Ausgewählte Security-Tools für UNIX

TCP-Wrapper und S/Key

TCP-Wrapper

Anliegen

- Zugangskontrolle für TCP/UDP-Dienste
- Logging von Verbindungsanforderungen
- kein Eingriff in installierte Software auf dem lokalen System
- kein Datenaustausch mit dem die Verbindung anfordernden Client

Autor

Wietse Venema (wietse@wzv.win.tue.nl)
Department of Mathematics and Computing Science
Eindhoven University of Technology

TCP-Wrapper

Problem

- Verbindungsanforderungen für Dienste wie `telnet, ftp, rlogin,` ... werden vom `inetd` aufgrund `/etc/inetd.conf` erfüllt

  ```
  telnet stream tcp nowait root /usr/etc/in.telnetd in.telnetd
  ```

  bzw. nicht erfüllt

  ```
  #telnet stream tcp nowait root /usr/etc/in.telnetd in.telnetd
  ```

- die meisten Systeme bieten keine Zugangskontroll- bzw. Loggingmöglichkeiten

TCP-Wrapper

Lösung:

- `tcpd` steht als *tiny daemon wrapper* zwischen dem `inetd` und dem zu startenden Server (z.B. `in.telnetd`)
- `tcpd` wertet Konfigurationsfiles aus, in denen Zugangsgenehmigungen erteilt bzw. entzogen werden können
- `tcpd` ermöglicht vor dem Start des entsprechenden Servers zusätzliche Aktionen
- `tcpd` sendet Logging-Informationen an `syslogd` (remote host, ggf. remote user, ausgeführte Aktion)

TCP-Wrapper

Aktivieren

- Installieren als `/etc/tcpd`
- Modifikationen in `/etc/inetd.conf`

```
telnet stream tcp nowait root /etc/tcpd /usr/etc/in.telnetd
```

Access Control Rules

- `/etc/hosts.allow` und/oder `/etc/hosts.deny`
  Definition von Regeln für die Zugangskontrolle

TCP-Wrapper

Access Control Rules – Auswertung

- erst `/etc/hosts.allow` anschließend `/etc/hosts.deny`
- sobald ein zutreffender Eintrag (daemon, client) gefunden wird, bestimmt dieser Eintrag die Zugangskontrolle
- wenn es die Files nicht gibt oder die Files leer sind, ist die Zugangskontrolle ausgeschaltet und der Zugang wird gewährt
- `tcpd` verweigert den Zugang, wenn gelieferter Hostname und IP-Adresse des Clients nicht zusammenpassen

TCP-Wrapper

Beispiel: `/etc/hosts.allow`

```
in.telnetd : 134.109.
in.telnetd : ALL : twist=/bin/echo Sorry, %c, not allowed
in.ftpd    : 134.109.
in.ftpd    : ALL : twist=/bin/echo Sorry, %c, not a ftp server
in.rlogind : @urz
in.rlogind : ALL : twist=/etc/rshd-echo Sorry, %c, not allowed
in.rshd    : @urz
in.rshd    : ALL : twist=/etc/rshd-echo Sorry, %c, not allowed
in.fingerd : 134.109.
in.fingerd : ALL : twist=/bin/echo Sorry, %c, not allowed
in.tftpd   : ALL : deny
```

TCP-Wrapper

Logging

- standardmäßig über das Logging-Level `mail.debug` (bei uns `user.info`)
- entsprechende Konfiguration des `syslogd` notwendig (`/etc/syslog.conf`)

Auszug aus Log-File:

```
Dec 14 14:46:11 rom.hrz.tu-chemnitz.de in.fingerd[1173]:
  twist kirke.zcu.cz to
  /bin/echo Sorry, kirke.zcu.cz, finger is not supported
Dec 15 06:53:21 atlas-f.hrz.tu-chemnitz.de in.fspd[2163]:
  connect from taurus.ludd.luth.se
Dec 15 07:13:28 sisyphus.hrz.tu-chemnitz.de in.telnetd[10654]:
  twist keeper.bosch.de to
  /bin/echo Nice to meet you, keeper.bosch.de;exec /usr/sbin/in.telnetd
```

TCP-Wrapper

weitere Tools/Libraries

`rshd-echo`	`echo` für Prozesse ohne steuerndes Terminal
`try, try-from`	Tests des Access Control Rules
`safe-finger`	sichere Implementation von `finger`
`libwrap.a`	hosts_access(3), hosts_ctl(3)

weitere Infos

Manuals: `tcpd`, `hosts_options`, `hosts_access`

Installation

in `/uni/global` für sun4, sun5, hp9k700-9, hp9k800, dec3000, dec5000, risc6000, sgi4000, linux

S/Key

Anliegen

- Schutz vor Netz-Abhörern (eavesdropping protection)
- Vermeidung der Übertragung von geheimen Klartext-Paßworten über das Netz z.B. bei `su` und `login` (`telnet, rlogin`)
- Verwendung von nur einmal gültigen Paßworten
- keine geheimen Algorithmen, keine Notwendigkeit geheime Informationen zu speichern

Autoren

Neil M. Haller (nmh@thumper.bellcore.com), Philip R. Karn (karn@chicago.qualcomm.com), John S. Walden (jsw@thumper.bellcore.com), Scott Chasin (chasin@crimelab.com)

S/Key

Prinzip

- Challenge/Response-Verfahren
- remote Host (z.B. `su`) sendet eine Challenge-Nachricht, daraus wird am lokalen Host (oder offline) die Response mit Hilfe des geheimen Paßworts berechnet und an den remote Host zurueckgesendet
- zur Berechnung der Response wird die *one-way function* MD4 verwendet
- MD4 berechnet zu jedem beliebigen Input-String einen *message digest* (16 Byte), wobei es nicht möglich ist zu einem vorgebenen message digest den zugehörigen Input-String zu bestimmen

S/Key

Prinzip

- Challenge besteht aus zwei Teilen:

sequence number (n)	wird bei jeder Benutzung dekrementiert
seed (s)	zufällige Zeichenkette
	wird mit geheimem Paßwort verknüpft

Umfang

`key`	berechnet Response aus Challenge
`keyinit`	initialisiert die Verwendung des Systems für einen Nutzer
`keysu`	ersetzt `su`

S/Key

Erzeugen des Einmal-Paßworts (Kommando `key`)

- Prinzip: mehrfache Anwendung der one-way Funktion
- das erste Einmal-Paßwort wird berechnet durch (n - 1) malige Anwendung der one-way Funktion auf den String *geheimes Paßwort + seed*
- das Ergebnis (16 Byte) wird auf 8 Byte (2^{64} Möglichkeiten) abgebildet und daraus wird eine Folge von sechs englischen Worten gebildet

S/Key

Überprüfen des Einmal-Paßworts (Kommando `keysu`)

- die one-way Funktion wird noch einmal auf das empfangene Einmal-Paßwort angewendet und mit dem beim letzten Mal abgespeicherten Einmal-Paßwort verglichen
- bei Übereinstimmung wird der Zugang gewährt
- die sequence number wird dekrementiert und gemeinsam mit dem empfangenen Einmal-Paßwort abgespeichert (`/etc/skeykeys`)

S/Key

Beispiel:

`remote% keysu`	`local% key 986 ka14343`
`[s/key 986 ka14343]`	`Enter secret password:`
`Response:`	`ROIL HOOF SOUL COW SHIN COAL`
	`local%`
`remote#`	

`keysu` basiert auf der GNU-Implementation von `su`, jedoch erweitert um einige Logging-Möglichkeiten (`auth.notice`) und um den `wheel`-Mechanismus aus dem BSD

S/Key

weitere Informationen:
Manuals zu `skey`, `key`, `keyinit`

Installation:
in `/uni/global` für sun4, sun5, hp9k700-9, hp9k800, dec3000, dec5000, risc6000, sgi4000, linux