Fachhochschule Karlsruhe – Hochschule für Technik
Fachbereich Informatik
Alexander Fiedler, Mat-Nr.: 011236

Studienarbeit I 5141

Apache mit SSL

Installation und Konfiguration mit Testanwendung

1 Inhaltsverzeichnis

2 Motivation

Das Internet gewinnt immer mehr an Popularität und mit ihm auch der elektronische Handel oder E-Commerce. Unter E-Commerce versteht man das Angebot von Dienstleistungen bzw. der Handel mit Gütern über das Medium Internet, wie z.B. Online-Banking, Internet-Auktionen oder auch die jüngsten Versuche von Behörden, Dienstleistungen wie Anträge, etc. Online abzuwickeln.

Dies bringt eine ganze Menge Vorteile und Erleichterungen mit sich. Gerade ältere, kranke und behinderte Menschen können so bei der Bewältigung ihres Alltags unterstützt werden. Auch die langen Wartezeiten in den Wartesälen der Behörden und oftmals auch der Gang zur Post können eingespart werden.

Doch leider hat der E-Commerce nicht nur gute Seiten, sondern birgt auch einige Nachteile und Risken. Ein solches Risiko ist, dass die Verbindungen über das Internet standardmäßig nicht gesichert sind. Daten, die über das Internet verschickt werden, können von jedermann abgefangen und gelesen werden. Gerade im Hinblick auf E-Commerce, wo vertrauliche Daten wie z.B. Konto- bzw. Kreditkartendaten, etc. ausgetauscht werden, ist diese Sicherheit der Daten aber unbedingt notwendig.

Eine Möglichkeit der Absicherung der Übertragung ist ein Protokoll namens Secure Sockets Layer (SSL), das Datenströme übers Internet verschlüsselt und somit gegen mitlesen und verändern absichert.

Diese Studienarbeit handelt von der Installation und Konfiguration eines Web-Servers mit einem SSL-Modul und soll für die Fachhochschule Karlsruhe, insbesondere für die Vorlesung „Elektronische Medien und Märkte" im Fachbereich Informatik die vorhandenen Erfahrungen erweitern und die Vorlesung aufwerten.

Weiter soll sie auch nachfolgenden Studentinnen und Studenten als unterstützendes Dokument bei der Installation und Konfiguration eines Web-Servers mit SSL-Unterstützung dienen.

Um die eigenen Erfahrungen im Umgang mit einem Web-Server mit SSL-Unterstützung zu erweitern, ist dieser Studienarbeit eine Testanwendung zugehörig, die mittels PHP Datenbankzugriffe tätigt und über eine sichere Verbindung mit SSL in einem Web-Browser anzeigt.

3 Secure Sockets Layer

3.1 Was ist SSL?

Secure Sockets Layer (SSL) ist ein Protokoll, das unabhängig vom Protokoll der Anwendungsschicht den Datenstrom über das Internet verschlüsselt. Es wurde im Jahr 1994 von Netscape entwickelt und 1999 von der IETF als TLS 1.0 1999 (Transport Layer Security) standardisiert. Genauere Infos zum IETF-Standard TLS sind im RFC 2246 (http://www.ietf.org/rfc/rfc2246.txt) hinterlegt.

Es ist im TCP/IP-Protokollstack zwischen der Anwendungs- und Transportschicht angeordnet.

Anwendungsschicht	SMTP	HTTP	FTP
Secure Socket Layer	SSL		
Transportschicht	TCP	UDP	
Internetschicht	IP		
Netzzugangsschicht	Ethernet, FDDI, ISDN		

Abbildung 1: TCP/IP-Protokollstack mit SSL

SSL sichert dabei nur die Verbindung zweier Hosts über das Internet. Auf den Hosts selbst liegen die Daten wieder in ungeschütztem Klartext vor, der durch gezielte Angriffe auf den Host gelesen werden kann.

SSL setzt eine Authentifizierung mittels X.509 Zertifikaten (4 Zertifikate, Seite 15) voraus.

Mittlerweile ist SSL der Standard für gesicherte Nachrichten, die über das Internet übertragen werden und wird besonders beim Austausch vertraulicher Daten, wie z.B. Kreditkarteninformationen, eingesetzt.

3.2 Funktion

Wie bereits erwähnt, ist SSL ein Sicherheitsprotokoll, das den Datenverkehr übers Internet verschlüsseln und damit sichern kann.

Das SSL-Protokoll ermöglicht eine sichere Verbindung mit drei Eigenschaften:

- **Die Verbindung ist gesichert und privat**:
  Durch das anfänglich genutzte Handshake-Protokoll wird ein geheimer Schlüssel vereinbart, der die weitere Kommunikation durch symmetrische Kryptographie sichert. Beispiele für von SSL eingesetzte symmetrische Verschlüsselungsverfahren sind Triple-DES, IDEA, RC4, RC2 und DES.
- **Authentifizierung der Kommunikationspartner:**
  Durch Zertifikate können sich die Kommunikationspartner authentifizieren. Standardmäßig tut dies der angefragte Server gegenüber dem anfragenden Client und optional ist auch eine Authentifizierung des Clients gegenüber dem Server möglich.
- **Verlässliche Verbindung:**
  Die über SSL verschickten Nachrichten werden zusätzlich durch eine mitgeschickte Prüfsumme, die mittels einer Hash-Funktion ermittelt wird, geschützt. Veränderungen an den empfangenen Daten können somit entdeckt werden. SSL benutzt dazu die Hash-Funktionen MD5 und SHA-1.

SSL selbst ist dabei wieder aus zwei Schichten aufgebaut, die wieder aus verschiedenen Protokollen bestehen.

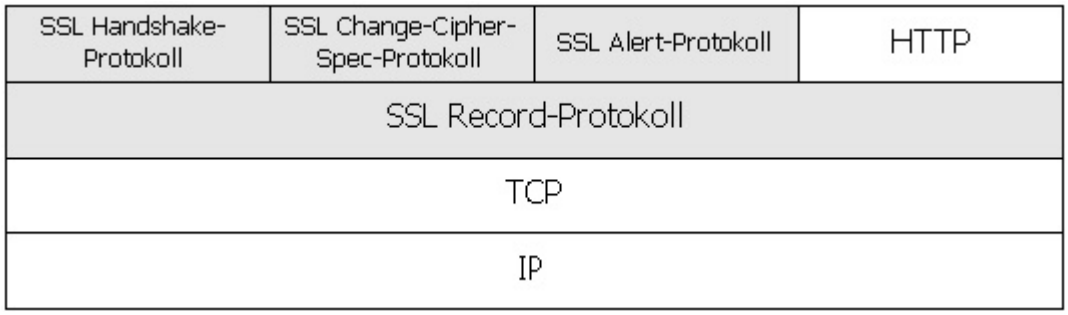

Abbildung 2: Aufbau der SSL-Protokollschichten

3.2.1 SSL Record-Protokoll

Das SSL Record-Protokoll empfängt auf dem sendenden Rechner von den übergeordneten SSL-internen Protokollen und den Protokollen der Anwendungsschicht (z.B. http, etc.) Daten in nichtleeren Blöcken beliebiger Größe.

Auf diese erhaltenen Daten wendet das Record-Protokoll dann die folgenden Operationen an:

- **Zerlegung der Daten:**
  Das Record-Protokoll zerlegt die erhaltenen Daten in Blöcke (sog. SSLPlaintext-Records), die eine Größe von 2^{14} Bytes oder weniger haben. Zusätzlich wird den SSLPlaintext-Records noch das Protokoll der Anwendungsschicht, das die Daten verarbeiten kann, als Information hinzugefügt.
- **Komprimierung der Daten:**
  Die SSLPlaintext-Records werden dann mit dem durch das Handshake-Protokoll ausgehandelten Kompressionsverfahren verlustfrei komprimiert. Der Kompressions-Algorithmus übersetzt die SSLPlaintext-Records dabei in SSLCompressed-Records.
- **Schutz der Records durch MAC:**
  Den Datenblöcken wird nun eine durch den in der CipherSpec (Sepzifikation der Chiffrierung) festgelegten MAC-Algorithmus errechnete Prüfsumme angehängt, die es dem empfangenden Rechner erlaubt, die Richtigkeit der Nachricht festzustellen.
- **Verschlüsseln der Nachricht:**
  Die Nachricht wird nun durch den ebenfalls in der aktuellen Cipher-Spec festgelegten Verschlüsselungsalgorithmus verschlüsselt. So wird aus den Datenblöcken SSLCiphertext.
- **Anfügen eines SSL-Headers:**
  Zum Schluss wird dem SSLCiphertext noch ein SSL-Header angehängt, der unter anderem eine Sequenznummer enthält, die es dem empfan-

genden Rechner erlaubt, fehlende, zusätzliche oder fehlerhafte Nachrichten zu erkennen.

Auf dem empfangenden Rechner wird das SSL Record-Protokoll in umgekehrter Reihenfolge durchlaufen. Die einzelnen Operationen werden dabei umgekehrt auf die empfangenen Daten angewendet, so dass nach Durchlaufen des Protokolls die Daten wieder für das Protokoll der Anwendungsschicht lesbar sind.

3.2.2 SSL Handshake-Protokoll

Mittels dem SSL Handshake-Protokoll verhandeln die beiden an der Kommunikation beteiligten Rechner die Bedingungen für den Datenaustausch, da verschiedene Verschlüsselungsverfahren und innerhalb dieser Verfahren wieder verschiedene Algorithmen unterstützt werden. Diese Verhandlungen werden mittels Nachrichten geführt, die folgend aufgebaut sind:

- Typ der Nachricht: 1Byte
- Länge der Nachricht: 3 Byte
- Inhalt der Nachricht
-

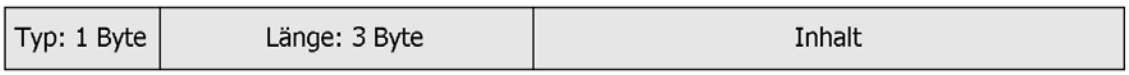

Abbildung 3: Aufbau der Nachrichten des SSL Handshake-Protokolls

Der Client startet die Verhandlung mit einer `ClientHello`-Nachricht. Mit dieser Nachricht übermittelt der Client dem Server folgende Informationen:

- Die höchste SSL-Version des Clients
- Eine 32 Byte Zufallszahl, die aus einer 4 Byte Zeitmarke und einer 28 Byte Zufallszahl besteht.

- Eine Session_ID:
  - Bei einer neuen Session ist diese ID Null.
  - Wird eine bereits existierende Session_ID mitgeschickt, werden die bereits verhandelten Parameter verändert.
- Die CipherSuite, die die kryptografischen Methoden des Clients nach Priorität geordnet enthält.
- Eine Liste der Kompressionsverfahren des Clients.

Der Server antwortet auf das `ClientHello` mit einem `ServerHello`, das eine Liste der identischen Parameter und die vom Server getroffene Auswahl für je ein asymmetrisches und ein symmetrisches Verschlüsselungsverfahren und ein Hash-Verfahren enthält.

Weiter sendet der Server mit `Certificate` ein oder mehrere X.509-Zertifikate an den Client, die den Server authentifizieren, und mit `ServerKeyExchange` eine Public Key Information, die gegebenenfalls mit Schlüsseln aus dem Zertifikat signiert ist.

Optional kann nun der Server mit `CertificateRequest` eine Authentifizierung des Clients verlangen.

Hat der Server alle diese Informationen an den Client geschickt, sendet er zum Abschluss die Nachricht `ServerHelloDone`.

Nun ist der Client wieder an der Reihe und kann, falls ein Client-Zertifikat angefordert wurde, mittels der Nachricht `Certificate` sein Client-Zertifikat an den Server schicken. Mit `CertificateVerify` sendet der Client nun noch die zum Zertifikat zugehörige Signatur an den Server, damit dieser die Echtheit des Zertifikats überprüfen kann. Sendet der Client kein oder ein ungültiges Zertifikat, bricht der Server die Kommunikation sofort ab.

Der Client prüft nun die mit der ServerHello-Nachricht erhaltenen Parameter und die Zertifikate. Nach erfolgreicher Prüfung generiert der Client einen zufälligen 48 Byte Wert, der `PreMasterSecret` genannt wird. Dieser `PreMaster-`

`Secret` wird mit dem Public Key des Servers verschlüsselt und mit der Nachricht `ClientKeyExchange` an den Server geschickt.

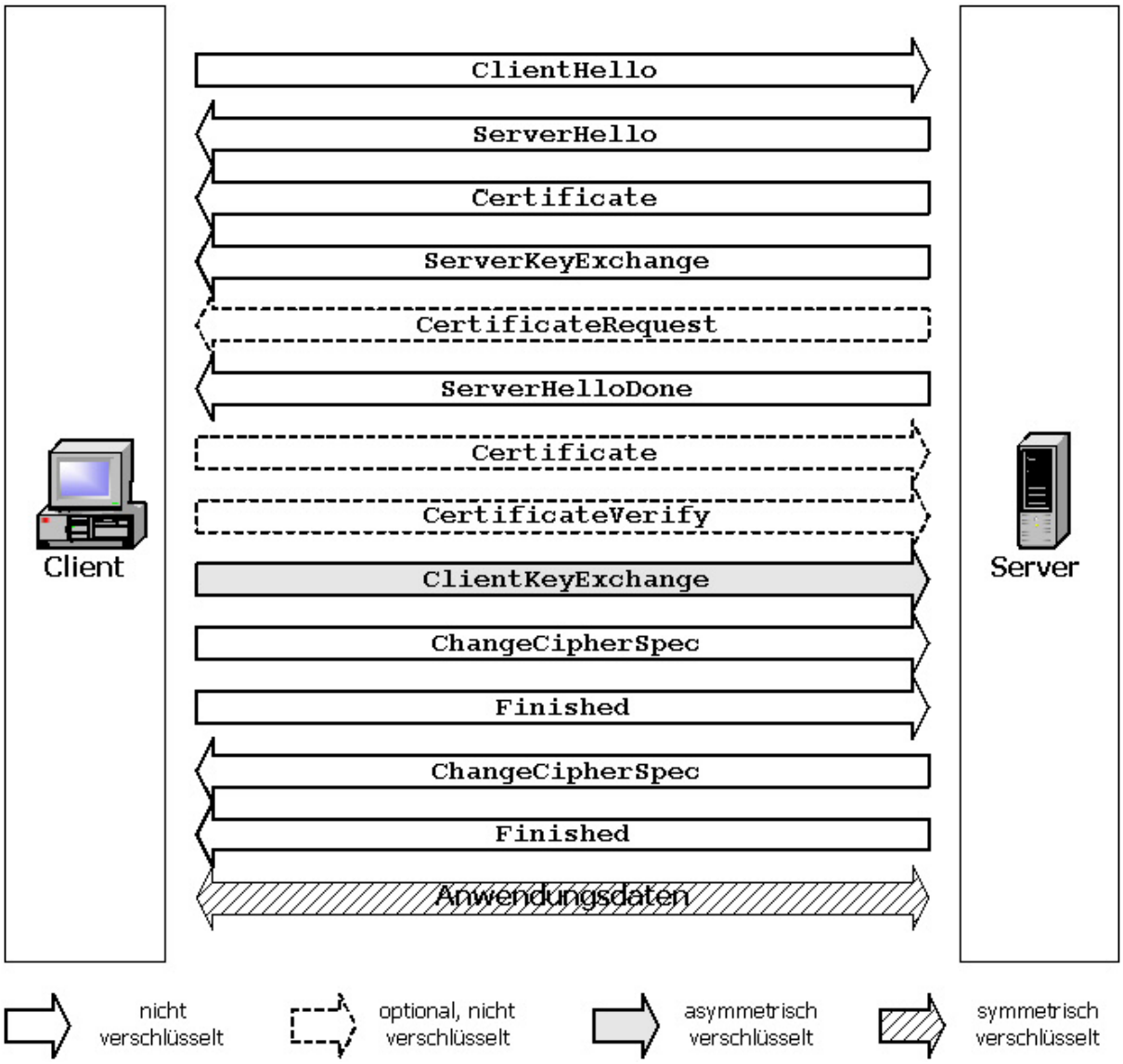

Abbildung 4: Das SSL Handshake-Protokoll

Aus allen Daten des Handshake-Protokolls wird nun auf beiden Seiten ein `MasterSecret` errechnet, der für die eigentliche symmetrische Verschlüsselung der Anwendungsdaten verwendet wird. Ist dies geschehen, bestätigt der Client mit der Nachricht `ChangeCipherSpec`, dass die Schlüsselgenerierung abgeschlossen ist und beendet mit der Nachricht `Finished` das Handshake-Protokoll. Der Server bestätigt dem Client mit den selben Nachrichten, dass

auch er den `MasterSecret` berechnet hat und zum verschlüsselten Datenaustausch bereit ist.

3.2.3 SSL Change-Cipher-Spec-Protokoll

Das Change-Cipher-Spec-Protokoll signalisiert Änderungen in der Chiffrierung. Es wird sowohl an den Client als auch an den Server geschickt. Es wird so dem Client und dem Server bekannt gemacht, dass ab sofort der Datenverkehr mit der mittels dem Handshake-Protokoll verhandelten ChiperSpec verschlüsselt wird.

3.2.4 SSL Alert-Protokoll

Mit dem Alert-Protokoll werden Unregelmäßigkeiten bei der Übertragung mittels SSL angezeigt. Alert-Nachrichten enthalten die Wichtigkeit und eine detaillierte Beschreibung des Alarms. So führen Alert-Nachrichten mit der Priorität `fatal result` zum sofortigen Abbruch der Verbindung. In diesem Fall können zwar bereits bestehende Verbindungen zur ungültigen Sitzung weiterlaufen, aber die Session_ID muss auf den Status ungültig gesetzt werden, damit keine neue Verbindung mit dieser Sitzung aufgebaut werden kann.

Grundsätzlich werden folgende Arten von Alert-Nachrichten unterschieden:

- **Closure Alerts**
  Sowohl der Client als auch der Server müssen Kenntnis über die Beendigung einer Sitzung haben. Dadurch können Truncation-Attacken, die die Verbindung unterbrechen und Nachrichtenteile ausschneiden, verhindert werden. Wird eine Verbindung ohne die `close_notify`-Nachricht abgebrochen, wird die aktuelle Sitzung sofort ungültig.

- **Error Alerts**

  Die Fehlerbehandlung im SSL Handshake-Protokoll funktioniert nach einem einfachen Prinzip: Wird von einer der verhandelnden Parteien ein Fehler festgestellt, so schickt diese Partei sofort eine Nachricht an die andere Partei. Hat diese Nachricht den Prioritätsstatus fatal result, so wird die Verbindung sofort abgebrochen und sowohl Client als auch Server müssen alle verhandelten Daten, wie z.B. Session_ID, Schlüssel und Geheimzahlen, vergessen, die mit der fehlerhaften Verbindung zu tun haben.

  Es werden folgende Fehler unterschieden:

  - *unexpected_message*

    Dieser Fehler wird ausgelöst, wenn anhand der Sequenz-Nummer ein fehlender, überflüssiger oder veränderter Datenblock festgestellt werden kann. Diese Fehler haben immer den Status `fatal result`.

  - *bad_record_mac*

    Diese Nachricht wird gesendet, wenn ein Datenblock mit einem ungültigen MAC empfangen wurde. Auch diese Fehler haben den Status `fatal result`.

  - *decompression_failure*

    Bei der Dekomprimierung eines empfangenen Datenblockes wurde ein Fehler festgestellt. Bei diesem Fehler wird die Verbindung ebenfalls sofort abgebrochen.

  - *handshake_failure*

    Diese Fehler-Nachricht sagt aus, dass der Sender dieser Nachricht mit den angebotenen Verschlüsselungs-Parametern keinen vernünftigen Satz für die Verschlüsselung bilden kann. Auch dieser Fehler hat den Status `fatal result`.

- *illegal_parameter*

  Ein Datenfeld innerhalb des Handshake-Protokolls ist ungültig. In diesem Fall wird die Verbindung ebenfalls sofort abgebrochen.

- *Certificate-Errors*

  Zu den Error Alerts gehören auch noch einige Certificate-Fehler-Nachrichten. Zu ihnen gehören: *no_certificate*, *bad_certificate*, *unsupported_certificate*, *certificate_revoked*, *certificate_expired* und *certificate_unknown*.

3.2.5 SSL Application-Data-Protokoll

Mit dem Application-Data-Protokoll werden die Application-Data-Nachrichten über das Record-Protokoll übertragen. Sie sind somit fragmentiert, komprimiert und verschlüsselt. Die Nachrichten werden vom Record-Protocoll als transparent behandelt.

4 Zertifikate

Wie bereits im vorherigen Kapitel über Secure Sockets Layer (3 SSL, Seite 6) beschrieben, verwendet SSL digitale Zertifikate zur Authentifizierung der Kommunikationspartner.

Zertifikate finden mittlerweile in der Welt der elektronischen Kommunikation regen Einsatz. Nicht nur beim gesicherten Datenverkehr über das Word Wide Web werden sie eingesetzt, sondern auch im Email-Verkehr (PGP).

Der Überbegriff für die Authentifizierung mittels digitaler Zertifikate ist Public Key Infrastructure (kurz: PKI). Wie der Name auch schon sagt, kommen bei der Identifikation auch Public Key Verfahren zum Einsatz.

Diese eingesetzten digitalen Zertifikate können mit Personalausweisen verglichen werden, die es einer Person erlauben, sich einem anderen Menschen gegenüber zu identifizieren. Allerdings gibt es bei den digitalen Zertifikaten oder digitalen Ausweisen ein Problem: Beim Personalausweis kann die zugehörige Person durch den Vergleich des Passbildes mit dem Gesicht identifiziert werden. Beim digitalen Zertifikat ist das nicht so einfach. Um trotzdem eine ausreichende Sicherheit zu gewährleisten, gibt es sogenannte Certification Authorities (CA), die mit ihrer Signatur die Echtheit des Zertifikats garantieren.

Digitale Zertifikate, wie sie im Datenverkehr über das SSL-Protokoll eingesetzt werden, sind mittlerweile unter dem Begriff X.509v3 Zertifikate standardisiert. Dieser Standard wurde im Januar 1999 von der IETF in der RFC 2459 (http://www.ietf.org/rfc/rfc2459.txt) festgelegt.

4.1 Aufbau der Zertifikate

Der Grundaufbau aller digitaler Zertifikate nach dem X.509-Standard ist gleich. Sie enthalten folgende Informationen:

- *Version:*

  Da es vom X.509-Protokoll bereits insgesamt drei Versionen gibt, wird hier die Versionsnummer der verwendeten Version hinterlegt.

- *Seriennummer (Serial Number):*

  Hier wird eine ganzzahlige Seriennummer hinterlegt, die innerhalb einer CA eindeutig sein muss.

- *Algorithmenindentifikation (Signature Algorithm):*

  An dieser Stelle werden dem Zertifikat die Spezifikationen des verwendeten Signaturalgorithmus und die eingesetzte Hash-Funktion als Information mitgegeben.

- *Aussteller (Issuer):*

  In diesem Teil des Zertifikates sind nähere Daten zur CA hinterlegt.

- *Geltungsdauer (Validity):*

  Da Zertifikate nur beschränkt haltbar sind, sind in diesem Punkt Informationen über den Zeitraum der Gültigkeit gespeichert.

- *Subject:*

  Hier sind die Indentifikations-Daten des zu sichernden Objektes bzw. der zu sichernden Person gespeichert.

- *Public-Key (Subject Public Key Info):*

  Unter dem Punkt *Public-Key* ist der öffentliche Schlüssel des *Subjects* abgelegt. Er enthält weiter noch den Namen des verwendeten Algorithmus, die Schlüssellänge, den Modul *n* und den öffentlichen Exponenten *e*.

- *X.509v3-Erweiterungen (Extensions):*

  An dieser Stelle des Zertifikates befindet sich Raum für spezielle Parameter der jeweiligen Anwendung.

- *Signatur (Signature):*

  Am Ende des Zertifikates befindet sich noch die Signatur des Hash-Wertes aller anderen Felder durch die CA.

4.2 Sicherheitsklassen

Zertifikate können nach verschiedenen Sicherheitsklassen ausgestellt werden. Die insgesamt fünf existierenden Sicherheitsklassen unterscheiden sich durch unterschiedliche Sicherheitsqualitätsmerkmale, die das Maß an Vertrauenswürdigkeit festlegen.

- Klasse 0:
  Bei den Zertifikaten der Klasse 0 handelt es sich um sogenannte Demo-Zertifikate, die nur zu Testzwecken ausgestellt werden und eine eingeschränkte Gültigkeitsdauer haben. Sie haben die niedrigste Sicherheitsstufe, die Zertifikaten zugeordnet werden kann.

- Klasse 1:
  Klasse 1-Zertifkate werden überwiegend für Privatkunden ausgestellt, die erste Erfahrungen mit sicherer Kommunikation über das Medium Internet sammeln wollen. Diese Zertifikate benötigen keine Identifikation des Kunden und sind dementsprechend auch nur eingeschränkt sicher. Es wird nur die Existenz der Email-Adresse überprüft.
  Zertifikate dieser Klasse sind meist kostenlos bei den CA´s erhältlich.

- Klasse 2:
  Auch die Zertifikate der Klasse 2 sind nur eingeschränkt sicher. Sie sind überwiegend für Geschäftskunden gedacht, die zur Ausstellung des Zertifikates lediglich einer Kopie des Handelsregistereintrages oder eines schriftlichen Auftrages bedürfen. Diese Zertifikate sind besonders für Geschäftspartner gedacht, die sich auch außerhalb des Internets kennen.

- Klasse 3:
  Zertifikate der Klasse 3 benötigen neben der Überprüfung der Email auch eine persönliche Identifikation der zu zertifizierenden Person oder Einrichtung. Mit diesen Zertifikaten wird von der CA bestätigt, dass die Person anhand des Ausweises identifiziert worden ist und dass die Angaben im Zertifikat mit denen im Ausweis übereinstimmen. Gleichzeitig wird natürlich bei der Zertifizierung einer juristischen Person diese auch geprüft.

Klasse 3-Zertifikate eigenen sich vor allem für Anwendungen im E-Business, wie z.B. Internet-Banking oder Online-Shopping.

- Klasse 4:
  In der Klasse 4 finden wir die Zertifikate mit der höchsten Sicherheitsstufe. Sie unterscheiden sich von Klasse 3-Zertifikaten durch ein aufwändigeres Identifikationsverfahren.
  Der deutsche Anbieter TrustCenter bietet diese Art der Zertifikate aufgrund der geringen Nachfrage nicht mehr an.

4.3 Funktion einer Public Key Infrastructure

Public Key Infrastructures können auf sehr vielfältige Art und Weise eingesetzt werden. Neben dem SSL-Protokoll ist z.B. auch eine sicher Übertragung von Emails über den S/MIME-Standard möglich. Da diese Studienarbeit über das SSL-Protokoll und seinen Einsatz handelt, wird die Funktion einer PKI anhand dessen erklärt.

Wird über eine mit SSL gesicherte Verbindung durch einen Client eine Webseite von einem Server angefordert, so sendet der Server mit seiner Antwort im Rahmen des SSL-Handshake-Protokolls (3.2.2 SSL Handshake-Protokoll, Seite 9) das Zertifikat mit. Dieses Zertifikat wird vom Client mit einem Directory verglichen, das in den gängigen Browsern mit den Daten der bekanntesten CA´s bereits vorhanden ist. Weiter prüft der Client die Gültigkeit des Zertifikates mittels einer Certificate Revocation List (CRL oder auf deutsch Rückruf- oder Sperrliste). Diese CRL´s können von den Internet-Seiten der CA´s heruntergeladen und in den jeweiligen Browser eingebunden werden.

Findet der Client die CA, die das empfangene Zertifikat ausgestellt hat, nicht oder ist das Zertifikat nicht mehr gültig, wird der User mit einer Bildschirmmeldung gefragt, ob er dem Zertifikat trotzdem vertrauen möchte. Lehnt der User das Zertifikat ab, wird auch die Verbindung zum Server abgebrochen.

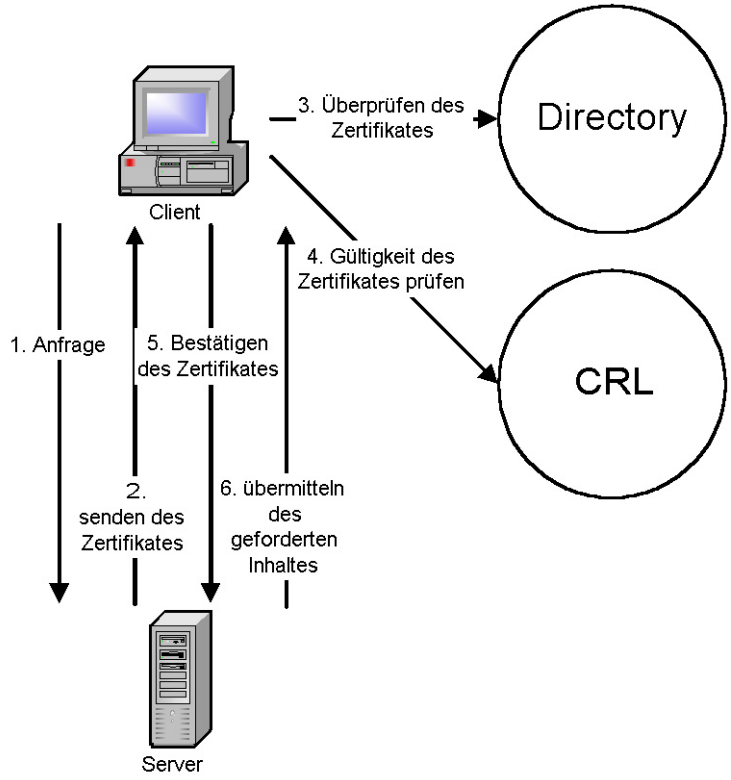

Abbildung 5: Public Key Infrastructure am Beispiel SSL

Wird das Zertifikat vom User angenommen, werden die Verbindungsparameter durch das SSL-Protokoll weiter ausgehandelt und schlussendlich der gewünschte Inhalt an den Client übertragen.

4.4 Certification Authorities

Wie schon zuvor erfahren, sind die sogenannten Certification Authorities für die Signierung der Zertifikate zuständig. Bei diesen CA´s können verschiedene Arten von Zertifikaten beantragt werden.

Zu den bekanntesten CA´s gehören TrustCenter (http://www.trustcenter.de) in Deutschland, VeriSign (http://www.verisign.com) in den USA und Thawte (http://www.thawte.com) in Südafrika. Auf den Internetpräsenzen der CA´s gibt es weitere Informationen und die Möglichkeit zur Beantragung eines Zertifikates.

5 Installation und Konfiguration

Das folgende Kapitel beschreibt den praktischen Teil dieser Studienarbeit:

Die Installation und Konfiguration eines Apache-Web-Servers mit SSL-Unterstützung auf den Betriebssystemen Linux und Windows 2000.

5.1 Notwendige Software

Bevor mit der Installation der einzelnen Komponenten begonnen werden kann, muss die dafür notwendige Software besorgt werden. Da es sich bei den eingesetzten Komponenten ausschließlich um freie Software handelt, ist eine Beschaffung über das Internet kein Problem.

Als Web-Server wurde in dieser Arbeit die derzeit aktuelle Apache-Version 1.3.22 benutzt, die unter der URL http://httpd.apache.org/dist/httpd in verschiedenen Versionen heruntergeladen werden kann. Für die Installation des Servers unter beiden Betriebssystemen wurde jeweils die selben Quellcodes aus dem Paket `apache_1.3.22.tar.gz` genommen. Unter der oben genannten URL stehen aber auch fertig kompilierte Apache-Versionen für verschiedene Betriebssysteme zum Download bereit.

Zur Installation der SSL-Funktionen sind zwei Software-Komponenten notwendig:

- Software, die die SSL-Funktionalität bietet.
  Im Weiteren wurde hierfür die freie SSL-Variante OpenSSL in der Version 0.9.6b verwendet, die ebenfalls im Internet zu finden ist. Unter der URL www.openssl.org/source können die notwendigen Quellcodes heruntergeladen werden.
- Software, mit der die SSL-Funktionalität als Modul in den Apache-Web-Server eingebunden werden kann.
  Hierfür wurde im Folgenden mod_ssl in der Version 2.8.5 für Apache 1.3.22 eingesetzt. Die notwendigen Quellcodes können unter

www.modssl.org/source heruntergeladen werden. Unter www.modssl.org/contrib liegen bereits kompilierte Softwarepakete für verschiedene Betriebssysteme zum Download bereit.

Für die Erstellung der Testanwendung sind nun noch PHP und MySQL notwendig, die ebenfalls frei im Internet verfügbar sind.

PHP in der aktuellen Version 4.1.1 kann unter der URL www.php.net heruntergeladen werden. Es finden sich dort auch verschiedene bereits für den Einsatz unter Windows kompilierte Versionen.

MySQL, in der aktuellen Version 3.23, liegt unter der URL www.mysql.com/downloads zum Download bereit. Auch hier finden sich noch verschiedene andere, zum Teil auch fertig kompilierte Versionen des Datenbanksystems.

Zur Konfiguration und Installation sind noch weitere Programme, wie ein Perl-Interpreter und ein C-Kompiler notwendig. Dazu aber bei den einzelnen Systemen mehr.

5.2 Linux-System

Für die folgende Installation des Apache-Servers wurde ein SUSE Linux 7.0 System benutzt.

Im Weiteren werden nun die notwendigen Schritte, die zur erfolgreichen Installation eines Apache-Web-Servers mit SSL unter Linux notwendig sind, beschrieben. Weiter finden sich hier auch Beschreibungen zur Installation eines PHP-Moduls und des Datenbanksystems MySQL, die beide zur Erstellung der Testanwendung notwendig sind.

5.2.1 Installation

Bei der Installation der Softwarekomponenten sind die Abhängigkeiten der einzelnen Software-Pakete voneinander zu beachten. Die hier aufgeführte Reihenfolge der Installation beachtet diese Abhängigkeiten.

Neben den eigentlichen Programmpaketen sind auch ein Perl-Interpreter und ein C-Kompiler notwendig, welche aber mit den meisten handelsüblichen Linux-Distributionen bereits automatisch installiert werden.

Für die Installation ist es wichtig, sich als `root` am System einzuloggen, da sonst einige Schritte aufgrund der beschränkten Nutzerrechte nicht ausgeführt werden können.

Als Pfad für die entpackten Softwarepakete wurde `/usr/local/src/lamp` gewählt.

Weiter ist noch zu erwähnen, dass alle Programm-Archive eine mehr oder minder gute Beschreibung in Form einer Text-Datei enthielten. Zusätzliche Installationsanleitungen können in gängigen Suchprogrammen durch die Eingabe des Begriffes „LAMP" (steht für Linux, Apache, MySQL und PHP) gefunden werden. Einige davon werden im Anhang (8.5.2.2 Installationsanleitungen, Seite 67) aufgeführt.

5.2.1.1 Installation von OpenSSL

Die erste Softwarekomponente, die installiert werden muss, ist die SSL-Software OpenSSL. Dies ist notwendig, da ein erfolgreiches Compilieren des Apache-Moduls mod_ssl OpenSSL benötigt.

Zuerst muss das aus dem Internet besorgte Softwarepaket entpackt werden. Dies erfolgt über den Aufruf

```
tar -xzf path_to/openssl-0.9.6b.tar.gz
```

aus dem Verzeichnis heraus, in dem die Software entpackt werden soll.

Als nächstes muss OpenSSL für die Installation konfiguriert werden. Dies erfolgt mit dem Aufruf

```
./configure --prefix=/usr/local/openssl-0.9.6b.
```

Mit dem übergebenen Parameter `--prefix` wird der Pfad festgelegt, in dem die Software installiert werden soll.

Nun kann die Software mit dem Aufruf des Makefiles durch

```
make
```

compiliert werden.

Mit der Eingabe von

```
make install
```

auf der Konsole wird die Software dann im übergebenen Verzeichnis installiert.

5.2.1.2 Vorbereiten von mod_ssl und Apache

Als nächster Schritt des Installationsvorganges wird das Apache-Modul mod_ssl auf seinen Einsatz vorbereitet. Dazu ist es erforderlich, dass die Apache-Quellcodes ausgepackt in einem Verzeichnis abgelegt werden. Dazu ist ein Wechsel in das Verzeichnis notwendig, in das die Software ausgepackt werden soll. In dieses Verzeichnis kann dann durch die Eingabe von

```
tar -xzf path_to/apache_1.3.22.tar.gz
```

das Auspacken der Dateien erfolgen.

Als nächstes wird ein Wechsel in das Verzeichnis notwendig, in das die Dateien für mod_ssl entpackt werden sollen. Dorthin können dann mit dem Aufruf

```
tar -xzf path_to/mod_ssl-2.8.5-1.3.22.tar.gz
```

die entsprechenden Dateien entpackt werden.

Da das Modul selbst später mit dem Apache-Web-Server compiliert wird, ist lediglich eine Ausführung des Konfigurationsskriptes notwendig. Das mod_ssl-Konfigurations-Skript übernimmt gleich die Konfiguration der Apache-

Quelldateien mit. Dies erfordert den Aufruf des Skriptes mit einer Menge an Parametern:

```
./configure --with-apache=path_to/apache_1.3.22
--with-ssl=path_to/openssl-0.9.6b
--prefix=/usr/local/apache-1.3.22
--datadir=/var/www
--enable-module=most
--enable-shared=max
--enable-module=ssl
```

Die bei dieser Installation verwendeten Parameter haben die folgende Bedeutung:

- `--with-apache= path_to/apache_1.3.22`

  Diese Angabe macht dem Konfigurations-Skript den Pfad der Apache-Quelldateien an.

- `--with-ssl=path_to/openssl-0.9.6b`

  Durch diese Angabe wird dem Konfigurations-Skript mitgeteilt, in welchem Verzeichnis die Installation von OpenSSL zu finden ist.

- `--prefix=/usr/local/apache-1.3.22`

  Über den Parameter `prefix` wird der Zielpfad der Apache-Installation an das Konfigurations-Skript übergeben.

- `--datadir=/var/www`

  Mittels dem `datadir`-Parameter wird dem Konfigurations-Skript das Verzeichnis übergeben, das später das Wurzel-Verzeichnis (`DocumentRoot`) für die HTML-Dateien befindet.

- `--enable-module=most`

  Dieser Konfigurations-Parameter bewirkt, dass die meisten Apache-Module bereits verfügbar sind. Sie müssen lediglich über die `LoadModule`-Direktive in der Konfigurationsdatei des Apache-Servers (`httpd.conf`) aktiviert werden.

- `--enable-shared=max`

  Dieser Parameter ermöglicht dem Apache-Web-Server eine maximale Flexibilität im Hinzufügen von Modulen. Die einzelnen Module können

dann als Dynamic Shared Objekts (DSO) ohne erneutes kompilieren dem Apache-Server zugefügt werden.

- `--enable-module=ssl`

  Der letzte in diesem Beispiel übergebene Parameter macht dem Konfigurations-Skript bekannt, dass das SSL-Modul im Apache-Web-Server Einsatz finden soll.

Neben diesen hier aufgeführten Parametern können bei der Konfiguration von mod_ssl und dem Apache-Server noch eine ganze Menge weitere Einstellungen via Parameter vorgenommen werden. Nähere Informationen dazu finden sich indem man mit ./configure –help die Hilfe-Funktion des jeweiligen Konfigurations-Skriptes aufruft oder im Falle mod_ssl und Apache-Server auch in der README.configure, die im Verzeichnis mit den entpackten Apache-Quellcodes zu finden ist.

Nach einem Wechsel in das Verzeichnis mit den Apache-Quelldateien kann nun durch den Aufruf

```
make
```

der Apache-Web-Server kompiliert werden.

Damit der Apache-Web-Server auch automatisch startet, sind noch einige Schritte notwendig. Die folgenden Schritte beziehen sich auf ein SUSE-Linux-System.

Zuerst muss ein symbolischer Link auf die Apache-Start-Datei gelegt werden.

```
ln –s /path_to/apache_1.3.22/bin/apachectl
      /etc/rc.d/init.d/apachectl.
```

Dann den Ordner wechseln

```
cd /etc/rc.d/rc2.d
```

und folgende symbolische Links anlegen:

```
ln –s ../init.d/apachectl S20apachectl
ln –s ../init.d/apachectl K20apachectl
```

Der K* -Link dient dabei dem Beenden von laufenden Prozessen und der S*-Link zum Starten neuer Prozesse.

Nun müssen noch einige Zeilen Code der Datei `apachectl` geändert werden, damit der Apache-Server auch mit SSL-Unterstützung starten kann. Betroffen ist die folgende Stelle im Code:

```
start)
if [ $RUNNING -eq 1 ]; then
     echo "$0 $ARG: http (pid $PID) already running"
     continue
fi
# Original
# if $HTTPD; then
# geändert, -DSSL eingefügt:
if $HTTPD -DSSL; then
     echo "$0 $ARG: httpd started"
else
     echo "$0 $ARG: httpd could not be started"
     ERROR = 3
fi
;;
```

Der Apache-Web-Server sollte nun beim nächsten Start des Linux-Systems mit SSL-Unterstützung starten.

5.2.1.3 Erzeugen eines Zertifikates

Da Server-Zertifikate einer Certification Authority meist einen aufwändigen Identifikationsprozess und einige Kosten mit sich bringen, wird hier nun gezeigt, wie man sich ein eigenes Zertifikat für den Heimgebrauch erstellen kann. Dieser Schritt muss noch vor der Installation des Web-Servers erfolgen.

Für die Erstellung des Zertifikates ist das `Makefile` im Apache-Quell-Verzeichniss zuständig. Über den Aufruf

```
make certificate TYPE=custom
```

kann dem `Makefile` mitgeteilt werden, dass ein anwenderspezifisches Zertifikat mit der Signatur einer "eigenen CA", die an dieser Stelle gleich mit erstellt wird, angelegt werden soll.

Im ersten Block der nun folgenden Fragen an den Anwender werden Informationen zur „eigenen CA" gesammelt:

```
Signature Algorithm: R (Verschlüsselung: RSA)
Country Name: "DE"
State or Province: "BW"
Localty Name: "Karlsruhe"
Organization Name: "FH Karlsruhe"
Organizational Unit Name: "CA"
Common Name: "FH Karlsruhe CA"
Email Address: "fial0011@fh-karlsruhe.de"
Certificate Validity: "365" (Gültigkeit in Tagen)
Certificate Version: 3 (X.509v3-Zertifikat)
```

Es wird nun nach den Angaben zum zu zertifizierenden Server gefragt:

```
Country Name: "DE"
State or Province: "BW"
Localty Name: "Rastatt"
Organization Name: "FH Karlsruhe"
Organizational Unit Name: "Webmaster"
Common Name: "192.168.10.102"
Email Address: "www@192.168.10.102"
Certificate Validity: "365" (Gültigkeit in Tagen)
Certificate Version: 3 (X.509v3-Zertifikat).
```

Hierbei ist es ganz wichtig, dass beim Punkt `Common Name` der Name eingetragen wird, unter dem der Server erreichbar ist. In diesem Fall ist es die IP-Adresse 192.168.10.102.

Als letzter Punkt zur Zertifikat-Erzeugung wird noch nachgefragt, ob die Schlüssel verschlüsselt werden sollen. Zu erst wird mit

```
Encrypt the private key now: n
```

nach dem Schlüssel der CA gefragt und dann nochmals mit der selben Frage nach dem Schlüssel des Servers.

In diesem Beispielfall wurde auf diese Verschlüsselung verzichtet. In einem Produktivsystem sollte sie aber in jedem Fall aktiviert werden.

Und so sieht das fertige Zertifikat nach einem Aufruf einer sicheren Verbindung mittels SSL im Internet Explorer aus:

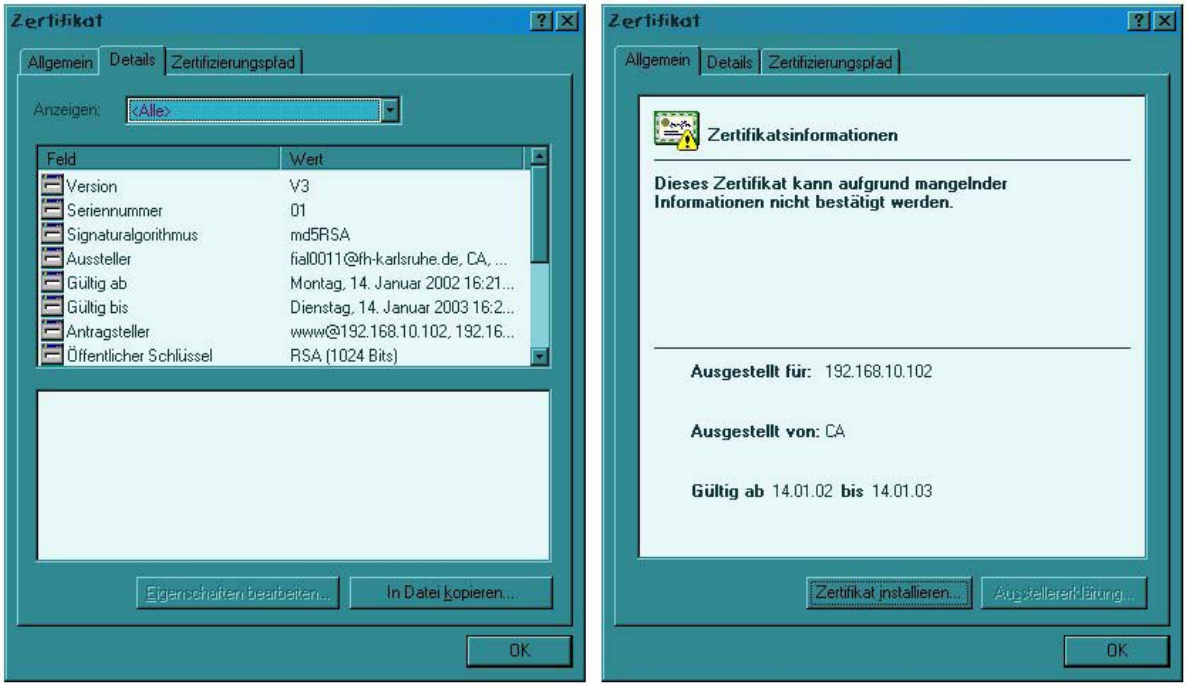

Abbildung 6: Das fertige Zertifikat

Nach der erfolgreichen Erstellung eines Zertifikates muss noch mit dem Aufruf

```
make install
```

der konfigurierte und kompilierte Apache-Web-Server installiert werden.

5.2.1.4 Installation der restlichen Komponenten

Für die Testanwendung ist es notwendig, noch weitere Komponenten zu installieren. Dies sind die Open Source-Datenbank MySQL, hier in der Version 3.23.43, und die freie, serverseitige Scriptsprache PHP, hier in der Version 4.0.6. Das DBS kann bei www.mysql.com kostenlos heruntergeladen werden.

Installation von MySQL

Bevor PHP installiert werden kann, ist es sinnvoll das Datenbanksystem MySQL zu installieren. Dazu muss zuerst das Tar-Archiv ausgepackt werden.

```
tar -xzf path_to/mysql-3.23.43.tar.gz
```

Dann muss mit dem Aufruf

```
cd path_to/mysql-3.23.43/
```

in den entsprechenden Ordner gewechselt werden. Dort muss dann zuerst das Konfigurationsskript ausgeführt werden.

```
./configure --prefix=/path_to/mysql
  --localstatedir=/var/mysql/data
make
```

Die beiden dem Konfigurationsskript übergebenen Optionen bewirken dabei folgendes:

- `--prefix=/path_to/mysql` :
  Setzt den gewünschten Installtionspfad als Präfix im Konfigurationsskript.
- `--localstatedir=/var/mysql/data` :
  Setzt den Datenpfad des Datenbanksystems.

Mit dem Befehl

```
make install
```

Kann das Datenbanksystem nun auf dem Linux-System installiert werden.

Als nächstes muss sichergestellt werden, dass die mit der Option `localstatedir` festgelegten Verzeichnisse für die Daten auch tatsächlich existent sind. Sind sie das nicht, so müssen sie mit dem `mkdir`-Befehl angelegt werden.

Mit dem Aufruf

```
scripts/mysql_install_db
```

müssen jetzt noch die für eine erfolgreiche Arbeit mit dem DBS erforderlichen Privilege-Tabellen angelegt werden. Diese Tabellen sind für die Verwaltung der Zugriffsrechte zuständig.

Nach der erfolgreichen Installation, sollte das Datenbanksystem mit dem Aufruf des Programms

```
/path_to/mysql/bin/safe_mysqld &
```

noch getestet werden.

Eine weitere Möglichkeit den Datenbank-Server zu testen ist das Programm `mysqladmin`. Nähere Informationen hierzu gibt es beim Aufruf der Hilfefunktion des Programms mit

```
/path_to/mysql/bin/mysqladmin –help.
```

Soll der Datenbank-Server nun noch automatisch beim Starten des Betriebssystem gestartet werden, sind noch einige weitere Schritte notwendig. Das nun folgende Beispiel ist für eine SUSE-Linux-Distribution geeignet.

Zuerst muss die Startup-Datei mysql.server in den Ordner `/etc/rc.d/init.d` kopiert werden. Dies erfolgt mit folgendem Aufruf:

```
cp /path_to/mysql/support-files/mysql.server
   /etc/rc.d/init.d
```

Dann müssen die Rechte für die kopierte Datei vollen Zugriff für den User und Leserechte für die restlichen Gruppen eingestellt werden:

```
chmod 744 /etc/rc.d/init.d/mysql.server
```

Dann mit dem Aufruf

```
Cd /etc/rc.d/rc2.d
```

den Ordner wechseln und mit

```
ln –s ../init.d/mysql.server S20mysql.server
ln –s ../init.d/mysql.server K20mysql.server
```

symbolische Links anlagen. Der K* -Link dient dabei dem Beenden von laufen-laufenden Prozessen und der S*-Link zum Starten neuer Prozesse.

Nun müsste der MySQL-Server automatisch beim Start des Linux-System verfügbar sein.

Installation von PHP

Zum Zugriff auf ein Datenbanksystem über eine Web-Anwendung wird eine serverseitige Skriptsprache benötigt. In diesem Beispiel, das überwiegend freie Software benutzt, wird die Skriptsprache PHP eingesetzt, die im Web unter www.php.net kostenfrei heruntergeladen werden kann.

Da bei der Konfiguration des Apache-Web-Servers (5.2.1.2 Vorbereiten von mod_ssl und Apache, Seite 23) das Hinzufügen von Modulen als DSOs eingestellt wurde, kann das PHP-Modul für sich kompiliert werden und muss nicht in den Apache-Web-Server hineinkompiliert werden.

Zuerst muss natürlich das PHP-Modul entpackt werden. Dies erfolgt mit folgendem Aufruf:

```
tar -xzf path_to/php-4.0.6.tar.gz.
```

Danach muss das Konfigurationsskript mit verschiedenen Optionen ausgeführt werden:

```
./configure
   --prefix=/path_to/php_4.0.6
   --with-apxs=/path_to/apache_1.3.22/bin/apxs
   --with-mysql=/path_to/mysql
   …
   --with-config-file-path=/etc
make.
```

Neben den im Beispiel aufgeführten Optionen gibt es noch eine ganze Menge mehr, die durch den Aufruf des Konfigurationsskriptes mit dem Optionsparameter `--help` am Bildschirm dargestellt werden können. Die oben genannten Parameter bedeuten folgendes:

- `--prefix=/path_to/php_4.0.6`
  Setzt den Installationspfad des PHP-Moduls.
- `--with-apxs=/path_to/apache_1.3.22/bin/apxs`
  Konfiguriert das PHP-Modul als DSO. Der Pfad gibt den Pfad zum apxs-Programm des Apache-Web-Servers an.
- `--with-mysql=/path_to/mysql`
  Konfiguriert das PHP-Modul mit den Funktionen für den MySQL-Datenbankzugriff. Der Pfad führt zum MySQL-Datenbanksystem.
- `--with-config-file-path=/etc`
  Setzt den Pfad, in dem das PHP-Modul die Konfigurationsdatei `php.ini` findet.

Mit dem Befehl

```
make install
```

wird das PHP-Modul nun als DSO in den mit der `prefix`-Option eingestellten Pfad installiert. Mit

```
cp /path_to/php_4.0.6/php.ini-dist /etc/php.ini
```

wird noch die PHP-Konfigurationsdatei `php.ini` in den zuvor eingestellten Pfad kopiert.

Bevor der Apache-Web-Server mit SSL nun eingesetzt werden kann, müssen noch die Konfigurationsdateien der einzelnen Komponenten überprüft und ergänzt werden. Dies geschieht im folgenden Kapitel.

5.2.2 Konfiguration

Bevor der Server gestartet werden kann, muss noch ein Blick auf die Konfigurationsdateien der einzelnen Komponenten, insbesondere der Apache-Konfigurationsdatei `httpd.conf` und der PHP-Konfigurationsdatei `php.ini`, geworfen werden.

`httpd.conf`

Werden dem Konfigurationsskript `configure` zu Beginn der Apache-Installation die richtigen Optionen übergeben, ist nur noch wenig Handarbeit bei der Konfiguration des Apache-Web-Servers notwendig. Lediglich einige Einstellungen zur richtigen Erkennung von PHP-Dateien sind notwendig.

Mit der Zeile

```
AddType application/x-httpd-php .php
```

wird der MIME-Typ .php dem Apache-Server bekannt gemacht. Diese Zeile kann zusätzlich noch um die für PHP-Dateien gebräuchlichen Endungen `.phtml` und `.php3` ergänzt werden. Auch die Dateiendungen `.html` und `.htm` können an diese Zeile angefügt werden, weil auch normale HTML-Dateien PHP-Codezeilen enthalten können, zu deren richtigen Ausführung der zu den Dateiendungen gehörende MIME-Typ bekannt sein muss.

Die zweite wichtige Einstellung ist die Erweiterung der Direktive, die dem Apache-Web-Server die Verzeichnis-Index-Dateien bekannt macht. Dies erfolgt mit der Zeile

```
DirectoryIndex index.html index.php index.htm,
```

die insbesondere um den Zusatz `index.php` ergänzt wurde.

Der Apache-Web-Server kann natürlich noch mit vielen weiteren Direktiven auf die individuellen Bedürfnisse angepasst werden. Eine genaue Beschreibung kann den Quellen **[A1]** und **[B3]** entnommen werden.

`php.ini`

Die PHP-Konfigurationsdatei `php.ini` ist eigentlich soweit durch das Konfigurationsskript in funktionsfähigem Zustand. Zumindest war dies in beschriebenem Beispiel so. Sollte es wieder erwarten an dieser Stelle Probleme geben, so hilft das PHP-Manual weiter, das meist unweit vom eigentlichen PHP-Modul zum Download bereit liegt.

MySQL

Das freie Datenbanksystem entpuppte sich als relativ einfach zu konfigurieren. Die Konfiguration der Tabellen wurde mit der Unterstützung des PHP-Tools PHPMyAdmin, das unter http://www.phpwizard.net/projects/phpMyAdmin/ frei im Internet verfügbar ist, vorgenommen. Über das auf der serverseitigen Skriptsprache basierenden Konfigurations-Tool lassen sich über einen Web-Browser sehr leicht Tabellen anlegen und administrieren. Es setzt deshalb auch einen funktionsfähigen Web-Server mit PHP-Unterstützung voraus.

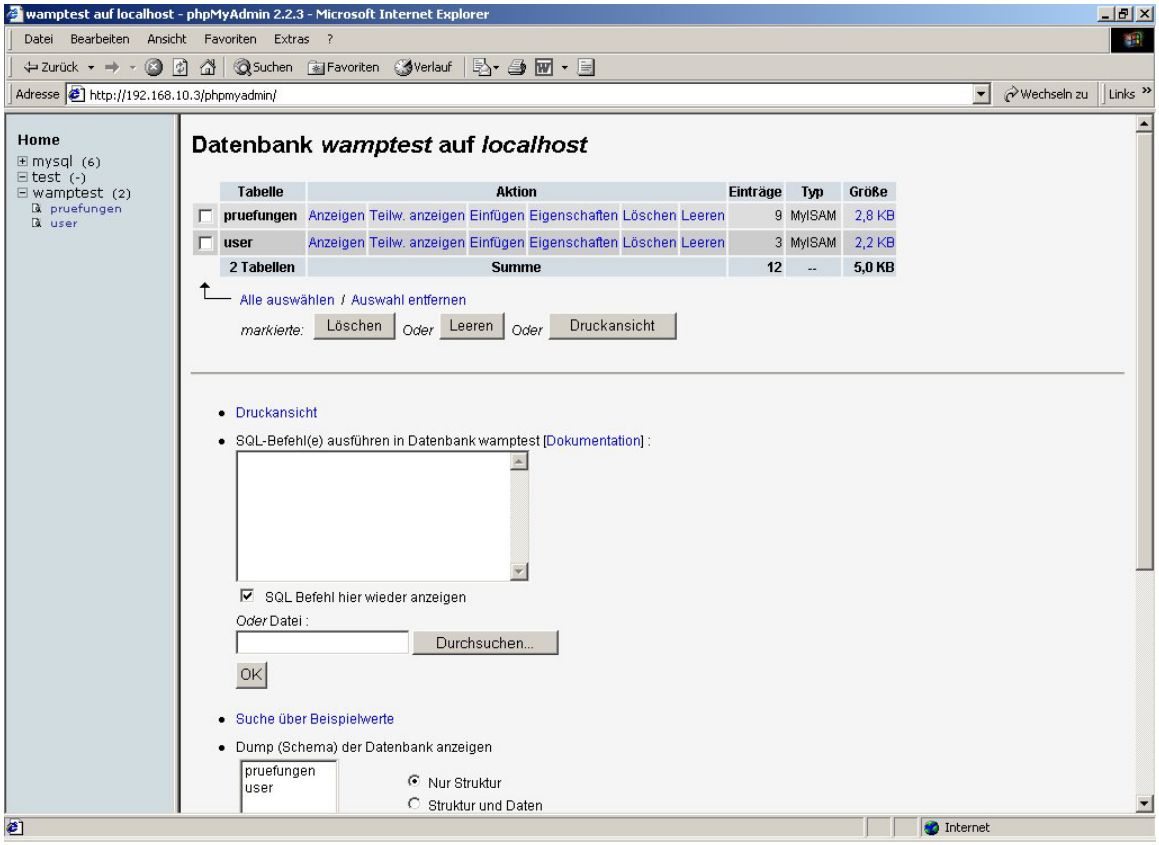

Abbildung 7: Oberfläche von PHPMyAdmin

Zur Eingabe und Korrektur der notwenigen Daten wurde über den ODBC-Treiber MyODBC für MySQL eine Microsoft-Access-Datenbank benutzt.

5.3 Windows 2000-System

Die Installation eines Apache-Web-Servers mit SSL auf einem Windows-System des Software-Giganten Microsoft entpuppte sich als weitaus komplizierter, als auf dem zuvor beschriebenen Linux-System. Nach vergeblichen Versuchen den Web-Server auf zwei unterschiedlichen Windows 98-Systemen zu installieren, wurde für dieses Beispiel eine neuere und für Server-Aufgaben konzipierte Windows 2000-Umgebung ausgewählt.

Auch unter Windows 2000 ist es wichtig, sich als Administrator anzumelden, da sonst nicht alle notwendigen Operationen durchgeführt werden können.

Obwohl für eine Vielzahl der eingesetzten Programme bereits fertig kompilierte Binary-Archive zum Download bereitstehen, wird hier eine ausführliche Installation mit Kompilierung der Source-Codes durchgeführt und beschrieben.

5.3.1 Installation

Vor der Installation unter Windows 2000 muss sichergestellt werden, dass einige Programme, die bei Linux-Distributionen serienmäßig dabei sind und nahezu automatisch installiert werden, auch unter Windows vorhanden sind.

Als erstes und wichtigstes Tool sollte ein Archivierungs-Werkzeug wie z.B. WinZIP (für den Privatgebrauch eingeschränkt nutzbar) oder ein freies Tool wie CygWin32 (Programmpaket, das Unix-Befehle auf einer Windows-Umgebung verfügbar macht) installiert werden, mit dem die Quell-Code-Archive dekomprimiert und entpackt werden können. In dem beschriebenen Beispiel wurde WinZIP verwendet.

Weiter muss ein Perl-Interpreter installiert sein, mit dem das Konfigurations-Skript bei der OpenSSL-Installation ausgeführt werden kann. Perl-Interpreter findet man kostenlos unter http://www.perl.com/CPAN-local. Den Perl-Archiven liegt eine Installations-Beschreibung bei, nach der der Interpreter mehr oder minder problemlos kompiliert und installiert werden kann.

Die letzte Software-Komponente, die abgesehen von den Server-Komponenten selbst, zur Installation benötigt wird, macht wahrscheinlich die größten Probleme, da sie nicht frei erhältlich ist. Insbesondere zur Kompilierung des Apache-Web-Servers unter Windows ist die Microsoft-Entwicklungsumgebung Visual C++ ab Version 5 notwendig. Für die beschriebene Beispiel-Installation wurde MS Visual C++ 6.0 eingesetzt.

Neben Visual Studio wird noch ein Programm namens `awk.exe` zur Kompilierung des Web-Servers benötigt. Dieses Tool bearbeitet in Textform vorliegende Datenbanken (mehr bei **[I11]**) und kann als `awk95.exe` von der Seite http://cm.bell-labs.com/cm/cs/who/bwk/ heruntergeladen werden. Anschließend muss `awk95.exe` in `awk.exe` umbenannt werden, damit es von Visual

C++ gefunden wird. Das Programm muss dann so abgelegt werden, dass zum einen die PATH-Variable den Weg zu `awk.exe` kennt und zum anderen auch Visual C++ die Datei finden kann. Im Beispiel wurde `awk.exe` im `bin`-Verzeichnis von Visual C++ abgelegt.

Nach der Installation des Perl-Interpreters und des C-Kompiliers müssen noch die Pfade zu beiden Programmen in der PATH-Umgebungsvariable des Systems eingetragen werden. Bei Visual C++ interessiert hier im besonderen das Programm `nmake`, das im `bin`-Verzeichnis des Visual C++-Ordners zu finden ist.

Die Installation kann abgesehen vom Entpacken der Archive mit WinZIP vollständig im DOS-Konsolen-Fenster durchgeführt werden.

Ein Hinweis noch: Im Archiv des SSL-Moduls mod_SSL befindet sich eine gute Beschreibung der Installationsvorgänge, der in diesem Beispiel gefolgt wurde.

5.3.1.1 Installation von OpenSSL

Die OpenSSL-Software kann unter http://www.openssl.org, der Homepage des OpenSSL-Projekts heruntergeladen werden.

Nach dem Entpacken und Dekomprimieren des Software-Archivs, kann nach dem Wechsel in das OpenSSL-Verzeichnis das Konfigurations-Skript mit dem Aufruf

```
perl Configure VC-WIN32 --prefix=c:/openssl
```

ausgeführt werden. Mit der Option `--prefix=c:/openssl` wird das Installationsverzeichnis an das Konfigurations-Skript übergeben.

```
ms\do_ms
```

startet dann den Assembler, der die OpenSSL-Makefiles erzeugt.

Mit dem Aufruf

```
nmake /f ms\ntdll.mak
```

werden dann die OpenSSL-Dateien kompiliert.

Zum Schluss der OpenSSL-Installation müssen noch einige Dateien in den zuvor mit `--prefix` festgelegten Pfad kopiert werden:

```
md c:\openssl
md c:\openssl\bin
md c:\openssl\lib
md c:\openssl\include
md c:\openssl\include\openssl
copy /b inc32\* c:\openssl\include\openssl
copy /b out32dll\ssleay32.lib c:\openssl\lib
copy /b out32dll\libeay32.lib c:\openssl\lib
copy /b out32dll\ssleay32.dll c:\openssl\bin
copy /b out32dll\libeay32.dll c:\openssl\bin
copy /b out32dll\openssl.exe  c:\openssl\bin
```

Das Kopieren der Dateien kann natürlich auch mit dem Windows-Explorer durchgeführt werden.

Jetzt muss noch der bin-Ordner des OpenSSL-Verzeichnisses als PATH-Umgebungsvariable des Systems gesetzt werden.

Nachdem OpenSSL nun installiert ist, kann mit dem nächsten Schritt, dem Erstellen eines Zertiflates nach dem X.509-Standard, fortgefahren werden.

5.3.1.2 Erstellen eines Zertifikates

Zur reibungslosen Funktion des SSL-unterstützenden Apache-Web-Servers im Testbetrieb ist auch unter Windows 2000 ein Zertifikat notwendig, welches mit den folgenden Schritten erstellt werden kann. Für den professionellen Einsatz des Servers sollte besser bei einer Certification Authority ein X.509-Zertifikat (4 Zertifikate, Seite 15) beantragt werden.

Bevor mit der Erzeugung des Zertifikates begonnen werden kann, muss zuerst noch eine `openssl.cnf` Konfigurationsdatei besorgt werden. Eine solche Datei findet man unter http://tud.at/programm/openssl.cnf oder in einer der unter http://www.modssl.org/contrib/ verfügbaren Apache-Openssl-Konplettpakete für Windows. Die Datei `openssl.cnf` muss in den Ordner kopiert werden, der die Datei `openssl.exe` enthält. Im aktuellen Beispiel befindet sich die Datei `openssl.exe` im Pfad `C:\openssl\bin`. Die Erstellung des Zertifikates muss natürlich dann ebenfalls in diesem Ordner stattfinden.

Aber nun zur Erstellung eines Test-Zertifikat, wie es für studentische Zwecke oftmals ausreicht:

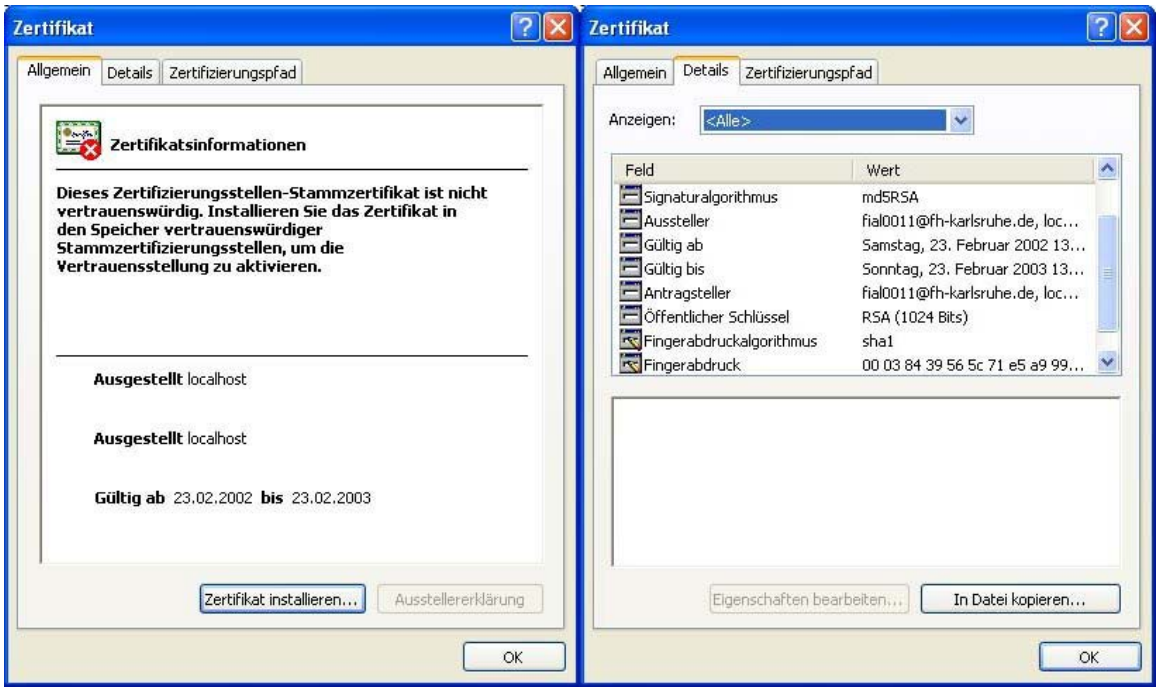

Abbildung 8: Das X.509-Zertifikat auf dem Windows-System

Durch die Eingabe von

```
openssl req    -config openssl.cnf
               -new
               -out name_of_server.csr
```

wird der private Schlüssel generiert und die das Zertifikat auszeichnenden Informationen gesammelt. Es müssen also nach dem obigen Aufruf einige Informationen für das Zertifikat angegeben werden. Dabei ist dringend zu beachten, dass beim Punkt `Common Name (eg, YOUR name)` der Name angegeben wird, unter dem der Server angesprochen wird. Hier wurde der Eintrag `localhost` gewählt.

Durch den Aufruf

```
openssl rsa    -in privkey.pem
               -out my-server.key
```

wird `my-server.key` nur noch durch den Apache-Web-Server und den Administrator lesbar.

Mit

```
Openssl x509   -in my-server.csr
               -out my-server.cert
               -req
               -signkey my-server.key
               -days 365
```

wird erst das eigentliche Zertifikat erzeugt. Das mit diesem Aufruf erzeugte Zertifikat hat allerdings nur eine Haltbarkeit von einem Jahr und sollte nur solange benutzt werden, bis ein von einer CA ausgestelltes Zertifikat für den Server vorliegt.

Zu guter Letzt sollte noch ein Verzeichnis `\ssl` im Pfad `\conf` des Apache-Root-Verzeichnisses angelegt werden. Dorthin müssen die Dateien `my-server.key` und `my-server.cert` kopiert werden.

5.3.1.3 Kompilieren von mod_ssl

Als nächster Schritt muss das Apache-Modul mod_ssl kompiliert werden, das als Schnittstelle zwischen Apache und OpenSSL fungiert.

Dazu muss zuerst das mod_ssl-Archiv, das unter http://www.modssl.org heruntergeladen werden kann, entpackt werden. Beim Download muss darauf geachtet werden, dass die Version heruntergeladen wird, die zu der Apache-Version passt, die zum Einsatz kommen soll.

Nach dem Wechsel in das Verzeichnis, in das mod_ssl entpackt wurde, kann das Konfigurations-Skript mit dem Aufruf

```
configure.bat  --with-apache=..\apache_1.3.x
               --with-ssl=c:\openssl
```

ausgeführt werden.

Nach der Konfiguration von mod_ssl kann zum nächsten Schritt übergegangen werden.

5.3.1.4 Installation des Apache-Servers

Als letzter Schritt der Server-Installation muss nun noch der eigentliche Web-Server von Apache installiert werden.

Zuerst müssen die Quell-Dateien des Apache-Web-Servers entpackt werden. Nach dem Wechsel in den `src`-Ordner des Apache-Verzeichnises erfolgt das Kompilieren des Servers mit dem Aufruf

```
nmake /f Makefile.win.
```

Die Installation des Web-Servers erfolgt nun noch mit dem Aufruf

```
nmake /f Makefile.win installr
```

Damit ist der Apache-Web-Server mit SSL-Unterstützung installiert. Es fehlen nun lediglich noch der PHP-Interpreter und das MySQ-Datenbanksystem, die beide für die Testanwendung notwendig sind.

5.3.1.5 Installation der restlichen Komponenten

Bei den beiden restlichen für diese Studienarbeit wichtigen Komponenten, dem freien Datenbanksystem MySQL und der ebenfalls freien server-seitigen Skript-Sprache PHP, wurde auf die bereits fertig kompilierten Windows-Binarys zurückgegriffen, die unter http://www.mysql.com bzw. http://www.php.net zum Download bereit liegen.

Installation von MySQL

Nach dem Auspacken des heruntergeladenen Binary-Archivs mittels WinZIP, kann mit dem Programm `setup.exe` ein Installer gestartet werden, der das Datenbanksystem unter Windows 2000 als Dienst installiert. Über die Diensteeinstellungen im Ordner Systemeinstellungen/Verwaltung lässt sich dieser Dienst dann entsprechend konfigurieren.

Installation von PHP

Auch PHP lässt sich relativ einfach unter Windows installieren. Auch hier muss das Binary-Archiv mit WinZIP entpackt werden. Danach muss lediglich die Datei php.ini-dist in den Windows-Ordner (bei 9x/Me/XP: C:\windows, bei NT/2000: C:\winnt) kopiert werden. Die Datei muss dann noch in php.ini umbenannt werden und schon kann mit der Konfiguration der einzelnen Komponenten begonnen werde.

5.3.2 Konfiguration

Die Konfiguration unter Windows ist etwas aufwändiger. So müssen im Vergleich zur Konfiguration unter Linux noch sämtliche SSL-relevanten Direktiven in die Apache-Konfigurationsdatei `httpd.conf` eingetragen werden.

`httpd.conf`

In der Konfigurationsdatei `httpd.conf` des Apache-Servers müssen Direktiven eingetragen werden, die zum einen das Modul mod_ssl konfigurieren und zum anderen das PHP-Modul verfügbar machen.

Zuerst muss dem Apache-Server mit

```
LoadModule ssl_module modules/mod_ssl.so
```

das SSL-Modul bekannt gemacht werden und die bereits vorhandene Diretktive

```
Port 80
```

durch die Direktiven

```
Listen 80
Listen 443
```

ersetzt werden. Diese beiden Zeilen weisen den Web-Server an, auf Anfragen über die Ports 80 (Standard http-Port) und 443 (Standard https-Port) zu achten.

Dann muss mit einigen `mod_ssl`-Direktiven die SSL-Engine konfiguriert werden:

Die Direktive

```
SSLMutex sem
```

konfiguriert die Semaphore der SSL-Engine. Durch den Parameter `sem` wird unter Windows ein Windows Mutex als Semaphore benutzt. Dieser Parameter ist nur dann möglich, wenn die System-Plattform sie unterstützt.

Mit der Direktive

```
SSLRandomSeed startup builtin
```

wird der SSL-Zufallszahlen-Generator konfiguriert. Die Parameter startup builtin besagen, dass der Generator beim Start der Engine konfiguriert wird und die immer verfügbare Zufallszahlen-Quelle benutzen soll.

Mit

```
SSLSessionCache none
```

Wird der optional zuschaltbare SSL Session Cache deaktiviert.

Um eventuelle Fehler der SSL-Engine verfolgen zu können, wird mit

```
SSLLog log/SSL.log
SSLLogLevel info
```

Der Pfad der Logdatei und der Detailtiefe der gespeicherten Nachrichten in der Logdatei angegeben. Die Detailtiefe sollte im Echtbetrieb des Servers auf den Level warn gestellt werden, da sonst der Web-Server ständig am protokollieren ist.

Bis hierher handelte es sich um sogenannte globale Direktiven, die für alle weiteren Einstellungen des Apache-Web-Servers gelten.

Nun muss noch für den im Zertifikat erwähnten Server namens localhost ein VirtualHost definiert werden:

```
<VirtualHost localhost:443>
     SSLEngine on
     SSLCertificateFile conf/ssl/my-server.cert
     SSLCertificateKeyFile conf/ssl/my-server.key
</VirtualHost>
```

Wird der `localhost` über den Port 443 aufgerufen, so startet die SSL-Engine. Mit den Direktiven `SSLCertificateFile` und `SSLCertificateKeyFile` wird dem Server der Pfad mitgeteilt, indem das Zertifikat und der zugehörige Schlüssel gefunden werden können.

Weitere Direktiven von `mod_ssl` werden unter http://www.modssl.org/docs/2.4/ssl_reference.html ausführlich beschrieben. Dort finden sich auch ausführlichere Beschreibungen der im Beispiel eingesetzten Direktiven.

Nach dem Einfügen der oben aufgeführten Direktiven ist der Apache-Web-Server mit SSL-Unterstützung lauffähig.

Nun fehlt nur noch die Einbindung des PHP-Moduls in den Apache-Web-Server:

Mit der Direktive

```
LoadModule php4_module c:/php/sapi/php4apache.dll
```

wird das PHP4-Module in den Web-Server geladen. Mit der Direktive

```
AddType application/x-httpd-php .php .php3
```

wird dem Web-Server mitgeteilt, dass alle Dateien mit den Endungen .php und .php3 dem PHP-MIME-Typ entsprechen.

Nun kann der Direktive `DirectoryIndex` noch mitgeteilt werden, dass es neben den `index.html`-Dateien auch `index.php`-Dateien als Verzeichnisindex akzeptiert. Dies sieht dann folgendermaßen aus:

```
DirectoryIndex index.html index.php
```

Somit ist der Apache-Web-Server mit SSL und PHP konfiguriert und einsatzbereit. Allerdings müssen noch einige Einstellungen in der PHP-Konfigurationsdatei vorgenommen werden.

`php.ini`

Wie schon bei der Installation von PHP erwähnt, muss die PHP-Konfigurationsdatei `php.ini` noch an den Rechner angepasst werden, auf dem die Skriptsprache laufen soll.

Dazu sind in erster Linie Pfadeinstellungen vorzunehmen. Der erste einzustellende Pfad betrifft die im PHP-Ordner enthaltenen Erweiterungen, zu denen auch die openSSL-Funktionalitäten gehören. Hinter `extensions_dir` verbirgt sich der Ordner zu den Erweiterungen:

```
extension_dir = D:\php\extensions
```

Vor deren Nutzung müssen noch zwei weitere Schritte durchgeführt werden:

1. Die gewünschten Erweiterungen müssen in der `php.ini` eingetragen werden. Eigentlich sind vor den gewünschten Erweiterungen (`extension=php_*.dll`) nur die Kommentarzeichen zu entfernen, da die mitgelieferten Funktionen bereits alle auskommentiert in die `php.ini` eingetragen sind.
2. Die im Unterordner `dll` befindlichen DLL-Deteien müssen in den `System32`-Unterordner des Windows-Ordners (bei 9x & ME: `Windows`; bei 2000 & XP: `Winnt`) kopiert werden.

Nun muss noch das Wurzelverzeichnis für die HTML-Dokumente als `doc_root` eingetragen werden.

```
doc_root = D:\apache\htdocs
```

Nun sollte der Apache-Web-Server auch fähig sein, PHP-Skripte zu interpretieren.

MySQL

Die Konfiguration und Administration des MySQL-Datenbanksystems verläuft auf einem Windows-System anlog zur Konfiguration und Administration auf einem Linux-System (5.2.2 Linux-System/Konfiguration; Seite 33).

6 Beschreibung der Testanwendung

Eigentlich sollte die zur Studienarbeit gehörige Testanwendung insbesondere die OpenSSL-Funktionen von PHP demonstrieren. Diese befinden sich derzeit laut http://www.php.net aber noch in einem Experimental-Stadium und sind nur schlecht dokumentiert.

Es war so lediglich möglich eine Testanwendung zu programmieren, die das Zusammenspiel zwischen PHP und MySQL demonstriert. Alleine die OpenSSL-Funktion `open_x509_parse()`, die ein bestehendes X.509-Zertifikat parst und die Informationen über ein Array auf dem Bildschirm zurückgibt, konnte mit Erfolg zum Einsatz kommen.

Als Test-Szenario wurde ein Server der FH Karlsruhe angenommen, der einem sich mit Vor-, Zunamen und Passwort einloggenden Studenten die Daten der zu erbringenden Leitungsnachweise des aktuellen Semesters auflistet.

Dazu wurde eine MySQL-Datenbank mit dem Namen `wampTest` angelegt, die nach dem folgenden Datenmodell aufgebaut ist:

6.1 Die MySQL-Datenbank wampTest

6.1.1 Konzeptueller Entwurf:

Um den konzeptuellen Entwurf der Anwendung zu entwickeln, wurde ein Entity-Relationship-Modell eingesetzt, das im Folgenden abgebildet ist.

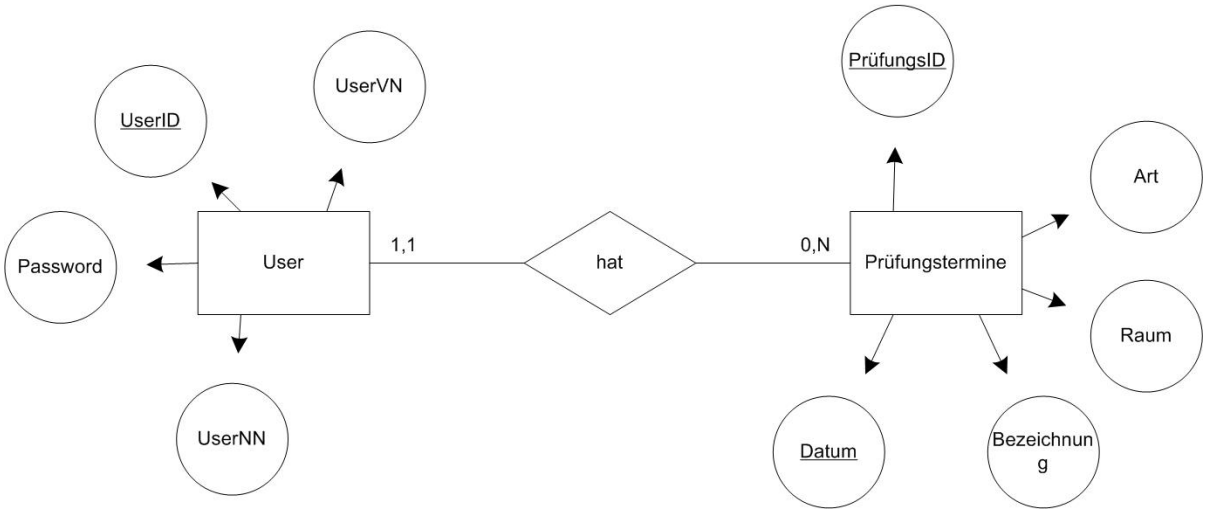

Abbildung 9: Entity-Relationship-Modell der Testanwendung

Die aus der Entität User resultierende Tabelle `user` enthält die Login-Daten der Studenten. Zu diesen Daten gehören die eindeutige User-ID, die auch den Schlüssel der Tabelle bildet, der Vor- und Zuname des Studenten und das Passwort.

Die aus der Entität Prüfungstermine resultierende Tabelle `pruefungen` enthält die Daten über die abzulegenden Leistungsnachweise. Zu den in dieser Tabelle gespeicherten Daten gehören die eindeutige Prüfungs-ID und das Prüfungs-Datum, die gemeinsam den Schlüssel dieser Tabelle bilden, eine genaue Klartext-Bezeichnung, den Raum der Prüfung und die Art des Leistungsnachweises.

Zwischen beiden Entitäten besteht eine 1:N-Beziehung. Das heißt, dass jeder Student keinen oder mehrere Leistungsnachweise ablegt und jeder in der Tabelle `pruefungen` abgelegte Leistungsnachweis genau einem Studenten zuzuweisen ist. Dies wird mit einem der Tabelle `pruefungen` angefügten Fremdschlüssel realisiert, der die User-ID aus der Tabelle `user` aufnimmt.

6.1.2 Logischer Entwurf

Im logischen Entwurf sind die beiden Tabellen folgendermaßen realisiert:

user				
Feld	**Bezeichnung**	**Typ**	**Null**	**Schlüssel**
userID	User-ID	Smallint(6)	Nein	Ja
userVN	Vorname	Char(20)	Ja	Nein
userNN	Nachname	Char(20)	Nein	Nein
password	Passwort	Char(20)	Nein	Nein

Tabelle 1: Die Tabelle user

pruefungen				
Feld	**Bezeichnung**	**Typ**	**Null**	**Schlüssel**
prID	Prüfungs-ID	Char(5)	Nein	Ja
prDatum	Prüfungsdatum	Date	Nein	Ja
prBez	Klartext-Bezeichnung	Char(50)	Nein	Nein
prRaum	Raum der Prüfung	Char(10)	Ja	Nein
prArt	Art der Leistung	Char(20)	Nein	Nein
prUID	Studenten-Nummer	Smallint(6)	Nein	Fremd

Tabelle 2: Die Tabelle pruefungen

6.2 Die Anwendung

6.2.1 `index.html`

Die Anwendung bietet dem Studenten beim Aufruf einen Login-Bildschirm, in dem er sich mit Vor-, Zuname und Passwort anmelden kann. Der Login-Bildschirm ist komplett mit HTML und CSS erstellt und enthält ein Formular, mit

dessen Hilfe Vor-, Zuname und Passwort per http-Post-Methode über eine sichere Verbindung an die PHP-Datei `login.php` geschickt wird.

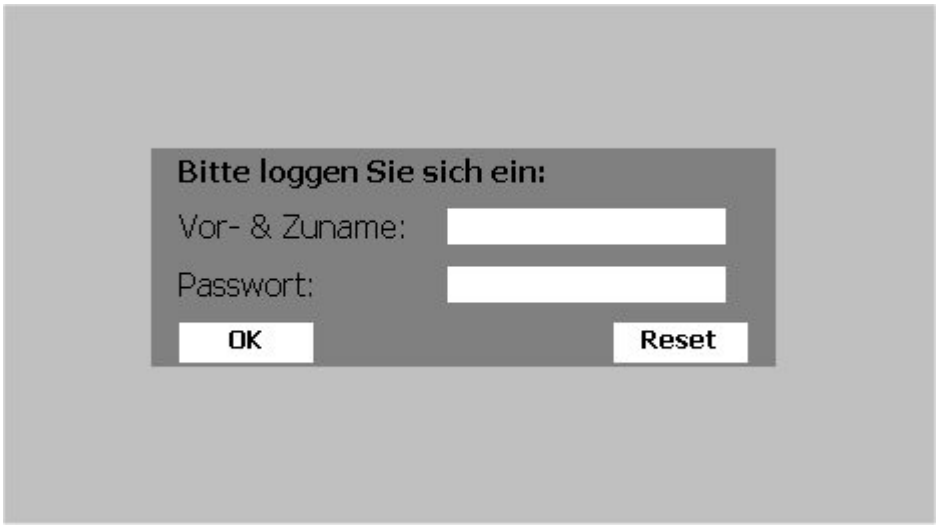

Abbildung 10: Der Login-Bildschirm

Nach der Anmeldung läuft der Rest der Anwendung im gesicherten SSL-Modus.

6.2.2 `login.php`

Die Datei `login.php` vergleicht die erhaltenen Daten mit den in der Tabelle `user` hinterlegten. Stimmen Vor- und Zuname und das zugehörige Passwort überein, wird eine HTML-Datei namens frameset.html geladen, die das Frame-Raster für den Bildschirmaufbau enthält.

Die Datei frameset.html lädt eine für ein Menü reservierte Seite in den linken Frame, die aber nicht weiter beschrieben wird, weil sie für dieses Beispiel leer blieb. Im rechten oberen Teil wird ein Header der Seite geladen, der den Namen der Fachhochschule und das Logo enthält.

Die Datei frameset.html lädt auch die PHP-Datei `exam.php`, die die Prüfungsdaten auf dem Bildschirm ausgibt, in den rechten unteren Frame.

Stimmen Vor- und Zuname und Passwort nicht überein, gibt das Programm eine Fehlermeldung aus:

Abbildung 11: Falsches Passwort

Bei mehreren Datensätzen zu einem Studentennamen (impliziert Vor- und Zunamen) wird ebenfalls eine Fehlermeldung ausgegeben:

Abbildung 12: Fehlermeldung bei mehreren Datensätzen

6.2.3 `exam.php`

Die Datei exam.php bekommt per http-Post-Methode die aus der Datenbank erhaltene User-ID weiter. Anhand dieser User-ID werden nun von der Datei die zugehörigen Daten aus der Tabelle `pruefungen` gesucht und auf dem Bildschirm dargestellt.

Weiter parst die Datei exam.php noch das verwendete X.509-Zertifikat und blendet die wichtigsten Informationen, wie z.B. die IP-Adresse des Hostes und dessen Standort, auf dem Bildschirm ein. Der Student kann durch einen Vergleich mit den im Browser abrufbaren Informationen zum eingesetzten Zertifikat die angezeigten Informationen vergleichen und so sicher gehen, dass er auch mit dem FH-Server verbunden ist.

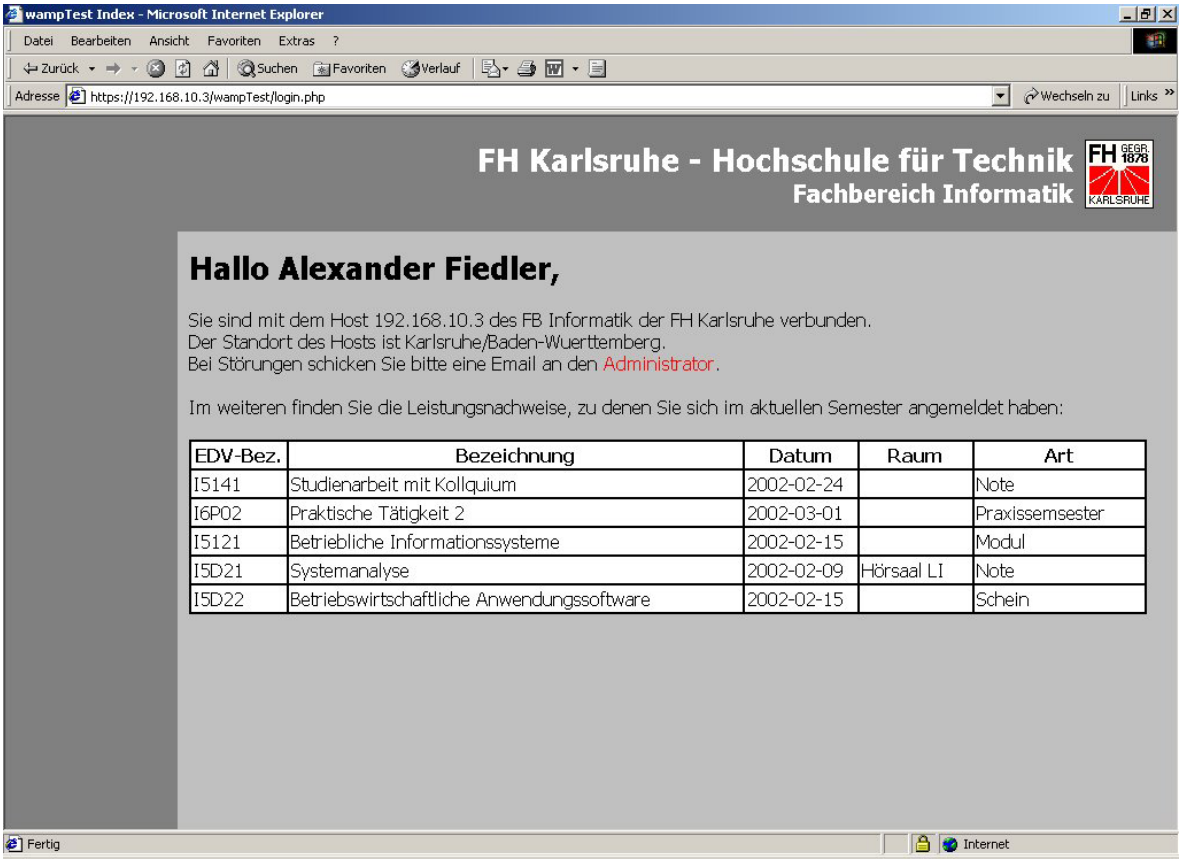

Abbildung 13: Die Ausgabe der Datei exam.php

Noch ein Hinweis:

Im Anhang (8.4 Anhang/Quellcodes, Seite 56) befinden sich sämtliche eingesetzte Quellcodes dieser Anwendung.

7 Fazit

Als Fazit bleibt zu sagen, dass die Installation eines Apache-Web-Servers mit SSL-Unterstützung viele positive und auch negative Überraschungen birgt.

So war es zum Beispiel äußerst erstaunlich, wie schwer es doch ist, einen Apache-Web-Server und das zusätzlich benötigte SSL-Modul für ein Windows-System zu kompilieren und installieren. Unter Windows 98 ist es auf zwei verschiedenen Systemen nicht gelungen, die Komponenten vollständig zu kompilieren. Bei beiden Systemen wurde der Kompiliervorgang an unterschiedlichen Stellen unterbrochen.

Dafür war die Überraschung umso größer, als die unter Windows XP vorgenommene Kompilierung und Installation zum Test des bisher Niedergeschriebenen umso reibungsloser von statten ging.

Im Vergleich zur ersten Installation unter Windows 2000 waren die erforderlichen Vorgänge unter Linux nahezu als Spaziergang anzusehen. Durch die standardmäßige Verfügbarkeit von C-Kompiler und Perl-Interpreter bei großen Linux-Distributionen ist hier ein großer Teil der vorbereitenden Arbeit weggefallen.

Es bleibt noch zu erwähnen, dass so gut wie jedes Programm-Archiv eine ausführliche Installations-Anleitung, meist in Englischer Sprache, enthält oder die zuständigen Organisationen eine gute Web-Seite mit Anleitungen zur Verfügung stellen, an denen man sich mit mehr oder minder großen Problemen entlang hangeln kann. Ohne diese Anleitungen wäre ein erfolgreicher Ausgang dieser Studienarbeit nicht sicher gewesen.

8 Anhang

8.1 Abkürzungsverzeichnis

CA	Certification Authority
CRL	Certificate Revocation List
CSS	Cascading Style Sheets
DSO	Dynamic Shared Objects
FDDI	Fiber Distributed Data Interface
FTP	File Transfer Protocol
HTML	HyperText Markup Language
HTTP	Hypertext Transfer Protocol
IETF	Internet Engineering Task Force
IP	Internet Protocol
ISDN	Integrated Services Digital Network
MAC	Message Authentication Codes
PGP	Pretty Good Privacy
PHP	Unterschiedliche Angaben in Fachliteratur: PHP: Hypertext Preprocessor oder Personal Home Page
PKI	Public Key Infrastructure
RFC	Request for Comments
SMTP	Simple Mail Transfer Protocol
SSL	Secure Sockets Layer
TCP	Transmission Control Protocol
TLS	Transport Layer Security
UDP	User Datagram Protocol
URL	Uniform Resource Locator

8.2 Abbildungsverzeichnis

8.3 Tabellenverzeichnis

8.4 Quellcodes

8.4.1 index.html

```
<!doctype html public "-//W3C//DTD HTML 4.0 //EN">
<html>
<head>
<title>wampTest Main-Template</title>
<meta name="author" content="fifi">
<meta name="generator" content="Ulli Meybohms HTML EDITOR">
<link rel="stylesheet" href="CSS/wampTest.css"
type="text/css">
</head>
<body class="main">
<div style="position: absolute; top: 45%; left: 40%; visi-
bility: visible;">
 <form action="https://192.168.10.3:443/wampTest/login.php"
method="post">
 <table class="login" border="0" cellapscing="0" cellpad-
ding="0" width="300" height="100">
  <colgroup>
   <col width="10">
   <col width="120">
   <col width="10">
   <col width="150">
   <col width="10">
  </colgroup>
  <tr>
   <td></td>
   <td class="label" colspan="3">Bitte loggen Sie sich
ein:</td>
   <td></td>
  </tr>
  <tr height="5">
   <td colspan="5"> </td>
  </tr>
  <tr>
   <td></td>
   <td>Vor- & Zuname:</td>
   <td></td>
   <td><input class="standard" type="Text" name="loginName"
value="" size="" maxlength=""></td>
   <td></td>
  </tr>
  <tr height="5">
   <td colspan="5"> </td>
  </tr>
  <tr>
   <td></td>
   <td>Passwort:</td>
```

```
    <td></td>
    <td><input class="standard" type="password"
name="password" value="" size="" maxlength=""></td>
    <td></td>
   </tr>
   <tr height="5">
    <td colspan="5"> </td>
   </tr>
   <tr>
    <td></td>
    <td align="left"><input class="buttonStandard"
type="Submit" name="" value="OK"></td>
    <td></td>
    <td align="right"><input class="buttonStandard"
type="reset" value="Reset"></td>
    <td></td>
   </tr>
  </table>
  </form>
</div>
</body>
</html>
```

8.4.2 wampTest.css

```
a:link { font-family: Tahoma, Arial; font-size: 12pt; font-
  weight:normal; color:#FF0000; text-decoration:underline }
a:visited { font-family: Tahoma, Arial; font-size: 12pt;
  font-weight:normal; color:#FF0000; text-decoration:none }
a:hover { font-family: Tahoma, Arial; font-size: 12pt;
  font-weight:bold; color:#FF0000; text-
  decoration:underline }
a:active { font-family: Tahoma, Arial; font-size: 12pt;
  font-weight:bold; color:#000000; text-
  decoration:underline }

body.frame { background-color: #808080; font-family: Ta-
  homa, Arial; font-size: 12pt; font-color:#000000; font-
  weight: normal;}
body.main { background-color: #C0C0C0; font-family: Tahoma,
  Arial; font-size: 12pt; font-color:#000000; font-weight:
  normal;}

frame.standard {border: 2pt; margin: 0pt; border-color:
  #808080;}

table.standard {border: 0; table-layout: fixed;}
table.login {border: 0; table-layout: fixed; background-
  color: #808080; font-family: Tahoma, Arial; font-size:
  12pt; font-color:#000000; font-weight: normal;}
```

```
table.exam {border: 1 #000000 solid; table-layout: fixed;
 background-color: #FFFFFF; font-family: Tahoma, Arial;
 font-size: 12pt; font-color:#000000; font-weight: nor-
 mal;}
th.examHead {border: 1px #000000 solid; background-color:
 #FFFFFF; font-family: Tahoma, Arial; font-size: 12pt;
 font-color:#000000; font-weight: bold; text-align: cen-
 ter;}
td.exam {border: 1px #000000 solid; background-color:
 #FFFFFF; font-family: Tahoma, Arial; font-size: 12pt;
 font-color:#000000; font-weight: normal;}

span.header1 {font-familiy: Tahoma, Arial; font-weight:
 bold; font-size: 20pt; color:#FFFFFF;}
span.header2 {font-familiy: Tahoma, Arial; font-weight:
 bold; font-size: 16pt; color:#FFFFFF;}

.label {font-weight: bold; font-size: 11pt;}

input.standard {border: 0;}

.buttonStandard {background-color: #FFFFFF; font-family:
 Tahoma, Arial, font-size: 10pt; font-weight: bold; width:
 50pt; border:0;}

.red {color:red; font-weight:bold;}
```

8.4.3 login.php

```
<?
include("database.inc.php");

$lk=connect();
$db="wampTest";
//print "db: $db<br>";

$name = explode(" ", $loginName);
$sql = "SELECT * FROM user WHERE userVN = \"".$name[0]."\"
      AND userNN = \"".$name[1]."\"";
//print "sql: $sql<br>";
$res = sendSQL($db, $sql);
$anz=mysql_num_rows($res);


for ($i=0; $i<$anz; $i++){
      $data[$i] = mysql_fetch_array($res, MYSQL_ASSOC);
}

if ($anz==1){
     if ($data[0]["password"]==$password){
```

```
            $fp1 = fopen("frameset.html", "r");
            $page = fread($fp1, 8192);
            fclose($fp1);
            $page=str_replace("%uid%",
"?uid=".$data[0]["userID"], $page);
            print $page;
      } else {
        include ("htmlFrameTop.inc.php");
        print "<h1 class=\"red\">Falsches Pass-
            wort!!</h1>\n";
        include ("htmlFrameBottom.inc.php");
      }
} else {
  include ("htmlFrameTop.inc.php");
  print "<h1 class=\"red\">Es wurde mehr als ein Datensatz
     zum Namen $loginName gefunden!!</h1>\n";
  include ("htmlFrameBottom.inc.php");
}

?>
```

8.4.4 frameset.html

```
<!doctype html public "-//W3C//DTD HTML 4.0 //EN">
<html>
<head>
<title>wampTest Index</title>
<meta name="author" content="fifi">
<meta name="generator" content="Ulli Meybohms HTML EDITOR">
<link rel="stylesheet" href="CSS/wampTest.css"
type="text/css">
</head>
<frameset border="0" cols="150,*">
   <frame class="standard" frameborder="0" scrolling="no"
      name="menu" src="menu.html">
   <frameset  rows="100,*">
      <frame class="standard" frameborder="0" scroll-
       ing="no" name="head" src="head.html">
      <frame class="standard" frameborder="0" scroll-
       ing="auto" name="main" src="exam.php%uid%">
   </frameset>
</frameset>
<noframes>
Ihr Browser unterstützt keine Frames!
</noframes>
</html>
```

8.4.5 head.html

```
<!doctype html public "-//W3C//DTD HTML 4.0 //EN">
<html>
<head>
<title>wampTest Head</title>
<meta name="author" content="fifi">
<meta name="generator" content="Ulli Meybohms HTML EDITOR">
<link rel="stylesheet" href="CSS/wampTest.css"
type="text/css">
</head>
<body class="frame">
<table class="standard" border="0" cellspacing="0" cellpad-
 ding="0" width="100%">
 <colgroup>
  <col width="10">
  <col width="*">
  <col width="10">
  <col width="60">
  <col width="10">
 </colgroup>
 <tr height="5">
  <td colspan="5"></td>
 </tr>
 <tr>
  <td></td>
  <td align="right"><span class="header1">FH Karlsruhe -
    Hochschule für Technik</span><br><span
    class="header2">Fachbereich Informatik</span></td>
  <td></td>
   <td><img src="images/fh-logo.gif" width="60" border="0"
    alt=""></td>
  <td></td>
 </tr>
 <tr height="5">
  <td colspan="5"></td>
 </tr>
</table>
</body>
</html>
```

8.4.6 exam.php

```
<?
include ("database.inc.php");
include ("htmlFrameTop.inc.php");

$lk=connect();
$db="wampTest";

$name = explode(" ", $loginName);
$userSql = "SELECT * FROM user WHERE userID = ".$uid;
//print "sql: $userSql<br>";
$userRes = sendSQL($db, $userSql);
$userAnz=mysql_num_rows($userRes);

for ($i=0; $i<$userAnz; $i++){
      $userData[$i] = mysql_fetch_array($userRes,
       MYSQL_ASSOC);
}

print "<h1>Hallo ".$userData[0]["userVN"]."
 ".$userData[0]["userNN"].",</h1>";

$fp3 = fopen("C:\\apache\\conf\\ssl\\fh-server.cert", "r");
$key = fread($fp3, 8192);
fclose($fp3);

$certInfo = openssl_x509_parse($key, FALSE);

$helloMsg = "Sie sind mit dem Host
 ".$certInfo["issuer"]["commonName"]." des
 ".$certInfo["issuer"]["organizationalUnitName"];
$helloMsg.= " der
 ".$certInfo["issuer"]["organizationName"]." verbun-
 den.<br>Der Standort des Hosts ist ";
$helloMsg.= $certInfo["issuer"]["localityName"]."/"
 .$certInfo["issuer"]["stateOrProvinceName"].".<br>";
$helloMsg.= "Bei Störungen schicken Sie bitte eine Email an
 den <a href=\"mailto:".$certInfo["issuer"]["emailAd-
 dress"]."\">";

$helloMsg.= "Administrator</a>.<br><br>Im weiteren finden
 Sie die Leistungsnachweise, zu denen Sie sich im aktuel-
 len Semester angemeldet haben:<br><br>";
print $helloMsg;

$examSql = "SELECT * FROM pruefungen WHERE prUID = ".$uid;
$examRes = sendSQL($db, $examSql);
$examAnzRows=mysql_num_rows($examRes);
$examAnzFields=mysql_num_fields($examRes);
```

```
for ($k=0; $k<$examAnzRows; $k++){
      $examData[$k] = mysql_fetch_array($examRes,
MYSQL_ASSOC);
}

print "<table class=\"exam\" cellspacing=\"0\" cellpad-
ding=\"2\" width=\"100%\">\n";
print "<colgroup>\n<col width=\"85\">\n<col
width=\"*\">\n<col width=\"100\">\n<col width=\"100\">\n";
print "<col width=\"150\">\n</colgroup>\n";
print "<tr>\n<th class=\"examHead\">EDV-Bez.</th>\n";
print "<th class=\"examHead\">Bezeichnung</th>\n";
print "<th class=\"examHead\">Datum</th>\n";
print "<th class=\"examHead\">Raum</th>\n";
print "<th class=\"examHead\">Art</th>\n</tr>\n";
for ($j=0; $j<$examAnzRows; $j++){
   print "<tr>\n<td
    class=\"exam\">".$examData[$j]["prID"]." </td>\n";
   print "<td
    class=\"exam\">".$examData[$j]["prBez"]." </td>\n"
    ;
   print "<td
    class=\"exam\">".$examData[$j]["prDatum"]." </td>\
    n";
   print "<td
    class=\"exam\">".$examData[$j]["prRaum"]." </td>\n
    ";
   print "<td
    class=\"exam\">".$examData[$j]["prArt"]." </td>\n<
    /tr>\n";
}
print "</table>\n";

include ("htmlFrameBottom.inc.php");
?>
```

8.4.7 PHP Include-Dateien

8.4.7.1 database.inc.php

```
<?
$wampHost="192.168.10.3";
$wampUser="root";
$wampPass="";

function connect(){
        global $wampHost, $wampUser, $wampPass;
        if (! $linkID=mysql_connect("$wampHost",
              "$wampUser", "$wampPass")){
               print "<span class=\"label\">Die Verbin-
                    dung zu $wampHost konnte nicht herge-
                    stellt werden!</span>\n";
               exit;
        }
        return $linkID;
}

function sendSQL($db, $sql){
        if(! $res=mysql_db_query($db, $sql)){
              print mysql_error();
              exit;
        }
        return $res;
}
?>
```

8.4.7.2 htmlFrameTop.inc.php

```
<?
print "<!doctype html public \"-//W3C//DTD HTML 4.0
 //EN\">\n";
print "<html>\n";
print "<head>\n";
print "<title>wampTest Main-Template</title>\n";
print "<meta name=\"author\" content=\"fifi\">\n";
print "<meta name=\"generator\" content=\"Ulli Meybohms
 HTML EDITOR\">\n";
print "<link rel=\"stylesheet\" href=\"CSS/wampTest.css\"
 type=\"text/css\">\n";
print "</head>\n";
print "<body class=\"main\">\n";
?>
```

8.4.7.3 htmlFrameBottom.inc.html

```
<?
print "</body>\n";
print "</html>\n";
?>
```

8.5 Quellenverzeichnis

8.5.1 Drucksachen

8.5.1.1 Bücher

[B1] **Ertel**, Wolfgang
Angewandte Kryptographie,
erschienen 2001 im Fachbuchverlag Leipzig

[B2] **Stoll**, Rolf D. und **Leierer**, Gudrun Anna
PHP 4 + MySQL
erschienen 2000 in 2. Auflage bei Data Becker

[B3] **Bowen**, Rich und **Coar**, Ken
Apache und CGI
erschienen 2000 im Markt + Technik Verlag

8.5.1.2 Andere Unterlagen

[A1] **Schäfer-Lorinser**, Frank
Vorlesungsunterlagen zur Vorlesung „elektronische Medien und Märkte" im Fachbereich Informatik der FH Karlsruhe

[A2] **Schäfer-Lorinser**, Frank
Vorlesungsunterlagen zur Vorlesung „elektronische Bezahlsysteme" im Fachbereich Informatik der FH Karlsruhe

8.5.2 Internet

8.5.2.1 Software

[I1] http://www.apache.org
Die Homepage der Apache-Foundation. Neben dem freien Web-Server Apache und zugehörigen Anleitungen finden sich auf dieser Seite links zu weiteren Projekten der Apache-Foundation (z.B. Jakarta, etc.).

[I2] http://www.modssl.org

Hier wird das Apache-Modul modssl, das das Open-Source SSL-Toolkit OpenSSL mit dem Apache-Server zusammenarbeiten lässt, zum download angeboten. Natürlich gibt es auch auf dieser Seite jede Menge Informationen und Hilfestellungen zum Modul.

[I3] http://www.openssl.org

Die Homepage des Open Source Toolkits OpenSSL. Neben den aktuellen Source-Codes gibt es hier auch jede Menge Information rund um das Toolkit selbst.

[I4] http://www.perl.com/CPAN-local

Auf der Homepage des Comprehensive Perl Archive Network finden sich der Quellcode und die Windows-Binary-Files für den Perl-Interpreter, der für die Installation von mod_ssl benötigt wird.

[I5] http://www.php.net

Hinter diesem URL verbirgt sich die Homepage der PHP Group. Neben den aktuellen PHP-Quellcodes finden sich hier auch jede Menge Tipps und Tricks rund um die freie serverseitige Skriptsprache.

[I6] http://www.mysql.com

Die Homepage der MySQL-Macher, die neben den aktuellen Versionen auch jede Menge Infos zum freien Datenbanksystem anbietet.

[I7] http://www.phpwizard.net/projects/phpMyAdmin/

Hier kann das freie MySQL-Administrationstool phpMyAdmin heruntergeladen werden, das eine Administration des Datenbanksystems per Internet bzw. Web-Server ermöglicht.

[I8] http://cygwin.com

Von dieser Homepage kann das freie Tool CygWin32 heruntergeladen werden, das unter Windows eine UNIX-Umgebung emuliert und so Funktionalitäten wie entpacken etc. auch unter Windows ermöglicht.

[I9] http://www.winzip.com

Auf dieser Homepage findet sich WinZip, das bekannte und weitverbreitete Archivierungs-Tool für Windows, zum Download. Die Evaluation Version ist ohne Einschränkungen zum Test des Programms nutzbar.

[I10] http://cm.bell-labs.com/cm/cs/who/bwk/

Seite von Brian Kernighan (Bell Labs), von der das für die Apache-Kompilierung notwenige Programm awk.exe heruntergeladen werden kann.

[I11] http://www.e-technik.fh-schmalkalden.de/labore/wspool/pages/soft/awk.html?fc=ok

Seite der FH Schmalkalden zum Programm awk.exe, das in Textform gespeicherte Datenbanken bearbeiten kann.

8.5.2.2 Installationsanleitungen

[I12] http://www.uni-koeln.de/RRZK/sicherheit/ssl-ca/apache.html

Seite mit Links und Beschreibungen zur Installation eines Apache-Web-Servers. Unter Anderem findet sich hier eine Installations-Kurzanleitung für ein Unix-System.

[I13] http://www.baach.de/lamp-tutorial.html

Sehr gute Installationsanleitung für einen LAMP-Server mit SSL.

8.5.2.3 Zertifkate

[I14] http://www.trustcenter.de

Internetpräsenz der deutschen Certification Authority. Neben allerlei Informationen über X.509-Zertifikate, gibt es hier auch die Möglichkeit CRL´s herunterladen oder Zertifikate zu beantragen.

[I15] http://www.verisign.com

Die wohl bekannteste CA mit Sitz in den USA, der ebenfalls mit einem großen Angebot an Informationen und Zertifikaten im Internet vertreten ist.

[I16] http://www.thawte.com

Thawte hat den Hauptsitz in Kapstadt/Südafrika und bietet ebenfalls ein großes Angebot rund um X.509-Zertifikate.

8.5.2.4 Sonstige Links

[I17] http://www.ietf.org

Homepage der Internet Engineering Task Force, die für die Festlegung von Internet-Standards zuständig ist. Hier sind auch alle RFCs hinterlegt, die die Internetstandards beschreiben.

[I18] http://www.suse.de

Die Homepage des deutschen Linux-Distributors, die neben Hilfe auch Programm-Updates bereitstellt.