Das TIS Firewall Toolkit unter Linux

Die Zahl, der im Netzwerk- und Kommunikationsbereich eingesetzten Linux Systeme steigt von Jahr zu Jahr an, sei es als Internet ISDN-Gateway-, Dial-In-, FTP- oder WWW-Server. Das legt nahe, dieses bewährte Betriebssystem auch als Firewall-Rechner zu nutzen. Der folgende Artikel soll anhand eines kurzen Beispiels das Zusammenspiel von Linux mit dem Firewall Toolkit (FWTK) der Firma TIS (Trusted Information Systems Inc.) darstellen.

Das FWTK ist ein modular aufgebautes Software-Paket. Die Komponenten sind so aufeinander abgestimmt, daß man mit ihnen einen einfachen, aber wirkungsvollen Bastion Host konfigurieren kann. Das Toolkit läuft unter UNIX-Systemen, die TCP/IP mit Berkeley Socket Interface unterstützen. Der damit geschaffene Bastion Host fungiert als Application-Level Gateway. Das FWTK bietet keine Packet Filter-Funktionen. Der Bastion Host sollte niemals ohne Schutz seine Arbeit im Netz verrichten. Es ist ratsam, mindestens einen weiteren Rechner oder Router im Netz zu plazieren, der die IP-Filterung vornimmt. Die alleinige Installation eines IP-Filters auf dem Bastion Host ist nicht ratsam. Möglich wäre eine klassische Installation in einem Screened Subnet. Weitere Installationsvarianten und allgemeine Hinweise zu Firewall-Systemen finden Sie auch im Heft 14 der RZ-Mitteilungen.

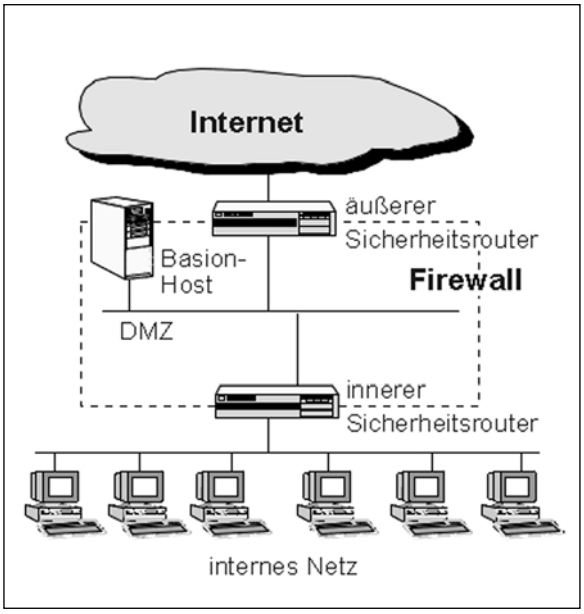

Abb1.: Screened Subnet

IP-Filter oder Packet Filter stellen einen einfachen Bestandteil eines Firewall-Systems dar. Sie überprüfen den Verkehr im Netzwerk und filtern die Pakete nach Quell-IP-Adresse, Ziel-IP-Adresse und Dienst. IP-Filter sind einfach zu installieren und zu konfigurieren. Ihr grundlegender Nachteil ist allerdings das Vertrauen in die IP-Adresse und die globale Abwehr von Paketen, ohne deren Inhalt zu kennen. Es sind also Mechanismen nötig, die die Selektion auf einer höheren Ebene des OSI-Layer Modells vornehmen.

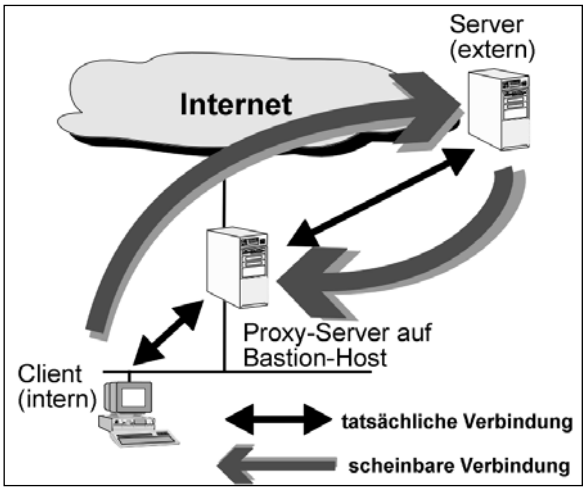

Abb.2: Proxy

Das **Application-Level Gateway** ist sinnvoller Bestandteil eines Firewall-Systems. In der Regel sind diese Gateways nach dem Grundsatz „Alles, was nicht ausdrücklich erlaubt ist, ist verboten“ konfiguriert. Diese Regel ist auf der Applikationsebene des OSI-Layer Modells angesiedelt und erlaubt eine genaue Kontrolle des den Bastion Host passierenden Netzverkehrs. Bastion Hosts können entweder mit mehreren Netzwerkkarten ausgerüstet werden oder in einem von speziellen IP-Filtern geschützten Subnetz stehen.

Das FWTK unterstützt die Funktionen eines Bastion Host durch mehrere kleine, überschaubare Programme, die wie in einem Baukasten zusammengesetzt und nur über eine Konfigurationsdatei administriert werden müssen. Für jeden Dienst, der das Firewall-System passieren soll, wird ein Application-Level **Proxy** installiert. Ein Proxy verhält sich für den Nutzer vollkommen transparent. Der Kommunikationspartner erliegt der Illusion, daß er nur mit dem Bastion-Host kommuniziert, und dem Nutzer im Netz wird eine direkte Internetverbindung vorgetäuscht (Abb. 2).

Das Toolkit enthält Proxies für `telnet`, `ftp`, `rlogin`, `SMTP-Mail`, `http`, `gopher`, `X11-Windows` und ein `plug-gw`, das als transparenter Pass-through Proxy für sehr viele TCP-Dienste einsetzbar ist.

Zusätzlich enthält das Softwarepaket die Programme `netacl`, welches den Zugriff auf Layer-3-Ebene kontrolliert, und `authsrv`, das für die Netzwerk Authentisierung verantwortlich ist.

Vorbereiten des Bastion Host

Der erste Schritt ist die ordnungsgemäße und sichere Installation einer aktuellen (!) Linux-Distribution. Der Bastion Host sollte nur diesem Zweck dienen, d. h. jede andere Funktion oder gar Nutzerverkehr ist unbedingt auszuschließen. Dem folgend sollten nicht benötigte Betriebssystem-Komponenten deinstalliert, alle nicht benötigten Dienste deaktiviert und die dazugehörige Software gelöscht werden. NIS und NFS sollten für Ihren Bastion Host unbekannte Begriffe sein.
Der Betriebssystem-Kernel (aktuelle stabile Version - kein Developer-Kernel) sollte mit einigen veränderten Optionen neu kompiliert werden. Deaktivieren Sie dabei die nicht benötigten Optionen und Module, auch die Option `CONFIG_IP_FORWARDING`. Lesen Sie im Zweifelsfall in einer aktuellen `FAQ` oder `README`-Datei nach.

Nachdem Sie die Datei `/etc/inetd.conf` von allen nicht benötigten Diensten bereinigt haben, sollten Sie die Startup Verzeichnisse `/etc/rc.*d/` überprüfen. Löschen Sie gnadenlos alle nicht benötigten Dienste, auch alle `rpc`-basierten Dienste nebst `portmapper`, von Ihrem Host. Sie können die laufenden Dienste mit dem Befehl `netstat -a` überprüfen. Im Unterverzeichnis `tools/admin/` der FWTK-Distribution befindet sich u. a. das einfache Testprogramm `portscan`, mit dem Sie nach laufenden Diensten suchen können.

Installieren Sie spezielle Software, die als Intruder Detection fungiert. Dies können Programme sein, die erkennen, wann der Host von einem Portscanner getestet wird. Es sollte auch Software installiert sein, die SYN-Flooding Attacken registriert. Um die Integrität der Installation zu gewährleisten, ist es ratsam, Tripwire einzusetzen. Mit diesem Programm können Sie im Dateisystem Veränderungen nach der Installation feststellen.

Jeglicher administrativer Zugriff auf den Bastion Host sollte über die Konsole oder über eine verschlüsselte Verbindung erfolgen.

Kompilieren des Firewall Toolkits

Besorgen Sie sich das FWTK von einer vertrauenswürdigen Quelle und überprüfen Sie die vorhandene digitale Signatur. Der FTP-Server von TIS scheint dazu am besten geeignet. Zusätzlich sollten Sie über die aktuellen Patches verfügen. Entpacken Sie das FWTK-Archiv, editieren Sie das `Makefile` nach Ihren Bedürfnissen und starten sie die Kompilierung mit `make`.

Konfiguration der Network Access Control List

Nach dem erfolgreichen Kompiliervorgang sollten Sie, neben ein paar Tools, über drei Komponenten verfügen: `netacl`, `authsrv` und die verschiedenen Proxies. `Netacl` ist ähnlich dem tcp_wrapper `tcpd`, der mittlerweile auf vielen UNIX-Systemen zu finden ist. `Netacl` überprüft definierte Regeln auf Diensteebene. `Netacl` kann auch als Tool für die generelle proxy-unabhängige TCP/IP-Zugangskontrolle dienen. Die meisten Toolkit-Komponenten werden vom `inetd` aufgerufen. Dafür muß als erstes die Datei `/etc/inetd.conf` editiert werden:

```
# Beispiel inetd.conf mit netacl und finger
  stream tcp nowait daemon /usr/local/etc/
  netacl in.fingerd
```

`Netacl` überprüft die Definition in der Konfigurationsdatei `netperm-table` (in der Regel in `/usr/local/etc`), nach dem angesprochenen Dienst und der auszuführenden Aktion. Diese Datei ist die einzige Konfigurationsdatei für das gesamte Firewall Toolkit. `Netacl` versteht die Ausdrücke `-permit-hosts` und `-deny-hosts` und verlangt als letzten Parameter `-exec`.

```
# Beispiel netperm-table für Netacl
  netacl-service: permit-hosts address
                  -exec program name
  netacl-service: deny-hosts address
                  -exec program name
```

Ein Dienst kann durch mehrere Zeilen in der Konfigurationsdatei definiert werden. Der Parameter *`address`* kann eine Liste von IP-Adressen oder Hostnamen sein (Wildcards wie z. B. `141.20.*` oder `*.hu-berlin.de` sind erlaubt). Die erste Regel, die auf eine IP-Adresse oder einen Hostnamen zutrifft, wird genommen.

```
# Beispiel netperm-table für finger
  netacl-in.fingerd: permit-hosts .intern.de
    -exec /usr/sbin/in.fingerd
  netacl-in.fingerd: permit-hosts *
    -exec /bin/cat/usr/local/etc/
    mein_finger_script
```

Die erste Zeile ruft den `finger`-Daemon auf, wenn der anfordernde Host zur Domain `intern.de` gehört. Trifft dies nicht zu, gilt die zweite Zeile, die dann ein eigenes Script startet. Dieses Script könnte z. B. einen Hinweis enthalten oder in eine Protokolldatei schreiben. Die Konfigurationsdatei `netperm-table` enthält außer den Zugriffsregeln auch allgemeine Konfigurationsoptionen für die Proxy-Clients.

```
# Beispiel netperm-table für das telnet-gateway
  (tn-gw)
  tn-gw: welcome-msg /usr/local/etc/telnet-
         welcome.txt
  tn-gw: deny-msg /usr/local/etc/telnet-
         deny.txt
  tn-gw: help-msg /usr/local/etc/telnet-
         help.txt
```

Konfiguration des Authentisierungsservers

Das Firewall Toolkit beeinhaltet den Authentisierungsserver `authsrv`. Er ist optional und verfügt über eine Vielzahl von Authentisierungsmechanismen, die sehr einfach konfiguriert werden können. `Authsrv` unterstützt Authentisierungen für alle interaktiven Proxies, die im Firewall Toolkit enthalten sind. Diese Authentisierung kann sowohl für eingehende, als auch für ausgehende Anforderungen durchgeführt werden. Der Server kann für jeden Proxy in der Konfigurationsdatei `netperm-table` de- bzw. aktiviert werden.

`Authsrv` unterstützt neben der internen einfachen Klartext-Paßwortauthentisierung folgende stärkere Authentisierungsmechanismen:

- S/Key (Bellcore)
- SecurID (Secure Dynamics)
- Silver Card (Enigma Logic)
- SNK Secure Net Key (Digital Pathways).

Die gewünschten Authentisierungsmethoden müssen beim Kompilieren aktiviert werden. Im folgenden soll die Konfiguration des Einmalpaßwort-Systems S/Key beschrieben werden. Alle Optionen können in den Dateien im Verzeichnis `../fwtk/auth/` definiert werden. S/Key ist ein sogenanntes Challenge-Response Einmalpaßwort-System, das dem Benutzer bei der Anmeldung eine Sequenz Nummer präsentiert. Der Benutzer muß vorher mit seiner Sequenznummer und einem geheimen Paßwort eine Liste mit sechs einmal gültigen Paßwörtern erzeugen. Für jede Anmeldung muß ein anderes Paßwort benutzt werden.

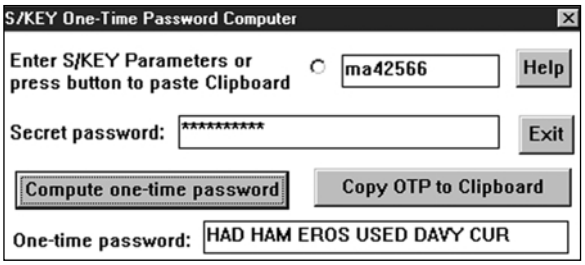

Abb. 3: S/Key Paßwortgenerierung unter Windows.

Da der Authentisierungsserver keinen registrierten TCP/IP-Port benutzt, muß ein unbenutzter Port für diesen Dienst in der Datei `/etc/services` konfiguriert werden. TIS schlägt den Port 7777 vor; dies hängt natürlich von den lokalen Ressourcen ab.

```
# Beispiel /etc/services
  authsrv        7777/tcp
```

Der korrespondierende Eintrag in der Datei `/etc/inetd.conf` könnte so aussehen:

```
# Beispiel /etc/inetd.conf für authsrv
  authsrv stream tcp nowait root /usr/local/
  etc/authsrv authsrv
```

Nach einem Neustart des `inetd` kann der Authentisierungsserver über die Datei `netperm-table` konfiguriert werden. Der Server akzeptiert folgende Parameter: `database`, `permit-hosts`, `nobogus`, `userid` und `badsleep`. Die Option `database` definiert, wo die Datenbank mit den Nutzerinformationen gespeichert werden soll, während mit `permit-hosts` Hosts definiert werden können, die den Server kontaktieren dürfen. Lediglich die Proxy-Server auf dem Bastion Host sollten den Authentisierungsserver abfragen dürfen.

```
# Beispiel netperm-table für authsrv
  authsrv: database /usr/local/etc/authsrv.db
  authsrv: permit-hosts localhost bastionhost
```

Zum Einrichten von S/Key-Accounts auf dem Bastion Host muß der Authentisierungsserver initialisiert werden. Der beste Weg ist das lokale Starten von `authsrv` als `root`. Dann müssen Administrationsnutzer hinzugefügt und aktiviert werden:

```
root@bastion%./authsrv
authsrv # adduser admin
authsrv # enable admin
```

Für diesen Nutzer muß nun S/Key als Standard eingestellt werden.

```
authsrv # proto admin skey
authsrv # superwiz admin
```

Nachdem der Nutzer auch zum Superuser der Datenbank „befördert “wurde, muß sein geheimes Paßwort gesetzt werden. Dieses Paßwort sollte ein ganzer Satz sein.

```
authsrv # password admin "mein supergeheimes
          passwort"
ID admin s/key is 99 ma42566
authsrv # exit
```

Die angezeigte S/Key Sequenz muß benutzt werden, um Einmalpaßwörter generieren zu können (Abb. 3). Um einen Benutzer mittels Authentisierungsserver identifizieren zu können, muß der FWTK-Proxy folgende Schritte durchführen:

1. den Login-Namen des Benutzers abfragen,
2. Kontakt zum Authentisierungsserver aufnehmen und diesem den Benutzernamen mitteilen,
3. eine Antwort des Servers mit der Eingabeaufforderung für den Benutzer empfangen,
4. die Eingabeaufforderung des Servers anzeigen,
5. die Antwort des Benutzers entgegennehmen und zum Server senden,
6. vom Server entweder Einverständis oder Ablehnung empfangen,
7. dem Benutzer Zugang gewähren oder die entsprechende Fehlermeldung ausgeben.

Dieser gesamte Vorgang wird über eine einzige TCP-Verbindung zwischen Client und Authentisierungsserver abgewickelt.

Problematisch beim Betrieb eines solchen Authentisierungsservers ist die Verbindung zwischen Client und Server. Ein Angreifer, der sich als Authentisierungsserver maskiert, kann sich selbst unter einer beliebigen Kennung authentisieren. Daher sollte `authsrv` immer auf dem eigentlichen Bastion Host laufen und fremder Zugriff deaktiviert werden.

Für den entfernten administrativen Zugriff auf die Datenbank des Authentisierungsservers wird das Programm `authmgr` bereitgestellt. Es können u. a. Gruppen eingerichtet und mit speziellen Rechten versehen werden. Die beiden Hilfsprogramme `authdump` und `authload` dienen als Werkzeuge zum Bearbeiten der Datenbank des Authentisierungsservers.

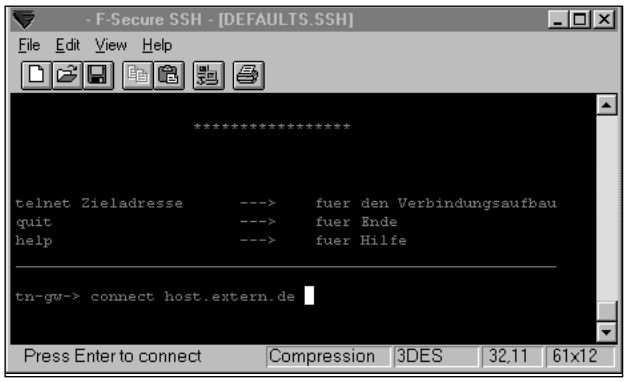

Abb. 4: tn-gw aus Benutzersicht

Konfiguration und Benutzung der Proxy Clients

Anhand der beiden klassischen Dienste `smtp` und `telnet` soll das generelle Konfigurationskonzept des FWTK dargestellt werden. Weiterführende Optionen entnehmen Sie bitte den Hilfedateien.

Wie ist die Konfiguration in Abb. 6 zu interpretieren? Zu Beginn einer Telnet-Sitzung startet `inetd` das Kontrollprogramm `netacl`. `Netacl` überprüft die Quell-IP-Adresse. Ist es nicht der Bastion Host selbst, wird der Telnet Proxy `tn-gw` gestartet. `Tn-gw` zeigt den Inhalt der Datei `telnet-deny.txt` an, wenn die Quell-Adresse `unknown` lautet. Desweiteren wird eine Verbindung von der Domain `.trusted.extern.de` zum Rechner `daten.bank.intern.de` ohne Abfrage des Authentisierungsservers erlaubt. Verbindungen aus der Domain `intern.de` können zu jedem externen Server aufgebaut werden. Den internen Nutzern ist auch erlaubt, das Bastion Host-Paßwort zu ändern. Wenn die Verbindung vom Bastion Host selbst kommt, wird der richtige `telnet`-Daemon gestartet. Diese Konfiguration soll nur als Anschauungsbeispiel dienen, denn `telnet`-Zugriffe auf das interne Netz sind im allgemeinen keine gute Idee.

Diesen Bildschirm (Abb. 4) bekommt jeder Nutzer angezeigt, der den Telnet-Proxy passieren muß. Folgende Meldungen werden an den `syslog`-Daemon – dessen modifizierte Version auch im Toolkit enthalten ist – geschickt (Abb. 5).

```
Nov 3 13:57:20 bastionhost tn-gw[ 21763] : permit host=unknown/111.111.111.111 use of gateway
Nov 3 13:57:29 bastionhost tn-gw[ 21763] : permit host=unknown/111.111.111.111
                                           destination=host.extern.de
Nov 3 13:57:29 bastionhost tn-gw[ 21763] : connected host=unknown/111.111.111.111
                                           destination=host.extern.de
Nov 3 14:05:13 bastionhost tn-gw[ 21763] : exit host=unknown/111.111.111.111 dest=host,extern.de
                                           in=51821 out=174 user=unauth duration=474
```

Abb. 5: `syslog`-Meldung

Eine sehr beeindruckende Lösung beinhaltet das FWTK für `smtp`-Mail. Es werden zwei Programme angeboten – `smap` und `smapd`.

Das folgende Beispiel soll die Konfiguration des `smtp`-Proxy demonstrieren. Beim Einsatz von `sendmail` als Mail-Relay muß dieses Programm gesondert konfiguriert werden.

```
# Beispiel /etc/inetd.conf für die Telnet-Proxy Konfiguration
telnet stream tcp nowait root /usr/local/etc/netacl in.telnetd

# Beispiel netperm-table für die Telnet-Proxy Konfiguration
netacl-in.telnetd: permit-hosts 127.0.0.1 -exec /usr/sbin/in.telnetd
netacl-in.telnetd: permit-hosts bastionhost -exec /usr/sbin/in.telnetd
netacl-in.telnetd: permit-hosts * -exec /usr/local/etc/tn-gw
tn-gw: deny-hosts unknown
tn-gw: permit-hosts .trusted.extern.de -dest daten.bank.intern.de
tn-gw: permit-hosts .intern.de -passok
tn-gw: permit-hosts * -auth -dest !bastionhost !127.0.0.1
tn-gw: authserver localhost 7777
tn-gw: denial-msg /usr/local/etc/telnet-deny.txt
tn-gw: welcome-msg /usr/local/etc/telnet-welcome.txt
tn-gw: help-msg /usr/local/etc/telnet-help.txt
```

Abb. 6: Beispielkonfiguration für Telnet

`Smap` ist ein kleiner, einfacher `smtp`-Client, der – von `inetd` gestartet – auf Port 25 `smtp`-Mail entgegennimmt und sie in ein besonderes Spool-Verzeichnis schreibt. `Smapd` wird beim Bootvorgang gestartet und untersucht dieses Spool-Verzeichnis. Je nach Konfiguration übergibt `smapd` die Mail an den lokalen `sendmail` zum Weitertransport oder `sendmail` kann – via `cron` gesteuert – das Spool-Verzeichnis selbst leeren.

```
# Beispiel inetd.conf für smap Konfiguration
  smtp stream tcp nowait mail /usr/local/
  etc/smap smap
```

`Smap` wird von `inetd` gestartet. Zu beachten ist, daß dies nicht mit `root`-Rechten geschieht (hier `mail`).

```
# Beispiel rc* -Script für den Start von smapd
if[ i-x /usr/local/etc/smapd] ; then echo
   "Starting sendmail proxy"
   /usr/local/etc/smapd
fi
```

Der automatische Start von `sendmail` sollte deaktiviert werden.

```
#Beispiel netperm-table für smap and smapd
 smap, smapd:  userid mail
 smap, smapd:  directory /var/spool/smap
 smap, smapd:  baddir /var/spool/smap/bad
 smapd:        executable /usr/local/bin/smapd
 smapd:        sendmail /usr/sbin/sendmail
 smap:         permit-hosts .intern.de
 smap:         maxbytes 2500000
 smap:         maxrecip 250
 smap:         timeout 240
```

Sie können mit der gleichen Syntax wie bei den anderen Proxy-Clients die Zugriffskontrolle definieren. Weiterhin können Sie hier Maximalgrenzen für die Mailgröße und Adressatenanzahl festlegen.

Nachdem Sie die entsprechenden Verzeichnisse `/var/spool/smap` und `/var/spool/smap/bad` erstellt und sie mit `chown` dem Nutzer `mail` übereignet haben, können Sie mit folgendem Eintrag in die `crontab` das Programm `sendmail` in regelmäßigen Abständen über das Spool-Verzeichnis schicken:

```
0,30 * * * * /usr/lib/sendmail -q>/dev/null 2>&1
```

Dies sollen nur einige Beispiele für die Einsatzmöglichkeiten des Firewall Toolkit sein. Der einfache modulare Aufbau dieses Paketes ermöglicht es dem Administrator, die Arbeitsweise seines Bastion Host zu verstehen. Die Verfügbarkeit der Quellen läßt auch eine Anpassung an die eigenen Bedürfnisse zu. Für sicherheitsrelevante Anwendungen – was ein Bastion Host in einem Firewall-System nun mal ist – ist es von Vorteil, wenn man vor dem Einsatz einen Blick in die Quellen geworfen hat und die Software selber kompilieren konnte.

Der Einsatz des Firewall Toolkit unter Linux stellt – abgesehen von der Installations- und Konfigurationszeit – keinen großen Kostenfaktor dar. Ein schneller Pentium-Rechner mit ausreichendem Hauptspeicher und leistungsfähiger Netzwerkkarte ist heutzutage nicht sehr teuer. Mit freier Software kann man sehr wirkungsvoll Sicherheitsinstallationen realisieren. Natürlich bedarf es geschulter Mitarbeiter, die sich mit Betriebssystem und Software auskennen. Es sollte niemand ein Firewall-System betreiben, der davon keine Ahnung hat.

Alexander Geschonneck
geschonneck@rz.hu-berlin.de
http://www.hu-berlin.de/~h0271cbj/