Inhaltsverzeichnis

43-1

Kapitel 43
Kommunikations-Sicherheit
Verschlüsselung: Der digitale Briefumschlag

43-2

Lernziele dieses Kapitels

1. Konzept der Verschlüsselung verstehen
2. Unterschied zwischen symmetrischen und asymmetrischen Verfahren kennen
3. Konzept der digitalen Unterschrift verstehen
4. Programme zur Verschlüsselung einsetzen können
5. Zertifikate erstellen und installieren können

Inhalte dieses Kapitels

Worum es heute geht

Verschlüsselung von Daten gehört seit der Antike zu den Grundmethoden der Kriegsführung. Eine sehr alte (und sehr einfache) Verschlüsselungsmethode ist beispielsweise die Cäsar-Chiffre, bei der einfach jeder Buchstabe durch einen Buchstaben etwas weiter hinten im Alphabet ersetzt wird. Ob sich Julius Cäsar dieses Verfahren allerdings selbst ausgedacht hat? Berthold Brechts lesender Arbeiter würde sich sicherlich fragen: »Cäsar eroberte ganz Gallien. Hatte er nicht wenigstens einen Kryptographen dabei?«

Im zweiten Weltkrieg wurde erstmals im großen Stil Kryptographie *technisiert*, insbesondere in Form der *Enigma*. Es entwickelte sich ein Katz-und-Maus-Spiel, bei dem die deutschen Truppen ihre Enigma kryptographisch immer sicherer machten, während die Alliierten ihre *Kryptoanalyse* gleichzeitig immer weiter verbesserten. Die meisten mit Hilfe der Enigma verschlüsselten Funksprüche der deutschen Truppen konnten von den Alliierten mitgelesen werden. Zur Bedeutung des Brechens des Enigma-Codes schreibt Wikipedia: »Die Kompromittierung der Enigma wird als ein strategischer Vorteil angesehen, der den Alliierten den Gewinn des Krieges erheblich erleichtert hat. Es gibt sogar Historiker, die diese Tatsache für kriegsentscheidend halten, denn die Entzifferungen waren nicht nur auf militärisch-taktischer Ebene (Heer, Luftwaffe und Marine) eine große Hilfe, sondern sie erlaubten aufgrund der nahezu vollständigen Durchdringung des deutschen Nachrichtenverkehrs auf allen Ebenen (Polizei, Geheimdienste, diplomatische Dienste, SD, SS, Reichspost und Reichsbahn) auch einen genauen Einblick in die strategischen und wirtschaftlichen Planungen der deutschen Führung. Speziell schätzten die Alliierten die Authentizität der aus Enigma-Funksprüchen gewonnenen Informationen, die aus anderen Quellen, wie Aufklärung, Spionage oder Verrat, nicht immer gegeben war. So konnten die Briten ihre zu Beginn des Krieges noch begrenzten Ressourcen optimal koordinieren und gezielt gegen die deutschen Schwächen einsetzen, und später, zusammen mit ihren amerikanischen Verbündeten, die Überlegenheit noch besser ausspielen.«

Einer der Hauptgründe, weshalb der Enigma-Code gebrochen werden konnte, war, dass das folgende schon 1883 von Kerckhoffs formulierte Prinzip nicht eingehalten wurde:

> Die Sicherheit eines Kryptosystems darf nicht von der Geheimhaltung des Algorithmus abhängen, sie gründet sich nur auf die Geheimhaltung des Schlüssels.

Die heute verwendeten Verschlüsselungsalgorithmen haben diese Schwachstelle nicht, man kann die Algorithmen in jedem Lehrbuch nachlesen. Trotzdem sei darauf hingewiesen, dass ihre Sicherheit nur *vermutet* wird – es gibt keinen Beweis, dass irgend eines der heute eingesetzten Standardverfahren wirklich sicher ist.

43.1 Ziele von IT-Sicherheit

Worum geht es bei IT-Sicherheit? 43-4

Bei *IT-Sicherheit* geht es um folgende Anliegen:

1. Schutz vor und die Aufrechterhaltung des Betriebs bei
   - Ausfall von Teilen des Systems (Stromausfall, Absturz)
   - Angriffen auf das System (durch Hacker, korrumpierte Mitarbeiter)

   Diese *Systemsicherheit* wird uns im nächsten Kapitel interessieren.
2. Schutz von Daten und Kommunikation vor
   - Spionage
   - Fälschung

   Diese *Daten- und Kommunikationssicherheit* wird uns in diesem Kapitel interessieren.

43.2 Verschlüsselung

43.2.1 Ziele

Ziele der Verschlüsselung von Daten und Kommunikation 43-5

Vertraulichkeit Es muss sichergestellt werden, dass Daten und Kommunikation nur von »den Guten« gelesen werden können.
(Meine Daten gehen niemand etwas an.)

Integrität Es muss sichergestellt werden, dass Daten und Kommunikation nicht verfälscht werden können.
(Aus 1000 Euro dürfen nicht 10000 Euro werden.)

Authentizität Es muss sichergestellt werden, dass Daten und Kommunikation wirklich von den behaupteten Personen stammen.
(Die Email mit Alices Absenderadresse wurde tatsächlich von Alice versandt; der Online-Banking-Server muss wirklich der Server meiner Bank sein.)

✎ Zur Übung

Beurteilen Sie Postkarten, versiegelte Briefe und die Eröffnung eines Bankkontos bei einer Online-Bank in Bezug auf die drei Kriterien.

43.2.2 Symmetrische Verschlüsselung

43-6

Symmetrisch Verschlüsselung mittels eines Schlüssels

- Zum Schutz vor unbefugtem Lesen kann man Daten und Kommunikation *verschlüsseln*. Dazu benutzt man ein *Verschlüsselungsverfahren* sowie einen *geheimen Schlüssel*.
- Beim *Verschlüsseln* (encryption) wird ein *Klartext* m (message) zusammen mit dem *Schlüssel* k (key) in ein *Chiffretext* $c = e(m,k)$ verwandelt.
  Dies entspricht dem »Abschließen« mit dem Schlüssel.
- Beim *Entschlüsseln* (decryption) wird der Chiffretext zusammen mit dem Schlüssel in den Klartext zurückverwandelt, d.h. $m = d(c,k)$.
  Dies entspricht dem »Aufschließen« mit dem Schlüssel.
- Da man zum »Auf- und Zuschließen« denselben Schlüssel verwendet, spricht man von *symmetrischen Verfahren*.

43-7

Verfahren I: Die Cäsar-Chiffre

Bereits in der Antike nutzte Julius Cäsar Verschlüsselungen, wenn er Befehle an seine Feldherren übermittelte. Das als *Cäsar-Chiffre* bekannte Verfahren funktioniert wie folgt: Jeder Buchstabe der Nachricht wird durch den Buchstaben ersetzt, der *drei Buchstaben später im Alphabet* kommt. Allgemeiner kann man statt »drei Buchstaben später« auch »k Buchstaben später« benutzen. Mathematisch ist also e die Funktion, die eine Nachricht und eine Zahl k nimmt und jeden Buchstaben der Nachricht um k viele Stellen im Alphabet vorwärts schiebt. Entsprechend schiebt d jeden Buchstaben um k viele Stellen zurück.

✎ Zur Diskussion

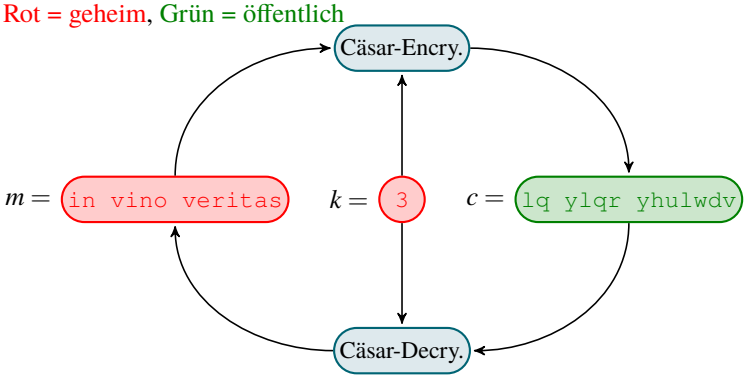
Betätigen Sie sich als *Kryptoanalytiker*! Wie kann man – ohne Kenntnis des Schlüssels k – eine mit einer Cäsar-Chiffre kodierte Nachricht entschlüsseln?

43-8

Beispiel einer Cäsar-Verschlüsselung

Rot = geheim, Grün = öffentlich

Public domain



43-9

Verfahren II: Der One-Time-Pad

Da die Cäsar-Chiffre offenbar nicht sonderlich sicher ist, benötigen wir ein besseres Verfahren:

One-Time-Pad-Algorithmus (Ver- und Entschlüsselung)

Eingaben seien die Nachricht m und der Schlüssel k *gleicher Länge*

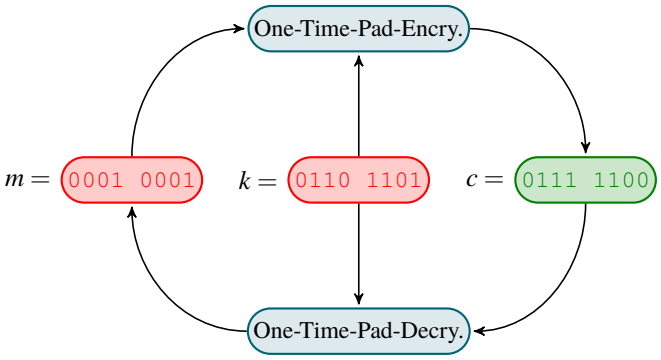
1. Schreibe Nachricht und Schlüssel als Bitstrings auf (wie im ersten Kapitel).
2. Bilde nun das bitweise XOR von Nachricht und Schlüssel. Dies bedeutet: Flippe das i-te Bit der Nachricht, wenn das i-te Bit des Schlüssels eine 1 ist.

Wie der Name schon sagt, kann man das Verfahren leider nur einmal pro Schlüssel (sicher) verwenden. Außerdem sind die Schlüssel schrecklich lang.

Beispiel einer One-Time-Pad-Verschlüsselung

43-10

Rot = geheim, Grün = öffentlich



State-of-the-Art in Sachen symmetrische Verschlüsselung.

43-11

Moderne symmetrische Verschlüsselungverfahren kann man sich als eine *sehr clevere Mischung* aus Cäsar-Verfahren und One-Time-Pad vorstellen. Lange Zeit war das amerikanische DES (data encryption standard) das wichtigste Verfahren. Wegen Problemen wie Ausfuhrverbot, zu kurze Schlüssellänge und Patenten wurde es vor gut 10 Jahren durch ein moderneres und sicheres Verfahren ersetzt. Der neue internationale Standard nennt sich AES (advanced encryption standard).

Sicherheit von Verschlüsselungsverfahren.

43-12

Wann ist ein Verfahren »sicher«?

Informationstheoretisch sicher heißt ein Verfahren, wenn man Chiffretexte ohne Kenntnis des Schlüssels nicht entschlüsseln kann. Leider kann man zeigen, dass nur One-Time-Pad-Verfahren in diesem Sinn sicher sind. *Komplexitätstheoretisch sicher* heißt ein Verfahren, wenn es für die Entschlüsselung kein wesentlich schnelleres Verfahren gibt, als alle Schlüssel durchzuprobieren. Bei Schlüssellängen ab 500 Bits ist solch eine Sicherheitsstufe dann *in diesem Universum nicht zu brechen*.

43.2.3 Asymmetrische Verschlüsselung

Eine ungewöhnliche Idee.

43-13

Symmetrische Verschlüsselungsverfahren haben den *Nachteil*, dass Kommunikationspartner erstmal einen *Schlüssel sicher austauschen müssen*. Rivest, Shamir und Adleman haben 1977 ein Verfahren vorgeschlagen, *bei dem man keinen Schlüssel auszutauschen braucht* – das RSA-Verfahren. (Es gab vorher auch schon ein schlechteres, geheimgehaltenes Verfahren.) Heute ist dies ein Spezialfall der *Klasse der asymmetrischen Verschlüsselungsverfahren*.

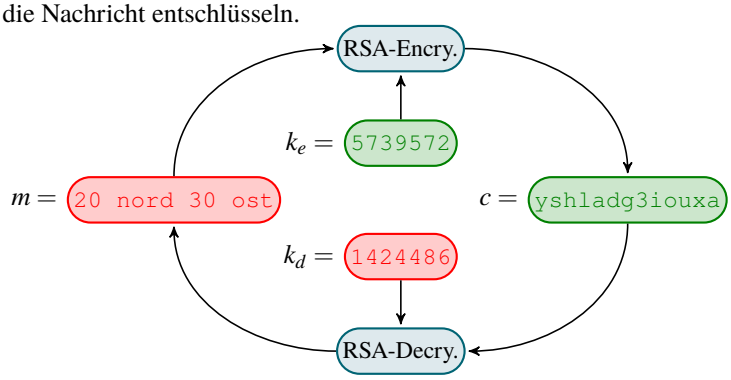
Idee der asymmetrischen Verfahren

Man benutzt *zwei Schlüssel*. Wenn man eine Nachricht mit einem ersten Schlüssel »abschließt«, kann man sie *nur mit dem zweiten Schlüssel »aufschließen«*. Kennt man einen der Schlüssel, so kann man daraus nicht mit vertretbarem Aufwand den anderen Schlüssel generieren.

43-14

Ablauf einer asymmetrischen Verschlüsselung.

Die Royal Airforce möchte den Aufenthaltsort von Prinz Harry an den Buckingham Palace schicken. Dazu braucht sie nur den *öffentlichen Schlüssel 5739572 des Buckingham Palace* zu kennen. Nur im Palast kennt man den *privaten Schlüssel 1424486 des Palastes* und kann die Nachricht entschlüsseln.



43.2.4 Das El-Gamal-Verfahren

Skript-Referenz

Im Folgenden soll das *El-Gamal-Verfahren* vorgestellt werden. Allerdings habe ich, der Darstellung von Bongartz und Unger folgend, zum besseren Verständnis die mathematischen Operationen zunächst durch (zu) radikal vereinfachte ersetzt. Am Ende wird dann noch angedeutet werden, wie es in Wirklichkeit geht.

Grundideen

- Zum *Verschlüsseln* benutzen wir eine *einfache mathematische Operation*. Konkret wird dies *im Wesentlichen eine Multiplikation sein*.
- Zum *Entschlüsseln* muss man diese Operation also rückgängig machen, man muss also im Wesentlichen dividieren, was aber prinzipiell *schwerer* ist als Multiplizieren.
- Genauer nehmen wir an, dass Addition, Subtraktion und Multiplikation *leicht* sind, Division hingegen *sehr schwer*.
- Der *private Schlüssel* hilft uns aber, *statt der Division* die Nachricht doch schnell zu entschlüsseln.

Da das El-Gamal-Verfahren mathematische Operationen benutzt, müssen sinnigerweise sowohl Nachrichten wie Schlüssel auch mathematische Objekte sein, konkret Zahlen:

- Die Nachricht ist eine Zahl; beispielsweise $m = 5$.
- Der private Schlüssel ist eine Zahl; beispielsweise $k_d = 13$.
- Der öffentliche Schlüssel ist ein Paar, bestehend aus einer Zufallszahl z und dem Produkt dieser Zahl mit dem privaten Schlüssel; beispielsweise $k_e = (z, k_d \cdot z) = (11, 143)$.
- *Könnte man dividieren*, so kann man offenbar den privaten Schlüssel k_d leicht aus dem öffentlichen Paar k_e errechnen. Wir nehmen aber in der radikalen Vereinfachung an, dass man nicht effizient dividieren kann.

Eine *Verschlüsselung* funktioniert nun wie folgt:

- Zur Verschlüsselung wählt man *eine zufällige Zahl aus*, beispielsweise $s = 9$.
- Die Verschlüsselung der Nachricht $m = 5$ mit dem Schlüssel $k_e = (z, k_d \cdot z) = (11, 143)$ besteht aus zwei Teilen:

$$t_1 = m + s \cdot k_d \cdot z = 5 + 9 \cdot 143 = 1292$$
$$t_2 = s \cdot z = 9 \cdot 11 = 99$$

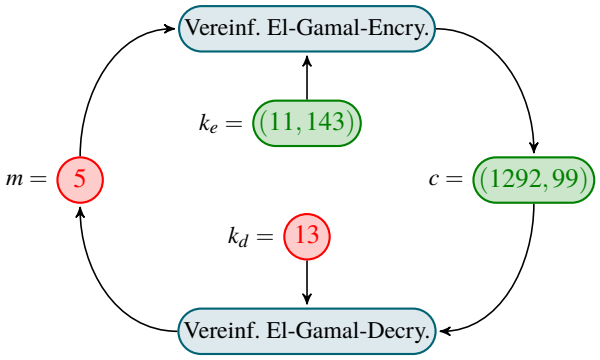
  Also ist die Verschlüsselung von 5 das Chiffrat $(1292, 99)$.

Umgekehrt läuft nun das Entschlüsseln wie folgt ab:

- Es gilt $t_1 = m + t_2 \cdot k_d$ und somit $m = t_1 - t_2 \cdot k_d$.
- Um also aus dem Chiffrat $(1292, 99)$ die Nachricht 5 zu rekonstruieren, kann man beim Entschlüsseln *unter Kenntnis von* k_d wie folgt rechnen:

$$m = t_1 - t_2 \cdot k_d = 1292 - 99 \cdot 13 = 5.$$

- Es wird nur addiert, subtrahiert und multipliziert.



Soweit zu dem vereinfachten El-Gamal-Verfahren. In Wirklichkeit müssen offenbar andere mathematische Operationen angewandt werden, da ja die Annahme, man könne nicht effizient dividieren, recht unrealistisch ist.:

- Im *echten El-Gamal-Verfahren* werden die Grundoperationen wie folgt ersetzt (p sei eine große Primzahl – es gibt Verfahren, solche Primzahlen schnell zu bestimmen):
  1. Addition wird ersetzt durch »Modulo-Multiplikation«: $a + b$ wird zu $(a \cdot b) \bmod p$.
  2. Subtraktion wird ersetzt durch »Modulo-Division«: $a - b$ wird zu $(ab^{p-2}) \bmod p$.
  3. Multiplikation wird ersetzt durch »Modulo-Exponentenbildung«: $a \cdot b$ wird zu $a^b \bmod p$.
  4. Division wird ersetzt durch den »Modulo-Logarithmus«.
- Man überlegt sich nun recht leicht, dass zunächst die Modulo-Multiplikation recht flott berechnet werden kann, selbst wenn die Zahlen a, b und p alle einige Hundert Stellen haben. Dazu benutzt man einfach das schriftliche Multiplizieren und Dividieren, wie man es aus der Schule kennt.
- Für die Modulo-Division und -Exponentenbildung ist zunächst nicht klar, wie diese schnell funktionieren sollen: Es ist in der Tat nicht möglich, in vertretbarer Zeit a^b zu berechnen, wenn a und b jeweils hunderte Ziffern haben. Der Grund ist einfach, dass in diesem Fall das Ergebnis länger ist, als es Atome im Universum gibt.
  Nun soll aber auch nicht a^b berechnet werden, sondern $a^b \bmod p$. Der Trick ist grob folgender: Will man beispielsweise $a^{256} \bmod p$ berechnen, so kann man dies schnell machen, indem man nacheinander a bin p, a^2 bin p, a^4 bin p, a^8 bin p, a^{16} bin p, a^{32} bin p, a^{64} bin p, a^{128} bin p und a^{256} bin p berechnet. Jede Zahl in der Reihe ergibt sich, indem man die vorherige Zahl quadriert und modulo p rechnet. Will man also a^b bin p berechnen für ein b mit tausend Stellen, so muss man diese Operation etwa eintausend Mal ausführen – was noch eine vertretbare Laufzeit ergibt.

Entscheidend für die Sicherheit des El-Gamal-Verfahrens ist nun, dass man für die Berechnung des Modulo-Logarithmus schlicht kein schnelles Verfahren kennt.

43.3 Sichere E-Mail

43.3.1 Vertraulichkeit: Digitale Briefumschläge

Eine Nachricht soll vertraulich sein.

43-15

Ziel: Vertraulichkeit

- Eine Nachricht soll einer Person zugestellt werden.
- Nur diese Person soll die Nachricht lesen können.

Dies nennt man auch einen *digitalen Briefumschlag*.

Methode

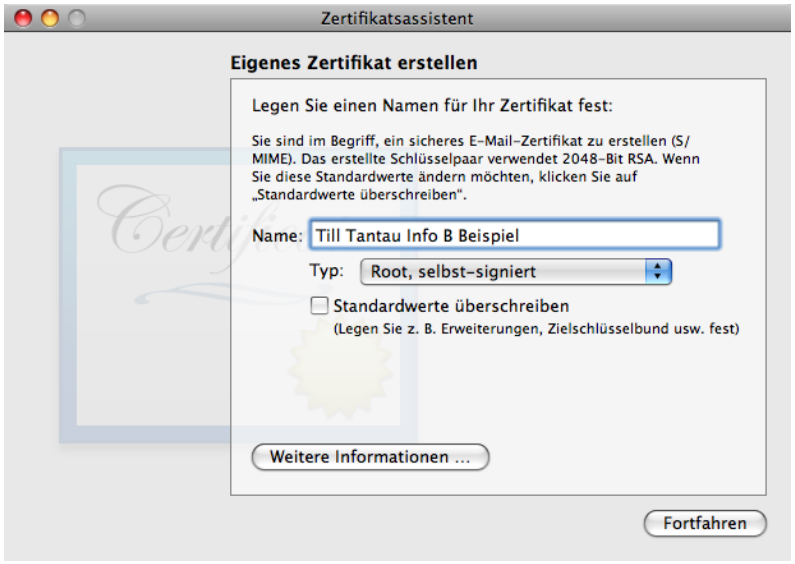
Die Person erzeugt ein RSA-Paar (k_e, k_d). Der Schlüssel k_e wird »allgemein bekanntgegeben« und heißt nun *öffentlicher Schlüssel*. Nachrichten werden mit dem öffentlichen Schlüssel der Person verschlüsselt. Effekt: Nur diese Person kann die Nachricht (mit dem nur ihr bekannten privaten Schlüssel k_d) entschlüsseln.

43-16

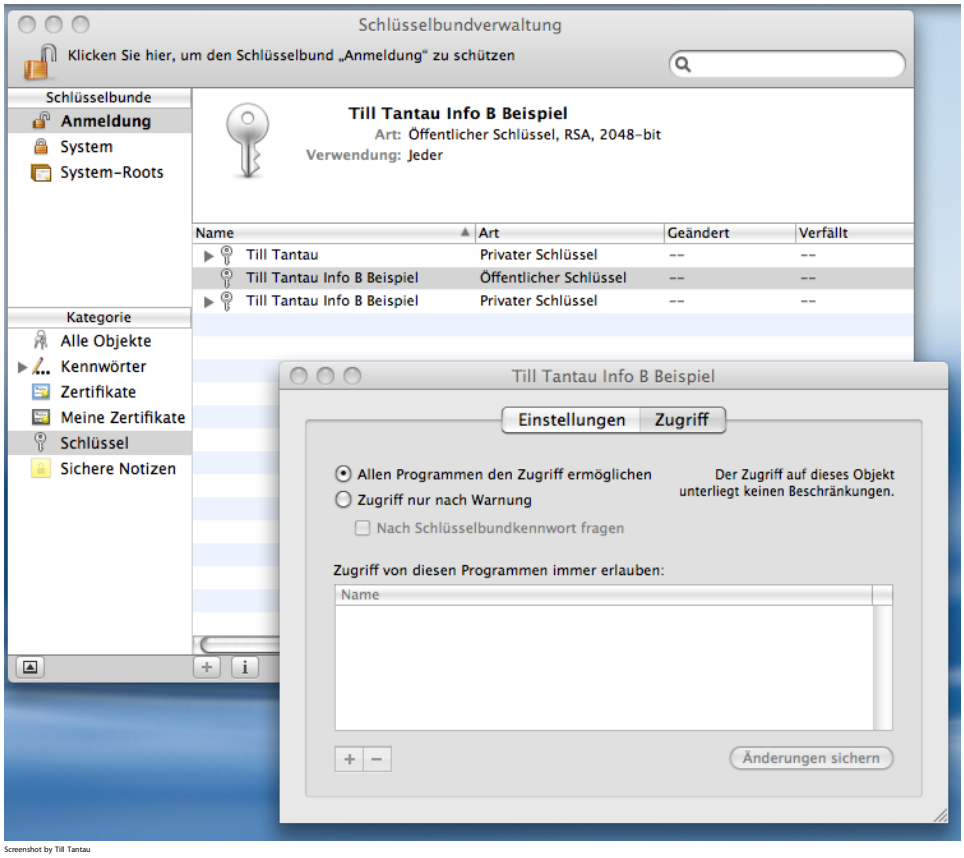
Praktische Umsetzung: Erzeugen des Schlüsselpaares.

Zur Erzeugung eines Schlüsselpaares benutzt man ein *geeignetes Programm* wie beispielsweise `openssl`. Bei MacOS kann man auch komfortabel den *Schlüsselbundverwaltung* nutzen.

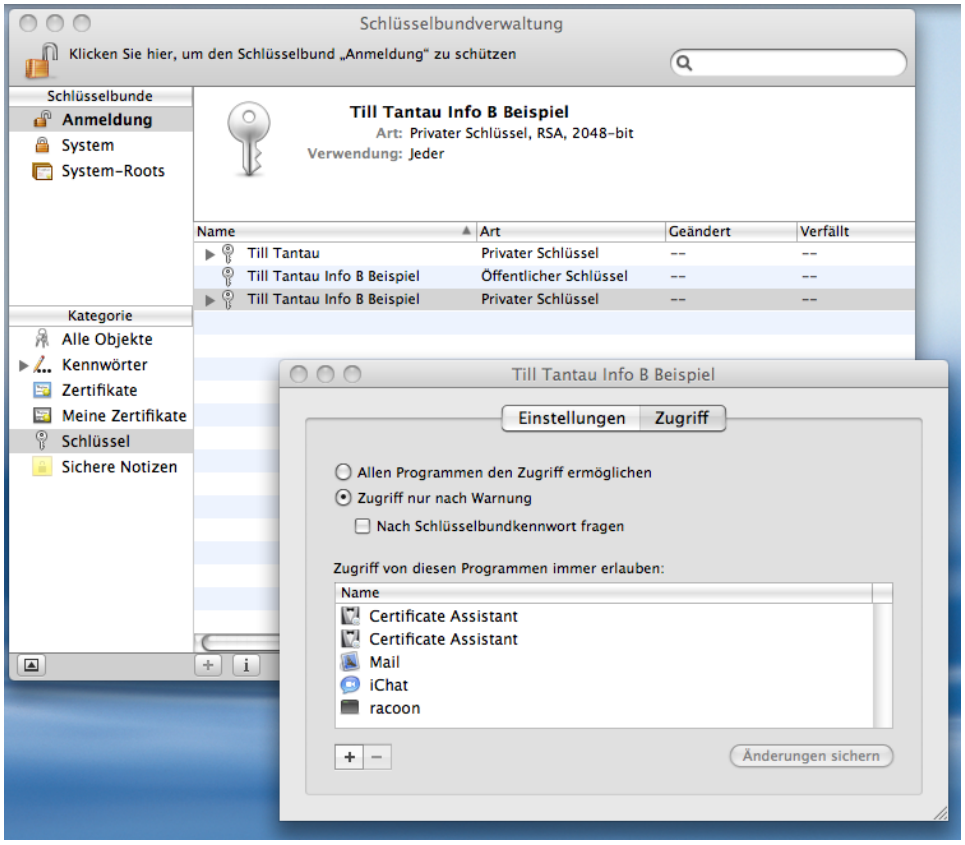
(Das Verfahren wird gleich noch etwas komplizierter werden, aber kümmern wir uns erstmal um den einfachen Fall.)



Screenshot by Till Tantau



Screenshot by Till Tantau



Screenshot by Till Tantau

Praktische Umsetzung: E-Mail an Johannes.

43-17

Schritt 1: Wir brauchen Johannes öffentlichen Schlüssel.

Um eine Mail zu verschlüsseln, muss das E-Mail-Programm den *öffentlichen Schlüssel der Person kennen, der man eine E-Mail schreiben möchte*. Diesen öffentlichen Schlüssel kann einem die Person beispielsweise vorher geschickt haben. E-Mail-Programme oder die Schlüsselverwaltung erlaubt es, solche öffentliche Schlüssel zu *importieren* (ein geeigneter Menüpunkt). So sieht der öffentliche Schlüssel von Johannes beispielsweise wie folgt aus:

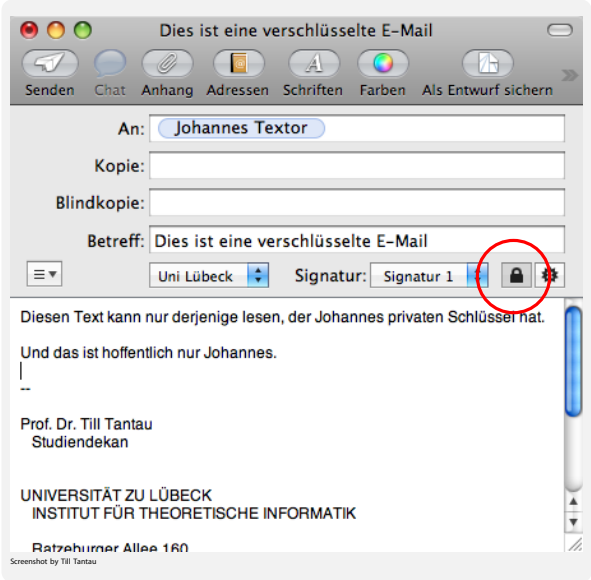
```
-----BEGIN CERTIFICATE-----
MIIFSTCCBDGgAwIBAgIECz1arTANBgkqhkiG9w0BAQUFADCBqzELMAkGA1UEBhMC
REUxIDAeBgNVBAoTF1VuaXZlcnNpdGFldCB6dSBMdWViZWNrMS4wLAYDVQQLEyVJ
bnN0aXR1dCBmdWVyIE1lZGl6aW5pc2NoZSBJbmZvcm1hdGlrMScwJQYDVQQDEx5D
QSBkZXIgVW5pdmVyc2l0YWV0IHp1IEx1ZWJlY2sxITAfBgkqhkiG9w0BCQEWEnBr
aUB1bmktbHVlYmVjay5kZTAeFw0wNzEwMjMxMTM4MDdaFw0xMDEwMjIxMTM4MDda
... 20 ausgelassene Zeilen ...
Q8RGGPHGKcAocX3kGB3VTWZptDCACiJ9E5Q5pD4mWMRPYAgYnjJwWv4KzF5Moboe
29IKgSvifR5ttcdqCFn5gfwVYrHWOLxxSZkbUpqGIsAhoGeWd65Z9u4FArI87UIw
3KCso+ohahGUeVqZQk6ZXU6zJX/hw6K4y2l1QfiBwVQuNjvudADM3Q6RSedECK4m
MsUhRqoUqvXegCZ7mA==
-----END CERTIFICATE-----
```

Er kann auf `https://pki.pca.dfn.de/uzl-ca/cgi-bin/pub/pki` heruntergeladen werden. Alternativ kann Johannes den Schlüssel zunächst von seinem Mail-Programm oder von der Schlüsselverwaltung *exportieren* und dann verschicken (zur Not per Post).

43-18

Praktische Umsetzung: E-Mail an Johannes.
Schritt 2: Schreiben und abschicken der E-Mail.

Hat man dem eigenen System erstmal Johannes öffentlichen Schlüssel beigebracht, so kann man ihm eine verschlüsselte E-Mail schreiben. Bei modernen E-Mail-Programmen muss man dazu einfach auf einen Verschlüsselungsknopf drücken.



An Johannes wird dann ein mit dem *öffentlichen Schlüssel von Johannes verschlüsselter* Text geschickt. Um den Text zu entschlüsseln, braucht man den privaten Schlüssel von Johannes. Folglich kann niemand außer Johannes – *nichtmal der Autor der Mail* – den Text wieder entschlüsseln.

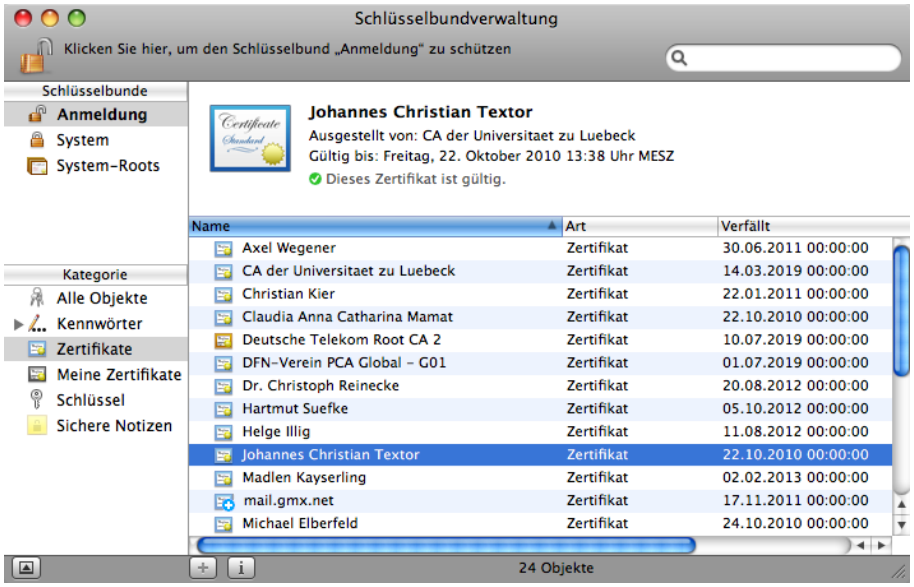
```
Message-Id: <AF944753-2276-487B-9BB1-995DA5D2058F@tcs.uni-luebeck.de>
From: Till Tantau <tantau@tcs.uni-luebeck.de>
To: Johannes Textor <textor@tcs.uni-luebeck.de>
Content-Type: application/pkcs7-mime;
        name=smime.p7m;
        smime-type=enveloped-data
Content-Transfer-Encoding: base64
X-Smtp-Server: theogate.tcs.uni-luebeck.de:tantau
Content-Disposition: attachment;
        filename=smime.p7m
Mime-Version: 1.0 (Apple Message framework v936)
Subject: =?ISO-8859-1?Q?Dies_ist_eine_verschl=FCsselte_E-Mail?=
Date: Fri, 11 Jun 2010 16:05:31 +0200

MIAGCSqGSIb3DQEHA6CAMIACAQAxggOiMIIBzQIBADCBtDCBqzELMAkGA1UEBhMCREUxIDAeBgNV
BAoTF1VuaXZlcnNpdGFldCB6dSBMdWVlZWNrMS4wLAYDVQQLEyVJbnN0aXR1dCBmdWVyIE1lZGl6
aW5pc2NoZSBJbmZvcm1hdGlrMScwJQYDVQQDEx5DQSBkZXIgVW5pdmVyc2l0YWV0IHplIEx1ZWJl
Y2sxITAfBgkqhkiG9w0BCQEWEnBraUB1bmktbHVlYmVjay5kZQIECz1arTANBgkqhkiG9w0BAQEF
AASCAQDbXJGvqKCu+Qd1xfTLSJKW6I8T2Eds7ks1qtvBVoD6RFzcHF7XOTllzcOmxgC2GS8frzNr
pVIC5VBssy/BEnm3BCpD8sw8HCxE0pcIm96/p5oHp9Qyk0FYs97JF3GLHVXPAs8AoHEMoVBXAuku
... 66 ausgelassene Zeilen ...
FbiGph3to8Q/xaLEGHplQNN245YGcAY3xPAKibQfs3725ctGA9bBUXliZnejMXR15DMWcd8VozAA
3MAQYwRIfJprLx7pYpUMEbSUN1Zdqi4WhTYS+6MvVEbj5ItZ68PWDx2OFsqm5enql58XViNLuS58
aTzsJ6uw86lgEgqbbiD8wpMExZxpBCBxX07+jmfobYe16/gFjInEB7XOV+VC95GEryi87pu3GAQI
qJB1HZWcdHQAAAAAAAAAAAA
```

43-19

Zusammenfassung: Wer bekommt welchen Schlüssel?

- Generell gilt: Ihr *privater* Schlüssel ist nur Ihnen bekannt (auch nicht Ihrem Lover!) und möglichst gut auf Ihrem Computer geschützt.
  - Am sichersten ist es, wenn der Schlüssel auf einer separaten Karte liegt, die ihrerseits durch eine PIN geschützt ist – dies ist mit dem neuen Personalausweis möglich.
  - In der Praxis liegt Ihr privater Schlüssel aber auf Ihrer Festplatte.
  - Dort ist er selbst wieder durch ein Master-Passwort symmetrisch verschlüsselt und damit selbst vor Diebstahl des Computers oder vor bösartigen Hackern geschützt.
- Generell gilt: *Öffentliche* Schlüssel sind möglichst breit gestreut und überall verfügbar. Sie sollten auf Ihrem System die öffentlichen Schlüssel aller Ihnen bekannter Personen haben.



Screenshot by Till Tantau

43.3.2 Authentizität: Digitale Unterschriften

Eine Nachricht soll digital unterschrieben werden. 43-20

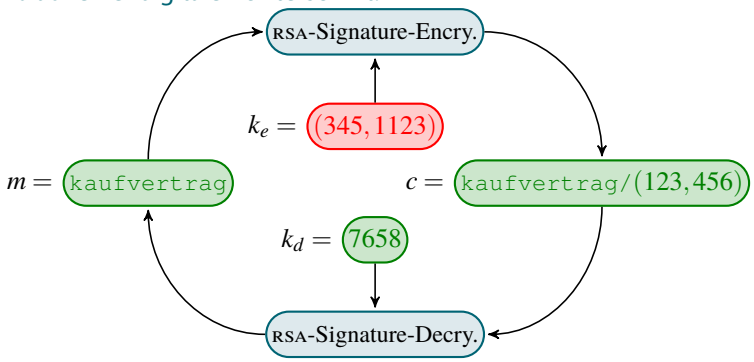
Ziel: Authentizität

Es soll garantiert werden, dass eine Nachricht von einer bestimmten Person stammt. Dies nennt man auch eine *digitale Unterschrift*.

Methode

Man benutzt dieselben Schlüssel wie beim Verschlüsseln von Nachrichten, *nur umgekehrt*: Die zu unterschreibende Nachricht wird mit dem *privaten Schlüssel* k_e verschlüsselt und dieser Text an die Originalnachricht angehängt. Überprüfung: Man entschlüsselt den verschlüsselten Teil mit dem öffentlichen Schlüssel k_d und vergleicht ihn mit dem behaupteten Text. Effekt: Nur die Person, die k_e kennt, kann die unterschriebene Nachricht erzeugen.

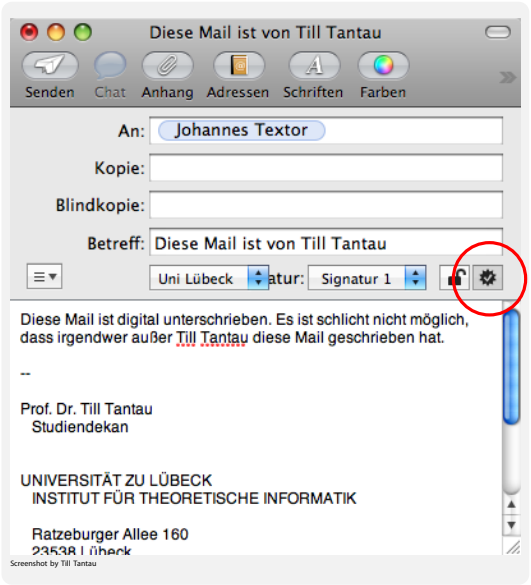
Ablauf einer digitalen Unterschrift. 43-21



43-22

Praktische Umsetzung: Unterschriebene E-Mail an Johannes.

Um eine E-Mail zu unterschreiben, braucht man lediglich den eigenen privaten Schlüssel – und den haben wir ja schon erzeugt. Damit jemand die E-Mail überprüfen kann, braucht er den öffentlichen Schlüssel – diese sind ja aber frei zugänglich.



Achtung: Mit »Signatur« wird auch manchmal der Standard-Text am Ende einer Mail bezeichnet. Dieser ist *keine* digitale Unterschrift.

```
Message-Id: <A15605B5-9389-442D-BD9E-FC65D76A18DD@tcs.uni-luebeck.de>
From: Till Tantau <tantau@tcs.uni-luebeck.de>
To: Johannes Textor <textor@tcs.uni-luebeck.de>
Content-Type: multipart/signed;
        boundary=Apple-Mail-11--309560619;
        micalg=sha1;
        protocol="application/pkcs7-signature"
X-Smtp-Server: theogate.tcs.uni-luebeck.de:tantau
Mime-Version: 1.0 (Apple Message framework v936)
Subject: Diese Mail ist von Till Tantau
Date: Mon, 14 Jun 2010 08:44:26 +0200

... 13 ausgelassene Zeilen ...
Diese Mail ist digital unterschrieben. Es ist schlicht nicht m=F6glich, =20=

dass irgendwer au=DFer Till Tantau diese Mail geschrieben hat.
... 62 ausgelassene Zeilen ...

--Apple-Mail-11--309560619
Content-Disposition: attachment;
        filename=smime.p7s
Content-Type: application/pkcs7-signature;
        name=smime.p7s
Content-Transfer-Encoding: base64

MIAGCSqGSIb3DQEHAqCAMIACAQExCzAJBgUrDgMCGgUAMIAGCSqGSIb3DQEHAQAAoIIOnzCCBCEw
ggMJoAMCAQICAgDHMA0GCSqGSIb3DQEBBQUAMHExCzAJBgNVBAYTAkRFMRwwGgYDVQQKExNEZXV0
c2NoZSBUZWxla29tIEFHMR8wHQYDVQQLExZULVRlbGVTZWMgVHJ1c3QgQ2VudGVyMSMwIQYDVQQD
... 79 ausgelassene Zeilen ...
DfpFZundteQqc6R/FdbTj67j23Y5h/8+qCIewT//LLWh4hvW/kEs7bi1Ibcm3yniFd4PzjkKI9gi
WuPJkqz3VrcuGYfjCHT3RGyAANNQOF6fiJQ5tipmN4dHkfoxWQ7nPJBobS13MLLd+fIvBrI78pTp
8mFDaOsObxdlyOXhm5RTBor2U/4uDGtZJddGOCNLpNjnCqEti3PWAHD/IAAAAAAAAA==

--Apple-Mail-11--309560619--
```

43.3.3 Echtheits-Zertifikate: Digitale Notare

43-23

Das Henne-Ei-Problem bei öffentlichen Schlüsseln

Damit ich Johannes eine sichere Mail schreiben kann, muss er mir seinen öffentlichen Schlüssel zukommen lassen. Nun könnte auch irgendjemand anderes mir einen Schlüssel schicken und behaupten, er sei Johannes und dies sei sein Schlüssel. Um das auszuschließen, würde ich gerne verlangen, dass die Mail von Johannes unterschrieben ist – aber dazu brauche ich ja gerade den öffentlichen Schlüssel, um den es gerade geht.

Certificate-Authorities zertifizieren die Echtheit von Schlüsseln.

43-24

Ziel: Echtheit von Schlüsseln bezeugen

A will sichergehen, dass ein vermeintlicher öffentliche Schlüssel von B tatsächlich von B stammt.

Methode

Eine *vertrauenswürdige Instanz*, »Certificate Authority (CA)« oder »Trust-Center« genannt, legt *Root-Schlüssel* (k_d, k_e) für digitale Unterschriften an. Der öffentliche Schlüssel k_d ist allgemein bekannt (er ist zum Beispiel in Ihren Browser schon fest eingebaut). Benutzer B lässt sich von der vertrauenswürdigen Instanz den folgenden Text unterschreiben: »Person B hat den öffentlichen Schlüssel 1234567.« Wenn A mit B zum ersten Mal redet, schickt B diesen unterschriebenen Text. A kann die Unterschrift der CA überprüfen (A kennt ja das Root-Zertifikat) und kann dann den öffentlichen Schlüssel von B benutzen.

Die privaten Schlüssel der Root-CAs sind *extrem gut gesichert*. Das darf man sich in etwa wie bei Mission Impossible vorstellen – inklusive gepanzerter Räume mit Servern, die vom Netz und auch von sonst allem getrennt sind. Selbst wenn Sie mit Ihrer Privatarmee das Gebäude stürmen, die wackeren Systemadministratoren überwältigen bevor diese die Schlüssel löschen können und den Schlüssel stehlen, würde das nicht viel nützen. Schlüssel können in zentralen Verzeichnissen als ungültig erklärt werden, was einige Minuten nach Ihrem Angriff der Fall wäre. Merke: Wenn man den privaten Schlüssel einer Root-CA klaut, so darf dies niemand merken. Ich empfehle, sich beispielsweise an Herrn Hunt zu wenden.

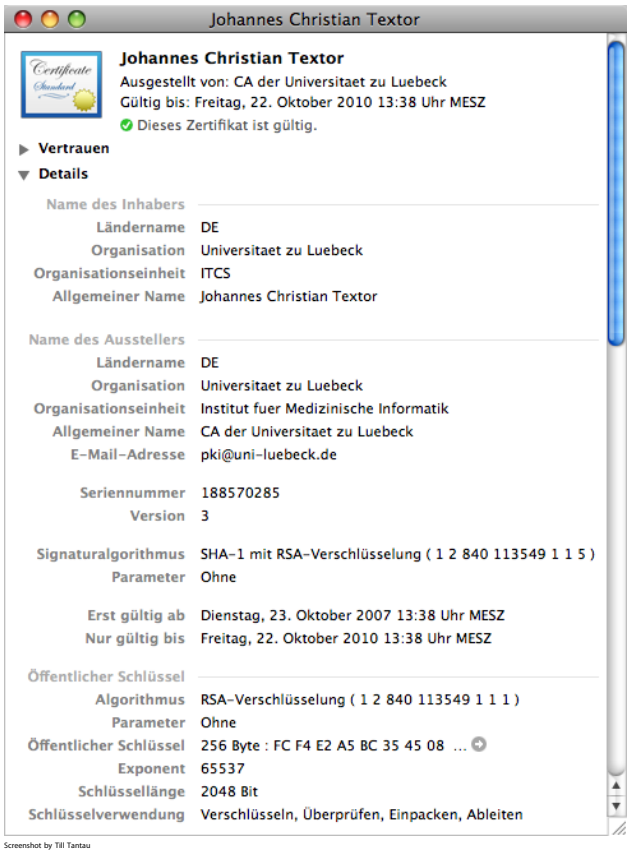
Wenn die Zeit es erlaubt, so würde hier die Schlüsselszene aus Mission Impossible gut passen, bei der Herr Cruise an Seilen schwebend Daten auf eine recht altertümliche Diskette kopiert und dabei Problem mit Schweißbildung hat.

Regie

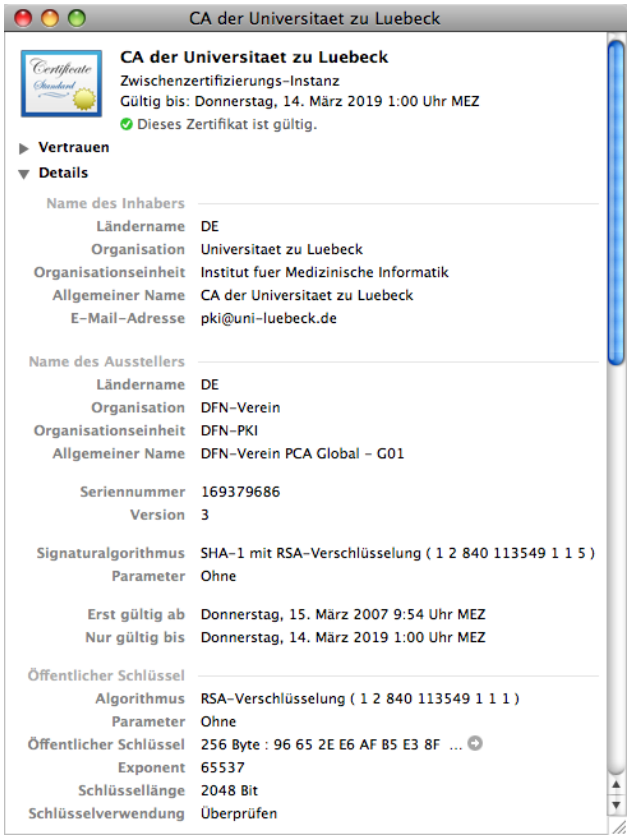
Kaskaden von Unterschriften

43-25

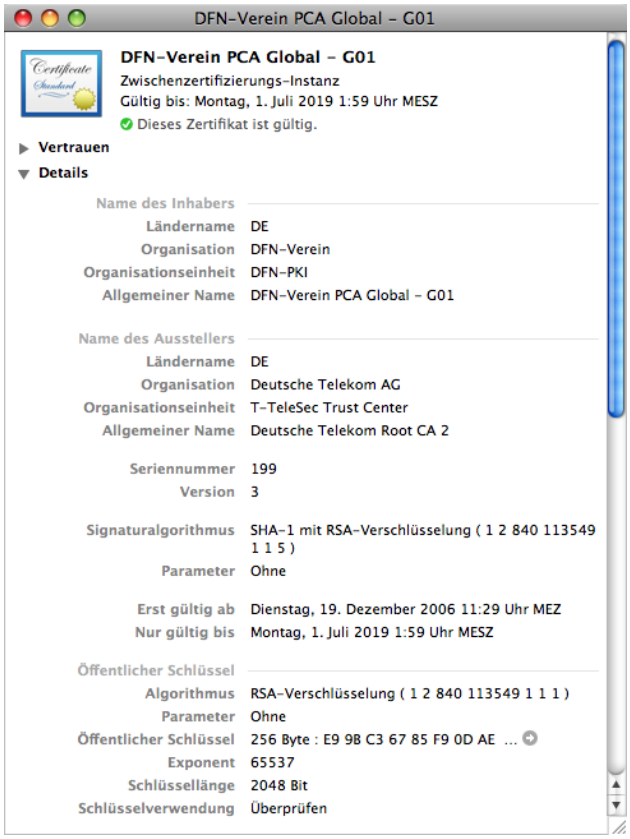
In der Praxis kann die Root-CA nicht die Schlüssel beispielsweise aller Deutschen unterschreiben. Stattdessen gibt es »Zwischen-CAs«.



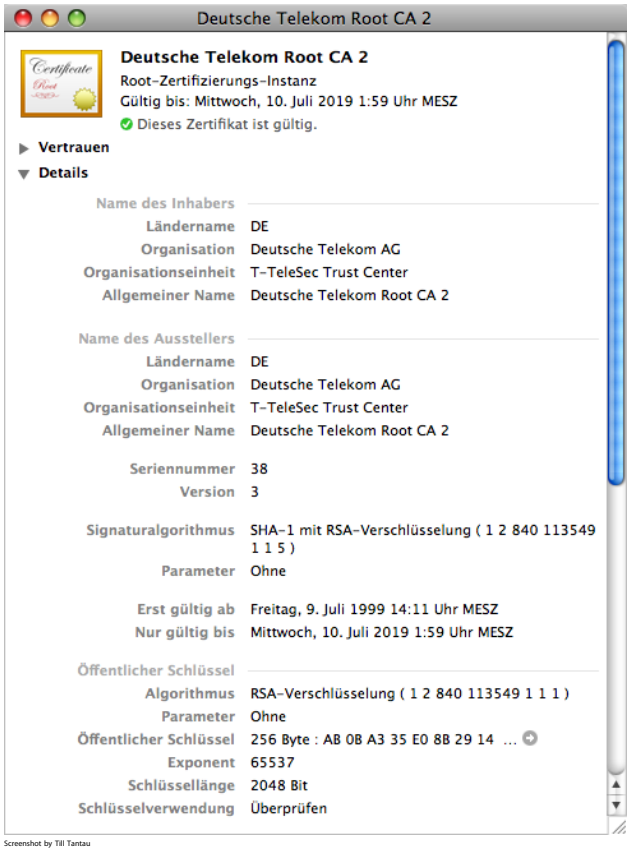
Screenshot by Till Tantau



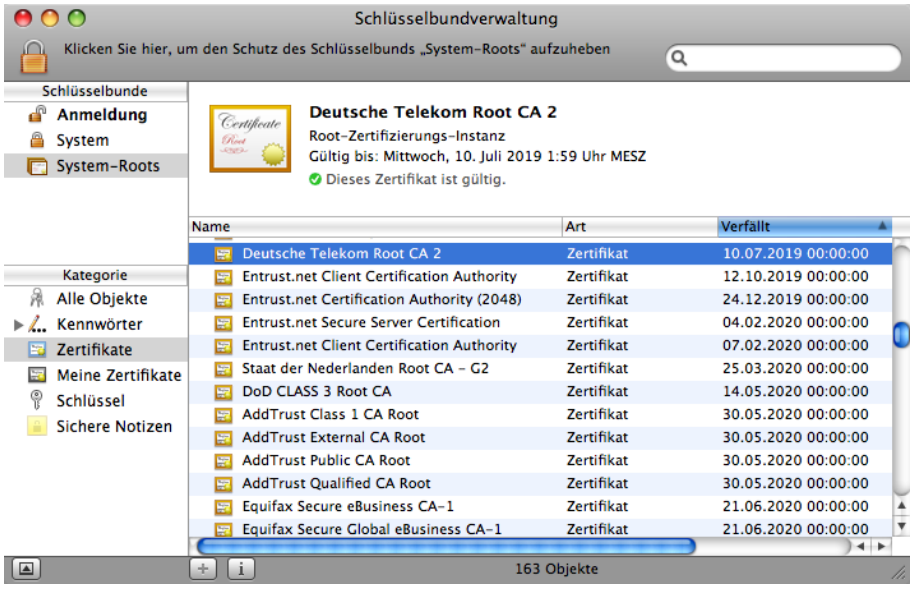
Screenshot by Till Tantau



Screenshot by Till Tantau



Screenshot by Till Tantau



Screenshot by Till Tantau

43.4 Sicheres Surfen

Sicheres Surfen funktioniert wie sichere E-Mail.

43-26

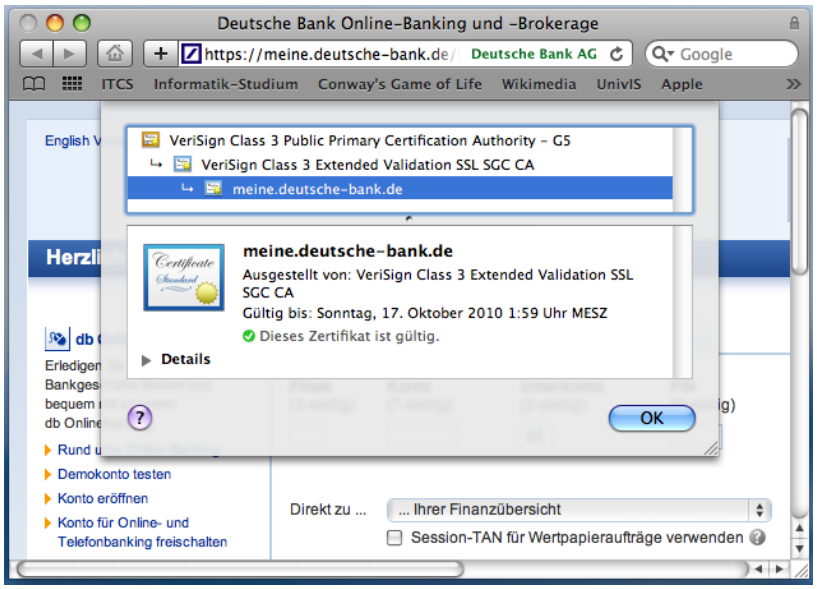
Wenn Sie mit einem Online-Shop oder Ihrer Bank kommunizieren, haben Sie *dieselben Problem wie bei E-Mail*:

- Niemand soll die Kommunikation abhören können.
- Sie müssen sicher sein können, dass Ihre Bank auch wirklich Ihre Bank ist.

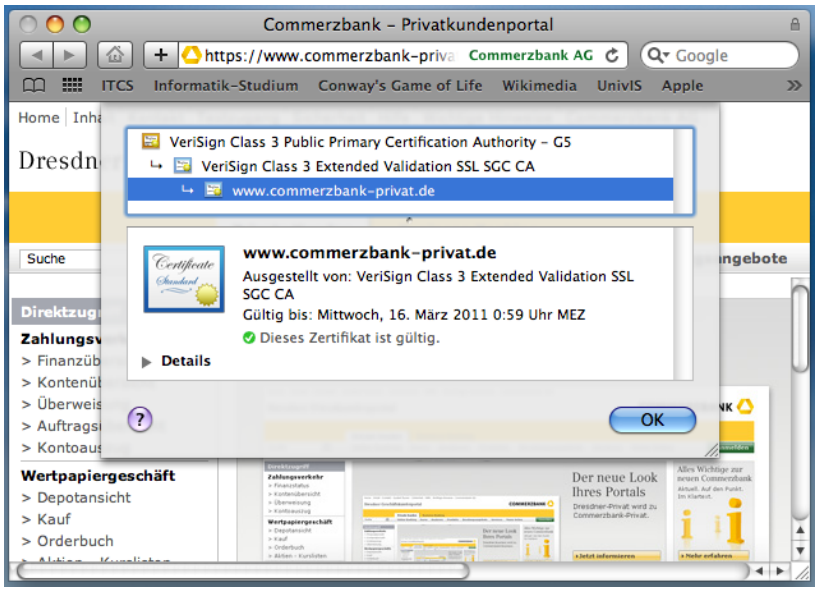
Diese Probleme werden auch genauso gelöst:

- Die Bank oder der Online-Shop hat ein Schlüsselpaar, dass nun aber *keine Person* identifiziert sondern *eine Webseite*.
- Das Schlüsselpaar ist über eine Kette von CAs unterschrieben.

Die Protokolle *https* und *ssh* bauen auf diese Art sichere Kanäle auf.



Screenshot by Till Tantau



Screenshot by Till Tantau

Zusammenfassung dieses Kapitels

43-27

Symmetrische Verschlüsselung
Eine Nachricht m wird mit einem Schlüssel k zu einem Chiffrat $c = e(m,k)$ verarbeitet. Mit *demselben* Schlüssel k lässt sich dann $m = d(c,k)$ berechnen. Das Standardverfahren heißt AES.

Asymmetrische Verschlüsselung
Eine Nachricht m wird mit einem *öffentlichen Schlüssel* k_e zu einem Chiffrat $c = e(m,k_e)$ verarbeitet. Mit einem *ganz anderen* Schlüssel k_d lässt sich dann $m = d(c,k_d)$ zurückgewinnen. Das Standardverfahren heißt RSA.

Vertraulichkeit
Die *Vertraulichkeit* von E-Mails und der Kommunikation mit Webservern (https) wird sichergestellt, indem mit dem *öffentlichen Schlüssel des Empfängers* verschlüsselt wird.

Authentizität
Die *Authentizität* einer Nachricht wird sichergestellt, indem mit dem *privaten Schlüssel des Absenders* verschlüsselt wird. Dies nennt man *digitale Unterschrift*.

▶ Zertifikate
Ein *digitales Zertifikat* ist ein von einer *Certificate Authority* unterschriebener Text, der bezeugt, dass ein öffentlicher Schlüssel zu einer bestimmten Person gehört.

Übungen zu diesem Kapitel

Übung 43.1 Sicherheit einer Briefzustellung bewerten, einfach

Wenn Ihre EC-Karte abläuft oder Sie den PIN vergessen haben, bekommen Sie eine neue PIN per normaler Briefpost zugestellt. Der Zettel mit der PIN wird in einem durchleuchtungssicheren Umschlag verschickt und ist darin nochmals in einem durchleuchtungssicheren Briefumschlag verschlossen. Um die Sicherheit zusätzlich zu erhöhen, wird auf dem Umschlag kein Absender angegeben, so dass man nicht sehen kann, dass der Brief von Ihrer Bank kommt.

Werden durch diese Vorsichtsmaßnahmen die Sicherheitsziele »Geheimhaltung«, »Integrität« und »Authentizität« erreicht?

Übung 43.2 Asymmetrische Verschlüsselung anwenden, einfach

Alice und Bob sind Geheimagenten, die regelmäßig über das Internet streng geheime Informationen austauschen. Dabei baut Alice eine Verbindung zu einem Rechner auf, der Bob gehört. Um sich zu identifizieren, übermittelt Alice am Anfang des Dialogs ein geheimes, zuvor vereinbartes Codewort.

Eva hat nun vom Treiben der Beiden Wind bekommen und einen finsteren Plan geschmiedet: Sie möchte Bob hinters Licht führen und sich selbst als Alice ausgeben. Dazu bemächtigt sie sich eines Routers im Internet, über den die Kommunikation zwischen Alice und Bob abläuft. Damit ist sie in der Lage, das geheime Codewort von Alice auszuschnüffeln und sich somit gegenüber Bob als Alice ausgeben zu können.

Da Alice und Bob bereits Verdacht geschöpft haben, ändern sie ihre Vorgehensweise wie folgt ab: Bob erzeugt ein Zertifikat und leitet den öffentlichen Schlüssel über einen sicheren Kanal an Alice weiter. Alice verschlüsselt dann ihr Passwort mit diesem öffentlichen Schlüssel, damit es sicher über das Netz übertragen wird. Beurteilen Sie, wie stark die Sicherheit der Kommunikation zwischen Alice und Bob durch diese Verschlüsselung erhöht wird!

Übung 43.3 Authentifizierungsprotokoll entwerfen, mittel

Angenommen, Alice und Bob aus Übung 43.2 sind beide im Besitz des öffentlichen Schlüssels des jeweils Anderen, und Eva kann nicht an die dazugehörigen privaten Schlüssel gelangen. Was könnten Alice und Bob dann tun, um Eva daran zu hindern, sich für Alice oder Bob auszugeben?

Übung 43.4 Replay-Attacken verhindern, mittel

Bei einer *Replay-Attacke* wird eine verschlüsselte Botschaft abgefangen und aufgezeichnet, um durch das erneute Abspielen dieser Botschaft zum Beispiel eine fremde Identität vorzutäuschen. Ein einfaches Beispiel für ein System, das potenziell für Replay-Attacken anfällig ist, sind moderne Autoschlüssel, bei denen man über Knopfdruck bereits aus einigen Metern Entfernung das Auto öffnen kann. Beim Entwurf eines solchen Systems muss darauf geachtet werden, dass ein Dieb nicht einfach das vom Schlüssel zum Auto übertragene Funksignal aufzeichnen und sich damit später Zugriff zum Auto verschaffen kann, ohne den Schlüssel zu besitzen.

Beschreiben Sie ein Kommunikationsprotokoll zwischen Schlüssel und Auto, das immun gegen Replay-Attacken ist! Gehen Sie dabei davon aus, dass sowohl der Schlüssel als auch das Auto kleine Computer enthalten, die über Funk in beide Richtungen miteinander kommunizieren können.

Übung 43.5 Sniffing, mittel

Für diese Übung sind im Arbeitsraum »Abhörstationen« aufgebaut, auf denen das Programm *wireshark* läuft. Mit diesem Programm kann man die Pakete, die im lokalen Netzwerk unterwegs sind, abfangen und damit sensible Informationen erschnüffeln. Sie können auch eigene Laptops mitbringen und abhören lassen, achten Sie aber darauf, keine wirklich sensiblen Daten preiszugeben.

Diskutieren Sie zu Beginn, wie realistisch der Versuchsaufbau Ihnen erscheint.

Teilen Sie sich dann in Gruppen auf. Während eine Gruppe am abgehörten Rechner im Internet surft (im Folgenden auch »Opfer« genannt), ist es die Aufgabe der Gruppe am Abhörrechner, so viel wie möglich über die Aktivitäten der anderen Gruppe herauszufinden. Sie können zum Beispiel versuchen, folgendes herauszufinden:

- Welche Programme auf dem Computer des Opfers kommunizieren mit dem Internet?
- Welches Betriebssystem und welchen Webbrowser verwendet das Opfer?
- Welche Webseiten wurden vom Opfer besucht?

- Wie lautet das GMX-Passwort des Opfers?
- Wie lautet die ICQ-Nummer des Opfers?
- Mit wem chattet das Opfer worüber?

Insbesondere beim GMX-Szenario sollten Sie nicht Ihre echten Passwörter verwenden.

Welche Protokolle spielen hier eine Rolle und auf welchen Schichten des Schichtenmodells spielen sich diese ab? Wie kann man verhindern, dass das eigene GMX-Passwort ausgeschnüffelt wird?

44-1

Kapitel 44
System-Sicherheit
Von Viren, Würmern und SQL-Spritzen

44-2

Lernziele dieses Kapitels

1. Bedrohungsszenarien der Sicherheit von IT-Systemen kennen und einschätzen können
2. Schutzmaßnahmen für die Sicherheit von IT-Systemen kennen und ergreifen können
3. Beispiel eines Sicherheitslochs verstehen

Inhalte dieses Kapitels

Worum es heute geht

In dem Film *War Games* aus dem Jahr 1983 löst der sympathische junge Held beinahe den Dritten Weltkrieg aus, indem er per Modem zufällig Telefonnummern anruft und dabei auf die des Zentralcomputers des Pentagons trifft. Der Computer gibt sich als Spieleserver aus, die wirklich spannenden Spiele wie *Global Thermonuclear War* sind aber durch ein Passwort geschützt. Mit ein paar Nachforschungen kommt der Held an das Passwort (der Name des Sohnes des Programmierers) und kann dann mit dem Computer eine Partie wagen. Leider setzt der Computer das Spiel gleich in die Realität um und nur der Held kann die Maschine in eine Endlosschleife schicken, woraufhin sie im wahrsten Sinne des Wortes durchbrennt. (Wenn die Uni-Computer jedesmal anfangen würden zu qualmen, wenn Sie mal wieder eine Endlosschleife programmiert haben, wäre die Universität schon längst abgefackelt.)

Wie sieht die Bedrohungslage 30 Jahre später aus? Sie werden sich nicht mehr mit einem Modem in das Pentagon einwählen können; darf man zumindest hoffen. Jedoch gibt es das moderne Äquivalent zu solchen »Hintertürchen« zu Systemen immer noch. Ein besonders krasses Beispiel aus dem Jahr 2008 werden Sie in diesem Kapitel kennen lernen: Die Webseite der Justizvollzugsanstalten des US-Staates Oklahoma. Über deren Webseiten hätte man zwar nicht den Dritten Weltkrieg auslösen, aber zumindest die Justizverwaltung gehörig durcheinander bringen können.

Bei Sicherheit in der Informationstechnologie (IT-Sicherheit) geht es aber nicht nur darum, Systeme möglichst gut abzuschotten. Seit Ende der neunziger Jahre hat ein Prozess begonnen, in dessen Rahmen Menschen immer mehr Persönliches digitalisieren und häufig anderen Menschen zugänglich machen. Dies gilt für digitale soziale Räume wie Facebook ebenso wie für die personalisierte Google-Seite, durch die die Firma Google die komplette Historie Ihrer Suchanfragen protokollieren und auswerten kann. Bei IT-Sicherheit geht es auch um die Frage, wie hier der Schutz der Daten vor Verlust und unbefugtem Zugriff zu gewährleisten ist. Dieses Problem hat viele Aspekte, unter anderem rechtliche, technische, algorithmische und auch soziale.

44-4

Wiederholung: Worum geht es bei IT-Sicherheit?

Bei *IT-Sicherheit* geht es um folgende Anliegen:

1. Schutz vor und die Aufrechterhaltung des Betriebs bei
   - Ausfall von Teilen des Systems (Stromausfall, Absturz)
   - Angriffen auf das System (durch Hacker, korrumpierte Mitarbeiter)

   Diese *Systemsicherheit* wird uns in diesem Kapitel interessieren.
2. Schutz von Daten und Kommunikation vor
   - Spionage
   - Fälschung

   Diese *Daten- und Kommunikationssicherheit* war Thema des letzten Kapitels.

44-5

Wie erreicht man IT-Sicherheit?

Es gibt verschiedene Maßnahmen, um die IT-Sicherheit zu erhöhen. *Perfekte Sicherheit kann es nicht geben.*
Mögliche Maßnahmen sind:

Redundanz Daten liegen mehrfach vor.
Stichwörter: Backups.

Abschottung Es wird schwierig gemacht, in das System hineinzukommen.
Stichwörter: Passwörter und Firewalls.

Aktive Kontrolle Es wird aktiv im laufenden Betrieb überprüft, ob das Systemverhalten normal ist.
Stichwort: Virenchecker, Vier-Augen-Prinzip, Intrusion-Detection

Verschlüsselung Alle Daten werden verschlüsselt. Ohne die Schlüssel sind die Daten nichts wert.
Stichwörter: ssh (secure shell), pgp (pretty good privacy), gpg (gnu privacy guard), https (http secure)

44.1 Systemsicherheit

44.1.1 Was ist zu schützen?

44-6

Rechner müssen geschützt werden.

1. Hardware kann ausfallen, was *Datenverlust* und/oder *Produktivitätsverlust* zur Folge hat.
2. Hardware kann sich *bösartig verhalten*: Moderne Rechner sind Mehr-Prozessor- und Mehr-Benutzer-Systeme. Sie können *selbstständig* mit anderen Rechnern kommunizieren. Dadurch kann ein Rechner *von außen übernommen werden*. Dies bedeutet, dass sich jemand als ein Benutzer ausgibt und dann dem Computer (bösartige) Befehle erteilt. Ist der Angreifer geschickt, so merkt man davon nichts. Solche Rechner heißen dann *Zombies*.

44-7

Daten sind wichtiger als Hardware.

Neulich auf einem Schild in einem kleinen Laden in Berlin:

> Gestern wurde in diesen Laden eingebrochen und mehrere Computer gestohlen. Die Computer sind uns egal, aber wir benötigen die Daten auf den Rechnern! Bitte, ihr Diebe, gebt uns die Daten zurück, Diskretion und eine Belohnung garantiert!

Praktisch alle Daten, die in den Systemen einer Firma lagern, sind schützenswert. Ein paar wichtige sind: Kundenkontaktdaten, Forschungsergebnisse, Lagerbestandsdatenbanken oder Buchhaltung. Diese Daten müssen nicht nur gegen Diebstahl, sondern auch gegen Verlust durch Feuer, etc. geschützt werden.

44-8

Private Daten müssen geschützt werden.

Zu Ihrer Privatsphäre gehört nicht nur Ihre Wohnung, sondern auch Ihre Festplatte. Das Bundesverfassungsgericht hat dies kürzlich sogar in einem Grundsatzurteil zu einem Grundrecht erhoben. Sie müssen aber Ihre Grundrechte auch selbst aktiv verteidigen. Viele Leute legen ihre Daten *völlig ungeschützt* und *für alle lesbar* ab.

Motto

Zeige mir deinen Web-Browser-Cache und ich sage dir, was für ein Mensch du bist.

Von jedem lesbare Daten von Informatikstudierenden an der TU Berlin. 44-9

```
murmel:~ tantau$ ssh conde.cs.tu-berlin.de
tantau@conde.cs.tu-berlin.de's password:
Last login: Tue Apr 29 13:31:48 2008 from murmel.tcs.uni-
Sun Microsystems Inc.   SunOS 5.10      Generic January 2005
conde tantau 1 (~): cslocate sex
...
/home/all/b/believer/.kde/share/apps/krn/alt.sex.homosexual
/home/all/b/belo/man/man1/sex.1
/home/all/m/mawia/bin/BORUSSIA/XP/profile/Cookies/mawia@counter6.sextracker[1].txt
/home/all/m/mawia/bin/BORUSSIA/XP/profile/Cookies/mawia@rb4.worldsex[2].txt
/home/all/m/mawia/bin/BORUSSIA/XP/profile/Cookies/mawia@sextracker[1].txt
/home/all/m/mayer/.kde/share/cache/favicons/sextensiv.com.png
/home/all/s/salou/.kde/share/cache/favicons/www.sexmosaic.com.png
/home/all/s/schaefni/ml_projekt/doku/emerkmalsextraktion.tex
/home/all/s/seemanns/.kde/share/cache/favicons/www.perfectsexteens.com.png
/home/all/s/seemanns/.kde/share/cache/favicons/www.sexycrazybabes.com.png
/home/all/t/tabet/BORUSSIA/DOTNET/profile/Cookies/tabet@counter.sexsuche[1].txt
/home/all/t/thommy/.sigfiles/sig.Linuxsex
/home/all/t/thommy/.sigfiles/sig.alt.sex
/home/all/t/thommy/.sigfiles/sig.computer.sex
/home/cis/cissoft/.netscape/xover-cache/host-/alt.homosexual.snm
/home/cis/cissoft/.netscape/xover-cache/host-/alt.politics.homosexuality.snm
/home/cis/cissoft/.netscape/xover-cache/host-/alt.politics.sex.snm
/home/cis/cissoft/.netscape/xover-cache/host-/alt.sex.bestiality.snm
/home/cis/cissoft/.netscape/xover-cache/host-/alt.sex.bondage.snm
/home/cis/cissoft/.netscape/xover-cache/host-/alt.sex.graphics.snm
/home/cis/cissoft/.netscape/xover-cache/host-/alt.sex.homosexual.snm
/home/cis/cissoft/.netscape/xover-cache/host-/alt.sex.masturbation.snm
...
```

44.1.2 Wovor ist zu schützen?

Die größte Sicherheitslücke: Der Mensch. 44-10

Die größte Gefahr für Systeme geht häufig von den *Benutzern* aus. In Unternehmen können die *eigenen Mitarbeiter* ihren Zugriff nutzen, um Daten oder gleich die ganze Hardware zu stehlen. Menschen können *auf Zettel aufgeschriebene Passwörter* ausspionieren. Menschen benutzen oft ganz *leicht zu erratende* Passwörter wie `gott` oder auch `26121975`.

Moral

Der »Faktor Mensch« muss in jedes Sicherheitskonzept einbezogen werden.

Die zweite Sicherheitslücke: Bösartige Programme. 44-11

Damit bösartige Software überhaupt zum Zuge kommt, müssen sie erstmal *ausgeführt* werden. Normalerweise geschieht dies *nicht freiwillig* durch den Benutzer oder das Betriebssystem. Vielmehr nutzt bösartige Software so genannten *Sicherheitslöcher* aus (dazu gleich mehr).

Glossar der Schädlinge. 44-12

Wurm Programm, das sich selbstständig über ein Netzwerk ausbreitet, indem es Kopien von sich selbst an andere Rechner schickt.

Trojaner Programm, das etwas sinnvolles oder hübsches macht, aber eine Schadensroutine enthält (normalerweise den Rechner zum Zombie macht).

Virus Programmteil, der eine Kopie von sich selbst an andere Programme anhängt und immer dann gestartet wird, wenn ein infiziertes Programm gestartet wird.

✎ Zur Diskussion

Wie bekommt man diese Schädlinge?

44-13


Public domain

Leider kein Science-Fiction: Der Wurm der Apokalypse.
Eine Studie der Wurmforscher des ICSI (International Computer Science Institute, Berkeley) ergab 2005 folgendes: Würmer sind in der Regel *extrem schlecht* und schlampig programmiert. Ein *sehr gut programmierter Wurm*, der alle bekannten Tricks nutzt und in ein einziges IP-Paket passt, könnte sich in ca. *30 Sekunden weltweit* ausbreiten. *In wenigen Minuten* könnte er das Internet komplett lahmlegen. Lädt er noch einen Firmware-Flasher nach (sehr schwierig), dann könnte er die weltweite IT-Infrastruktur *für Wochen lahmlegen*. Die Folgen für die Weltwirtschaft könnte man wohl als apokalyptisch bezeichnen.

44.1.3 Welche Maßnahmen helfen?

44-14

Gegen Datenverlust helfen nur Sicherungskopien.
Daten müssen regelmäßig gesichert werden. Punkt.

44-15

Glossar der Sicherungsmaßnahmen.

Kennwörter Systeme werden zum Schutz in verschiedene *Bereiche* aufgeteilt, zu denen man nur mittels des richtigen Kennworts Zugriff bekommt.

Firewall Programm oder Rechner, der die Verbindung eines Rechners oder eines Teilnetzes zum Internet überwacht. Er lässt nur als *sicher eingestufte* und *vertrauenswürdige* Kommunikation zu.

Virenscanner Programm, das Speicher und Festplatte nach den ihm bekannten Viren, Würmern oder Trojanern durchsucht.

IDS Intrusion-Detection-Systeme beobachten das Verhalten von Rechner(netzen). Im Falle von auffälligem Verhalten (beispielsweise massenhafte E-Mails) wird der Rechner gestoppt oder verlangsamt.

44.2 Fallbeispiel eines Sicherheitslochs

44.2.1 Die Methode: SQL-Injection

44-16

Das Sicherheitsloch »SQL-Injection«.
SQL-Injection ist ein Methode, Schwachstellen von *Web-Servern* auszunutzen, die auf eine SQL-*Datenbank* zugreifen. Die *Angreifer* sind Menschen oder Computer. Die *Angegriffenen* sind Web-Server. Ziel des Angreifers ist es, den Web-Server dazu zu bringen, dass er SQL-*Code des Angreifers ausführt*. Man sagt, der Angreifer *»injiziert* SQL-*Code in den Web-Server«*, daher der Name.

44-17

Ablauf einer SQL-Injection.

Normale Kommunikation

1. Nutzer trägt Daten in ein Web-Formular ein
2. Web-Client schickt Formular-Daten an den Web-Server
3. Web-Server führt daraufhin einen SQL-Befehl aus, um Daten aus der Datenbank zu holen
4. Web-Server schickt Antwort an den Web-Client

Kommunikation mit SQL-Injection

1. Nutzer trägt *ungewöhnliche Daten* in ein Web-Formular ein
2. Web-Client schickt Formular-Daten an den Web-Server
3. Web-Server führt SQL-Befehl aus, der aber aufgrund der ungewöhnlichen Daten *ungewöhnliche Effekte* hat.
4. Web-Server schickt *nicht gewollte* Antwort an den Web-Client

44.2.2 Fiktives Beispiel: Firmen-Intranet

Das Intranet von Molecular Sheep.

44-18

Die Firma Molecular Sheep verfügt über ein *Intranet*. Dieses bietet Zugang auf firmeninterne Daten (wie die Fellfarben von Dolly und Flauschi). Man muss sich *authentifizieren*, um Einlass zu erhalten. Die Liste der berechtigten Personen und deren Passwörter ist in einer Datenbanktabelle gespeichert. Leider wurde an der Sicherheit gespart, weshalb die Eingangskontrolle schlampig programmiert wurde.

Der Login-Vorgang von Molecular Sheep.

44-19

```
<!-- HTML-Login-Seite --!>
<form action="http://molecular-sheep.com/login.java" method="post">
  <p>User:     <input name="user" type="text"/> </p>
  <p>Password: <input name="pass" type="text"/> </p>
  <p><input name="submitButton" value="Login" type="submit"/></p>
</form>
```

Wenn sich User `ich` mit dem Passwort `gott` einloggt, wird folgende Anfrage an den Web-Server von Molecular-Sheep geschickt:

```
http://molecular-sheep.com/login.java?user=ich&pass=gott
```

Daraufhin ruft der Web-Server das Programm `login.java` auf mit den Parametern `ich` und `gott` auf. Darin:

```
boolean checkPassword (String user, String password) {
  ...
  String sqlQuery =
    "select␣*␣from␣password_table␣where␣user_name=\"" +
    user + "\"␣and␣password=\"" + password + "\"";

  Statement statement = connection.createStatement();
  statement.executeQuery (sqlQuery);
  if (statement.getMoreResults () == false)
    return false;
  else
    return true;
}
```

Das Sicherheitsloch von Molecular-Sheep.

44-20

Login für User `ich` mit Passwort `gott`

Der `sqlString` lautet

```
select * from password_table where
  user_name="ich" and password="gott";
```

Dies liefert genau fann mehr als null Treffen, wenn es den passenden Eintrag in der Tabelle `password_table` gibt.

Login mit SQL-Injection als User mit leerem Passwort

Ein Angreifer tippt nun als »Benutzernamen« folgendes ein:

```
egal" or true; --
```

Dann wird folgender SQL-Befehl ausgeführt:

```
select * from password_table where
  user_name="egal" or true; --" and password="";
```

Da `--` einen Kommentar in SQL beginnt, liefert dies immer Treffer.

Moral

44-21

1. Vertraue *niemals* Benutzereingaben!
2. Benutzereingaben dürfen *niemals* ohne besondere Vorkehrungen mit Befehlen vermischt werden.

44.2.3 Reales Beispiel: www.doc.state.ok.us

44-22

Little Shop of Horror für Datenschützer: Webseite des Department of Corrections, Oklahoma.

Im US-Staat Oklahoma sind (im Jahr 2008) die Insassen *aller Gefängnisse* über das Internet zugreifbar. Für *jeden Gefangenen* sind über ein *komfortable Suchfunktion* folgende Informationen bequem abrufbar:

- Name, Geburtsdatum, Rasse (!),
- Foto(s) des Gefangenen (!!),
- Komplettes Vorstrafenregister.

Für ehemalige Sexualstraftäter sind auch *nach der Entlassung (für mindestens 15 Jahre bis lebenslang)* verfügbar:

- aktuelle Anschrift,
- Telefonnummer.

Sexualstraftaten sind dort neben Vergewaltigung auch »distribution of obscene videos« (= Verkauf von Pornos). Sehr liebevoll gemacht ist auch die Seite mit dem *Hinrichtungs-Countdown* für die Menschen im Todestrakt. (Mit dem Betreiben einer solchen Seite würden Sie sich in Deutschland strafbar machen.)

44-23

Das Sicherheitsloch des DOC Oklahoma.

Die Informationen über die (ehemaligen) Gefangenen stehen in einer relationalen Datenbank. Die Suche in diesen Daten wird durch eine SQL-Anfrage bewerkstelligt. Das (absolut vollkommen unvorstellbar gigantische) Sicherheitsproblem besteht darin, dass die SQL-Anfrage in einen Link eingebettet ist. Dieses Sicherheitsproblem bestand zwischen den Jahren 2005 und 2008. Das DOC wurde auf das Problem aufmerksam gemacht, reagierte durch (völlig nutzlose) Änderungen der SQL-Anfrage. Das DOC löste das Problem erst, als man ihm die (ebenfalls in der Datenbank gespeicherte) Liste der medizinischen Behandlungen des Personals des DOC schickte.

44-24

Der Link im Original, umformatiert.

```
<a href="http://docapp8.doc.state.ok.us/pls/portal30/url/page/sor_roster?sqlString=
     select distinct
       o.offender_id,doc_number,o.social_security_number, o.date_of_birth,
       o.first_name,o.middle_name,o.last_name,o.sir_name,sor_data.getCD(race) race,
       sor_data.getCD(sex) sex,l.address1 address,l.city,l.state stateid,l.zip,
       l.county,sor_data.getCD(l.state) state,l.country countryid,
       sor_data.getCD(l.country) country,decode(habitual,'Y','habitual','')
       habitual,decode(aggravated,'Y','aggravated','') aggravated,
       l.status,x.status,x.registration_date,x.end_registration_date,l.jurisdiction
     from registration_offender_xref x, sor_last_locn_v lastLocn, sor_offender o,
       sor_location l, (select distinct offender_id
                        from sor_location
                        where status = 'Verified' and upper(zip) = '73064' ) h
     where lastLocn.offender_id(%2B) = o.offender_id and
           l.location_id(%2B) = lastLocn.location_id and
           x.offender_id = o.offender_id and
           x.status not in ('Merged') and x.REG_TYPE_ID = 1 and
           nvl(x.admin_validated,to_date(1,'J')) >= nvl(x.entry_date,to_date(1,'J'))
           and x.status = 'Active' and x.status <> 'Deleted' and
           h.offender_id = o.offender_id
     order by o.last_name,o.first_name,o.middle_name&sr=yes">
Print Friendly </a>
```

44-25

✎ Zur Übung

Das Sicherheitsloch (eher Sicherheitsabgrund) dieser Webseite lässt sich ausnutzen, indem Sie einfach eine eigene Web-Seite erstellen, auf der Sie den Link kopieren, dann aber den Link-Text an entscheidenden Stellen ändern.
Was müssen Sie ändern, um

1. die Liste der Gefangenen zu bekommen, die eigentlich von der Liste gestrichen sind?
2. herauszubekommen, welche anderen interessanten Tabellen lesbar sind?
3. Gefangene aus der Datenbank zu löschen?
4. fiktive Gefangen in die Datenbank einzutragen?
5. dem Spuk ein Ende zu bereiten und die ganze Datenbank zu löschen?

Zusammenfassung dieses Kapitels

44-26

1. *Daten, Kommunikation und Rechner* sind vielfältigen Gefahren ausgesetzt, die hauptsächlich von (unvorsichtigen, bösartigen oder dummen) *Menschen* ausgehen sowie von *Würmern*.
2. Gute Passwörter, regelmäßiges Schließen von Sicherheitslöchern und die Benutzung von Verschlüsselung bieten *guten, aber längst nicht perfekten Schutz*.
3. Seine Daten zu schützen sollte genauso selbstverständlich sein, wie das Abschließen der Wohnungstür oder des Fahrrades.
4. Der Staat Oklahoma ist allerdings der Meinung, dass all dies für ihn nicht gilt.

Übungen zu diesem Kapitel

Übung 44.1 SQL-Injection-Schwachstelle erkennen und beheben, mittel

Auf dem Server eines Online-Buchversands liegen Java-Programme, die Benutzereingaben eines Formulars im HTML-Format auswerten und entsprechende Datenbankabfragen ausführen. Dabei wird unter anderem folgende Methode aufgerufen:

```
String buecherliste( String suchbegriff, String maximale_anzahl ){
        ResultList r = Database.executeQuery( "SELECT␣*␣FROM␣books␣"
                +"WHERE␣title␣LIKE␣'%"+suchbegriff+"%'␣"
                +"LIMIT␣"+maximale_anzahl );
        String html = formatiere_resultate( r );
        return html;
}
```

1. Geben Sie drei verschiedene Möglichkeiten an, wie durch Benutzereingaben im HTML-Formular die Tabelle `books` aus der Datenbank gelöscht werden kann.
2. Erweitern Sie die Methode `buecherliste` um zusätzliche Befehle zur Vorverarbeitung der Eingabe, durch die die SQL-Injection-Schwachstelle behoben wird. Legitime Benutzereingaben sollen allerdings durch diese zusätzlichen Sicherheitsmaßnahmen nicht beeinträchtigt werden.

Prüfungsaufgaben zu diesem Kapitel

Übung 44.2 Absichten von E-Mail-Scams einschätzen, leicht, original Klausuraufgabe

Sie erhalten drei E-Mails, die in böswilliger Absicht an Sie geschrieben wurden.

1. Absender ist angeblich Ihre Bank. Sie werden aufgefordert, eine Website anzusurfen und Ihre Konto-Daten zu überprüfen beziehungsweise zu aktualisieren.
2. Absender ist angeblich Toni. Sie werden aufgefordert, die Datei Geburtstagsfeier.exe im Anhang der Mail zu öffnen.
3. Absender ist angeblich ein Königssohn aus Nigeria. Sie werden herzlich gebeten zu helfen, einen Geldtranfer nach Europa, den die nigerianische Regierung behindert, möglich zu machen. Nehmen Sie bitte E-Mail-Kontakt zum Absender auf.

Geben Sie für jede der drei E-Mails an, ob eine oder einige der folgenden drei Absichten jeweils dahinter stecken könnte.

1. Ausspionieren von Passwörtern.
2. Verbreiten eines Wurms.
3. Missbrauch des PC für eine DoS-Attacke.