Denn es geht um Ihr Geld :
[A]symmetrische Verschlüsselung,
Hashing, Zertifikate, SSL/TLS

Verschlüsselungsverfahren: Symmetrische Verschlüsselung

- 1 Key, der zum Ver- und Entschlüsseln benutzt wird (beliebig)
- Problem: Der Key muss allen (berechtigten!) Teilnehmern bekannt sein
- Lösung: Diffie-Hellman-Schlüsselaustausch (anfällig für Man-in-the-Middle -Angriffe)

Verschlüsselungsverfahren: Asymetrische Verschlüsselung

- 1 Schlüsselpaar (Keys voneinander abhängig)
- Je ein öffentlicher (allen bekannt) und ein privater (nur 1 Person bekannt) Key
- Prinzip: Wird ein Text mit dem öffentlichen Key verschlüsselt, kann er nur noch mit dem privatem Key entschlüsselt werden

Schlüsselgenerierung (RSA)

1. Wähle 2 Primzahlen p und q
2. Berechne $n = p * q$ sowie $\varphi(n) = (p-1) * (q-1)$
3. Wähle eine natürliche Zahl e mit
 $1 < e < \varphi(n)$ und $\text{ggT}(e, \varphi(n)) = 1$
4. Berechne d mit
 $1 < d < \varphi(n)$ und $e * d \bmod \varphi(n) = 1$
5. Dann ist (n, e) der private Key und (n, d) der öffentliche

Hashing

- Bildet beliebig große Eingabedaten auf einen kleinen Wert ab (md5 128 bit, SHA-xxx xxx bit)
- Einsatz
 - Datenbanken (Hashtabellen)
 - Ähnliche Eingabe → Ähnlicher Hash
 - Prüfsummen (schützt vor Übertragungsfehlern und gezielter Manipulation)
 - Als kryptographische Hashfunktion
 - Ähnliche Eingabe → sehr unterschiedlicher Hash
 - Schwer, gleichen Hash zu finden

Signatur

- Ver-/Entschlüsselung sind nur Umkehrfunktionen zueinander
- Prinzip: Hash der Nachricht mit privatem Key verschlüsseln
- Empfänger öffnet die Signatur mit öffentlichem Key und vergleicht mit Hash der Nachricht
 - Bei Ungleichheit: Manipulationsversuch

Zertifikat

- Wird üblicherweise von einer Zertifizierungsstelle (zb. Symantec) beglaubigt
- Enthält Public Key und Informationen über Eigentümer
- Ist signiert

Beispiel für ein X.509-Zertifikat

Certificate:

Data:

Version: 3 (0x2)

Serial Number: 1 (0x1)

Signature Algorithm: md5WithRSAEncryption

Issuer: C=AT, ST=Steiermark, L=Graz, O=TrustMe Ltd, OU=Certificate Authority, CN=CA/Email=ca@trustme.dom

Validity

Not Before: Oct 29 17:39:10 2000 GMT

Not After : Oct 29 17:39:10 2001 GMT

Subject: C=AT, ST=Vienna, L=Vienna, O=Home, OU=Web Lab, CN=anywhere.com/Email=xyz@anywhere.com

Subject Public Key Info:

Public Key Algorithm: rsaEncryption

RSA Public Key: (1024 bit)

Modulus (1024 bit):

00:c4:40:4c:6e:14:1b:61:36:84:24:b2:61:c0:b5:

d7:e4:7a:a5:4b:94:ef:d9:5e:43:7f:c1:64:80:fd:

9f:50:41:6b:70:73:80:48:90:f3:58:bf:f0:4c:b9:

90:32:81:59:18:16:3f:19:f4:5f:11:68:36:85:f6:

1c:a9:af:fa:a9:a8:7b:44:85:79:b5:f1:20:d3:25:

7d:1c:de:68:15:0c:b6:bc:59:46:0a:d8:99:4e:07:

50:0a:5d:83:61:d4:db:c9:7d:c3:2e:eb:0a:8f:62:

8f:7e:00:e1:37:67:3f:36:d5:04:38:44:44:77:e9:

f0:b4:95:f5:f9:34:9f:f8:43

Exponent: 65537 (0x10001)

X509v3 extensions:

X509v3 Subject Alternative Name:

email:xyz@anywhere.com

Netscape Comment:

mod_ssl generated test server certificate

Netscape Cert Type:

SSL Server

Signature Algorithm: md5WithRSAEncryption

12:ed:f7:b3:5e:a0:93:3f:a0:1d:60:cb:47:19:7d:15:59:9b:

3b:2c:a8:a3:6a:03:43:d0:85:d3:86:86:2f:e3:aa:79:39:e7:

82:20:ed:f4:11:85:a3:41:5e:5c:8d:36:a2:71:b6:6a:08:f9:

cc:1e:da:c4:78:05:75:8f:9b:10:f0:15:f0:9e:67:a0:4e:a1:

4d:3f:16:4c:9b:19:56:6a:f2:af:89:54:52:4a:06:34:42:0d:

d5:40:25:6b:b0:c0:a2:03:18:cd:d1:07:20:b6:e5:c5:1e:21:

44:e7:c5:09:d2:d5:94:9d:6c:13:07:2f:3b:7c:4c:64:90:bf:

ff:8e

State of the Art

- Symmetrische Verschlüsselung schnell, aber (beim Schlüsselaustausch) unsicher
- Asymmetrische Verschlüsselung (besonders für den Server) langsam und teuer, aber sicher und mit Authentifizierungsmöglichkeit
- Lösung: Kombination von beidem:
Authentifizieren und Bestimmen eines Schlüssels über Asymmetrische Verschlüsselung, danach für die eigentliche Daten Symmetrische Verschlüsselung

SSL/TLS

- 1994 SSL 1.0 (Netscape Communications)
- Durch RFC 2246 1999 in TLS umbenannt (SSL 3.0 zu TLS 1.0)
- 2008 TLS 1.2 (RFC 5246)
- OSI-Schicht 5 (Sitzungsschicht) / im TCP/IP Modell zwischen Transportschicht und Anwendungsschicht

SSL/TLS (2)

- Besteht selbst aus zwei Schichten:
 1. TLS Record Protocol: Untere Schicht, dient zum Verschlüsseln und Fragmentieren der Daten
 - Fragmentiert die Daten in maximal 16.384 Byte
 - komprimiert sie und hängt einen MAC (Message Authentication Code) an (0, 16 oder 20 Byte)
 - Dann folgt Verschlüsselung und Header
 - Header: Je 2 Byte Content Type, Major- und Minorversion sowie 4 Byte Compressed length
 2. TLS Handshake Protocol

SSL/TLS (3)

- TLS Handshake Protocol besitzt 3 Unterprotokolle:
 - Change-Cipher-Spec-Protokoll: Schickt ein Byte mit Wert 1 um zur vereinbarten Verschlüsselung zu wechseln
 - Alert-Protokoll: Sendet Warn- und Fehlermeldungen sowie Befehl für ordnungsgemäßen Abbruch (In modernen Implementationen quasi unbenutzt)
 - Das eigentliche TLS-Handshake-Protokoll

TSL Handshake

- hello_server/ hello_client mit grundsätzlichen Informationen (TLS-Version, Verschlüsselungsmethode, ...) sowie je einer Zufallszahl
- Server und Client schicken sich gegenseitig ihre Zertifikate und lassen diese überprüfen (Client optional)
- Client hasht über alle bisherigen Nachrichten und signiert diese mit seinem privaten Key → Schutz vor Man-in-the-Middle
- Client generiert ein zufälliges Pre-Master-Secret und verschlüsselt dies an den

TLS-Handshake

Quellen

- Diverse Wikipedia-Artikel (
http://de.wikipedia.org/wiki/Digitales_Zertifikat,
http://en.wikipedia.org/wiki/Certificate_authority,
http://de.wikipedia.org/wiki/Symmetrisches_Kryptosystem,
,
http://de.wikipedia.org/wiki/Transport_Layer_Security,
,
http://de.wikipedia.org/wiki/Symmetrisches_Kryptosystem,
,
http://de.wikipedia.org/wiki/Asymmetrisches_Kryptosystem,
,
<http://de.wikipedia.org/wiki/Diffie-Hellman-Schl%C3%9Cssel>