

Einsatz von Firewall-Systemen zum Schutz von Netzwerken.

Aufbau eines Laborversuchs zur Absicherung eines Webservers mit der Paketfilter-Firewall „iptables“.

Projektarbeit vorgelegt von: Christian Heinrich und Mario Tönse im Fach „Netzwerksicherheit 1“

Problemdarstellung:

Ein Webserver soll unter Verwendung der Linux Firewall „iptables“ gesichert werden.

Der Server soll

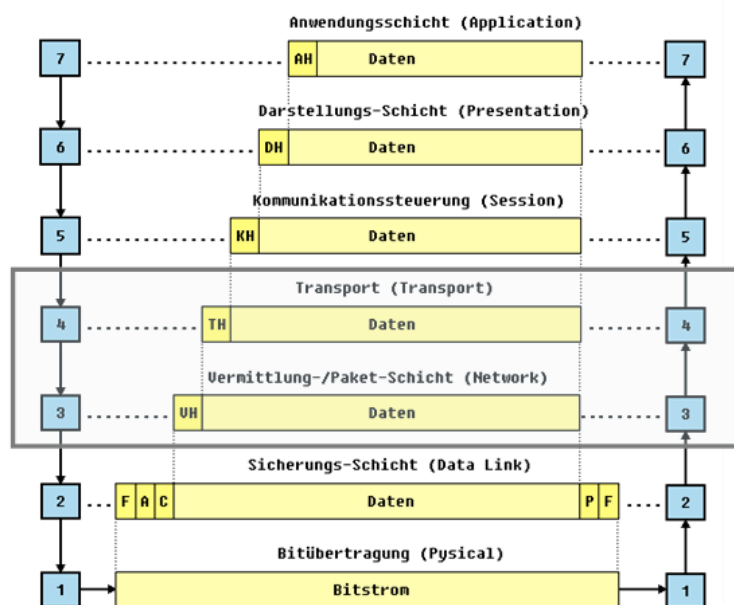
1. WEB-Server-Zugriffe aus dem Internet sowie aus dem internen Netz zulassen,
2. den SSH-Zugriff aus dem internen Netz erlauben und
3. Zeitsynchronisation mit einem spezifizierten Server durchführen.

Hierbei sind alle nicht notwendigen Ports zu sichern und die nicht autorisierten IP-Adressbereiche auszuschließen.

Theoretischer Ansatz:

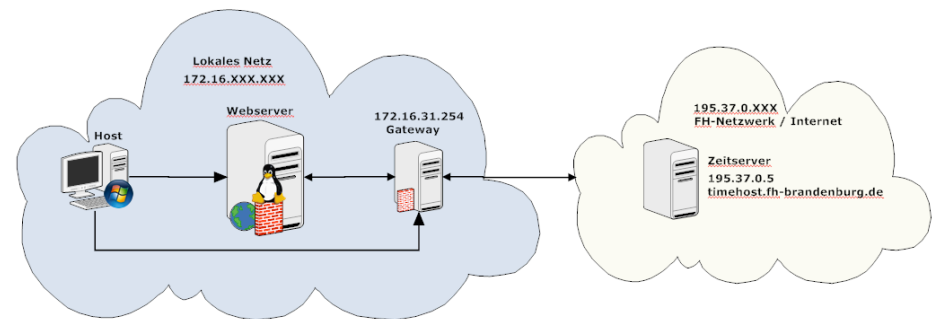
iptables

- ist ein Kommandozeilenprogramm zur Konfiguration einer Firewall auf Basis von Paketfiltern sowie zur Konfiguration von NAT.
- ist seit Kernel Version 2.4.0 fester Bestandteil von Linux-Betriebssystemen.
- arbeitet auf den OSI-Schichten 3 und 4 (Transport- und Vermittlungsschicht) und fällt in die Kategorie der Transport-Gateways.
- führt Paketfilterung auf Basis von IP-Adressen, Portnummern und Protokollen durch.



Versuchsaufbau:

Im internen Netzwerk (172.16.xxx.xxx) befindet sich ein Host (Windows Vista), ein virtueller Webserver (Ubuntu 8.10 unter VMWare) und ein Gateway (172.16.31.254), der die Verbindung zum externen Netzwerk herstellt. Der Host und der virtuelle Webserver laufen auf ein und demselben Laptop. Das externe Netzwerk repräsentiert das Internet und enthält den zugelassenen Zeitserver (195.37.0.5), mit dem der Webserver die Zeit synchronisieren darf.



Versuchsdurchführung:

Testdurchlauf vor und nach Aktivierung der Firewall:

- Ping zum Webserver erlaubt (icmp)?
- Webserver online (http)?
- SSH Zugriff auf den Server (ssh)
- Zeitsynchronisation (ntp)
- Port-Scan zum Test des Zugriffs auf den Server.

Aktivierung der Firewall

- Formulierung des Regelwerkes für den Webserver

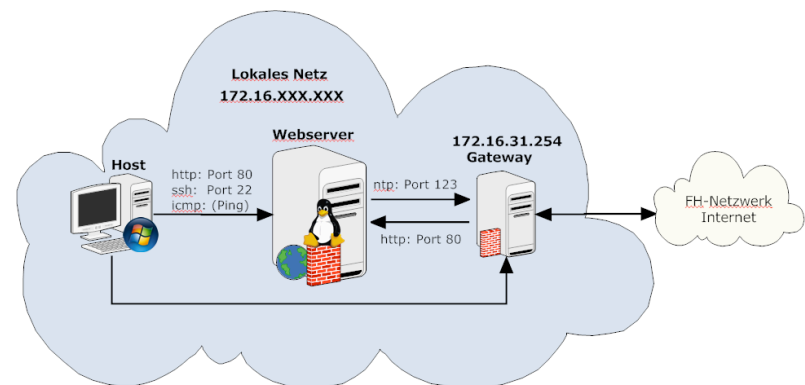
```
## Serverkonfiguration
## SSH zulassen (zweite Zeile bewirkt immer das der Rechner hier auch antworten kann)
## Hinzukommt der Punkt, dass manche Dienst auch über UDP kommunizieren müssen.
$IPTABLES -A INPUT -p tcp -s 172.16.0.0/16 --sport 1024:65535 --dport 22 -j ACCEPT;
$IPTABLES -A OUTPUT -p tcp -d 172.16.0.0/16 --sport 22 --dport 1024:65535 -j ACCEPT

## HTTP
$IPTABLES -A INPUT -p tcp --dport 80 --sport 1024: -j ACCEPT;
$IPTABLES -A OUTPUT -p tcp --sport 80 --dport 1024: -j ACCEPT;

## eingehenden Ping zulassen
$IPTABLES -A INPUT -p icmp --icmp-type echo-request -j ACCEPT;
$IPTABLES -A OUTPUT -p icmp --icmp-type echo-reply -j ACCEPT;

## NTP - Zeitsynchronisation
$IPTABLES -A INPUT -p udp -s 195.37.0.5 --sport 123 -j ACCEPT;
$IPTABLES -A OUTPUT -p udp -d 195.37.0.5 --dport 123 -j ACCEPT;
```

Auszug aus dem Regelwerk des Shell-Skripts.



Fazit:

Mit der vorliegenden Projektarbeit wurde gezeigt, dass das Betriebssystem Linux mit iptables über einen wirkungsvollen Paketfilter verfügt, um einen Webserver abzusichern. Diese Absicherung wurde durch eine einfache restriktive Konfiguration mit Protokollen, Portnummern und IP-Adressen umgesetzt. Durch die vorgestellten Regeln sind nur die geforderten Zugriffe per SSH, HTTP und NTP von und zu den definierten IP-Adressbereichen möglich.