Anwendung kryptographischer Programme
am Beispiel von NetBSD

5. Chemnitzer Linuxtag 2004

Stefan Schumacher, <stefan@net-tex.de>, PGP Key <0xB3FBAE33>

http://www.net-tex.de/krypt/

$Header: /home/stefan/daten/cvs/lectures/crypt/folien.tex,v 1.15 2004/02/27 13:09:22 stefan Exp $

Gliederung

- Motivation
- Prüfsummenverfahren
- symmetrische Verfahren
- asymmetrische Verfahren
- kryptographische Signaturen
- weitere Anwendungen

Motivation

- Mittel der Kriegsführung
- seit kurzem frei verfügbar
- Schutz von Geheimnissen (privat / kommerziell)
- Verifikation von Daten oder Kommunikation
- Grundgesetz, Artikel 10, Abs 1.

Prüfsummen

- http://www.pruefziffernberechnung.de/
- Basiszahl (Information) und Prüfziffer
- Prüfziffer wird aus Basiszahl berechnet
- beide Werte werden übertragen
- Empfänger berechnet Prüfziffer und vergleicht
- Authenzität (Integrität der Prüfziffer?)

Anwendung `md5(1)`

- RFC1321, erzeugt 128 Bit Prüfsummen
- $2^{128} = 340.282.366.920.938.463.463.374.607.431.768.211.456$
- Falltürfunktion
- als Programm bei (fast) jedem Unix dabei
- Einsatz u.a. als Passworthash
- Prüfsumme für FTP / pkgsrc

Anwendung: aide

http://www.cs.tut.fi/~rammer/aide.html

- reaktives HIDS / Dateisystemintegritätsprüfer
- Datenbank mit Dateistatus & Prüfsummen
- ermöglicht Vergleich des aktuellen Systems
- deckt Manipulationen an Dateien auf

Anwendung: `verifiedexec(4)`

- Ab NetBSD 2.0 (1.6-current)
- proaktives HIDS
- Abhängig von Kernel Securitylevel
- Prüfsummenvergleich im `exec()` Pfad des Kernel
- verweigert Ausführung bei unpassender Prüfsumme

symmetrische Verfahren

- verwendet genau einen Schlüssel
- wandelt Klartext mit Schlüssel k in Code um
- wandelt Code mit Schlüssel k in Klartext um
- Anwendung: `mcrypt(1)`
- `mcrypt -k GEHEIM -a rijndael-256 Brief.tex`
- `mcrypt -d -k GEHEIM Brief.tex.nc`

symmetrische Verfahren

- Alice schickt `Brief.tex.nc` an Bob
- Alice schickt Schlüssel k an Bob **GEFAHR**
- Schlüssel kann abgefangen werden $\rightarrow$
  sicherer Kanal nötig
- Integrität?
- Gruppenkommunikation: $n * (n-1)/2$ Schlüssel nötig
- 10 Personen $\rightarrow$ 45 Schlüssel

Anwendung: `cgd(4)`

- Ab NetBSD 2.0 (1.6-current)
- verschlüsselt Blöcke auf dem Weg zur Partition
- Daten liegen verschlüsselt auf Platte
- über Pseudodevice ver-/entschlüsselt eingemountet

Anwendung: `cfs`

- `cryptographic filesystem` von Matt Blaze
- implementiert im Userland über NFS ähnlichen `cfsd`
- Daten liegen verschlüsselt im Dateisystem
- bei Zugriff über `cfsd` auf `localhost` ver-/entschlüsselt
- verschlüsselte Daten können über Netz gehen und lokal entschlüsselt werden
- portabel

Exkurs: Daten löschen

- `rm(1)` löscht Daten nicht
  sondern gibt die Datenblöcke frei
- Daten können rekonstruiert werden
- Curietemperatur oder Degausser, vernichten Datenträger
- Daten müssen mehrfach überschrieben werden
- Wiper (bcwipe, wipe, ncrypt ...)
- `dd if=/dev/urandom of=/dev/wd0d bs=1k`

asymmetrische Verfahren

- Schlüsselpaar aus öffentlichem und privatem Schlüssel
- privater Schlüssel ist **geheim** zu halten
- Alice und Bob haben je öffentlichen $(A_{\ddot{o}}, B_{\ddot{o}})$ und privaten (A_p, B_p) Schlüssel
- Alice verschlüsselt für Bob $schematisch$ so:
  $Chiffrierung(Klartext + A_p + B_{\ddot{o}}) \rightsquigarrow Code$
- Bob entschlüsselt so:
  $Dechiffrierung(Code + B_p + A_{\ddot{o}}) \rightsquigarrow Klartext$

asymmetrische Verfahren

- Alice schickt $Code$ und $A_{\ddot{o}}$ an Bob
- Sniffer kann nichts damit anfangen, benötigt B_p
- Bob besitzt B_p und kann $Code$ dechiffrieren
- Bob kann Integrität prüfen, da nur Alice A_p hat

kryptographische Signatur

- Sonderfunktion der asymmetrischen Verfahren
- sichert Integrität und Authenzität
- Alice bildet Prüfsumme über Nachricht $h(N)$
- verschlüsselt Prüfsumme mit A_p: $S = \{h(N)\}_{A_p}$
- Bob empfängt Nachricht, Signatur und $A_{\ddot{o}}$
- Bob bildet selbst Prüfsumme $h'(N)$
- Bob entschlüsselt S mit $A_{\ddot{o}} \rightsquigarrow h(N)$
- gilt $h'(N) = h(N)$?

Beispielsignatur

```
-----BEGIN PGP SIGNED MESSAGE-----
Hash: SHA1

Von Zeit zu Zeit seh ich den Alten gern,
Und hüte mich, mit ihm zu brechen.
Es ist gar hübsch von einem großen Herrn,
So mit dem Teufel selbst zu sprechen.
-----BEGIN PGP SIGNATURE-----
Version: GnuPG v1.2.4 (NetBSD)

iD8DBQFAA8ySEfTEHrP7rjMRAmLxAJ0ePbnStfzK4NVSZFMSpvhhIp2NqACdEuFA
NChplstKIDO3nVEDEDdOUmA=
=+lsL
-----END PGP SIGNATURE-----
```

kryptographische Signatur

- Gehört $A_{\ddot{o}}$ wirklich Alice? (`0xDEADBEEF`)
- Ist Alice überhaupt Alice?
- Jemand signiert $A_{\ddot{o}}$
- Prüfe Alice Identität (Ausweis)
- Prüfe $A_{\ddot{o}}$ (Name, eMail ...)
- Schicke Signatur verschlüsselt per eMail

kryptographische Signatur

- **Baumhierarchie**
- zentrale Zertifizierungsinstanz
- anfälliger (Kann ich der CA trauen?)
- **Web of Trust**
- Netz, hat daher keine Wurzel
- baut Kette auf (über Signatur der Signierer ...)
- weniger anfällig bei einem Fehler
- keine CA nötig

Bsp.: http://www.uni-magdeburg.de/steschum/local-web.jpg

PGP/GnuPG

- **P**retty **G**ood **P**rivacy von Phil Zimmermann
- "Killerapplikation" der Kryptographie
- Hauptsächlich für eMail entwickelt
- resultierte in OpenPGP (RFC2440)
- GnuPG als Alternative
- Standard auf Unixsystemen

Steganographie

- Verstecken von Information
- Bspw. "Geheimtinte"
- Wasserzeichen, "Kopierschutzverfahren"
- nützlich wenn Kryptographie verboten
- Dateisystem: StegFS