Konzepte von Betriebssystem-Komponenten

Schwerpunkt Internetsicherheit

Secure Socket Layer v. 3.0

(SSLv3)

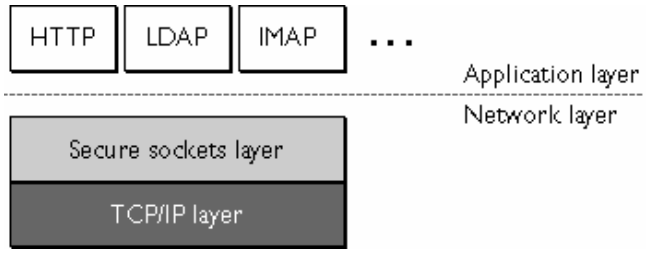
Zheng Yao

05.07.2004

1. Was ist SSL?

SSL steht für Secure Socket Layer, ein Protokoll zur Übertragung von verschlüsselten Informationen. Ursprünglich wurde es von Netscape im Jahr 1994 entwickelt, um mehr Sicherheit für die Online-Transaktion anzubieten. Es wurde im Netscape's Webbrowser und Webserver implementiert. Jetzt wird SSL schon bei den meisten Webbrowsern und -servern verwendet

Das SSL-Protokoll befindet sich zwischen die Anwendungsschicht und die Transportschicht und bietet den Schutz für Transportsschicht an. Es läuft auf TCP/IP Protokoll, und wird von den Protokollen wie HyperText Transport Protocol (HTTP), Lightweight Directory Access Protocol (LDAP), Internet Messaging Access Protocol(IMAP) etc. benutzt.



Quelle:www.netscape.com

SSL wurde für den Einsatz zwischen Client und Server entwickelt. Das Protokoll besteht wesentlich aus drei Teilen [1]:

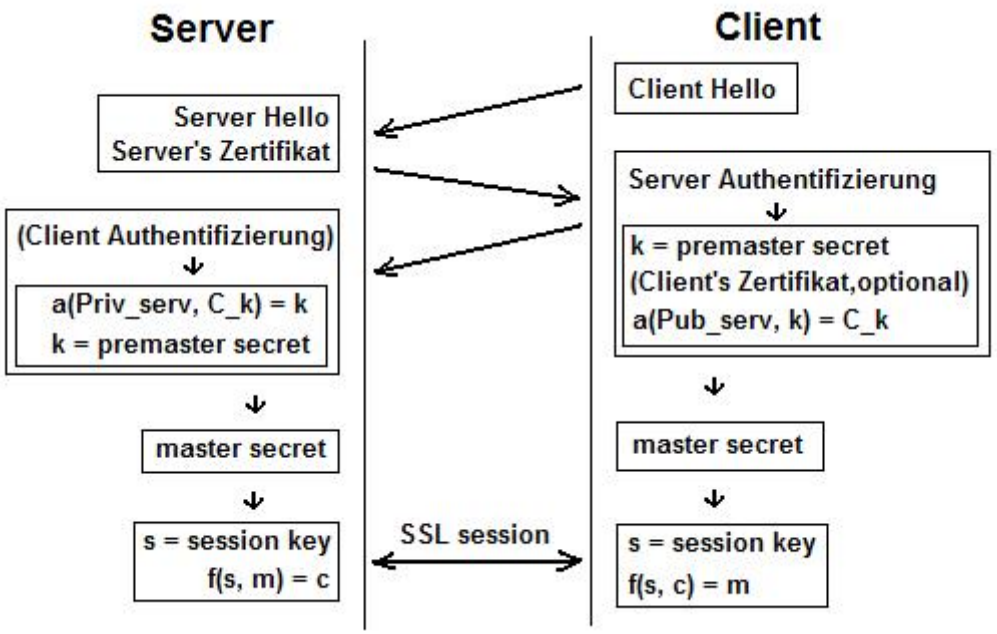
- **Das Record-Protokoll,** dieses Protokoll ist wie eine zusätzliche Schicht über allen SSL-Nachrichten. Es gibt die angewandeten Verschlüsselungs- und Authentifizierungsfunktionen an.
- **Das Handshake-Protokoll,** mit diesem Protokoll behandeln Client und Server den Einsatz von kryptographischen Algorithmen und Schlüsseln.
- **Das Alert-Protokoll,** dieses Protokoll signalisiert das Auftreten von Fehlern oder den Abbruch einer Kommunikationsverbindung zwischen Server und Client.

SSL Handshake *„SSL Hello"*

Wie wird eine Kommunikationsverbindung mit SSL hergestellt? Abbildung-2 zeigt, wie die SSL-Nachrichten zwischen Client und Server ausgetauscht werden, um eine SSL-Verbindung aufzubauen. Diesen Vorgang nennt man „SSL-Hello".

Der Vorgang geliedert sich in folgenden Schritte [4]:

1. **„Client Hello"** Client sendet die client's SSL Versionnummer, Cipher Einstellung, Zufallsdaten, und die andere Informationen, die bei den Kommunikationen mit SSL für Server nötig sind.
2. **„Server Hello"** Server sendet die server's SSL Versionnummer, Cipher Einstellung, Zufallsdaten, und die andere Informationen, die bei den Kommunikationen mit SSL für Client nötig sind. Der Server sendet auch seinen Zertifikat. Wenn Client die Resource in Server anfordert , die Authentikation von Client brauchen, anfordert der Server auch das Zertifikat von Client.



3. **„Server Authentifizierung“** Client bestätigt den Server. Wenn der Server nicht bestätigt werden kann, wird der Vorgang fehlergeschlagen.
4. **„Premaster Secret“** Mit den vorhandenen Daten erzeugt Client eine **premaster secret** für diese Session, und verschlüsselt sie mit Server's public key (von Server's Zertifikat) und dann sendet die verschlüsselte **premaster secret** an den Server.
5. **„Client Authentifizierung“**(optional) Wenn Client Authentifizierung von Server angefordert wird, erzeugt Client zusätzlich einige Zufallsdaten und signiert sie mit selben Private Key. Dann werden alle Daten - Client signierte Daten, Client’s Zertifikat und die verschlüsselte premaster secret dem Server gesendet.
6. **„Master Secret“** Wenn Server die Anforderung von Client Authentifizierung hatte, dann bestätigt Server den Client. Wenn Client nicht bestätigt wird, wird der Vorgang fehlergeschlagen. Wenn Client bestätigt wird, entschlüsselt der Server mit selben Private Key die Nachricht und bekommt die premaster secret. Mit der premaster secret können Server und Client die gleichen **master secret** erzeugen.
7. **„Session Keys“** Server und Client erzeugen mit der master secret die Session Keys**,** die symmetreisch Keys sind und mit der die Informationen bei der Kommunikation mit SSL verschlüsselt und entschlüsselt werden können und auch die Vollständigkeit von Daten geprüft werden kann.
8. **„Client Finishing“** Client sendet ein Nachricht, um den Server zu informieren, dass alle weitere Nachrichten von Client mit Session Key verschlüsselt werden. Und dann sendet Client ein verschlüsselt Finishing-Nachricht an den Server.
9. **„Server Finishing“** Server sendet ein Nachricht, um den Client zu informieren, dass alle weitere Nachrichten von Server mit Session Key verschlüsselt werden. Und dann sendet Server ein verschlüsselt Finishing-Nachricht an den Client.
10. **„Handshake Finishing“** Der Vorgang von SSL handshake ist hier schon fertig , und die SSL session hat began. Client and Server dann verschlüsseln und entschlüsseln die Daten mit den Session Keys, und auch prüfen mit den Keys die Vollständigkeit von Daten.

SSL Versionen

SSL wurde für das Einsetzung in Netscape Navigator entworfen. So wurde SSLv1 nur in Netscape verwendet. Nach SSLv2 hat Microsoft auch eine SSLv2 Inplementierung, die als PCT genannt wird. Die neuste Version von SSL ist 3.0. In 1996 hat die Arbeitsgruppe der IETF(Internet Engineering Task Force) ein neues standardisiertes Sicherheitsprotokoll TLS(Transport Layer Security) entwickelt. TLS basisiert auf SSLv3 und ist eigentlich die Version 3.1 von SSL Protokoll.

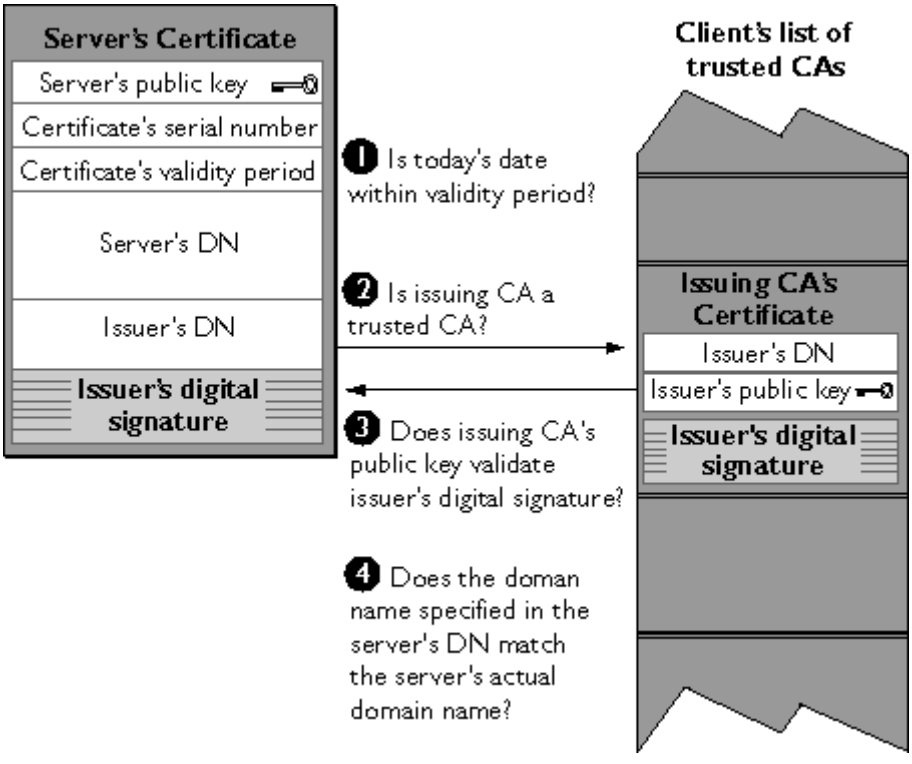
2. Digitale Authentifizierung

SSL benutzt **Public Key Zertifikate** für die Authentifizierung von Server und Client.
SSL unterstützt die folgende Zertifikate [2]:

- RSA public key Zertifikate, die Keys können beliebige Länge haben.
- RSA public key Zertifikate, die keys sind nur 512 bits. Für besondere Anwendung mit Beschränkung von Kryptographie.
- Signing-only RSA Zertifikate, die nur für Signieren der Daten sind. Sie enthalten RSA public keys, die nur für Signieren der Daten , nicht für Verschlüsselung.
- DSS Zertifikate
- Diffie-Hellman Zertifikate

Die folgenden zwei Abbilgungen zeigen uns, wie ein Zertifikat von Server oder Client authentifiziert wird.

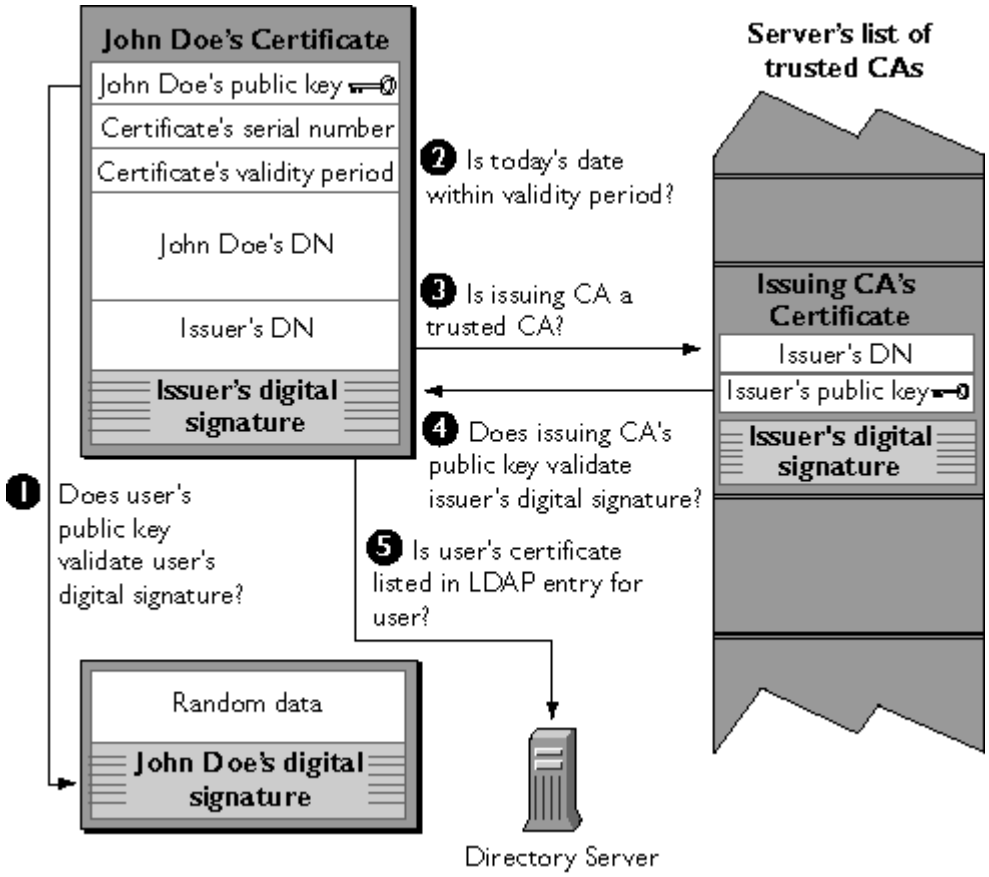
Server Authentifizierung:



Quelle:www.netscape.com

Abbildung-4 Wie wird ein Server Zertifikat authtifiziert?

Client Authentifizierung:
Die Authentifizierung von Client ist optional.



Quelle:www.netscape.com

Abbildung-5 Wie wird ein Client Zertifikat authtifiziert?

CA: (Certification Authority) eine Organisation, die digitale Zertifikate herausgibt.
DN: (Distinguished Name) bekannter Name
LDAP:(Lightweight Directory Access Protocol) Es ist ein Protokoll, das für die Kommunikation zwischen Client und X.500-Verzeichnisdienst gedacht ist.

3. SSL/TLS Features

- **Verschiedene Algorithmen für die Aufgaben**

SSL benutzt verschiedene Algorithmen für Verschlüsselung, Authentifizierung und Vollständigkeit der Daten, die jede einen eigenen Schlüssel haben kann. Die Länge von Schlüsseln kann auch verschieden sein. SSL erlaubt die nicht verschlüsselte aber authentifizierte Verbindung. Dies ist nützlich, wenn die Anwendung von Verschlüsselung verboten ist.

- **Effizienz**

Die Operation von Verschlüsselung und Entschlüsselung der Public Keys kostet realtiv viel Zeit. Um mehr Male Wiederholung von dieser Operation zu vermeiden, wird die SSL/TLS erst zwischen Server und Client eine „master secret“ generieren, dann wird die weitere Kommunikation mit symmetrischen Verfahren sicher durchgeführt werden.

- **Auf Zertifikate basierende Authentifizierung**

Durch die Anwendung von digitallen Zertifikationen bietet SSL die Authentifizierung von Client und Server an. SSLv3 und TLS benutzen X.509 Zertifikate. Aber Authentifizierung ist ein optionales Teil des SSL-Protokolles.

- **Protokoll agnostisch**

SSL kann nicht nur über TCP/IP, sondern auch über viele andere zuverlässige Protokolle in der Transportschicht arbeiten. Z.B.: X.25. Aber nicht über die unzuverlässigen Protokollen wie IP User Datagram Protocoll(UDP).

- **Schutz vor man-in-the-middle- und replay- Angriff**

Bei dem man-in-the-middle-Angriff fängt der Hacker alle Infomationen zwischen Webserver und -benutzer und läßt die beiden immer glauben, dass sie nur miteinander kommunizieren. Bei dem replay-Angriff fängt der Hacker die Bezahlungsnachrichten und schafft die illegale Transaktion durch Wiederholung dieser Nachrichten.
SSL benutzt **digitale Authentifizierung** gegen diese Angriffe.

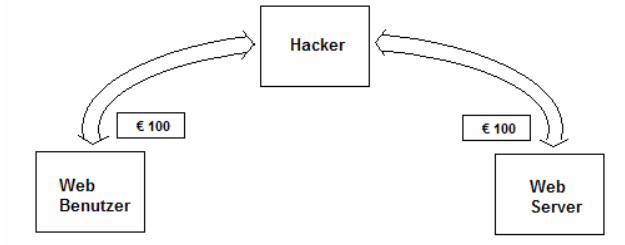


Abbildung-6 man-in-middle-Angriff

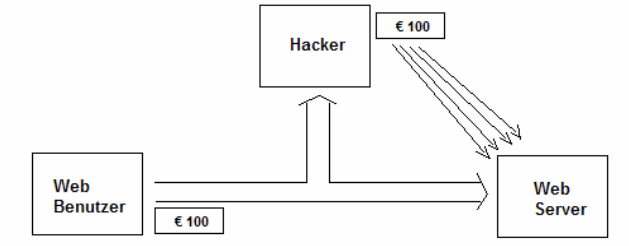


Abbildung-7 replay-Angriff

nach Web Security, Privacy, and Commerce (2nd Edition), Simson Garfinkel with Gene Spafford: O'Reilly, 2002

- **Unterstützung von Kompression**

Normalerweise können die gut verschlüsselten Daten nicht mehr komprimiert werden. So macht das SSL/TLS Protokoll eine Komprimierung von Daten, bevor diese Daten verschlüsselt werden.

- **Ruckwärts Kompatibilität mit SSL 2.0**

SSLv3 Server nimmt auch die Verbindung von SSLv2 Client auf und kann sofort die SSL-Nachrichten behandeln, ohne eine neue Verbindung mit Client aufzubauen.

4. SSL Implementierungen

SSL wurde erst in Juli 1994 von Netscape entwickelt und gehörte zu den Netscape´s Business Plans. Dises Protokoll gibt Web-Benutzern die Möglichkeit, sicher auf Web-Server zuzugreifen, um damit kommerzielle Transkationen durchzuführen.

- **SSL Netscape**

Die erste SSL Implementierung wurde nur in Netscape´s Webbrowser und Webserver verwendet.

- **SSLRef und Mozilla Network Security Services(NSS)**

Nach der Einsetzung von Netscape Navigator, hatte Netscape eine Referenz-implementierung von SSL produziert. Sie wurde in C geschrieben und hat den Name SSLRef. Jetzt ist SSLRef nicht mehr verfügbar. Mozilla NSS hat sie ersetzt und bietet eine große Menge von Bibliotheken für die Entwicklung an. Die mit NSS entwickelte Anwendungsprogramme unterstützen SSLv2, SSLv3, TLS und viele andere Internetsicherheitsstandards.

- **SSLeay und OpenSSL**

SSLeay ist eine selbständige Implementierung von SSL 3.0 und wurde von zwei australische Programmer Eric A.Young und Tim Hudson entwickelt. OpenSSL basisiert auf SSLeay und ist jetzt eine extensiv angewandete SSL Implementierung. Es ist auch die Basis von SSL Implementierung in Apache Web Server.

- **SSL Java**

Es gibt auch midestents zwei SSL Implementierungen in Java: Cryptography Extensions von Sun´s Java und JSAFE von RSA Data Security.

Quellen:

[1] Richard E.Smith Boon, Internet-Kryptographie, Addison-Wesley-Longman, 1998

[2] Simson Garfinkel with Gene Spafford, Web Security, Privacy, and Commerce (2nd Edition), O'Reilly, 2002

[3] Alan O. Freier, Philip Karlton, Paul C. Kocher, Netscape, The SSL Protocol Version 3.0, http://wp.netscape.com/eng/ssl3/ssl-toc.html, stand: March 1996

[4] Netscape, Introduction to SSL, http://developer.netscape.com/docs/manuals/security/sslin/, stand: Sept. 1998

[5] T. Dierks, C. Allen, The TLS 1.0 Protocol, http://www.ietf.org/rfc/rfc2246.txt, stand: January 1999

[6] The OpenSSL Project, http://www.openssl.org, stand: Juni 2004

[7] The Network Security Services Project (NSS), http://www.mozilla.org/projects/security/pki/nss/, stand: Juni 2004.