16: Aufwand bei Auslagerung von PKI Komponenten *(Quelle: DFN [Pm05])*



3.1.6. Übersicht des DFN-PKI Dienstangebots

Der DFN betreibt seit 1997 die Wurzelzertifizierungsstelle DFN-PCA für das deutsche Forschungsnetz und bietet den DFN-Anwendern Dienstleistungen im Bereich PKI an. Diese Dienstleistungen[12] beinhalten unter anderem:

- Zertifizierung vom DFN-Anwender betriebenen Zertifizierungsstellen *(CA Certification)*
- Direkte Ausstellung von Zertifikaten für Benutzer und Server der DFN-Anwender
- Betrieb der obersten Zertifizierungsstelle DFN-PCA
- Betrieb eines LDAP-Servers als *Directory Service* für die Veröffentlichung der Zertifikate
- Unterstützung von *Pretty Good Privacy* (PGP), durch Ausstellung von PGP-Zertifikaten und Betrieb eines *PGP-Key-Servers*
- Support der DFN-Anwender

[12] http://www.dfn.de/content/dienstleistungen/dfnpki/dfnpki1/ (Stand 15.09.05)

Um flexiblere Dienstleistungen anbieten zu können und auf neue Anforderungen seitens der DFN-Anwender (zum Beispiel aus dem Bereich GRID-Computing) eingehen zu können, hat der DFN das Angebot der Dienstleistungen überarbeitet. Das bisherige Angebot wird unter dem Namen *DFN-PKI-1*[13] weiterhin angeboten, während das neue Angebot den Namen *DFN-PKI-2*[14] erhalten hat.

3.1.7. Eckpunkte der Neustrukturierung der DFN-PKI

In [Pm04] werden die Eckpunkte der Neustrukturierung der DFN-PKI wie folgt zusammengefasst:

- Klare Ausrichtung auf fortgeschrittene Zertifikate auf Basis des X.509-Standards
- Angebot von vier unterschiedlichen Betreibermodellen, wobei insbesondere eine Trennung von Zertifizierungsstelle und Registrierungsstelle möglich ist, ebenso wie eine teilweise oder vollständige Übernahme dieser Funktionen durch Dritte
- Möglichkeit der Schlüsselerzeugung durch Dritte und der Schlüsselhinterlegung bei Dritten
- Unterschiedliche Sicherheitsniveaus für Identifizierungsverfahren, wobei zwei Sicherheitsstufen mit bzw. ohne Vorlage eines gültigen Ausweispapiers durch zwei entsprechend formulierte Policies unterstützt werden
- Betrieb erweiterter Infrastrukturkomponenten (Verzeichnisdienst, Sperrlistenmanagement, OCSP zur Online-Prüfung von Zertifikaten)
- Untersuchung und Bewertung der Möglichkeiten zur Integration der Wurzelzertifikate in Standard-Browser
- Einbindung in relevante nationale und internationale PKI
- Ausbau und Pflege eines umfassenden Support-Systems (Handbücher, Informationssysteme, Training)
- Weitere Unterstützung von PGP und Beobachtung sowie Bewertung neuer Verfahren

Der Sachverhalt wird auf den Webseiten[15] des DFN ausführlich beschrieben.

[13] http://www.dfn.de/content/dienstleistungen/dfnpki/dfnpki1/ (Stand 15.09.05)
[14] http://www.dfn.de/content/de/dienstleistungen/dfnpki/dfnpki2/ (Stand 15.09.05)
[15] http://www.dfn.de/content/dienstleistungen/dfnpki/ (Stand 15.09.05)

3.1.8. DFN-PKI-2

Das Dienstleistungspaket DFN-PKI-2 bietet die in Abschnitt 3.1.5 erläuterte Möglichkeit, Kernkomponenten an den DFN-Verein auszulagern. Zur Zeit befindet sich DFN-PKI-2 in einem Pilotbetrieb, an dem sich die Fachhochschule Düsseldorf zusammen mit anderen Einrichtungen beteiligt. Die Beteiligung am Pilotprojekt entstand im Rahmen dieser Diplomarbeit und in Absprache mit den Verantwortlichen aus der Datenverarbeitungszentrale der Fachhochschule Düsseldorf, Dipl.-Ing. Ernst Schawohl und dem betreuenden Professor dieser Diplomarbeit, Prof. Günther Franke. Das Ziel dieser Zusammenarbeit ist es, der Fachhochschule Düsseldorf den Einstieg in die PKI Technologie zu ermöglichen, indem die FH Düsseldorf zukünftig eine eigene Zertifizierungsstelle nutzen kann, die vom DFN in Hamburg betrieben wird. Damit lassen sich Zertifikate ausstellen, um vorhandene und zukünftige Netzwerkdienste besser schützen zu können (siehe Abschnitt 2) und um den Angehörigen der FH Düsseldorf den Einsatz von verschlüsselter und signierter Emailkommunikation auf Basis von *S/MIME* zu ermöglichen.

Derzeit (Oktober 2005) wird die Nutzung der FH eigenen Zertifizierungsstelle FH-D-CA von der Abteilung DFN-PCA der DFN-CERT Services GmbH implementiert. Der Autor und der Verantwortliche der DVZ, Herr Dipl.-Ing. Schawohl, hatten während der Bearbeitung der Diplomarbeit bereits Zugriff auf eine "Test-PKI", um den *Workflow* des Betriebs einer Zertifizierungsstelle durch den DFN kennen zu lernen. Die Test-PKI wurde im Rahmen dieser Arbeit intensiv genutzt. Die Zertifikate für die Demonstration eines zertifikatsbasierten VPN in Abschnitt 4.2 wurden mit der Test-PKI ausgestellt. Der Arbeitsablauf wird im Abschnitt 3.4 eingehend beschrieben und unterscheidet sich vom *Workflow* der FH eigenen Zertifizierungsstelle nur geringfügig.

Das Pilotprojekt DFN-PKI-2 wird voraussichtlich im Januar 2006 in den Produktivbetrieb übergehen.

3.2. Organisation zur Teilnahme am DFN-Projekt

3.2.1. Richtlinien *(Policies)*

Der Betrieb einer Public Key Infrastruktur erfordert Richtlinien *(Policies)* zur Zertifizierung und zum Betrieb der Zertifizierungsstelle. Die Richtlinien wurden durch die *PKI-X.509 Working Group* (PKIX) der IETF definiert. Die *Certificate Policy* (CP) enthält die Richtlinien zur Zertifizierung und das *Certificate Practice Statement* (CPS), die Richtlinien zum Betrieb der Zertifizierungstelle. Die

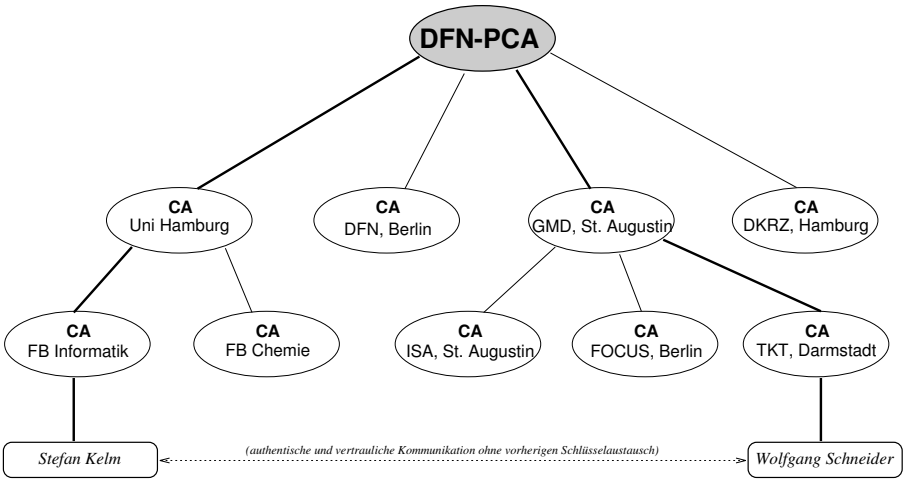
Policies der DFN-PCA und der FH-D-CA orientieren sich am internationalen Standard RFC 3647[16] bzw. RFC2527[17].

Von der DFN-PCA werden zwei Sicherheitsniveaus unterstützt (*Classic* und *Basic*), die je eine eigene Policy erfordern. Vom Sicherheitsniveau *Classic* unterscheidet sich das Sicherheitsniveau *Basic* hauptsächlich durch schwächere Identifizierungsverfahren bei der Registrierung[18].

3.2.2. Hierarchie

Die Zertifizierungsstelle der Fachhochschule Düsseldorf (FH-D-CA) erhält ihr CA-Zertifikat von der Wurzelzertifizierungsstelle, der DFN-PCA und ist somit ein Teil der Hierarchie der DFN-PCA (Abbildung 17).

Abbildung 17: Hierarchie der DFN-PCA *(Quelle: DFN)*



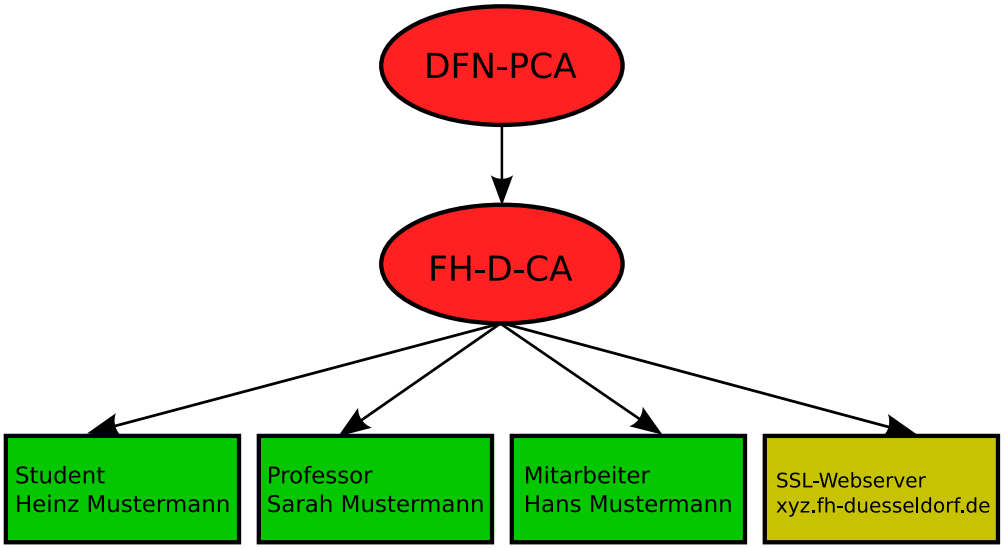
Mit dem CA-Zertifikat der FH-D-CA werden die weiteren Zertifikate für Benutzer und Systeme der FH-Düsseldorf ausgestellt bzw. signiert. (Abbildung 18)

3.2.3. Policies der FH-D-CA

Die Fachhochschule Düsseldorf hat sich für das Sicherheitsniveau *Classic* entschieden, um die Zertifikate auch dort anwenden zu können, wo starke Identi-

[16] http://www.ietf.org/rfc/rfc3647.txt (Stand 15.09.05)
[17] http://www.ietf.org/rfc/rfc2527.txt (Stand 15.09.05)
[18] http://www.pca.dfn.de/dfn-pki/certification/cp/ (Stand 15.09.05)

Abbildung 18: Hierarchie der FH-D-CA



fizierungsverfahren bei der Registrierung benötigt werden. Ein Beispiel ist die Einführung von *Selfservice-Funktionen* in der Hochschulverwaltung (siehe Abschnitt 2.3).

Nachfolgend wird die Erstellung der *Policies* für die Zertifizierungsstelle der FH-Düsseldorf (FH-D-CA) erläutert.

Die *Policies* wurden in Zusammenarbeit mit dem DFN auf Grundlage einer Vorlage für die Zertifizierungstellen in der Hierarchie der PKI des Deutschen Forschungsnetzes (DFN-PCA) erstellt. Die Grundlage der Policies der FH-D-CA bilden die Policies der DFN-PCA im Sicherheitsniveau *Classic*.[19] [20] Die Policies der FH-D-CA enthalten daher nur die notwendigen Ergänzungen, Einschränkungen und Abweichungen zu den Policies der DFN-PCA.

3.2.4. CP der FH-D-CA

Bedarf an Ergänzungen bei den Zertifizierungsrichtlinien (CP) der DFN-PCA bestand bei den Punkten:

- Pseudonymität/Anonymität des im Zertifikat enthaltenen Namens

  Für die Anwendungen bei Email *(S/MIME)* und Authentifizierung durch Zertifikate ist der vollständige Name wichtig. Die FH-D-CA schließt deshalb diese Möglichkeit aus.

[19] http://www.pca.dfn.de/dfn-pki/certification/cp/classic/x509/ (Stand 15.09.05)
[20] http://www.pca.dfn.de/dfn-pki/certification/cps/classic/x509/ (Stand 15.09.05)

- Nutzergruppen der FH-D-CA

  Die FH-D-CA bietet ihre Dienstleistungen allen Angehörigen der FH-Düsseldorf an.

- Möglichkeit der Schlüsselerzeugung durch die Zertifizierungstelle

  Die FH-D-CA bietet keine Möglichkeit zur Schlüsselerzeugung durch die Zertifizierungsstelle, weil diese Möglichkeit seitens des DFN noch nicht implementiert ist, und weil dies auch ein großes Vertrauen seitens der Anwender erfordert. Das Schlüsselpaar aus *private & public Key* wird bei der Beantragung eines Zertifikats im Webbrowser des Anwenders erzeugt und mit einer Passphrase geschützt im Software-Kryptomodul des Browsers gespeichert.

- Veröffentlichung des Zertifikats

  Die FH-D-CA veröffentlicht die gemäß der Zertifizierungsrichtlinien der DFN-PKI angeforderten Zertifikate über die in den Policies der FH-D-CA angegebenen Informationssysteme (HTTP/S & LDAP Server). Zertifikate für natürliche Personen werden immer durch die FH-D-CA veröffentlicht.

- Richtlinien und Praktiken zur Schlüsselhinterlegung und -wiederherstellung

  Die FH-D-CA bietet keine Möglichkeit zur Schlüsselhinterlegung, da dies seitens des DFN nicht implementiert ist und einen großen Aufwand zum Schutz der hinterlegten Schlüssel bedarf.

3.2.5. CPS der FH-D-CA

Bedarf an Ergänzungen bei der *Erklärung zum Zertifizierungsbetrieb* (CPS) der DFN-PCA:

Es mussten Kontaktinformationen wie Anschriften, Telefon-, Email- und Internetadressen für die Organisation der FH-D-CA in die Policies aufgenommen werden. Diese beinhalten unter anderem:

- Die Anschrift der Zertifizierungsstelle:

FH-D-CA (Abt. DVZ)	Telefon: +49 4351-522
Josef-Gockeln-Strasse 9	Telefax: +49 4351-523
	E-Mail: pki@fh-duesseldorf.de
D - 40474 Düsseldorf	WWW: http://www.fh-duesseldorf.de/pki

- Kontaktperson für die Zertifizierungsrichtlinien und die Erklärung zum Zertifizierungsbetrieb:

Herr	Telefon: +49 4351-522
Dipl.-Ing. Ernst Schawohl	Telefax: +49 4351-523
Josef-Gockeln-Strasse 9	E-Mail: pki@fh-duesseldorf.de
D - 40474 Düsseldorf	WWW: http://www.fh-duesseldorf.de/pki

- Informationen zum Verzeichnisdienst der FH-D-CA:
  - https://www.pca.dfn.de/fh-duesseldorf
  - http://www.pca.dfn.de/fh-duesseldorf
  - ldap://ldap.pca.dfn.de/c=DE/o=DFN-Verein/ou=DFN-PKI/o=FH-Duesseldorf

- Angaben wo Informationen zur FH-D-CA publiziert werden:
  - *Zertifikat und Fingerabdruck:*
    https://www.pca.dfn.de/fh-duesseldorf sowie
    http://www.pca.dfn.de/fh-duesseldorf
  - *Zertifizierungsrichtlinien:*
    https://www.pca.dfn.de/fh-duesseldorf sowie
    http://www.pca.dfn.de/fh-duesseldorf
  - *Erklärung zum Zertifizierungsbetrieb:*
    https://www.pca.dfn.de/fh-duesseldorf sowie
    http://www.pca.dfn.de/fh-duesseldorf
  - *Liste der Registrierungsstellen:*
    https://www.pca.dfn.de/fh-duesseldorf sowie
    http://www.pca.dfn.de/fh-duesseldorf

- Definition und Information zur Namensform der Zertifikate der FH-D-CA:

  Die Namensform bezieht sich auf die Attribute eines X.509 Zertifikats und muss fest definiert werden. Dies ist beispielsweise für eine *Client-Authentifizierung* auf Basis der Zertifikate notwendig, wo gezielt auf bestimmte Attribute des Zertifikats hin geprüft werden kann, um nur einer bestimmten Gruppe Zugang zu einer Ressource zu ermöglichen. Ein einfaches Beispiel hierfür ist die Annahme, dass nur die Professoren und Angestellten Webseiten und Dokumente eines für sie bestimmten Webservers nutzen sollen.

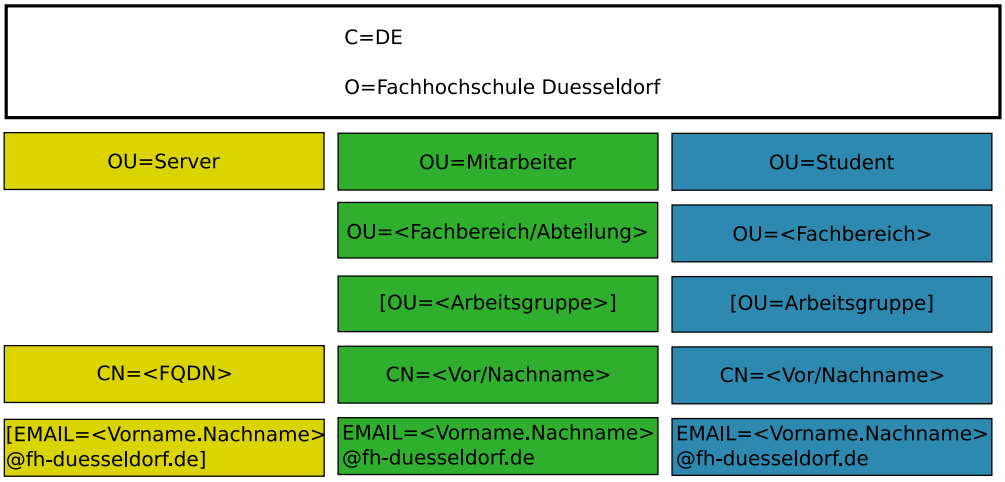
Es galt also eine Namensform zu definieren, die Hierarchien und Gruppenzugehörigkeiten in der Fachhochschule berücksichtigt und damit solche Anwendungen ermöglicht und vereinfacht. Die Policy der FH-D-CA berücksichtigt dies, indem ein Namensraum definiert wurde, der neben Professoren, Mitarbeitern und Studenten weitere Gruppenzugehörigkeiten wie den Fachbereich und die Abteilung festlegt. Erreicht wird dies durch das Zertifikats-Attribut *Organisational Unit* (OU), welches in die auszustellenden Zertifikate der FH-D-CA mehrfach aufgenommen werden kann und damit eine Differenzierung von verschiedenen Gruppen ermöglicht. (Abbildung 19)

Alle Zertifikate der FH-D-CA bekommen im Attribut *Country* (C) den Wert *DE* und im Feld *Organisation* (O) den Wert *Fachhochschule Duesseldorf*. Im ersten *Organisational Unit* (OU) Attribut wird zwischen *Server* für Serverzertifikate und *Student* oder *Mitarbeiter* unterschieden. Im zweiten OU Attribut wird der *Fachbereich* des Studenten bzw. die/der *Abteilung/Fachbereich* des Mitarbeiters aufgenommen. Um eine weitere Gruppenzugehörigkeit von Benutzern in die Zertifikate aufnehmen zu können, wurde ein optionales drittes OU Attribut definiert, welches als Wert beispielsweise eine *Arbeitsgruppe* enthalten kann, jedoch im Gegensatz zu den anderen Attributen optional ist.

Im Attribut *Common Name* (CN) wird bei Benutzerzertifikaten *Vorname* und *Nachname* des Benutzers aufgenommen und gegebenenfalls noch ein eindeutiges Kürzel. Bei Serverzertifikaten enthält das CN Feld den *Full Qualified Domain Name* (FQDN) des Servers, also den eindeutigen Namen im *Domain Name System* (DNS).

In das Feld *Email* wird die *Emailadresse* des Benutzers in der Form `<Vorname>.<Name>@fh-duesseldorf.de` aufgenommen, d.h. es muss hier die Emailadresse aus dem Emailsystem der Fachhochschule Düsseldorf angegeben werden. Emailadressen von anderen *Email Service Providern* können nicht aufgenommen werden. Bei Serverzertifikaten ist die Angabe der Emailadresse der Serveradministrators vorgesehen, jedoch nicht zwingend erforderlich und kann eine beliebige Emailadresse enthalten.

Abbildung 19: Namensform der FH-D-CA



3.3. PKI-Software

Es existieren viele Implementierungen von PKI mit unterschiedlicher Komplexität, angefangen von einfachen freien *Frontends* zu OpenSSL (z.B. TinyCA und pyCA) mit Grundfunktionalitäten, bis hin zu komplexen Software Produkten von Herstellern wie RSA Security, Entrust und Microsoft, die auch speziellen Anforderungen seitens der Anwender bedienen. Dementsprechend variieren auch die Preise für PKI Software und Support. *RSA Keon* kostet beispielsweise ca. 25000 Euro für 2000 Benutzer, wohingegen andere Produkte inklusive Quellcode frei verfügbar sind (OpenCA). Für den Endanwender bieten sich allerdings derzeit nur 2 "bezahlbare" Alternativen: OpenCA und die Microsoft CA. Die PKI Software von Microsoft ist mittlerweile als Komponente im Serverbetriebssystem Windows 2000/2003 Server enthalten. Die Software hat im Vergleich zu OpenCA folgende Vor- und Nachteile:

Vorteile:

- einfache Administration über GUI
- *Active Directory* Integration
- *automatic enrollment* of Certificates & CRLs

NACHTEILE:

- CA *private key* unverschlüsselt im Dateisystem (Sicherheitsrisiko!)
- *Closed Source*
- geringere Modularität

Die "Kosten" sind jedoch nicht direkt vergleichbar, da beispielsweise für freie Software auch kein kommerzieller Support für die Kunden verfügbar ist, bzw. nur in Form von Mailinglisten existiert.

3.3.1. DFN-PKI-2 Software

Der DFN setzt für sein Angebot DFN-PKI-2 auf das *Open Source* Projekt *OpenCA*, dass von Massimilano Pala initiiert worden ist und von einer internationalen Entwicklergemeinde weiterentwickelt wird. OpenCA ist in *Perl* implementiert und greift für die kryptographischen Funktionen auf das *OpenSSL* Projekts[21] zurück. Als Datenbasis kann jede Datenbank genutzt werden, die kompatibel zur DBI-Schnittstelle ist, also u.a. *MySQL, PostgreSQL, Oracle* und *DB2*.

Die in Abschnitt 3.1.3 beschriebenen Komponenten und deren Interaktion werden bei *OpenCA* durch ein flexibles Konzept von so genannten Interfaces abgebildet. Interfaces bedeuten hier eine logische Sammlung von Kommandos des zentralen *OpenCA Daemons*.

Weitere Informationen zu *OpenCA* liefert der von den Entwicklern verfasste Artikel "Ausweisvergabe" [BW05] aus der Sonderausgabe *Security Edition* des *Linux Magazins* im Jahr 2005. Der Artikel ist auch in Form eines PDF Dokuments [22] auf der Webseite des *OpenCA* Projekts zu finden.

3.3.2. Warum Open Source?

Die Sicherheit eines Systems lässt sich am Besten durch unabhängige Bewertungen von Experten belegen und verbessern. Ein System lässt sich jedoch nur zuverlässig beurteilen, wenn die Details des Systems, also der Quellcode, veröffentlicht sind. Durch den offenen Zugang zu den Quelltexten eines Algorithmus und einer Implementierung reduziert sich das Geheimnis auf den verwendeten Schlüssel beim Einsatz des Algorithmus bzw. Produkts. Die Antithese *"security*

[21] http://www.openssl.org (Stand 15.09.05)
[22] http://www.openca.info/docs/LinuxMagazinSpecial.pdf (Stand 15.09.05)

by obscurity", zu deutsch "Sicherheit durch Unklarheit", wird durch die kommerziellen Anbieter von *Closed Source* Software Produkten repräsentiert und verfochten. Diese argumentieren, dass die Veröffentlichung den Angreifern nur Informationen liefert, um Schwachstellen zu finden und auszunutzen. Es gibt jedoch genügend Beispiele aus der Praxis, wo das System der Geheimhaltung versagt hat [Sb00]:

- Das DVD Verschlüsselungssystem CSS *(Content Scrambling System)*
- Microsofts VPN Protokoll PPTP
- die ehemals proprietäre Mobilfunk Sicherheitssysteme des GSM Standards

Open Source bedeutet jedoch KEINE Garantie für die Sicherheit von Code, sie bietet nur bessere Bedingungen, um die Sicherheit durch Experten zu prüfen. Zuerst muss der Quellcode ja mal von einem fähigen Experten gelesen und beurteilt werden, und dann müssen eventuelle Schwachstellen auch behoben werden.

Gerade beim Thema Sicherheit dauert es auch extrem lange bis sich ein neues Produkt oder ein neuer Algorithmus durchsetzt, denn die Experten misstrauen erst einmal einem neuen Produkt. Hat es sich dann über Jahre im produktiven Einsatz und bei Bewertungen von Experten bewiesen, dann wird es auch von den Experten als sicher oder besser *well tested* eingestuft.

Für weitere Informationen sei an dieser Stelle auf das Buch *Secrets and Lies* [Sb00] des international anerkannten Experten Bruce Schneier verwiesen, der diesem Thema ein eigenes Kapitel widmet.

3.4. Workflow der FH-D-CA / Test-PKI

In diesem Abschnitt soll der *Workflow* bei der Nutzung von OpenCA anhand der Test-PKI des DFN gezeigt werden. Es wird darauf eingegangen, wie ein Zertifikat beantragt wird, der Antrag vom RA-Operator bearbeitet wird und sich die Ausgabe des Zertifikats für den Anwender darstellt. Die Nutzung des Zertifikats ist anwendungsspezifisch und kann daher nicht pauschal dargestellt werden. Der *Workflow* wird anhand von *Screenshots* mit entsprechenden Erläuterungen gezeigt und gibt die Interaktion der Komponenten einer PKI aus Abschnitt 3.1.3 im Fall von OpenCA wieder.

3.4.1. Zertifizierungsantrag stellen (Anwender)

Der Workflow für ein Serverzertifikat unterscheidet sich von dem für ein Benutzerzertifikat im Punkt der Beantragung des Zertifikats. Die Erstellung des Schlüsselpaars *(private & public key)* erfolgt bei einem Benutzer während des Antrags vollautomatisch im Webbrowser und wird in einem Software Kryptomodul des Browsers vor unbefugtem Zugriff geschützt gespeichert (browserspezifisch). Eine Hardware Lösung des Kryptomoduls in Form von Chipkarten oder *USB-Token* wird auch unterstützt. Der public Key wird dann zusammen mit den persönlichen Angaben des Anwenders automatisch als Zertifizierungsantrag (CSR) an die RA übermittelt.

Der Antrag wird, wie folgt beschrieben, an die RA gestellt. Die Abbildungen *(Screenshots)* für den Anwenderfall stammen von der englischen Version von Microsoft WindowsXP, mit dem Microsoft Internet Explorer. Daher sind die Dialoge in englischer Sprache.

1. Der Anwender füllt auf der Webseite die Angaben für das Zertifikat aus. (Abbildung 20)
2. Die Angaben werden auf der folgenden Seite bestätigt. (Abbildung 21)
3. Es empfiehlt sich, die hohe Sicherheitsstufe zu wählen. Damit wird für den Zugriff auf den privaten Schlüssel immer nach dem Passwort für das Kryptomodul des Browsers verlangt. (Abbildung 22)
4. Das Passwort für das Kryptomodul wird festgelegt, wenn nicht bereits vorhanden. (Abbildung 23)
5. Das Schlüsselpaar aus privatem und öffentlichem Schlüssel wird im Kryptomodul des Browsers erzeugt und abgelegt
6. Der öffentliche Schlüssel wird mit den persönlichen Angaben an die RA übermittelt.
7. Der Anwender wird aufgefordert, das Formblatt zur Vorlage beim Personal der Registrierungsstelle auszudrucken. (Abbildung 24)

Abbildung 20: Angaben für den Antrag

Abbildung 21: Bestätigung der Angaben für den Antrag

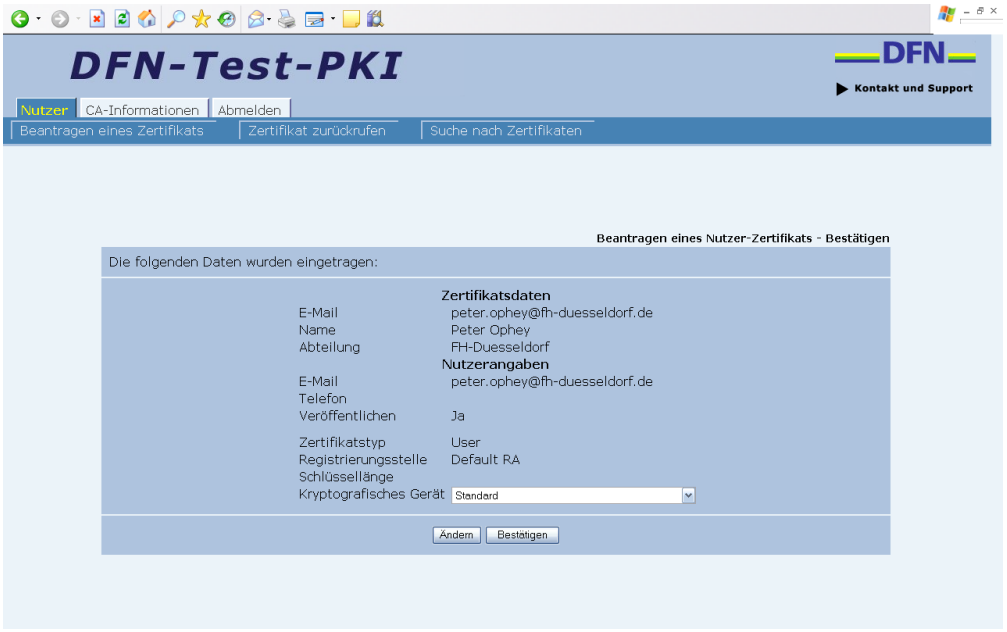


Abbildung 22: Sicherheitsstufe für den Zugriff auf den private Key

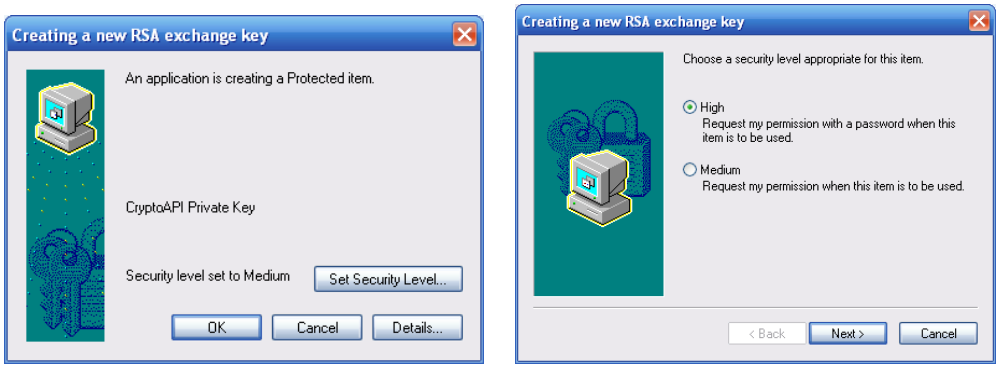
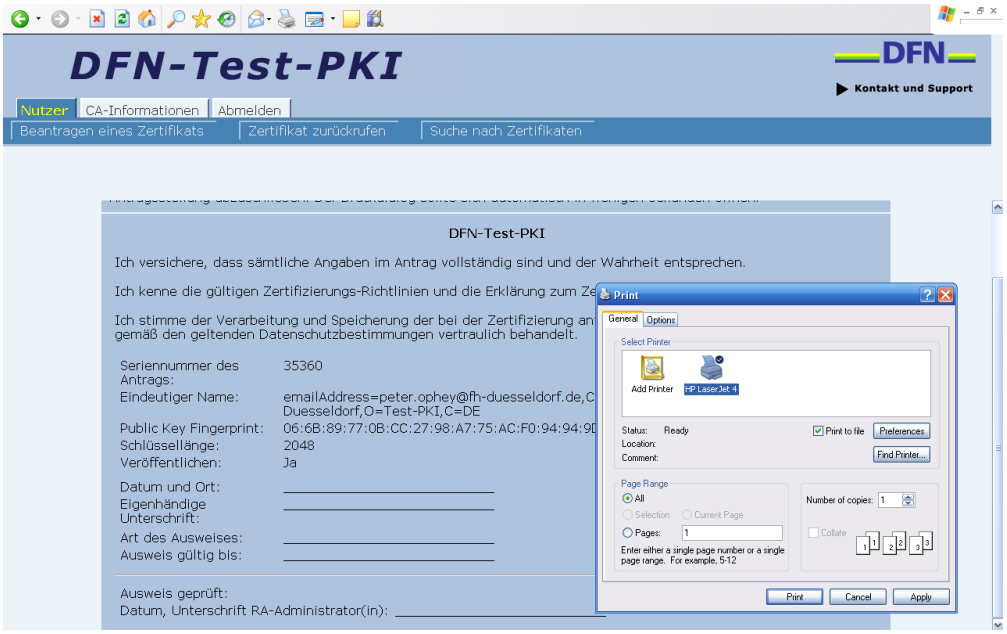


Abbildung 23: Passwort für den Zugriff auf das Kryptomodul

Abbildung 24: Formblatt zur Vorlage bei der Registrierungsstelle



3.4.2. Zertifizierungsantrag stellen (Server)

Für ein Serverzertifikat muss der CSR zusammen mit dem privaten Schlüssel von OpenSSL produziert werden und als *PKCS#10* Datei an die RA übermittelt werden. OpenSSL bietet "nur" ein Kommandozeilenprogramm, um das Schlüsselpaar und den CSR zu erzeugen, aber es existieren grafische Benutzeroberflächen (GUI)[23], die die Benutzung erleichtern. Der Autor stellt im Anhang C & D die Kommandos und eine OpenSSL Konfigurationsdatei zur Verfügung, mit denen ein CSR für den in Abschnitt 4.2 vorgestellten VPN Server erzeugt wurde. Der private Schlüssel wird der Serversoftware später zusammen mit dem Zertifikat mit restriktiven Zugriffsrechten im Dateisystem des Rechners zur Verfügung gestellt.

Der logische Ablauf zur Beantragung des Serverzertifikats ist weitestgehend identisch mit der zuvor beschriebenen Beantragung eines Benutzerzertifikats:

1. Der Anwender füllt auf der Webseite die Angaben für das Zertifikat und wählt die zuvor generierte PKCS#10 Datei aus. (Abbildung 25)

2. Die Angaben werden auf der folgenden Seite bestätigt. (Abbildung 26)

[23] http://www.sweb.cz/mycert/ (Stand 15.09.05)

3. Der öffentliche Schlüssel wird mit den persönlichen Angaben an die RA übermittelt.

4. Der Anwender wird aufgefordert, das Formblatt zur Vorlage beim Personal der Registrierungsstelle auszudrucken. (Abbildung 27)

Abbildung 25: Antrag für Server

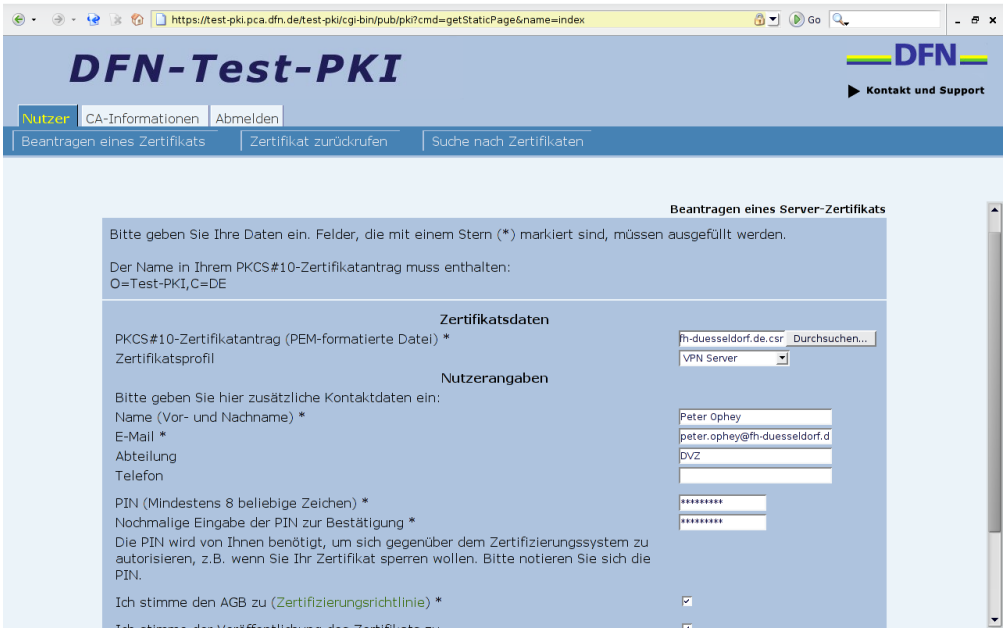
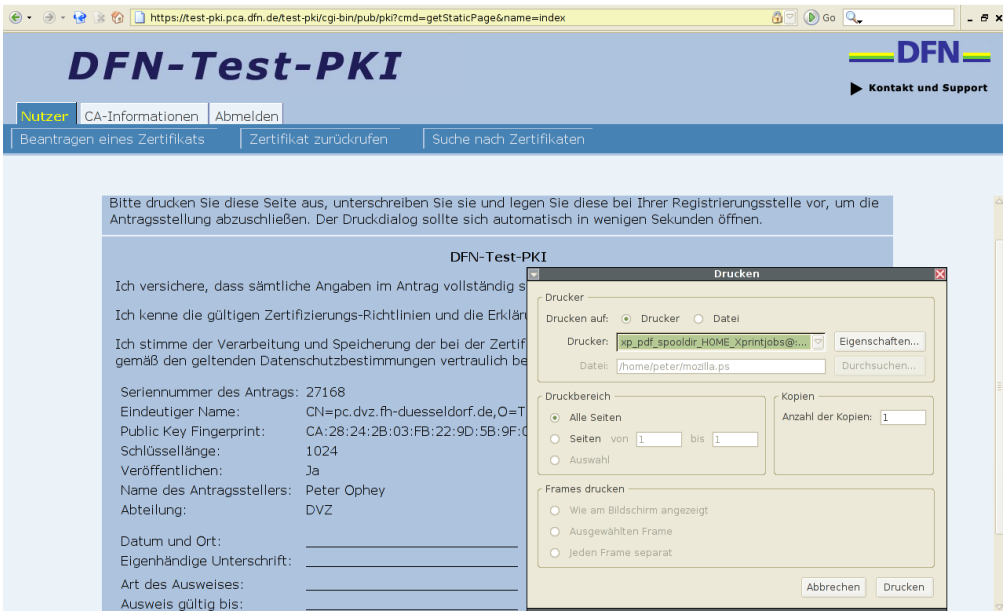


Abbildung 26: Bestätigung

Abbildung 27: Aufforderung zum Druck des Formblatts



3.4.3. Zertifizierungsantrag bearbeiten (RA)

Die Bearbeitung eines Zertifizierungsantrages durch das Personal *(RA-Operator)* ist für Server- und Benutzerzertifikate gleich. Nach dem Eingang des Zertifi-

zierungsantrags bei der RA muss sich der Anwender mit dem zuvor ausgedruckten Formblatt und einem Dokument, welches die eindeutige Identifizierung ermöglicht, an das Personal der Registrierungsstelle wenden. Dieses prüft und korrigiert gegebenenfalls die Angaben des Benutzers und dessen Identität. Die Bearbeitung eines Antrags erfolgt folgendermaßen:

1. Der Anwender wendet sich persönlich an die Registrierungsstelle, die dessen Identität prüft und den schriftlichen Antrag entgegennimmt.
2. Der RA-Operator sucht den Antrag über die Webschnittstelle der RA. (Abbildung 28)
3. Der RA-Operator prüft die Angaben im Antrag. (Abbildung 29)
4. Der RA-Operator korrigiert mögliche Fehler im Antrag. (Abbildung 30)
5. Der RA-Operator genehmigt und signiert den Antrag mit seinem persönlichen *private key*. (Abbildung 31)
6. Bestätigung der Genehmigung an den RA-Operator durch die RA Software (Abbildung 32)

Abbildung 28: neue Anträge der RA

Abbildung 29: prüfen der Angaben

Abbildung 30: korrigieren der Angaben

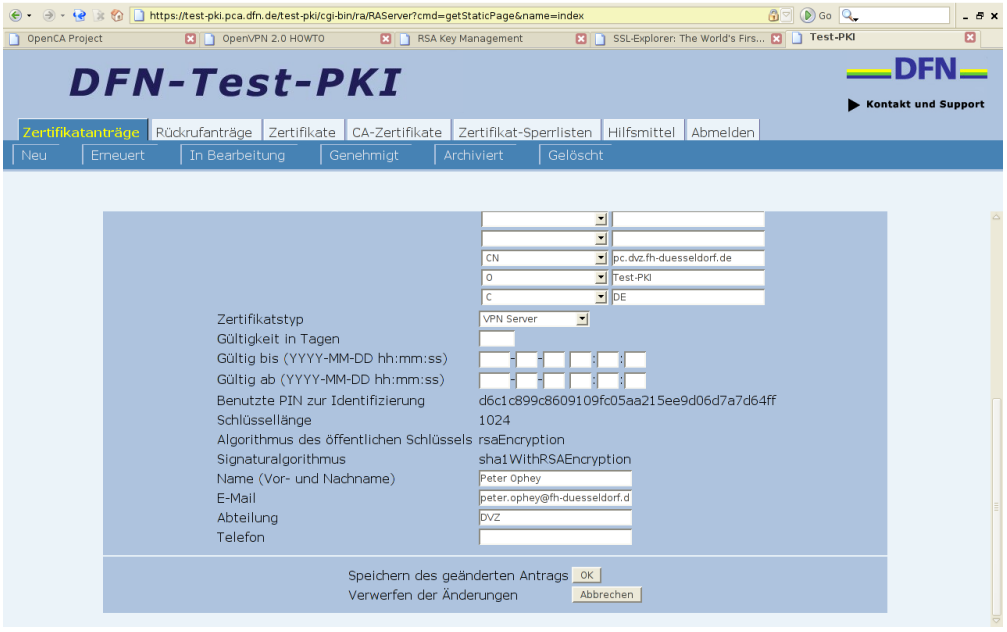


Abbildung 31: Genehmigung & Signatur des Antrags

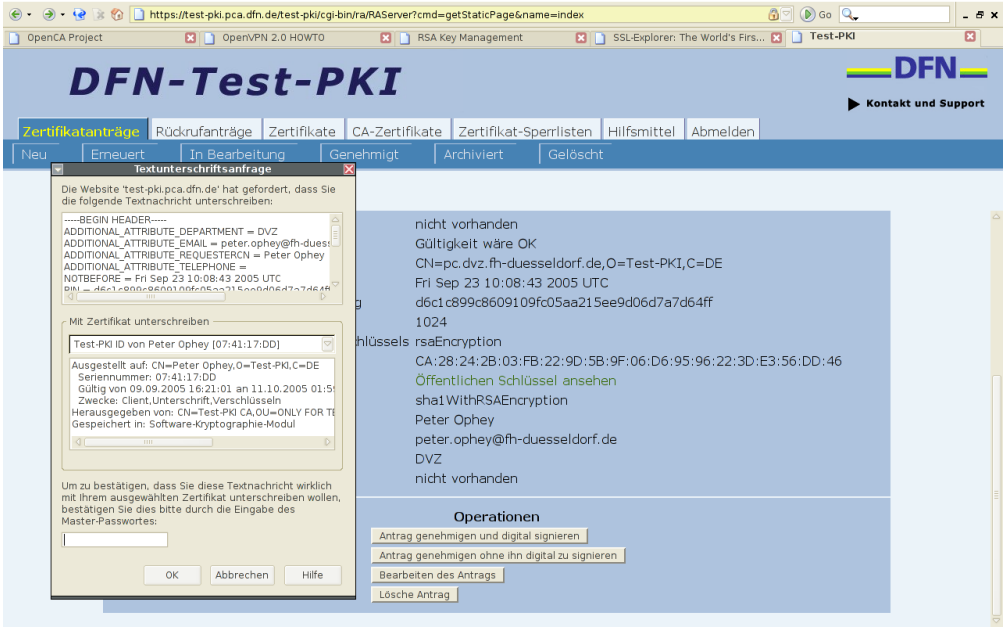


Abbildung 32: Bestätigung an den RA-Operator

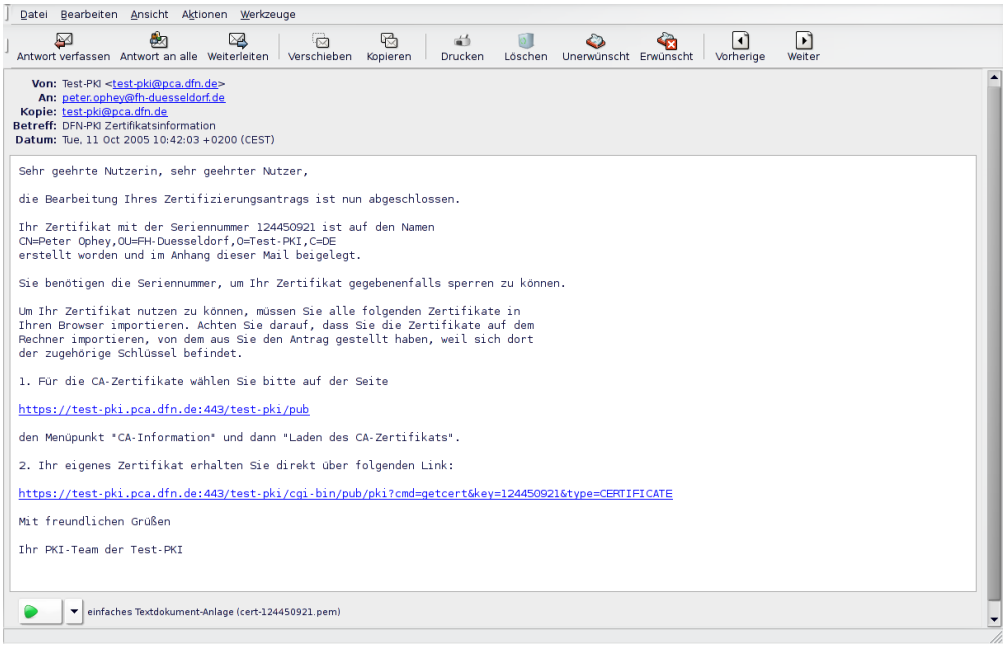
3.4.4. Zertifikat entgegennehmen

Nachdem der Antrag vom RA-Operator genehmigt worden ist, wird das Zertifikat durch die CA erstellt. Dazu werden die Daten des Anwenders aus dem Zertifizierungsantrag (CSR), also der öffentliche Schlüssel und weitere Angaben wie Name und Emailadresse oder im Fall eines Servers der *Common Name*, in einem X.509 Zertifikat zusammengefasst, welches mit dem privaten Schlüssel der Zertifizierungsstelle signiert wird. Die Signatur des Zertifikats kann von jedem Anwender mit dem frei verfügbaren öffentlichen Schlüssel der CA geprüft werden.

Das Zertifikat wird an die im Zertifizierungsantrag angegebene Emailadresse geschickt und im Verzeichnisdienst der PKI veröffentlicht. In der automatisch generierten Email (Abbildung 33) wird der Anwender aufgefordert, -

1. - das CA-Zertifikat der Zertifizierungsstelle über einen Link zu importieren.
2. - sein persönliches Zertifikat über einen Link zu importieren.

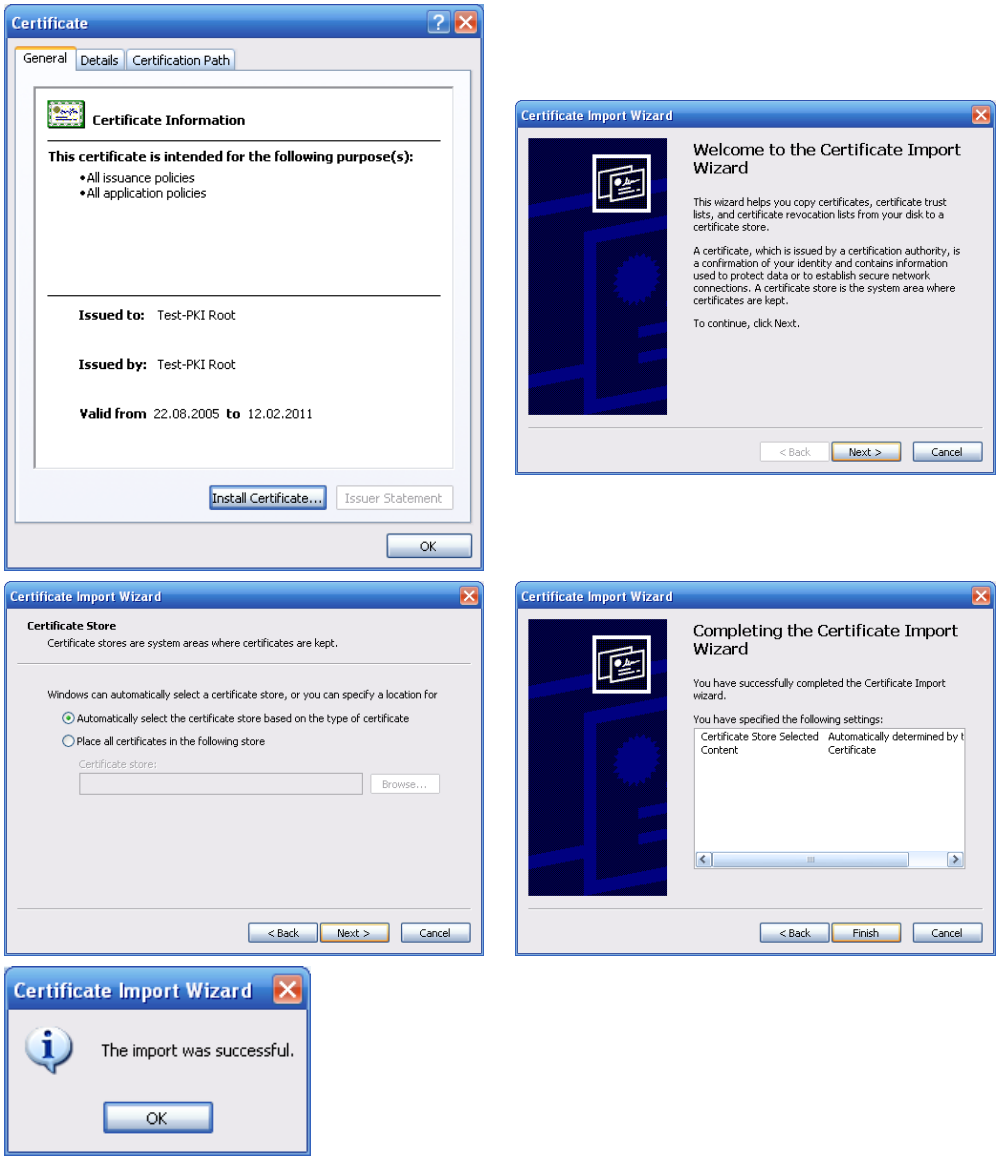
Abbildung 33: Email der Zertifizierungsstelle an den Anwender



Der Import der Zertifikate ist anwendungsabhängig und ist hier wiederum unter Mircrosoft WindowsXP und Internet Explorer 6 dargestellt. (Abbildung 34) Der Import in andere Browservarianten wie Mozilla Firefox, Opera, Apple Safari

und Konqueror wurde vom Autor unter diversen Betriebssystemen erfolgreich durchgeführt.

Abbildung 34: Import des Wurzelzertifikats



Es empfiehlt sich, ein Backup des Zertifikats inklusive des privaten Schlüssels auf ein externes Medium (z.B. Diskette, CD-ROM) anzulegen. Dazu kann das PKCS#12 Format genutzt werden, welches Zertifikat und privaten Schlüssel mit einem Passwort geschützt in eine Datei ablegt. Dieses Format wird von den meisten Anwendungen (Emailclients, VPN-Clients, usw.) standardmäßig für den Import und Export in den Zertifikatsspeicher bzw. die Kryptomodule unterstützt.

3.4.5. Zertifikate in Anwendungen

Für die Anwendung in einem Emailclient zur vertraulichen Emailkommunikation kann diese PKCS#12 Datei dann genutzt werden. Der Import und Export kann aufgrund der Vielzahl von Anwendungen und deren Funktionen hier nicht gezeigt werden. Der Autor hat die verwendeten Zertifikate jedoch in unterschiedlichsten Browsern und Emailclients auf den Betriebssystemen Microsoft WindowsXP, Linux und MacOS X genutzt und getestet. An dieser Stelle soll aber zumindest beispielhaft eine signierte Email an eine Mailingliste in drei verbreiteten Emailclients gezeigt werden; Evolution (Abbildung 35) und Mozilla Thunderbird (Abbildung 36) unter Linux und Microsoft Outlook (Abbildung 37) unter WindowsXP.

Abbildung 35: signierte Email in Evolution

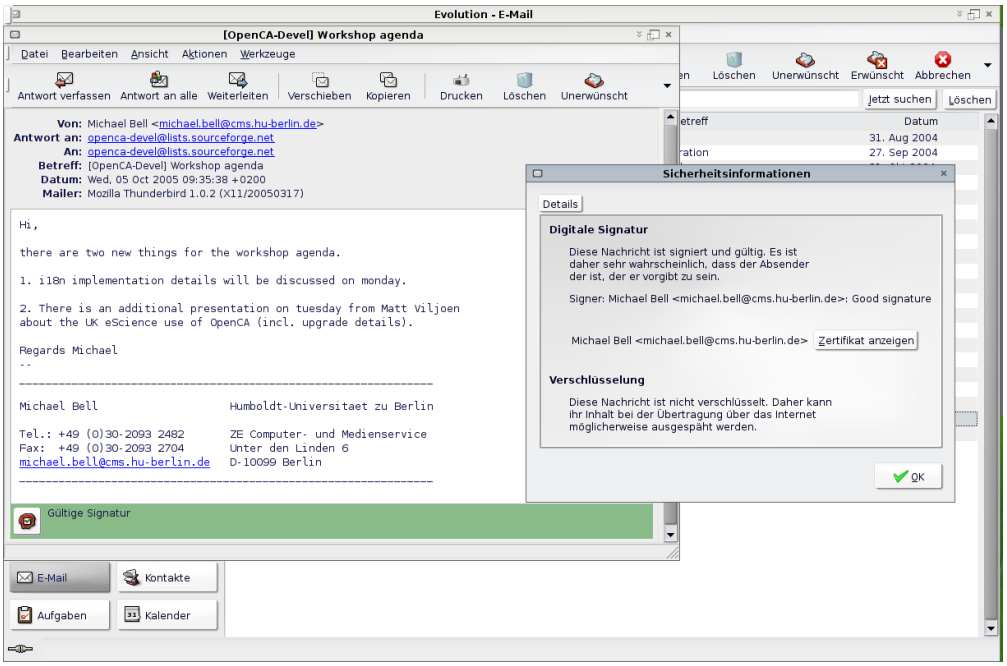


Abbildung 36: signierte Email in Mozilla Thunderbird

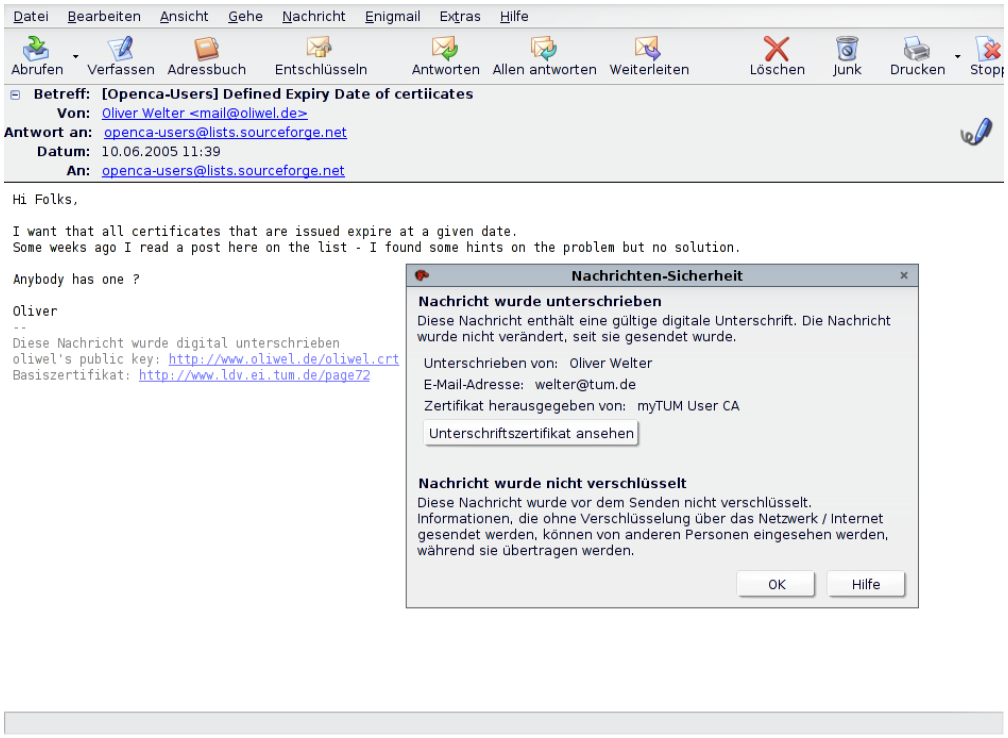
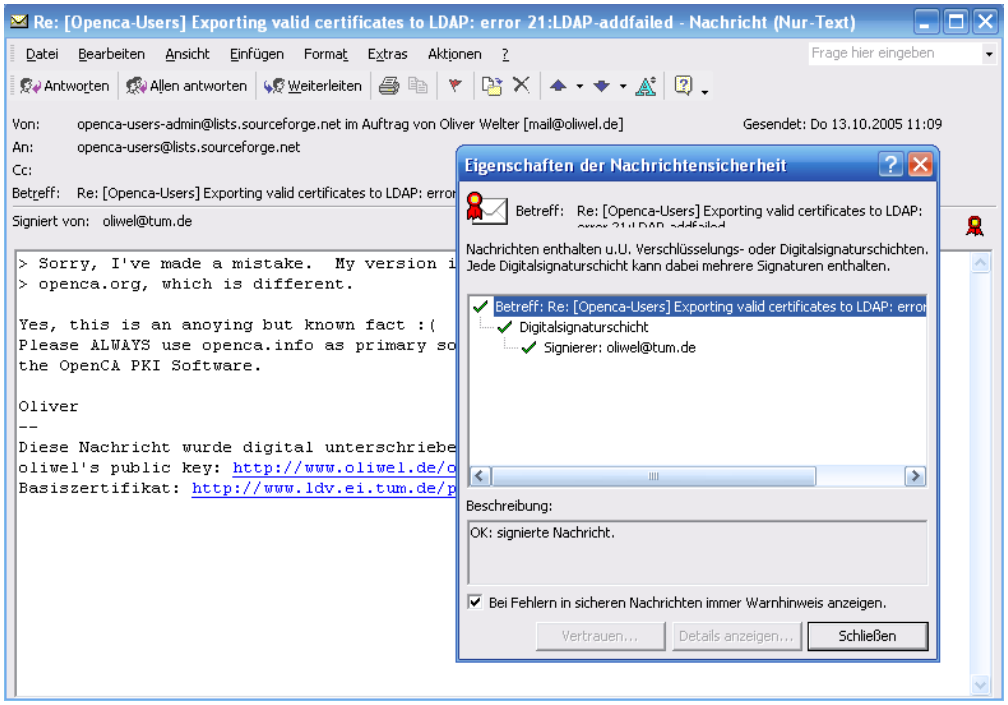


Abbildung 37: signierte Email in Microsoft Outlook



3.4.6. Informationen für den Anwender

Für die Anwender muss neben den Informationen durch die PKI Software weitere Hilfe für die Nutzung der Zertifikate in Anwendungen bereitgestellt werden. Der DFN plant ein solches Support-System für den Produktivbetrieb des DFN-PKI-2 Angebotes.

4. Virtual Private Network (VPN)

4.1. Definition eines VPN

Der Begriff *Virtual Private Network*, zu deutsch virtuelle private Netze, bezieht sich auf die privaten Netze von Großunternehmen, die ihre Standorte über Standleitungen vernetzen. Diese Standleitungen müssen von den Telefongesellschaften angemietet werden und verursachen enorme Kosten, bieten dafür allerdings die exklusive und sichere Nutzung. Der Erfolg von öffentlichen Datennetzen, insbesondere dem Internet, brachte diese teils weltweit agierenden Unternehmen auf den Plan, diese öffentlichen Netze zu nutzen, um ihre Standorte zu vernetzen und so Mittel zu sparen. Das Schlagwort *Virtual Private Network* war geboren. Virtuell, weil das private Netz hier nicht mehr physisch in Form von dezidierten Leitungen existiert sondern eben "nur" als *Overlay-Network*. Es wird zwischen zwei VPN-Topologien unterschieden:

- *site-to-site*: Hier werden lokale Netze über *VPN-Gateways* vernetzt. (Abbildung 38)
- *remote access:* Clients (so genannter *Roadwarrior*s) greifen auf ein internes Netz zu. (Abbildung 39)

Abbildung 38: *site-to-site* VPN

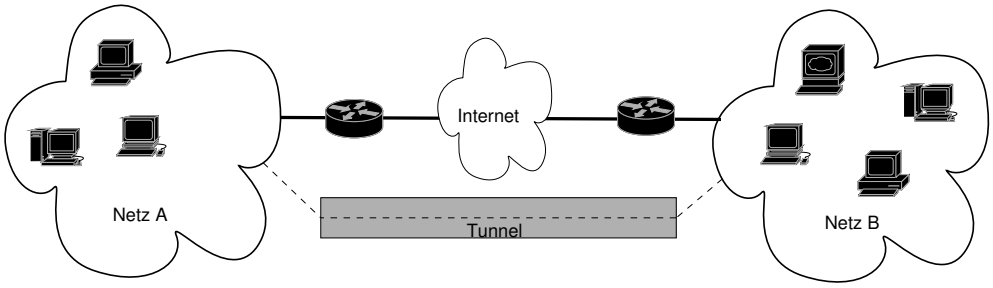
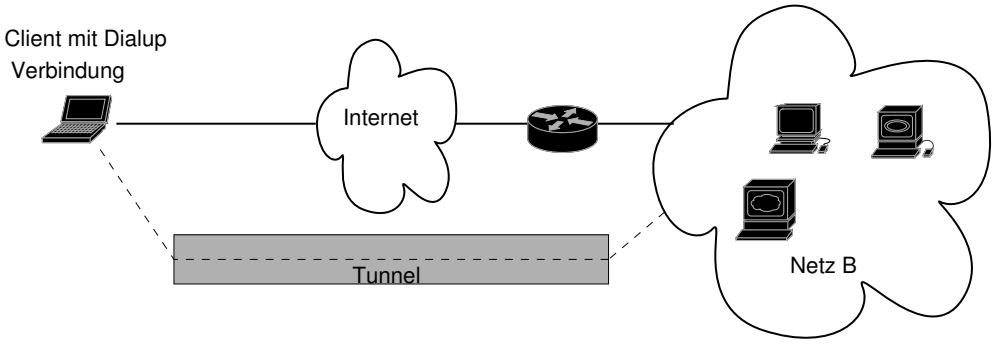


Abbildung 39: *remote access*



Eine große Herausforderung war, die Geheimhaltung und Integrität der übertragenen Daten zu gewährleisten. Daher wurden Sicherheitsprotokolle entwickelt, die dieses garantieren. Es existiert eine Vielzahl an Protokollen, die für den Einsatz in VPNs entwickelt wurden; neben Eigenentwicklungen wie PPTP von Microsoft und in Gremien (IETF) ausgearbeiteten Entwicklungen wie IPsec, natürlich auch proprietäre Entwicklungen, die nur in den Produkten des Unternehmens verfügbar sind. Nachdem in PPTP zahlreiche Schwachstellen aufgezeigt wurden, hat sich IPsec als Industriestandard durchgesetzt. Der Aufbau und Einsatz von IPsec ist sehr komplex, und diese Tatsache wurde von führenden Experten wie Bruce Schneier und Niels Ferguson in einer kryptographischen Evaluation[FS99] kritisiert:

> "IPsec was a great disappointment to us. Given the quality of the people that worked on it and the time that was spent on it, we expected a much better result. ... Our main criticism of IPsec is its complexity. IPsec contains too many options and too much exibility; there are often several ways of doing the same or similar things. This is a typical committee effect. Committees are notorious for adding features, options, and additional flexibility to satisfy various factions within the committee. As we all know, this additional complexity and bloat is seriously detrimental to a normal (functional) standard. However, it has a devastating effect on a security standard. ... IPsec is too complex to be secure."[FS99]

Aufgrund des langen Standardisierungsprozesses und der Komplexität von IPsec wurden andere Lösungen gesucht und gefunden. Durch die intensive Nutzung im *World-Wide-Web* empfahl sich SSL als Alternative zu IPsec und es begann die Entwicklung von SSL-VPNs.

Wie schon im Abschnitt 3.3.2 angesprochen wurde nimmt das Konzept *Open Source* beim Einsatz in Sicherheitstechnologie einen besonderen Platz ein. Weitere Informationen zum diesem Thema liefert das Buch *Secrets and Lies* von Bruce Schneier [Sb00] und *VPN mit Linux* von Ralf Spenneberg [Sr03].

4.2. OpenVPN

OpenVPN ist eine OpenSource Software von James Yonan, ist unter der *GNU-General-Public-License (GPL)* lizensiert und damit frei verfügbar. Im Gegensatz zu traditionellen VPNs auf Basis von IPsec in der Netzwerkschicht des OSI-Modells, setzt OpenVPN zum Aufbau eines VPN auf die SSL/TLS Protokollfamilie und nutzt den vorhandenen *TCP/IP Stack* über die Transportschicht. Dieser Ansatz bietet den Vorteil, dass OpenVPN als *Daemon* bzw. Dienst im *Userspace* laufen kann und im Gegensatz zu IPsec keinen speziellen *Kernelcode* zum Beispiel in Form von Kernelmodulen benötigt. Daher ist die Portierung von OpenVPN auf andere Plattformen relativ einfach. Zur Zeit läuft OpenVPN auf: Linux, Windows 2000/XP, OpenBSD, FreeBSD, NetBSD, MacOS X und Solaris. Ein weiterer Vorteil dieses Konzepts ist die einfachere Konfiguration und Administration des Programms im *Userspace*. Zur Authentifizierung der Nutzer wird *Public-Key-Kryptographie* auf Basis von X.509 Zertifikaten eingesetzt, und zur symmetrischen Verschlüsselung der Daten können die leistungsfähigen Algorithmen (AES, Blowfish, 3DES usw.) der OpenSSL Bibliothek genutzt werden.

4.2.1. Virtuelle Netzwerkschnittstellen

Um Datenverbindungen unabhängig vom Kernel des Betriebssystems zu verschlüsseln, richtet OpenVPN ein virtuelles Netzwerkinterface ein. Dieses so genannte *TUN-Interface* stellt für das Betriebssystem ein *Point-to-Point* Interface dar. Die beiden Endpunkte sind der VPN-Server und der VPN-Client. OpenVPN kann auf dieses Interface wie eine Datei lesend und schreibend zugreifen und IP-Pakete zum entfernten Endpunkt senden und von diesem empfangen. Diese virtuelle *Point-to-Point* Verbindung ist die logische Verbindung zwischen Client und Server.

Der verschlüsselte Tunnel zwischen VPN Server und Client wird über die reale Netzwerkschnittstelle in Form einer UDP Verbindung hergestellt. Jedes Paket mit TUN-Interface als Quelle wird von OpenVPN in ein neues UDP-Paket "verpackt" und dann verschlüsselt durch den Tunnel an die Gegenstelle geschickt. Dort wird es vom lokalen OpenVPN Prozess entschlüsselt, entpackt und an das

TUN-Interface weitergereicht. Der verbindungslose Charakter von UDP bietet einen Performancevorteil, da man bei einer getunnelten TCP Verbindung über einen UDP Tunnel keine Verluste durch "doppelte Empfangsbestätigungen und Flusssteuerung" hat. [To01]

Dieses Konzept bietet den Vorteil, dass man den normalen Datenverkehr auf dem realen Interface und den auf dem virtuellen "VPN" Interface einfach unterscheiden kann. Damit ergibt sich die Möglichkeit spezielle *routing* und/oder Firewall Regeln aufzustellen.

Weitere Informationen findet man in einer Presentation[Yj04] von James Yonan und auf den Webseiten[24] des OpenVPN Projekts.

4.2.2. Routing vs. Bridging

OpenVPN implementiert zwei Betriebsmodi um ein VPN zu errichten.

Bridging ist eine Technik, um die lokalen Netze auf OSI-Schicht 2 (Sicherungsschicht) bzw. der MAC-Teilschicht zu verbinden und auf OSI-Schicht 3 (Netzwerkschicht) in EINEM IP-Subnetz zu betreiben. Dadurch entsteht der Eindruck eines *wide-area Ethernet*. In diesem Fall werden sogar alle *Broadcasts* (z.B. Windows NetBIOS) weitergereicht. Der Vorteil von *Bridging* ist die einfache Konfiguration und der mögliche Transport von anderen Protokollen neben IP, also IPX, Netware oder AppleTalk. Nachteilig ist, dass es deswegen auch nicht gut skaliert.

Routing verbindet die lokalen Netze wie der Name schon sagt über Routen auf der Netzwerkschicht (IP). Diese Routen müssen konfiguriert werden, erlauben dafür aber auch eine bessere Kontrolle des erwünschten Datenverkehrs über das VPN. *Routing* ist aufgrund von fehlenden *Broadcasts* effizienter und skaliert daher auch besser als *Bridging*.

Der Autor hat sich daher bei der Demonstration eines VPN für *Routing* entschieden. Es konnten jedoch keine Tests der Performance bzw. des Durchsatzes durchgeführt werden. [Hc04] gibt jedoch Anhaltspunkte für den Durchsatz von OpenVPN.

4.2.3. Entscheidung für OpenVPN

Für den Autor gaben mehrere Gründe den Ausschlag, auf OpenVPN zur Demonstration eines VPN-Gateways für die Fachhochschule Düsseldorf zu setzen:

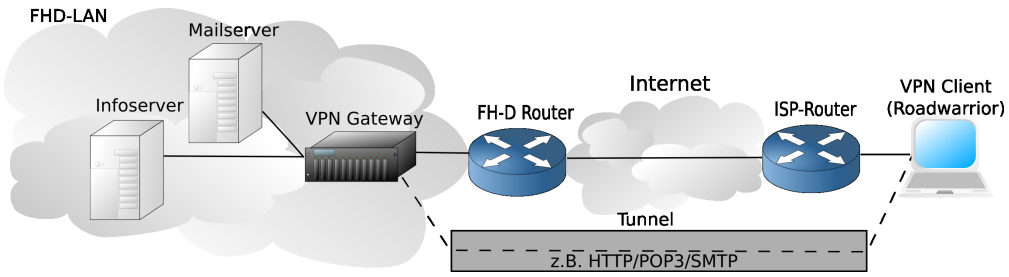
[24] http://openvpn.net/ (Stand 15.09.05)

- Bewertungen von VPN Implementierungen unter Linux durch Peter Gutmann in [Gp03]
- Quellcode verfügbar *(Open Source)*
- *cross plattform* Kompatibilität
- Einsatz von X.509 Zertifikaten zur Authentifizierung
- gute Dokumentation
- freie Verfügbarkeit und keine Lizenzkosten
- Performance

4.3. FH-D VPN Gateway

Die Topologie des VPN für die FH Düsseldorf entspricht grundsätzlich dem in Abschnitt 4.1 angesprochenen VPN für *remote access* (Abbildung 40). Das

Abbildung 40: *remote access* Topologie

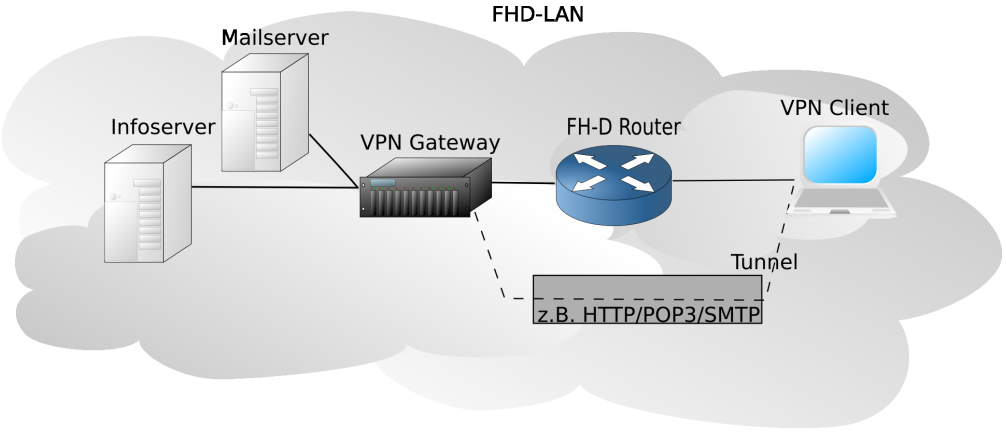


VPN soll allerdings nicht nur dazu genutzt werden, um die in Abschnitt 2.2.3 beschriebenen internen Dienste außerhalb der IP-Netze der Fachhochschule Düsseldorf nutzbar zu machen, sondern auch, um vorhandene (nichtöffentliche) Netzwerkdienste aus Abschnitt 2.2.2 abzusichern. Dieser zweite wichtige Aspekt lässt sich aber nicht nur *remote* nutzen, sondern eben auch aus den internen IP-Netzen der FH Düsseldorf. (Abbildung 41) Insofern ist der Begriff *remote access* hier nicht ganz richtig und man kann hier eher von *secure access* sprechen.

Ein weiteres Anliegen des Autors war es, die Verwendung der X.509 Zertifikate aus der Kooperation mit dem DFN (Abschnitt 3.1.8) zur Authentifizierung der Benutzer des VPN zu demonstrieren.

Diese Ziele wurden mit einem VPN Gateway auf Basis von OpenVPN erreicht und werden in den folgenden Abschnitten erläutert.

Abbildung 41: interne Nutzung des VPN Gateways



4.3.1. Testplattform

Dem Autor wurde von der Datenverarbeitungszentrale (DVZ) ein PC zur Demonstration des VPN Gateways bereitgestellt. Es handelt sich um einen PC mit standard Komponenten:

- Intel Pentium III Prozessor (500MHz)
- 512 MB RAM
- SCSI Controller mit 60GB Harddisk
- 100 MBit Ethernet Netzwerkkarte

Das VPN Gateway braucht eine ”offizielle” IP Adresse aus einem der IP-Subnetze der Fachhochschule Düsseldorf. Die Adresse 193.23.168.120 wurde mir kurzfristig von Herrn Dipl.-Ing. Ralf Hartenstein zur Verfügung gestellt.

Als Betriebssystem hat sich der Autor für GNU Linux in Form der Debian basierten Distribution Ubuntu entschieden. Den Ausschlag für diese Entscheidung gab die Erfahrung des Autors mit Debian basierten Linuxsystemen und die gute Hardwareerkennung des Ubuntu Installationssystems. Für den Produktivbetrieb empfiehlt sich allerdings eher eine für Serversysteme konzipierte Distribution ohne grafische Benutzeroberfläche und ”unnütze” Dienste und Software, zum Beispiel Debian GNU/Linux. Die Installation verlief wie erwartet problemlos und es wurden noch einige unwichtige Dienste des Systems deaktiviert. Weiterhin wurde ein *Secure Shell (SSH)* Dienst eingerichtet, um dem Autor die weitere Administration und Konfiguration des Systems *remote* vom Heimarbeitsplatz zu ermöglichen. Der SSH Dienst wurde aus Sicherheitsgründen mit

Public-Key-Authentication eingerichtet und eine Passwort basierte Anmeldung ausgeschlossen.

4.3.2. Server Installation & Konfiguration

Die Installation von OpenVPN wurde über das distributionsspezfische Paketmanagement APT durchgeführt, welches neben der Software auch die automatischen Start-/Stopskripte für den Betriebssystemstart bereitstellt. Die Konfigurationsdateien wurden im Verzeichnis /etc/openvpn angelegt und orientieren sich an den mitgelieferten Beispielkonfigurationen. Die angepassten Konfigurationsdateien für den Server sind im Anhang E dargestellt und an den entscheidenden Stellen vom Autor kommentiert. Die wichtigsten Fakten:

- IP Adresse des VPN-Servers: 193.23.168.120
- es wird UDP für den Tunnel genutzt (bessere Performance)
- Pfade zum Zertifikat der CA, des VPN-Servers und der private Schlüssel müssen angegeben werden
- Für die TUN Interfaces der Clients wird ein IP-Pool aus einem privaten Bereich genommen 10.8.0.0/24
- Es werden dezidierte Routen zu folgenden Servern beim Client eingerichtet:
  - Mailserver: 193.23.168.6/32 & 193.23.168.7/32
  - Infoserver: 193.23.168.26/32
  - WINS-Server der DVZ: 193.23.168.181/32
  - Sophos Antivirus Update Service: 193.23.168.37/32
  - Windows Update Service (SUS): 193.23.168.202/32
  - NT-Webserver-Bibliothek (Authentifizierung): 193.23.168.59/32
  - Bibliothek / Brockhaus ext. bei Firma Tanto: 80.237.180.40/32
  - Bibliothek / EZB ext. bei Uni-Regensburg: 132.199.144.214/32
  - Bibliothek / HZB-NRW ext. Subnetz: 193.30.112.0/24
- Das TUN Interface der Clients bekommt per DHCP den WINS Server der DVZ zugewiesen
- Die Clients können sich nicht gegenseitig über das VPN erreichen

- Es wird standardmäßig *Blowfish* als Verschlüsselungsalgorithmus verwendet (Performance)
- *LZO-Link-Compression* wird verwendet
- OpenVPN Dienst läuft mit eingeschränkten Benutzerrechten
- *Replay* Warnungen werden unterdrückt (Stichwort: WLAN)

Die Wahl des Betriebsmodus *Routing* (Abschnitt 4.2.2) erfordert neben der Konfiguration von OpenVPN zusätzlich den Einsatz von *Network Address Translation (NAT)* und *IP-Forwarding*. Dies wird notwendig, damit die IP-Pakete der Clients mit Quell-Adressen aus dem privaten Bereich 10.8.0.0/24 mit der gültigen IP-Adresse des Servers (193.23.168.120) maskiert werden. Ohne NAT würden die Antworten eines Servers (z.B. des Infoservers) an dessen *Default-Gateway* gerichtet und nicht wie vorgesehen an den VPN-Server zum weiteren Transport an den Client. Das *Default Gateway* würde diese nicht routbaren IP-Pakete dann verwerfen.

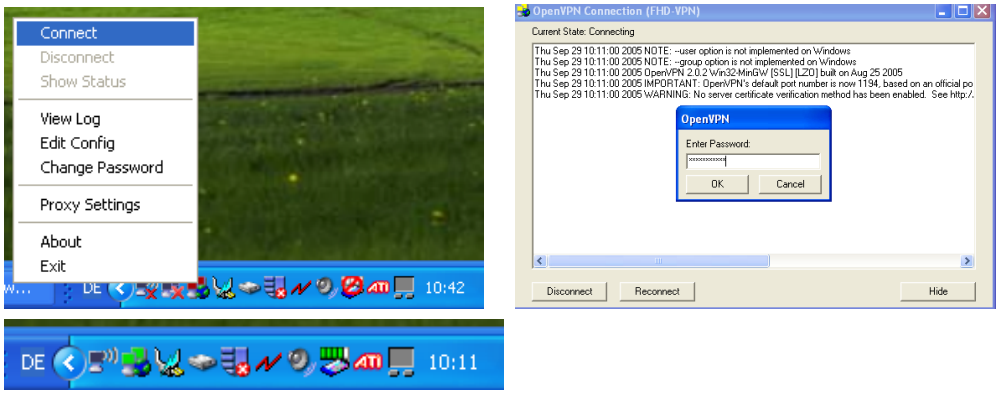
Der Linux Kernel besitzt einen leistungsfähigen zustandsorientierten Paketfilter *(netfilter)* zur Manipulation von IP-Paketen und ein *Userspace* Programm *(iptables)* zu dessen Kontrolle. Der Autor hat ein einfaches *Shellscript* für *iptables* geschrieben, um NAT und *IP-Forwarding* einzurichten und sämtlichen Datenverkehr, außer den Verkehr des OpenVPN Servers und des SSH Servers, zu blockieren. Damit dieses Script beim Start des Betriebssystems automatisch ausgeführt wird, wurde vom Autor noch ein *Init-Script* für die *Runlevel* von Ubuntu geschrieben und eingerichtet. Beide Scripte werden im Anhang G & H aufgeführt.

4.3.3. Clients

OpenVPN wird über die Kommandozeile gestartet bzw. gestoppt, und die Konfiguration kann mit jedem beliebigen Texteditor bearbeitet werden. Dies ist für einen Server praktikabel, für den Endanwender auf der Clientseite jedoch nicht. Abhilfe schaffen grafische Benutzeroberflächen (GUIs) als *Frontend* zu OpenVPN, die die Bedienung für den Anwender erleichtern. Diese GUIs[25] existieren derzeit für Betriebssysteme wie Windows, Linux und MacOS X. (Abbildung 42)

[25] http://openvpn.net/gui.html (Stand 15.09.05)

Abbildung 42: OpenVPN-GUI (Windows)



Durch dieses modulare Konzept und aufgrund der offenen Quellen lassen sich "maßgeschneiderte" OpenVPN Installationspakete in beliebigen Sprachen für den Endanwender erstellen und inklusive passender Konfigurationsdatei (Anhang F) und grafischer Benutzeroberfläche. Der Autor hat ein angepasstes Windows-Installationspaket mit dem *Nullsoft Installersystem*[26] für den OpenVPN Server in der Fachhochschule Düsseldorf erfolgreich getestet. (Abbildung 43)

Nach der Installation muss der Anwender nur noch seine PKCS#12 Datei mit privatem Schlüssel und Zertifikat im Installationspfad von OpenVPN im Unterverzeichnis "config" ablegen. Danach kann er sich, wie Abbildung 42 zeigt, mit dem OpenVPN Server verbinden. Beim Aufbau der Verbindung erhält das TUN Interface des Clients vom VPN Server über DHCP eine IP Adresse zugewiesen, und es werden die Routen für die Zielsysteme im Netzwerk der FH Düsseldorf vom Server zum Client gesendet und von diesem eingerichtet. (Abbildung 44) Ab diesem Zeitpunkt laufen alle Verbindungen zu den Zielsystemen über den VPN Server. Der Autor hat, wie im Abschnitt 4.3.2 bereits erwähnt, auf dem VPN Server unter anderem Routen für die internen Dienste *Sophos Antivirus Update Service* und *Windows Update Service* konfiguriert. Da diese Dienste auf dem *Server Message Block* (SMB)-/Windows Netzwerk-Protokoll und dessen Namensauflösung (WINS) basieren, wird dem Client bei Verbindungsaufbau per DHCP auch die Adresse des WINS Servers mitgeteilt. Damit können die Anwender des VPN Servers auch außerhalb des FH Netzwerks die Signaturen des Sophos Antivirus Programms und das Betriebssystem Windows 2000/XP aktualisieren. (Abbildung 45)

[26] *http://nsis.sourceforge.net/* (Stand 15.09.05)

Abbildung 43: angepasstes Installationspaket

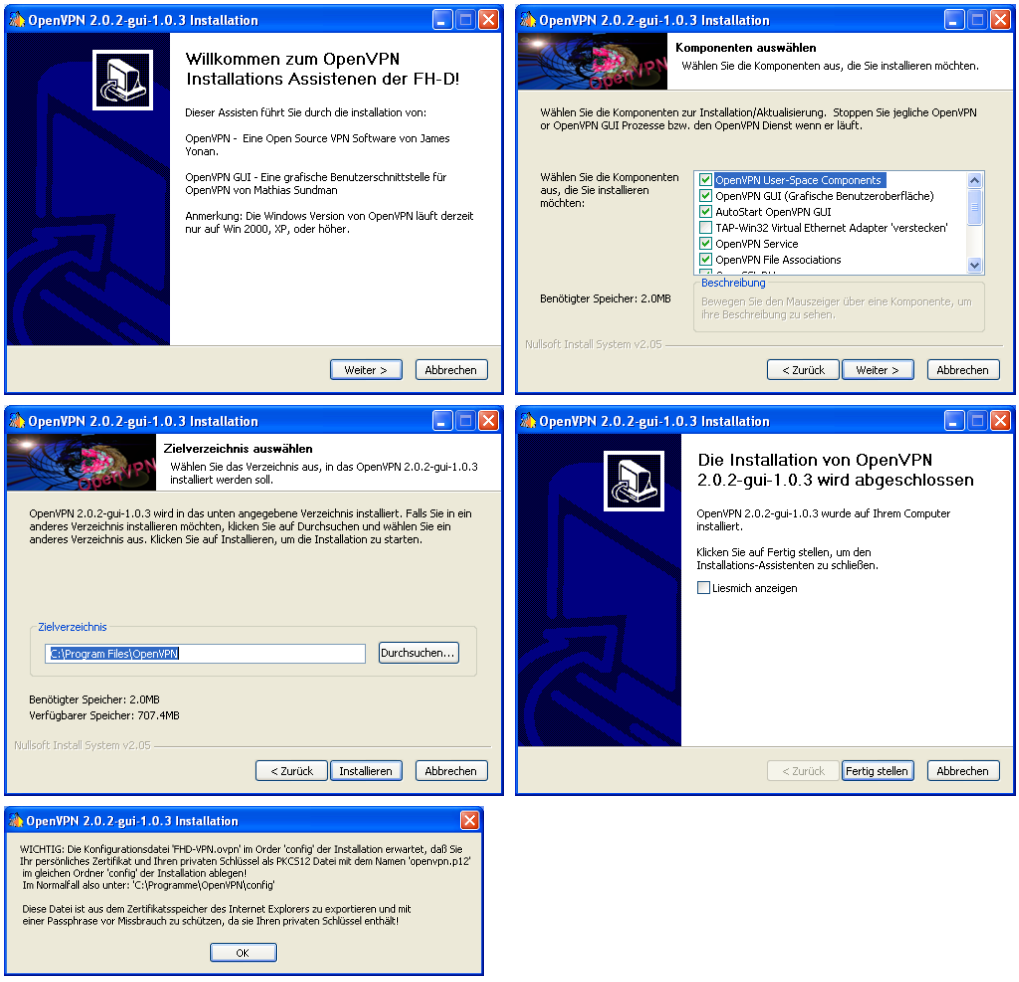
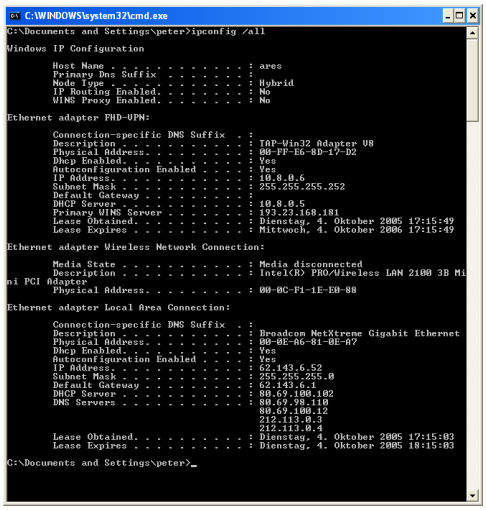


Abbildung 44: IP Konfiguration & Routing des VPN Clients



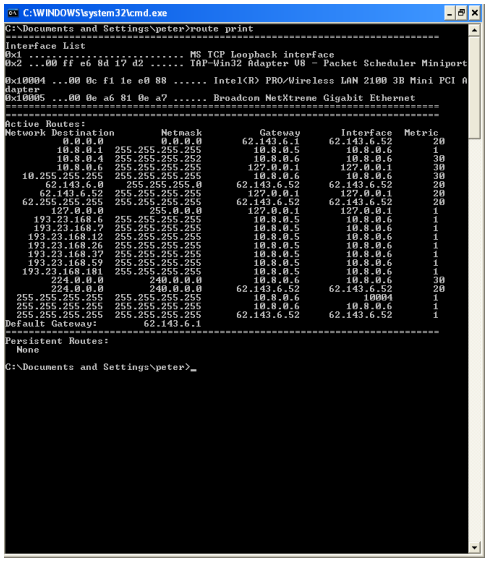
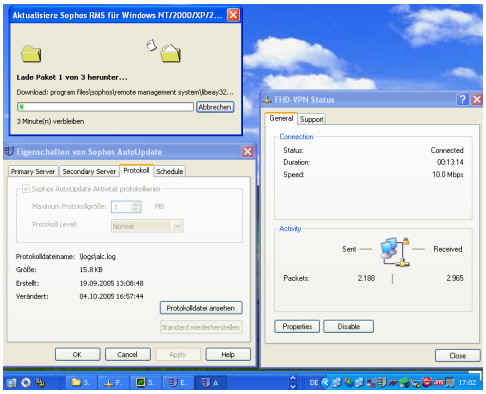
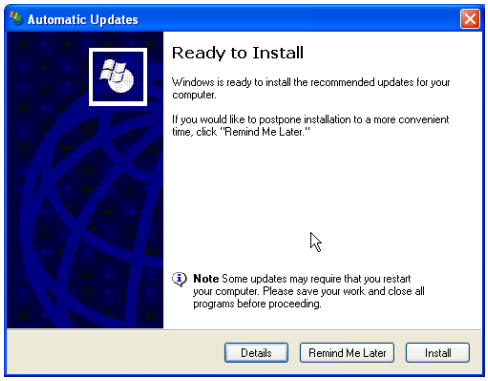
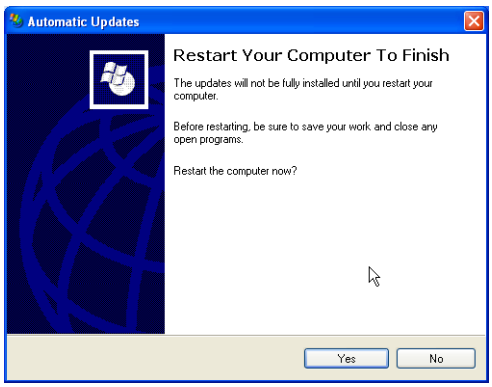


Abbildung 45: Sophos & Windows Aktualisierung







4.4. SSL-Explorer

Bei den weiteren Recherchen zum Thema VPN wurde der Autor auf ein interessantes junges *open source* Projekt der Firma *3SP* mit dem Namen SSL-Explorer aufmerksam. SSL-Explorer ist ein webbasiertes SSL-VPN, welches keinen zuvor installierten VPN-Client benötigt. Dazu wird ein Java basierter VPN-Client eingesetzt, der nach Verbindung zur Webschnittstelle von SSL-Explorer über das *Java Plug-in* des Webbrowsers gestartet wird. Dieses Konzept setzt eine installierte Java Laufzeitumgebung *(Java Runtime Environment)* voraus, was jedoch bei einem Großteil der modernen Betriebssysteme schon im Auslieferungszustand der Fall ist. Vorteil von Java ist die Plattformunabhängigkeit.

Die wichtigsten Merkmale von SSL-Explorer:

- entfernter Zugriff auf Windowsfreigaben (SMB) über *Microsoft- Webfolders*
- SSL/TLS Tunnel für beliebige TCP/UDP Verbindungen
- *Reverse Proxy Web Forwarding*
- *Active Directory* Authentifizierung
- RADIUS Authentifizierung
- UNIX Authentifizierung
- webbasierter Zugriff auf Windowsfreigaben
- *remote Desktop* Zugriff
- Zugriff über HTTP oder SOCKS Proxy
- Server lauffähig auf WindowsXP/2000/2003 und Linux

Durch die Erweiterung *SSL-Explorer Xtra* können fortgeschrittene Funktionalitäten und dezidierter kommerzieller Support durch die Firma 3SP erworben werden. Der Preis richtet sich nach Anzahl der gleichzeitigen Clientverbindungen zum VPN Server.

SSL-Explorer Xtra wird wie die Basissoftware derzeit sehr aktiv weiterentwickelt. Der derzeitige Entwicklungsstand kann auf den Webseiten von 3SP[27] und dem *Sourceforge* Projekt[28] eingesehen werden. Die geplante Funktionalität und Umfang der Erweiterung umfasst unter anderem:

[27] http://3sp.com/showSslExplorerXtra.do (Stand 15.09.05)
[28] http://sourceforge.net/projects/sslexplorer/ (Stand 15.09.05)

- Multifaktor Authentifizierung durch LDAP, SSL Client Zertifikate, SMS, public key und PIN
- Remote Management Java Applets für SSH, VNC und Telnet
- Kommerzieller Support

Der Autor konnte die Basisfunktionalitäten von SSL-Explorer mit einer Testinstallation auf einem PC im Netzwerk der Fachhochschule Düsseldorf testen. Einzig die Authentifizierung der Anwender über das RADIUS Plug-in konnte nicht erfolgreich getestet werden, sodass die Authentifizierung über die SSL-Explorer interne Datenbank vorgenommen werden musste. Das RADIUS Plug-in stand zum Zeitpunkt der Tests nicht zur Verfügung. Bei der Erstellung des schriftlichen Teils der Diplomarbeit wurde eine neue Version von SSL-Explorer mit RADIUS Funktionalität veröffentlicht.

4.4.1. Netzwerkdienste über SSL-Explorer

Die folgenden Dienste konnten über den SSL-Explorer VPN Server verfügbar gemacht bzw. geschützt werden:

- SSL Tunnel für die Mailserver (Abbildung 46)
- SSL Proxy für den Zugriff auf den Infoserver (Abbildung 47)
- SSL Proxy für die Authentifizierung der webbasierten Benutzerkonten der Bibliothek (Abbildung 47)
- SSL Proxy für den Zugriff auf Datenbanken (hier Brockhaus) der Bibliothek (Abbildung 47)
- SSL Proxy für den Zugriff auf das Webinterface des FH Mailservers (Abbildung 47)
- Webbasierter Zugriff auf die Windows Freigabe mit Sophos AV Signaturen (Abbildung 48)

Abbildung 46: SSL Tunnel Mailserver

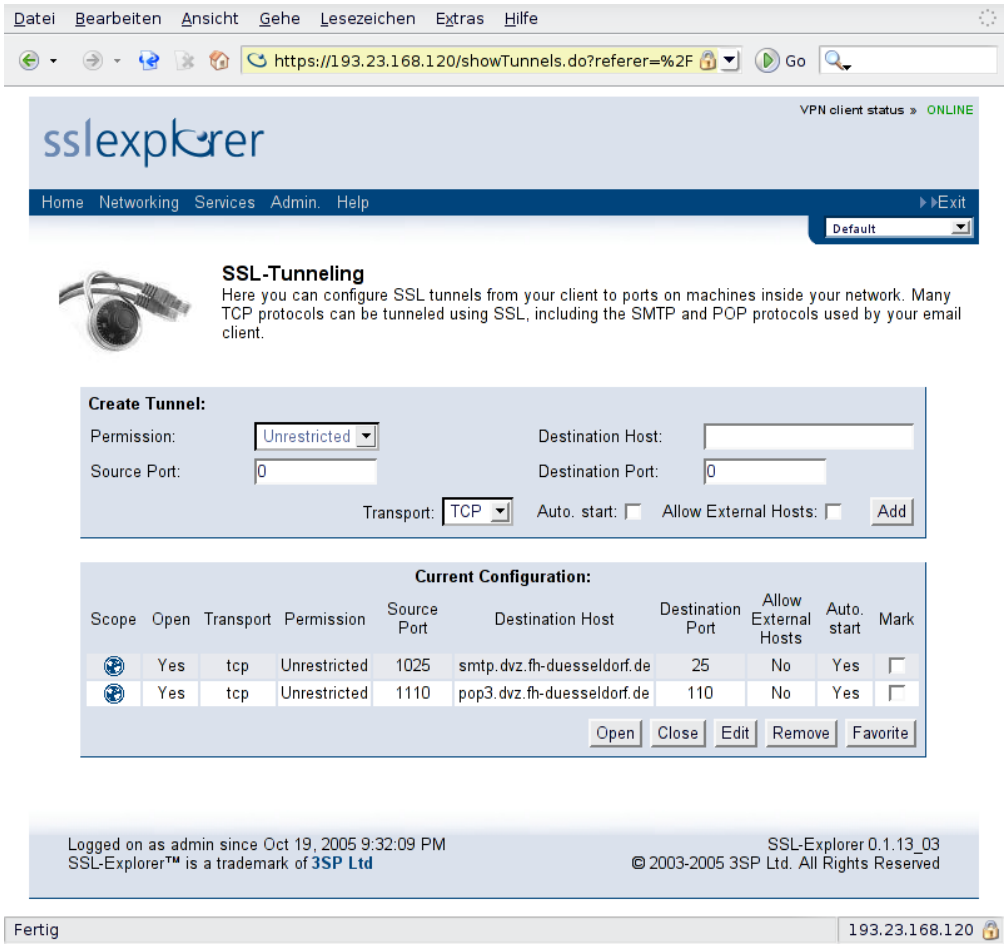


Abbildung 47: SSL Webproxies

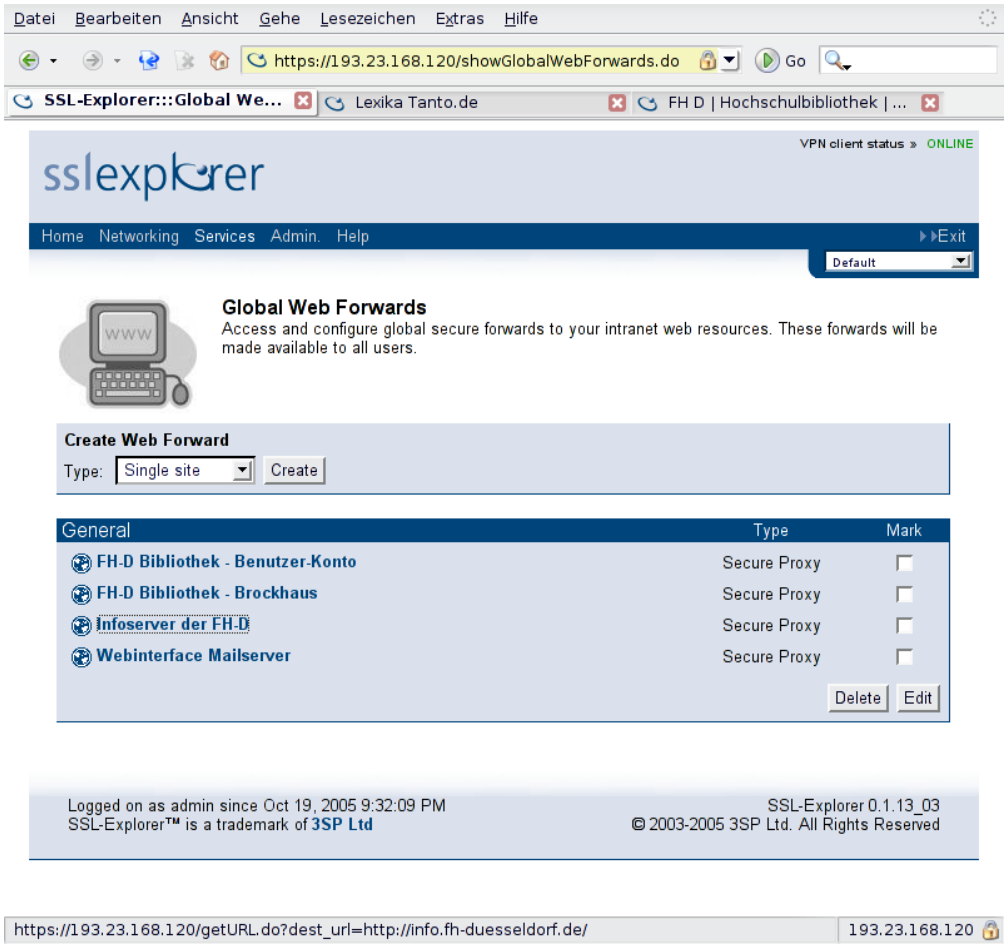
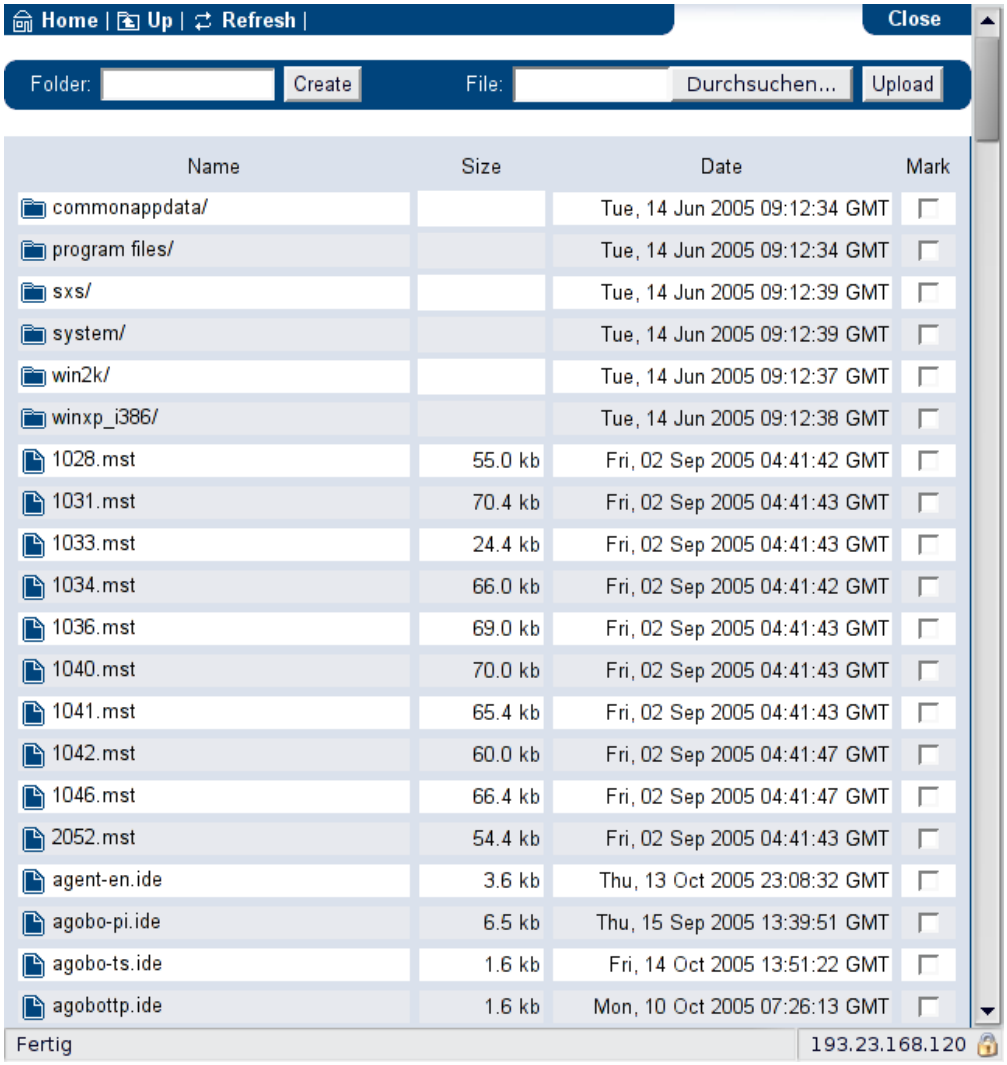


Abbildung 48: Sophos AV - SMB Freigabe



4.4.2. Vergleich mit standard VPN Konzepten

Das Konzept eines webbasierten VPN wie SSL Explorer hat Vorteile und Nachteile, die im Folgenden dargestellt werden.

Vorteile:

- Der Java VPN-Client von SSL Explorer bedarf keiner Installation und persönlichen Konfiguration durch den Anwender.
- Plattformunabhängigkeit durch Java Technologie

- Durch den *Reverse Proxy* können alle Datenbanken der Bibliothek direkt verfügbar gemacht werden. Bei OpenVPN muss für jeden Zielrechner eine eigene IP-Route auf dem Server eingerichtet werden.
- Die Anwendung der Webproxies und Windows Freigaben ist für den Anwender simpel und intuitiv.

Nachteile:

- Die Benutzung der SSL Tunnel für die Mailserver erfordert eine erneute Konfiguration von IP-Adresse und Port im Emailclient der Anwender. Bei OpenVPN entfällt dies durch die IP-Routen über die virtuelle Netzwerkschnittstelle.
- Client-Authentifizierung über Zertifikate ist nur mit kommerzieller Erweiterung möglich.
- SSL-Explorer ist nicht soweit verbreitet und getestet wie OpenVPN.
- Der Nutzer muss mit dem VPN Server über eine Webschnittstelle interagieren. Die Nutzung von OpenVPN erfordert dagegen nur die Interaktion mit dem Client beim Verbindungsaufbau /-abbau

Nach Meinung des Autors ist SSL-Explorer noch nicht ausgereift. Dies wird auch an der Versionsnummer deutlich, derzeit 0.1.13 . Das webbasierte Konzept und der Umstand der Quelloffenheit bieten aber enormes Potential.

5. Abschließendes Fazit und Perspektiven

Die vorliegende Arbeit soll den IT Verantwortlichen helfen einen Überblick über die Sicherheit der Netzwerkdienste zu erhalten und bei der Einführung neuer Dienste als Entscheidungshilfe dienen. Es wurde nicht nur Dienste im Sinne der Netzwerktechnik untersucht, sondern auch der Zugang zum Netzwerk der Fachhochschule Düsseldorf. Weiterhin wurde das Thema Hochschulverwaltungssoftware und die erweiterten Funktionen der HIS-QIS Module beschrieben, weil dies eine besonders kritische Netzwerkanwendung darstellt.

Die Einführung einer *Public Key Infrastruktur* im Rahmen des DFN-PKI Angebots ermöglicht, die Sicherheit der vorhandenen und zukünftigen Netzwerkdienste entscheidend zu verbessern. Die Anwendung der DFN-PKI Software bei

der Beantragung von Zertifikaten und die Bearbeitung dieser Anträge, wurde im Abschnitt 3.4 erläutert.

Eine direkte Anwendung der PKI Technologie wurde im Abschnitt 4.3 demonstriert. Mit dem vorgestellten VPN Server wurde die Nutzung von Diensten wie dem Mailserver abgesichert. Des Weiteren wurden ausschließlich intern nutzbare Dienste wie die Literaturdatenbanken der Bibliothek und der Infoserver außerhalb der IP Netze der Fachhochschule Düsseldorf verfügbar gemacht. Die Authentifizierung der VPN Clients durch den VPN Server basiert auf den Zertifikaten der Test-PKI.

Literatur

[AF05] Apache Foundation, *Apache Module mod_ auth_ digest*, URL (Stand 15.09.2005): http://httpd.apache.org/docs/2.0/mod/mod_auth_digest.html

[BW05] Bell Michael, Welter Oliver, *Ausweisvergabe*, Linux Magazin, Sonderheft 1/2005, URL (Stand 15.09.2005): http://www.openca.info/docs/LinuxMagazinSpecial.pdf

[Bp02] Burkholder Peter, *SSL Man-in-the-Middle Attacks*, URL (Stand 15.09.2005): http://www.sans.org/rr/whitepapers/threats/480.php

[DF05] DFN-PCA, Digitale Signatur: *Rechtliche Rahmenbedingungen*, URL: http://www.dfn-pca.de/bibliothek/sigg/

[Dd01] Ducamp Denis, *The monkey in the middle attacks*, URL (Stand 15.09.2005): http://www.groar.org/pres/MonkeyInTheMiddle/MonkeyInTheMiddle-en.htm

[FS99] Ferguson Niels, Schneier Bruce, *A Cryptographic Evaluation of IPsec*, 1998, URL (Stand 15.09.2005): http://www.schneier.com/paper-ipsec.html

[Gm05] Gast Matthew, *802.11 Wireless Networks: The Definitive Guide*, O'Reilly, 2005

[Gp03] Gutmann Peter, *Linux's answer to MS-PPTP*, 2003, URL (Stand 15.09.2005): http://www.cs.auckland.ac.nz/~pgut001/pubs/linux_vpn.txt

[Ht04] Hoeren Thomas, *Modellvergleich zur Vergabe fortgeschrittener Signaturen nach § 2 Nr. 2 Signaturgesetz,* 2004

[Hm00] Howard Michael, *Designing Secure Web-Based Applications for Microsoft® Windows® 2000*, Microsoft Press, 2000

[Hc04] Hosner Charlie, *OpenVPN and the SSL VPN Revolution*, 2004, URL (Stand 15.09.2005): http://www.sans.org/rr/whitepapers/vpns/1459.php

[Hc02] Hunt Craig, *TCP/IP Network Administration*, O'Reilly, 2000

[ND01] Nash Andrew, Duane William, Joseph Celia, Brink Derek, *PKI: e-security implementieren*, Mitp, 2001

[Pm04] Pattloch Marcus, *DFN-PKI-Strategie*, URL (Stand 15.09.2005): http://www.dfn.de/content/fileadmin/1Dienstleistungen/GWIN/sonstiges/DFNPKIStrategie.pdf

[Pm05] Pattloch Marcus, *PKI Strategie des DFN - ein neuer Dienst für DFN-Anwender*, URL (Stand 15.09.2005): http://www.hochschulverwaltung.de/tagung/braunschweig/Vortraege/MPattloch20050511DFN-PKIHV.pdf

[RF99] RFC 2617, *HTTP Authentication: Basic and Digest Access Authentication*, URL (Stand 15.09.2005): http://www.faqs.org/rfcs/rfc2617.html

[Sk98] Schmeh Klaus, *Kryptografie und Public-Key-Infrastrukturen im Internet*, d-Punkt, 1998

[Sb00] Schneier Bruce, *Secrets & Lies*, d-Punkt, 2000

[SWE98] Scott Charlie, Wolfe Paul, Erwin Mike, *Virtual Private Networks*, O'Reilly, 1998

[Sr03] Spenneberg Ralf, *VPN mit Linux*, Addison-Wesley, 2003

[Sm05] Stern Marc, *mod_ssl_error: Certificate validation error trapping*, URL (Stand 15.09.2005): http://marcstern.tripod.com/mod_ssl_error/

[Ss02] Strobel Stefan, *Firewalls und IT-Sicherheit*, d-Punkt, 2002

[Ta03] Tanenbaum Andrew S., *Computernetzwerke*, Pearson Studium, 2003

[To01] Titz Olaf, *Why TCP Over TCP Is A Bad Idea*, 2001, URL: (Stand 15.09.05):http://sites.inka.de/sites/bigred/devel/tcp-tcp.html

[VMC02] Viega John, Messier Matt, Chandra Pravir, *Network Security with OpenSSL*, O'Reilly, 2002

[Yj04] Yonan James, *Understanding the User-Space VPN*, 2004 , URL (Stand 15.09.2005):http://openvpn.net/papers/BLUG-talk/

A. Apache Konfiguration (SSL-Client-Auth.)

```
#Minimalkonfiguration eines Apache Virtualhost für
#SSL-Client-Authentifizierung mit X.509 Zertifikaten
#die wichtigen SSL Parameter sind der Dokumentation
#des MOD-SSL Moduls dokumentiert
Listen 443
<VirtualHost _default_:443>
DocumentRoot /var/www/sec
ServerName localhost
ServerAdmin peter.ophey@fh-duesseldorf.de
SSLEngine on
#Zertifikat und private key des servers
SSLCertificateFile /etc/apache/cert/pc.dvz.fh-duesseldorf.de.pem
SSLCertificateKeyFile /etc/apache/cert/pc.dvz.fh-duesseldorf.de.key
#Zertifikat der CA um Clients zu prüfen
SSLCACertificateFile /etc/apache/cert/rootcert.crt
#Clients benötigen zwingend ein gültiges Zertifikat der CA
SSLVerifyClient require
#SSLVerifyClient optional
SSLVerifyDepth 2
SSLLog /var/log/apache/ssl_engine.log
SSLLogLevel warn
</VirtualHost>
#bestimmte Pfade des Servers erfordern definierte Zertifikatsattribute
(hier OU)
<Location /~ipaudit>
SSLRequire %{SSL_CLIENT_S_DN_OU} in {"ONLY FOR TESTING PURPOSES"}
</Location>
```

B. Policies der FH-D-CA

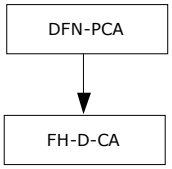
Zertifizierungsrichtlinie und Erklärung zum Zertifizierungsbetrieb der FH-D-CA in der DFN-PKI

Fachhochschule Düsseldorf
CP & CPS V1.1, 09.09.2005

1 Einleitung

Die FH-D-CA ist eine von der FH Düsseldorf betriebene Zertifizierungsstelle innerhalb der DFN-PKI.

Zur Diensterbringung wird eine Zertifizierungshierarchie verwendet, bei der das Zertifikat der FH-D-CA von der Wurzelzertifizierungsstelle der DFN-PKI, der DFN-PCA, ausgestellt wird.



2 Identifikation des Dokuments

- Titel: „Zertifizierungsrichtlinien und Erklärung zum Zertifizierungsbetrieb der FH-D-CA in der DFN-PKI"
- Version: 1.1

3 Zertifizierungsrichtlinien und Erklärung zum Zertifizierungsbetrieb

Für den Betrieb der Zertifizierungsstelle FH-D-CA gilt das folgenden Dokument uneingeschränkt:

Zertifizierungsrichtlinie der Public Key Infrastruktur im Deutschen Forschungsnetz - Classic",
Version 1.1, Februar 2005, OID 1.3.6.1.4.1.22177.300.1.1.1.1.1

Das folgende Dokument ist nicht als verbindlich anzusehen, setzt aber den grundlegenden Rahmen für den Betrieb der Zertifizierungsstelle:

"Erklärung zum Zertifizierungsbetrieb der Public Key Infrastruktur im Deutschen Forschungsnetz - Classic", Version 1.1, Februar 2005, OID 1.3.6.1.4.1.22177.300.2.1.1.1.1

Die Abweichungen zu den oben genannten Dokumenten sind in den folgenden Abschnitten beschrieben.

4 Abweichungen zu den Zertifizierungsrichtlinien der DFN-PCA

Die Zertifizierungsrichtlinien der DFN-PCA werden durch die FH-D-CA in einigen Bereichen erweitert bzw. dort, wo für die untergeordneten Zertifizierungsstellen Freiräume existieren, konkretisiert. Dies resultiert in abweichenden und ergänzenden Texten, die hier mit Verweis auf das Kapitel der Zertifizierungsrichtlinien aufgeführt sind.

Kapitel 3.1.3 Pseudonymität / Anonymität

Die FH-D-CA bietet keine Möglichkeit an, auf Verlangen einer natürlichen Person anstelle des Namens im Zertifikat ein Pseudonym aufzuführen.

Kapitel 4.1.1 Wer kann ein Zertifikat beantragen

Die FH-D-CA bietet ihre Dienstleistungen allen Angehörigen der FH Düsseldorf an.

Kapitel 4.1.2 Registrierungsprozess

Die FH-D-CA bietet bietet keine Möglichkeit zur Schlüsselerzeugung durch die Zertifizierungsstelle.

Kapitel 4.4.2 Veröffentlichung des Zertifikats

Die FH-D-CA veröffentlicht die gemäß der Zertifizierungsrichtlinien der DFN-PKI geforderten Zertifikate über die unten angegebenen Informationssysteme.

Zertifikate für natürliche Personen werden immer durch die FH-D-CA veröffentlicht.

Kapitel 4.12.1 Richtlinien und Praktiken zur Schlüsselhinterlegung und -wiederherstellung.

Die FH-D-CA bietet keine Möglichkeit zur Schlüsselhinterlegung für Schlüssel.

5 Abweichungen zu der „Erklärung zum Zertifizierungsbetrieb der DFN-PCA"

Die Erklärung zum Zertifizierungsbetrieb der DFN-PCA ist für die untergeordneten Zertifizierungsstellen nicht verbindlich, dient aber als „Best Practice". Außerdem kann eine untergeordnete Zertifizierungsstelle sich entscheiden, die Erklärung sinngemäß zu übernehmen. In diesem Fall – der für die durch den DFN-Verein im Auftrag eines DFN-Anwenders betriebenen Zertifizierungsstellen der Normalfall ist – sind nur geringfügige Abweichungen bzw. Ergänzungen notwendig.

Kapitel 1.3.1 Zertifizierungsstellen

Die Anschrift der Zertifizierungsstelle ist:

FH-D-CA (Abt. DVZ)	Telefon: +49 4351-522
Josef-Gockeln-Strasse 9	Telefax: +49 4351-523
	E-Mail: pki@fh-duesseldorf.de
D – 40474 Düsseldorf	WWW:http://www.fh-duesseldorf.de/pki

Kapitel 1.3.2 Registrierungsstellen

Die ausgezeichneten Registrierungsstellen für die zuvor genannten Zertifizierungsstellen befinden sich in den Räumen der FH-D-CA (DVZ???).

Darüber hinaus sind keine weiteren Registrierungsstellen verfügbar.

Kapitel 1.5.1 Organisation

Die Verwaltung der Richtlinien erfolgt durch:

FH Düsseldorf CA (Abt. DVZ)	Telefon: +49 4351-522
Josef-Gockeln-Strasse 9	Telefax: +49 4351-523
	E-Mail: pki@fh-duesseldorf.de
D – 40474 Düsseldorf	WWW: http://fh-duesseldorf.de/pki

Der Betrieb der unter Abschnitt 1.3. aufgeführten Zertifizierungsstellen erfolgt durch:

DFN-Verein	Telefon: +49 30 884299-23/24
Stresemannstr. 78	Telefax: +49 30 884299-70
	E-Mail: pki@dfn.de
D - 10963 Berlin	WWW: http://www.dfn.de/pki

Kapitel 1.5.2 Kontaktperson

Die verantwortliche Person für die Zertifizierungsrichtlinien und die Erklärung zum Zertifizierungsbetrieb ist:

Herr	Telefon: +49 4351-522
Dipl.-Ing. Ernst Schawohl	Telefax: +49 4351-523
Josef-Gockeln-Strasse 9	E-Mail: pki@fh-duesseldorf.de
D – 40474 Düsseldorf	WWW: http://fh-duesseldorf.de/pki

Kapitel 2.1 Verzeichnisdienst

Der Verzeichnisdienst der FH-D-CA ist unter der folgenden Bezugsadresse online zu erreichen:

- https://www.pca.dfn.de/fh-duesseldorf
- http://www.pca.dfn.de/fh-duesseldorf
- ldap://ldap.pca.dfn.de/c=DE/o=DFN-Verein/ou=DFN-PKI/o=FH-Duesseldorf

Kapitel 2.2 Veröffentlichung von Informationen

Die FH-D-CA publiziert die folgenden Informationen über den Web-Server http://www.dfn-pca.de:

- Zertifikat und Fingerabdruck:
  https://www.pca.dfn.de/fh-duesseldorf
  sowie
  http://www.pca.dfn.de/fh-duesseldorf
- Zertifizierungsrichtlinien:
  https://www.pca.dfn.de/fh-duesseldorf
  und
  http://www.pca.dfn.de/fh-duesseldorf
- Erklärung zum Zertifizierungsbetrieb
  https://www.pca.dfn.de/fh-duesseldorf
  und
  http://www.pca.dfn.de/fh-duesseldorf
- Liste der Registrierungsstellen
  https://www.pca.dfn.de/fh-duesseldorf
  und
  http://www.pca.dfn.de/fh-duesseldorf

Kapitel 3.1.1 Namensform

Die DNs aller Zertifikatnehmer unterhalb der Zertifizierungsstelle enthalten die Attribute *„C=DE*" und *„O=Fachhochschule Duesseldorf*". Im Folgenden wird zwischen **Serverzertifikaten** und **Benutzerzertifikaten** unterschieden.

Die Namensform des Zertifikatnehmers für **Serverzertifikate** entspricht grundsätzlich dem folgenden Schema:

```
C=DE
O=Fachhochschule Duesseldorf
OU=Server
CN=<FQDN>
[EMAIL=<serveradmin>@fh-duesseldorf.de]
```

Zertifikate für Server enthalten im Attribut *„OU=*" den Wert *„Server*" und im Attribut *„CN=*" einen voll qualifizierten Hostnamen (FQDN), einer der mit der Fachhochschule Düsseldorf verbundenen Domains. Das optionale Attribut *„EMAIL=*" sollte eine gültige und funktionsbezogene Emailadresse im Emailsystem der FH Düsseldorf enthalten, vorzugsweise die des Administrators des Servers, und kann einmal angegeben werden.

Bei den **Benutzerzertifikaten** wird zwischen **Studenten & Mitarbeitern** unterschieden.

Die Namensform des Zertifikatnehmers für **Studentenzertifikate** entspricht grundsätzlich dem folgenden Schema:

```
C=DE
O=Fachhochschule Duesseldorf
OU=Student
OU=<Fachbereich>
[OU=<Arbeitsgruppe>]
CN=<Vorname Name - ggf. eindeutiges Kürzel>
EMAIL=<Vorname>.<Name>@fh-duesseldorf.de
```

Zertifikate für Studenten enthalten im ersten Attribut *„OU=*" den Wert *„Student*" und im zweiten Attribut *„OU=*" den Fachbereich des Students (*FB1-Architektur, FB2-Design, FB3-Elektrotechnik, FB4-Maschinenbau&Verfahrenstechnik, FB5-Medien, FB6-Sozial&Kulturwissenschaften, FB7-Wirtschaft*). Das dritte Attribut *„OU=<Arbeitsgruppe>*" ist optional und kann einmal angegeben werden. Zur Aufnahme des dritten Attributs *„OU=<Arbeitsgruppe>*" in ein Benutzerzertifikat, ist ein entprechender Nachweis der Zugehörigkeit der Arbeitsgruppe zu erbringen, der von der Registrierungsstelle geprüft werden kann, z.B. eine schriftliche Bestätigung durch den Arbeitsgruppenleiter. Das Attribut *„CN=*" enthält den Vor- und Nachnamen des Studenten und gegebenenfalls ein eindeutiges Kürzel. Das Attribut *„EMAIL=*" enthält die gültige Emailadresse des Studenten im Emailsystem der FH Düsseldorf.

Die Namensform des Zertifikatnehmers für **Mitarbeiterzertifikate** entspricht grundsätzlich dem folgenden Schema:

```
C=DE
O=Fachhochschule Duesseldorf
OU=Mitarbeiter
OU=<Abteilung/Fachbereich>
[OU=<Arbeitsgruppe>]
CN=<Vorname Name - ggf. eindeutiges Kürzel>
EMAIL=<Vorname>.<Name>@fh-duesseldorf.de
```

Zertifikate für Mitarbeiter, Angestellte und Professoren enthalten im ersten Attribut *„OU=*" den Wert *„Mitarbeiter*" und im zweiten Attribut *„OU=*" den Fachbereich des Mitarbeiters (*FB1-Architektur, FB2-Design, FB3-Elektrotechnik, FB4-Maschinenbau&Verfahrenstechnik, FB5-Medien, FB6-Sozial&Kulturwissenschaften, FB7-Wirtschaft*) bzw. die Abteilung des Mitarbeiters (*DVZ, MKI, Verwaltung, usw.*) Das dritte Attribut *„OU=<Arbeitsgruppe>*" ist optional und kann einmal angegeben werden. Zur Aufnahme des dritten Attributs *„OU=<Arbeitsgruppe>*" in ein Benutzerzertifikat, ist ein entprechender Nachweis der Zugehörigkeit der Arbeitsgruppe zu erbringen, der von der Registrierungsstelle geprüft werden kann, z.B. eine schriftliche Bestätigung durch den Arbeitsgruppenleiter. Das Attribut *„CN=*" enthält den Vor- und Nachnamen des Mitarbeiters und gegebenenfalls ein eindeutiges Kürzel. Das Attribut *„EMAIL=*" enthält die gültige Emailadresse des Mitarbeiters im Emailsystem der FH Düsseldorf.

Kapitel 4.4.2 Veröffentlichung des Zertifikats

Die FH-D-CA veröffentlicht die gemäß der Zertifizierungsrichtlinien der DFN-PKI geforderten Zertifikate über die oben angegebenen Informationssystemen.

Zertifikate für natürliche Personen werden immer durch die FH-D-CA veröffentlicht.

Kapitel 5.8 Einstellung des Betriebs

Falls es zur Einstellung des Zertifizierungsbetriebs kommen sollte, werden folgende Maßnahmen ergriffen:

- Information der DFN-PCA mindestens drei Monate vor Einstellung der Tätigkeit.
- Information aller Zertifikatnehmer, Registrierungsstellen und betroffenen Organisationen mindestens drei Monate vor Einstellung der Tätigkeit.
- Rechtzeitiger Widerruf aller Zertifikate.
- Sichere Zerstörung der privaten Schlüssel der Zertifizierungsstelle nach Widerruf aller Zertifikate.

Die FH-D-CA stellt den Fortbestand der Archive und die Abrufmöglichkeit einer vollständigen Widerrufsliste für den zugesicherten Aufbewahrungszeitraum sicher.

Kapitel 6.1.2 Übermittlung des privaten Schlüssels an den Zertifikatnehmer

Die FH-D-CA bietet keine Möglichkeit zur Schlüsselerzeugung durch die Zertifizierungsstelle.

Kapitel 6.2.3 Hinterlegung privater Schlüssel

Die FH-D-CA bietet keine Möglichkeit zur Schlüsselhinterlegung für Schlüssel.

C. OpenSSL (Konfigurationsdatei)

```
# OpenSSL Beispielkonfigurationsdatei um die Erzeugung von CSR
# für die TEST-PKI des DFN zu erleichtern. (siehe Abschnitt [ req ]
in dieser Datei)
```

```
# Die Kommentare aus der Beispielkonfiguration sind zu Dokumentationszwecken
# beibehalten worden.
# This definition stops the following lines choking if HOME isn't
# defined.
HOME = .
RANDFILE = $ENV::HOME/.rnd
# Extra OBJECT IDENTIFIER info:
#oid_file = $ENV::HOME/.oid
oid_section = new_oids
# To use this configuration file with the "-extfile" option of the
# "openssl x509" utility, name here the section containing the
# X.509v3 extensions to use:
# extensions =
# (Alternatively, use a configuration file that has only
# X.509v3 extensions in its main [= default] section.)
[ new_oids ]
# We can add new OIDs in here for use by 'ca' and 'req'.
# Add a simple OID like this:
# testoid1=1.2.3.4
# Or use config file substitution like this:
# testoid2=${testoid1}.5.6
####################################################################
[ ca ]
default_ca = CA_default # The default ca section
####################################################################
[ CA_default ]
dir = ./demoCA # Where everything is kept
certs = $dir/certs # Where the issued certs are kept
crl_dir = $dir/crl # Where the issued crl are kept
database = $dir/index.txt # database index file.
#unique_subject = no # Set to 'no' to allow creation of
# several ctificates with same subject.
new_certs_dir = $dir/newcerts # default place for new certs.
```

```
certificate = $dir/cacert.pem # The CA certificate
serial = $dir/serial # The current serial number
#crlnumber = $dir/crlnumber # the current crl number must be
# commented out to leave a V1 CRL
crl = $dir/crl.pem # The current CRL
private_key = $dir/private/cakey.pem# The private key
RANDFILE = $dir/private/.rand # private random number file
x509_extensions = usr_cert # The extentions to add to the cert
# Comment out the following two lines for the "traditional"
# (and highly broken) format.
name_opt = ca_default # Subject Name options
cert_opt = ca_default # Certificate field options
# Extension copying option: use with caution.
# copy_extensions = copy
# Extensions to add to a CRL. Note: Netscape communicator chokes on
V2 CRLs
# so this is commented out by default to leave a V1 CRL.
# crlnumber must also be commented out to leave a V1 CRL.
# crl_extensions = crl_ext
default_days = 365 # how long to certify for
default_crl_days= 30 # how long before next CRL
default_md = sha1 # which md to use.
preserve = no # keep passed DN ordering
# A few difference way of specifying how similar the request should
look
# For type CA, the listed attributes must be the same, and the optional
# and supplied fields are just that :-)
policy = policy_match
# For the CA policy
[ policy_match ]
countryName = match
stateOrProvinceName = match
organizationName = match
organizationalUnitName = optional
```

```
commonName = supplied
emailAddress = optional
# For the 'anything' policy
# At this point in time, you must list all acceptable 'object'
# types.
[ policy_anything ]
countryName = optional
stateOrProvinceName = optional
localityName = optional
organizationName = optional
organizationalUnitName = optional
commonName = supplied
emailAddress = optional
####################################################################
[ req ]
default_bits = 1024
default_keyfile = privkey.pem
default_md = sha1
distinguished_name = req_distinguished_name
attributes = req_attributes
x509_extensions = v3_ca # The extentions to add to the self signed cert
# Passwords for private keys if not present they will be prompted for
# input_password = secret
# output_password = secret
# This sets a mask for permitted string types. There are several options.

# default: PrintableString, T61String, BMPString.
# pkix : PrintableString, BMPString.
# utf8only: only UTF8Strings.
# nombstr : PrintableString, T61String (no BMPStrings or UTF8Strings).
# MASK:XXXX a literal mask value.
# WARNING: current versions of Netscape crash on BMPStrings or UTF8Strings
# so use this option with caution!
```

```
string_mask = nombstr
# req_extensions = v3_req # The extensions to add to a certificate request
[ req_distinguished_name ]
countryName = Country Name (2 letter code)
countryName_default = DE
countryName_min = 2
countryName_max = 2
stateOrProvinceName = State or Province Name (full name)
stateOrProvinceName_default =
localityName = Locality Name (eg, city)
0.organizationName = Organization Name (eg, company)
0.organizationName_default = Test-PKI
# we can do this but it is not needed normally :-)
#1.organizationName = Second Organization Name (eg, company)
#1.organizationName_default = World Wide Web Pty Ltd
organizationalUnitName = Organizational Unit Name (eg, section)
#organizationalUnitName_default =
commonName = Common Name (eg, YOUR name)
commonName_max = 64
emailAddress = Email Address
emailAddress_max = 64
# SET-ex3 = SET extension number 3
[ req_attributes ]
challengePassword = A challenge password
challengePassword_min = 4
challengePassword_max = 20
unstructuredName = An optional company name
[ usr_cert ]
# These extensions are added when 'ca' signs a request.
# This goes against PKIX guidelines but some CAs do it and some software
# requires this to avoid interpreting an end user certificate as a CA.
basicConstraints=CA:FALSE
# Here are some examples of the usage of nsCertType. If it is omitted
```

```
# the certificate can be used for anything *except* object signing.
# This is OK for an SSL server.
# nsCertType = server
# For an object signing certificate this would be used.
# nsCertType = objsign
# For normal client use this is typical
# nsCertType = client, email
# and for everything including object signing:
# nsCertType = client, email, objsign
# This is typical in keyUsage for a client certificate.
# keyUsage = nonRepudiation, digitalSignature, keyEncipherment
# This will be displayed in Netscape's comment listbox.
nsComment = "OpenSSL Generated Certificate"
# PKIX recommendations harmless if included in all certificates.
subjectKeyIdentifier=hash
authorityKeyIdentifier=keyid,issuer:always
# This stuff is for subjectAltName and issuerAltname.
# Import the email address.
# subjectAltName=email:copy
# An alternative to produce certificates that aren't
# deprecated according to PKIX.
# subjectAltName=email:move
# Copy subject details
# issuerAltName=issuer:copy
#nsCaRevocationUrl = http://www.domain.dom/ca-crl.pem
#nsBaseUrl
#nsRevocationUrl
#nsRenewalUrl
#nsCaPolicyUrl
#nsSslServerName
[ v3_req ]
# Extensions to add to a certificate request
basicConstraints = CA:FALSE
```

```
keyUsage = nonRepudiation, digitalSignature, keyEncipherment

[ v3_ca ]

# Extensions for a typical CA

# PKIX recommendation.

subjectKeyIdentifier=hash

authorityKeyIdentifier=keyid:always,issuer:always

# This is what PKIX recommends but some broken software chokes on critical

# extensions.

#basicConstraints = critical,CA:true

# So we do this instead.

basicConstraints = CA:true

# Key usage: this is typical for a CA certificate. However since it
will

# prevent it being used as an test self-signed certificate it is best

# left out by default.

# keyUsage = cRLSign, keyCertSign

# Some might want this also

# nsCertType = sslCA, emailCA

# Include email address in subject alt name: another PKIX recommendation

# subjectAltName=email:copy

# Copy issuer details

# issuerAltName=issuer:copy

# DER hex encoding of an extension: beware experts only!

# obj=DER:02:03

# Where 'obj' is a standard or added object

# You can even override a supported extension:

# basicConstraints= critical, DER:30:03:01:01:FF

[ crl_ext ]

# CRL extensions.

# Only issuerAltName and authorityKeyIdentifier make any sense in a
CRL.

# issuerAltName=issuer:copy

authorityKeyIdentifier=keyid:always,issuer:always
```

D. OpenSSL (Kommando)

Das folgende Kommando erzeugt (für Server) ein RSA Schlüsselpaar und einen *Certificate Signing Request (CSR)*:

```
openssl req -new -sha1 -newkey rsa:1024 -nodes -keyout server.key -out
request.pem
```

E. Konfigurationsdatei OpenVPN Server

```
##################################################
# OpenVPN 2.0 Konfigurationsdatei für den VPN-Server der FH-D
# Die Kommentare aus der Beispielkonfiguration sind zu Dokumentationszwecken
# beibehalten worden.
# multi-client server.
# #
##################################################
# Which local IP address should OpenVPN
# listen on? (optional)
local 193.23.168.120
# Which TCP/UDP port should OpenVPN listen on?
# If you want to run multiple OpenVPN instances
# on the same machine, use a different port
# number for each one. You will need to
# open up this port on your firewall.
port 1194
# TCP or UDP server?
;proto tcp
proto udp
# "dev tun" will create a routed IP tunnel,
# "dev tap" will create an ethernet tunnel.
# Use "dev tap" if you are ethernet bridging.
# If you want to control access policies
# over the VPN, you must create firewall
```

```
# rules for the the TUN/TAP interface.
# On non-Windows systems, you can give
# an explicit unit number, such as tun0.
# On Windows, use "dev-node" for this.
# On most systems, the VPN will not function
# unless you partially or fully disable
# the firewall for the TUN/TAP interface.
;dev tap
dev tun
# Windows needs the TAP-Win32 adapter name
# from the Network Connections panel if you
# have more than one. On XP SP2 or higher,
# you may need to selectively disable the
# Windows firewall for the TAP adapter.
# Non-Windows systems usually don't need this.
;dev-node MyTap
# SSL/TLS root certificate (ca), certificate
# (cert), and private key (key). Each client
# and the server must have their own cert and
# key file. The server and all clients will
# use the same ca file.
# Any X509 key management system can be used.
# OpenVPN can also use a PKCS #12 formatted key file
# (see "pkcs12" directive in man page).
ca rootcert.crt
cert new/193.23.168.120.pem
key new/193.23.168.120.key # This file should be kept secret
# Diffie hellman parameters.
# Generate your own with:
# openssl dhparam -out dh1024.pem 1024
# Substitute 2048 for 1024 if you are using
# 2048 bit keys.
dh dh1024.pem
```

```
# Configure server mode and supply a VPN subnet
# for OpenVPN to draw client addresses from.
# The server will take 10.8.0.1 for itself,
# the rest will be made available to clients.
# Each client will be able to reach the server
# on 10.8.0.1. Comment this line out if you are
# ethernet bridging. See the man page for more info.
server 10.8.0.0 255.255.255.0
# Maintain a record of client <-> virtual IP address
# associations in this file. If OpenVPN goes down or
# is restarted, reconnecting clients can be assigned
# the same virtual IP address from the pool that was
# previously assigned.
ifconfig-pool-persist ipp.txt
# Configure server mode for ethernet bridging.
# You must first use your OS's bridging capability
# to bridge the TAP interface with the ethernet
# NIC interface. Then you must manually set the
# IP/netmask on the bridge interface, here we
# assume 10.8.0.4/255.255.255.0. Finally we
# must set aside an IP range in this subnet
# (start=10.8.0.50 end=10.8.0.100) to allocate
# to connecting clients. Leave this line commented
# out unless you are ethernet bridging.
;server-bridge 10.8.0.4 255.255.255.0 10.8.0.50 10.8.0.100
# Push routes to the client to allow it
# to reach other private subnets behind
# the server. Remember that these
# private subnets will also need
# to know to route the OpenVPN client
# address pool (10.8.0.0/255.255.255.0)
# back to the OpenVPN server.
#Routen für "alten" und "neuen" Mailserver
```

```
push "route 193.23.168.6 255.255.255.255"
push "route 193.23.168.7 255.255.255.255"
#und für den Infoserver zum Client pushen
push "route 193.23.168.26 255.255.255.255"
#WINS-Server der DVZ (Namensauflösung für SMB-Protokoll/Windows Netzwerk)
push "route 193.23.168.181 255.255.255.255"
#Sophos Antivirus Update Service auf \\ZNTS10
push "route 193.23.168.37 255.255.255.255"
#Windows Update Service auf \\ZNTS07
push "route 193.23.168.202 255.255.255.255"
#Server-Bibliothek
;push "route 193.23.168.12 255.255.255.255"
push "route 193.23.168.59 255.255.255.255"
#cdroms.digibib.net & digilink.digibib.net usw. im /24 Subnetz des Hochschulbibliotheks
des Landes Nordrhein-Westfalen
push "route 193.30.112.0 255.255.255.0"
#lexika.tanto.de , u.a. Brockhaus (Bibliothek), Zugriff IP beschränkt
push "route 80.237.180.40 255.255.255.255"
#Elektronische Zeitschriftenbibliothek über Uni-Regensburg (Bibliothek),
teilweise Zugriff IP beschränkt
push "route 132.199.144.214 255.255.255.255"
# To assign specific IP addresses to specific
# clients or if a connecting client has a private
# subnet behind it that should also have VPN access,
# use the subdirectory "ccd" for client-specific
# configuration files (see man page for more info).
# EXAMPLE: Suppose the client
# having the certificate common name "Thelonious"
# also has a small subnet behind his connecting
# machine, such as 192.168.40.128/255.255.255.248.
# First, uncomment out these lines:
;client-config-dir ccd
;route 192.168.40.128 255.255.255.248
# Then create a file ccd/Thelonious with this line:
```

```
# iroute 192.168.40.128 255.255.255.248
# This will allow Thelonious' private subnet to
# access the VPN. This example will only work
# if you are routing, not bridging, i.e. you are
# using "dev tun" and "server" directives.
# EXAMPLE: Suppose you want to give
# Thelonious a fixed VPN IP address of 10.9.0.1.
# First uncomment out these lines:
;client-config-dir ccd
;route 10.9.0.0 255.255.255.252
# Then add this line to ccd/Thelonious:
# ifconfig-push 10.9.0.1 10.9.0.2
# Suppose that you want to enable different
# firewall access policies for different groups
# of clients. There are two methods:
# (1) Run multiple OpenVPN daemons, one for each
# group, and firewall the TUN/TAP interface
# for each group/daemon appropriately.
# (2) (Advanced) Create a script to dynamically
# modify the firewall in response to access
# from different clients. See man
# page for more info on learn-address script.
;learn-address ./script
# If enabled, this directive will configure
# all clients to redirect their default
# network gateway through the VPN, causing
# all IP traffic such as web browsing and
# and DNS lookups to go through the VPN
# (The OpenVPN server machine may need to NAT
# the TUN/TAP interface to the internet in
# order for this to work properly).
# CAVEAT: May break client's network config if
# client's local DHCP server packets get routed
```

```
# through the tunnel. Solution: make sure
# client's local DHCP server is reachable via
# a more specific route than the default route
# of 0.0.0.0/0.0.0.0.
;push "redirect-gateway"
# Certain Windows-specific network settings
# can be pushed to clients, such as DNS
# or WINS server addresses. CAVEAT:
# http://support.microsoft.com/default.aspx?scid=kb;en-us;311218&Product=winxp
;push "dhcp-option DNS 10.8.0.1"
push "dhcp-option WINS 193.23.168.181"
# Uncomment this directive to allow different
# clients to be able to "see" each other.
# By default, clients will only see the server.
# To force clients to only see the server, you
# will also need to appropriately firewall the
# server's TUN/TAP interface.
;client-to-client
# Uncomment this directive if multiple clients
# might connect with the same certificate/key
# files or common names. This is recommended
# only for testing purposes. For production use,
# each client should have its own certificate/key
# pair.
#
# IF YOU HAVE NOT GENERATED INDIVIDUAL
# CERTIFICATE/KEY PAIRS FOR EACH CLIENT,
# EACH HAVING ITS OWN UNIQUE "COMMON NAME",
# UNCOMMENT THIS LINE OUT.
;duplicate-cn
# The keepalive directive causes ping-like
# messages to be sent back and forth over
# the link so that each side knows when
```

```
# the other side has gone down.
# Ping every 10 seconds, assume that remote
# peer is down if no ping received during
# a 120 second time period.
keepalive 10 120
# For extra security beyond that provided
# by SSL/TLS, create an "HMAC firewall"
# to help block DoS attacks and UDP port flooding.
#
# Generate with:
# openvpn --genkey --secret ta.key
#
# The server and each client must have
# a copy of this key.
# The second parameter should be '0'
# on the server and '1' on the clients.
;tls-auth ta.key 0 # This file is secret
# Select a cryptographic cipher.
# This config item must be copied to
# the client config file as well.
;cipher BF-CBC # Blowfish (default)
;cipher AES-128-CBC # AES
;cipher DES-EDE3-CBC # Triple-DES
# Enable compression on the VPN link.
# If you enable it here, you must also
# enable it in the client config file.
comp-lzo
# The maximum number of concurrently connected
# clients we want to allow.
;max-clients 100
# It's a good idea to reduce the OpenVPN
# daemon's privileges after initialization.
#
```

```
# You can uncomment this out on
# non-Windows systems.
user nobody
group nogroup
# The persist options will try to avoid
# accessing certain resources on restart
# that may no longer be accessible because
# of the privilege downgrade.
persist-key
persist-tun
# Output a short status file showing
# current connections, truncated
# and rewritten every minute.
status openvpn-status.log
# By default, log messages will go to the syslog (or
# on Windows, if running as a service, they will go to
# the "\Program Files\OpenVPN\log" directory).
# Use log or log-append to override this default.
# "log" will truncate the log file on OpenVPN startup,
# while "log-append" will append to it. Use one
# or the other (but not both).
;log openvpn.log
;log-append openvpn.log
# Set the appropriate level of log
# file verbosity.
#
# 0 is silent, except for fatal errors
# 4 is reasonable for general usage
# 5 and 6 can help to debug connection problems
# 9 is extremely verbose
verb 4
# Silence repeating messages. At most 20
# sequential messages of the same message
```

```
# category will be output to the log.
mute 20
#Wireless networks often produce a lot
# of duplicate packets. Set this flag
# to silence duplicate packet warnings.
--mute-replay-warnings
```

F. Konfigurationsdatei OpenVPN Clients

```
################################################
# Konfigurationsdatei für den OpenVPN Client
# der Fachhochschule Düsseldorf.
# Die Kommentare aus der Beispielkonfiguration sind zu Dokumentationszwecken
# beibehalten worden.
# ----------------------------------------
# Die Optionen in dieser Datei bedürfen
# keiner Änderung.
# Windows:
# Die persönliche PKCS12 Datei mit darin
# enthaltenem private Key und Zertifikat muss
# unter dem Namen "openvpn.p12" im Ordner
# "config" der OpenVPN Installation
# gespeichert werden.
# (z.B. "C:\Programme\OpenVPN\config" )
# Um den OpenVPN Dienst zu starten sind unter
# Windows Administrator Rechte nötig. Eine
# Lösung für unpreviligierte Nutzer ist
# unter der folgenden URL beschrieben:
# http://openvpn.se/install.txt
################################################
# description: VPN Verbindung ins FH-D Netz
# Specify that we are a client and that we
```

```
# will be pulling certain config file directives
# from the server.
client
# Use the same setting as you are using on
# the server.
# On most systems, the VPN will not function
# unless you partially or fully disable
# the firewall for the TUN/TAP interface.
;dev tap
dev tun
# Windows needs the TAP-Win32 adapter name
# from the Network Connections panel
# if you have more than one. On XP SP2,
# you may need to disable the firewall
# for the TAP adapter.
;dev-node MyTap
# Are we connecting to a TCP or
# UDP server? Use the same setting as
# on the server.
;proto tcp
proto udp
# The hostname/IP and port of the server.
# You can have multiple remote entries
# to load balance between the servers.
remote 193.23.168.120 1194
;remote my-server-2 1194
# Choose a random host from the remote
# list for load-balancing. Otherwise
# try hosts in the order specified.
;remote-random
# Keep trying indefinitely to resolve the
# host name of the OpenVPN server. Very useful
# on machines which are not permanently connected
```

```
# to the internet such as laptops.
resolv-retry infinite
# Most clients don't need to bind to
# a specific local port number.
nobind
# Downgrade privileges after initialization (non-Windows only)
user nobody
group nogroup
# Try to preserve some state across restarts.
persist-key
persist-tun
# If you are connecting through an
# HTTP proxy to reach the actual OpenVPN
# server, put the proxy server/IP and
# port number here. See the man page
# if your proxy server requires
# authentication.
;http-proxy-retry # retry on connection failures
;http-proxy [proxy server] [proxy port #]
# Wireless networks often produce a lot
# of duplicate packets. Set this flag
# to silence duplicate packet warnings.
;mute-replay-warnings
# SSL/TLS parms.
# See the server config file for more
# description. It's best to use
# a separate .crt/.key file pair
# for each client. A single ca
# file can be used for all clients.
#ca cacert.crt
#cert cert-121706461.pem
#key client.key
pkcs12 openvpn.p12
```

```
# Verify server certificate by checking
# that the certicate has the nsCertType
# field set to "server". This is an
# important precaution to protect against
# a potential attack discussed here:
# http://openvpn.net/howto.html#mitm
#
# To use this feature, you will need to generate
# your server certificates with the nsCertType
# field set to "server". The build-key-server
# script in the easy-rsa folder will do this.
;ns-cert-type server
# If a tls-auth key is used on the server
# then every client must also have the key.
;tls-auth ta.key 1
# Select a cryptographic cipher.
# If the cipher option is used on the server
# then you must also specify it here.
;cipher x
# Enable compression on the VPN link.
# Don't enable this unless it is also
# enabled in the server config file.
comp-lzo
# Set log file verbosity.
verb 3
# Silence repeating messages
;mute 20
```

G. iptables shellscript

```
#!/bin/sh
# iptables Script
# Eingehend SSHD, https (SSL-Explorer) und Openvpn erlauben, ''Rest''
blocken
```

```
# OpenVPN-Clients aus 10.8.0.0/24 maskieren (NAT), ip-forwarding aktivieren
#
IPT="/sbin/iptables"
MOD="/sbin/modprobe"
status="0"
startFirewall() {
echo
echo "Starting firewall..."
echo -n "Loading needed modules... "
$MOD ip_tables
$MOD ip_conntrack
$MOD ipt_LOG
$MOD ipt_limit
$MOD ipt_state
$MOD ip_conntrack_ftp
$MOD ip_conntrack_irc
$MOD iptable_filter
$MOD iptable_nat
$MOD iptable_mangle
echo "Done."
# Define all custom chains
echo -n "Create custom chains... "
echo "Done."
# Rules:
echo "Settup Rules in Table FILTER:
"
# Define Rules for Chain: INPUT
echo -n "Create Rules for Chain: INPUT "
$IPT -t filter -A INPUT --in-interface lo -j ACCEPT || { status="1";
echo "Setting up Rule: Loopback FAILED !!!"; exit 1; }
$IPT -t filter -A INPUT --match state --state RELATED,ESTABLISHED -j
ACCEPT || { status="1"; echo "Setting up Rule: statefull FAILED !!!";
exit 1; }
$IPT -t filter -A INPUT --protocol tcp --destination-port 1122 -j ACCEPT
|| { status="1"; echo "Setting up Rule: sshd FAILED !!!"; exit 1; }
```

```
$IPT -t filter -A INPUT --protocol tcp --destination-port 443 -j ACCEPT
|| { status="1"; echo "Setting up Rule: SSL-Explorer FAILED !!!"; exit
1; }

$IPT -t filter -A INPUT --protocol udp --destination-port 1194 -j ACCEPT
|| { status="1"; echo "Setting up Rule: OpenVPN FAILED !!!"; exit 1;
}

$IPT -t filter -P INPUT DROP || { status="1"; echo "Setting up Rule:
Chain: INPUT Default Target FAILED !!!"; exit 1; }

echo "Done."

# Define Rules for Chain: OUTPUT

echo -n "Create Rules for Chain: OUTPUT "

$IPT -t filter -A OUTPUT --out-interface lo -j ACCEPT || { status="1";
echo "Setting up Rule: Loopback FAILED !!!"; exit 1; }

$IPT -t filter -A OUTPUT --match state --state NEW,RELATED,ESTABLISHED
-j ACCEPT || { status="1"; echo "Setting up Rule: statefull FAILED !!!";
exit 1; }

$IPT -t filter -P OUTPUT ACCEPT || { status="1"; echo "Setting up Rule:
Chain: OUTPUT Default Target FAILED !!!"; exit 1; }

echo "Done."

# Define Rules for Chain: FORWARD

echo -n "Create Rules for Chain: FORWARD "

$IPT -t filter -P FORWARD ACCEPT || { status="1"; echo "Setting up Rule:
Chain: FORWARD Default Target FAILED !!!"; exit 1; }

echo "Done."

echo "Settup Rules in Table NAT:

"

# Define Rules for Chain: OUTPUT

echo -n "Create Rules for Chain: OUTPUT "

$IPT -t nat -P OUTPUT ACCEPT || { status="1"; echo "Setting up Rule:
Chain: OUTPUT Default Target FAILED !!!"; exit 1; }

echo "Done."

# Define Rules for Chain: PREROUTING

echo -n "Create Rules for Chain: PREROUTING "

$IPT -t nat -P PREROUTING ACCEPT || { status="1"; echo "Setting up Rule:
Chain: PREROUTING Default Target FAILED !!!"; exit 1; }

echo "Done."

# Define Rules for Chain: POSTROUTING
```

```
echo -n "Create Rules for Chain: POSTROUTING "

$IPT -t nat -A POSTROUTING --source 10.8.0.0/24 -j MASQUERADE || { status="1";
echo "Setting up Rule: NAT4OpenVPN-Clients FAILED !!!"; exit 1; }

$IPT -t nat -P POSTROUTING ACCEPT || { status="1"; echo "Setting up
Rule: Chain: POSTROUTING Default Target FAILED !!!"; exit 1; }

echo "Done."

echo "Settup Rules in Table MANGLE:

"

# Define Rules for Chain: INPUT

echo -n "Create Rules for Chain: INPUT "

$IPT -t mangle -P INPUT ACCEPT || ( status="1" && echo "Setting up Rule:
Chain: INPUT Default Target FAILED !!!" && exit 1 )

echo "Done."

# Define Rules for Chain: OUTPUT

echo -n "Create Rules for Chain: OUTPUT "

$IPT -t mangle -P OUTPUT ACCEPT || ( status="1" && echo "Setting up
Rule: Chain: OUTPUT Default Target FAILED !!!" && exit 1 )

echo "Done."

# Define Rules for Chain: FORWARD

echo -n "Create Rules for Chain: FORWARD "

$IPT -t mangle -P FORWARD ACCEPT || ( status="1" && echo "Setting up
Rule: Chain: FORWARD Default Target FAILED !!!" && exit 1 )

echo "Done."

# Define Rules for Chain: PREROUTING

echo -n "Create Rules for Chain: PREROUTING "

$IPT -t mangle -P PREROUTING ACCEPT || ( status="1" && echo "Setting
up Rule: Chain: PREROUTING Default Target FAILED !!!" && exit 1 )

echo "Done."

# Define Rules for Chain: POSTROUTING

echo -n "Create Rules for Chain: POSTROUTING "

$IPT -t mangle -P POSTROUTING ACCEPT || ( status="1" && echo "Setting
up Rule: Chain: POSTROUTING Default Target FAILED !!!" && exit 1 )

echo "Done."

echo -n "Enable IP Forwarding. "

echo 1 > /proc/sys/net/ipv4/ip_forward
```

```
echo "Done."
echo -n "Enable Reverse Path Filtering "
for i in /proc/sys/net/ipv4/conf/*/rp_filter ; do
echo 2 > $i
done
echo "Done."
echo -n "Disable log_martians (logging). "
for i in /proc/sys/net/ipv4/conf/*/log_martians ; do
echo 0 > $i
done
echo "Done.
"
echo -n "Enable Syn Cookies. "
echo 1 > /proc/sys/net/ipv4/tcp_syncookies
echo "Done."
}
stopFirewall() {
echo -n "Shutdown Firewall... "
$IPT -t filter -F || status="1"
$IPT -t filter -X || status="1"
$IPT -t filter -P INPUT ACCEPT || status="1"
$IPT -t filter -P OUTPUT ACCEPT || status="1"
$IPT -t filter -P FORWARD ACCEPT || status="1"
$IPT -t nat -F || status="1"
$IPT -t nat -X || status="1"
$IPT -t nat -P OUTPUT ACCEPT || status="1"
$IPT -t nat -P PREROUTING ACCEPT || status="1"
$IPT -t nat -P POSTROUTING ACCEPT || status="1"
$IPT -t mangle -F || status="1"
$IPT -t mangle -X || status="1"
$IPT -t mangle -P INPUT ACCEPT || status="1"
$IPT -t mangle -P OUTPUT ACCEPT || status="1"
$IPT -t mangle -P OUTPUT ACCEPT || status="1"
```

```
$IPT -t mangle -P PREROUTING ACCEPT || status="1"
$IPT -t mangle -P POSTROUTING ACCEPT || status="1"
echo "Done."
echo -n "Disable IP Forwarding. "
echo 0 > /proc/sys/net/ipv4/ip_forward
echo "Done."
}
case $1 in
start)
stopFirewall
startFirewall
;;
stop)
stopFirewall
;;
restart)
stopFirewall
startFirewall
;;
*)
echo "Usage: sh SSL-VPN-Firewall.sh { start | stop | restart } "
;;
esac
if [ "$status" = "1" ]; then
exit 1
else
exit 0
fi
```

H. init-script für iptables shellscript

```
#!/bin/sh
# einfaches rc-Script für die Firwall
```

```
#
status="0"
case $1 in
start)
/bin/sh /etc/firewall/SSL-VPN-Firewall.sh start || status="1"
;;
stop)
/bin/sh /etc/firewall/SSL-VPN-Firewall.sh stop || status="1"
;;
reload)
/bin/sh /etc/firewall/SSL-VPN-Firewall.sh stop || status="1"
/bin/sh /etc/firewall/SSL-VPN-Firewall.sh start || status="1"
;;
esac
if [ "$status" = "0" ]; then
exit 0
else
exit 1
fi
```

Eidesstattliche Erklärung

Ich erkläre hiermit an Eides statt, dass ich die vorliegende Diplomarbeit selbstständig und ohne fremde Hilfe verfasst habe.

Alle Stellen, die wörtlich oder sinngemäß aus veröffentlichten oder nichtveröffentlichten Schriften entnommen sind, habe ich als solche kenntlich gemacht.

Düsseldorf, 20.10.2005