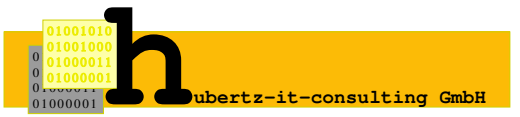
simple security policy editor

Johannes Hubertz

hubertz-it-consulting GmbH

FrOSCon

St. Augustin, 25.6.2006



Vorstellung: Johannes Hubertz

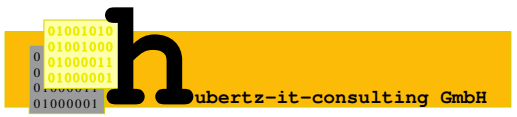
- 1954 in Köln-Lindenthal geboren
- 1973 Abitur in Köln-Mülheim
- bis 1980 E-technik RWTH und FH Aachen
- ab 1980 bei europ. IT-Hersteller
- ab 2002 bei europ. IT-Dienstleister
- seit 1973 Bundesanstalt THW Köln-Porz
- seit 2001 Segeln auf Salzwasser

Vorstellung: Johannes Hubertz

- 1986 Erstkontakt mit Unix (SCO-Xenix)
- 1994 Erstkontakt mit IP
- 1996 root@www.bull.de, root@www.bundestag.de, ...
- 1997 SSLeay, ipfwadm mit shell-scripts
- 1998 Ins Allerheiligste, iX 1/1998, Heise Verlag
- 1998 ipfwadm mit LPFC
- 1999 IT-Security Mgr. D-A-CH
- 2001 Gibraltar, FreeSwan, iptables ...
- 2001 Erste Gedanken zu sspe, Reinraum

Vorstellung: hubertz-it-consulting GmbH

- Gründung am 8.August 2005
- Sitz: Köln
- Geschäftsinhalt: Dienstleistungen im Umfeld der IT-Sicherheit
- Logo: Johannes Hubertz Certificate Authority als ASCII-7Bitmuster
- Diese Bitkombination findet sich in einigen 10000 Anwenderzertifikaten in der Seriennummer wieder :-)
- Wir sind käuflich 8-)



Vorstellung: simple security policy editor

- 2002 April: Online mit 2 Standorten
- 2002 6 Standorte voll vermascht
- 2003 erster Kunde mit eigenem sspe in Q1
- 2003 Veröffentlichung bei Sourceforge im März
- 2003 zweiter und dritter Kunde
- 2003 Trennung des internen Netzes vom Internet
- 2004 fünf Installationen, drei Personen

Vorstellung: Wichtigster Impuls

- Bellovin and Cheswick: Firewalls and Internet Security
- Fazit: keep it simple!
- Oder mit Einstein: So einfach wie möglich, aber nicht einfacher!

Übersicht: Ziele

- zentrale Administration mit minimalem Aufwand
- Faulheit stärkt die Glieder
- verteilte Firewall für beliebig viele Server und User-PC
- mehrere Standorte am Internet mit internen privaten Netzen
- voll vermaschtes IPSec-VPN mit FreeSwan

Übersicht: Randbedingungen

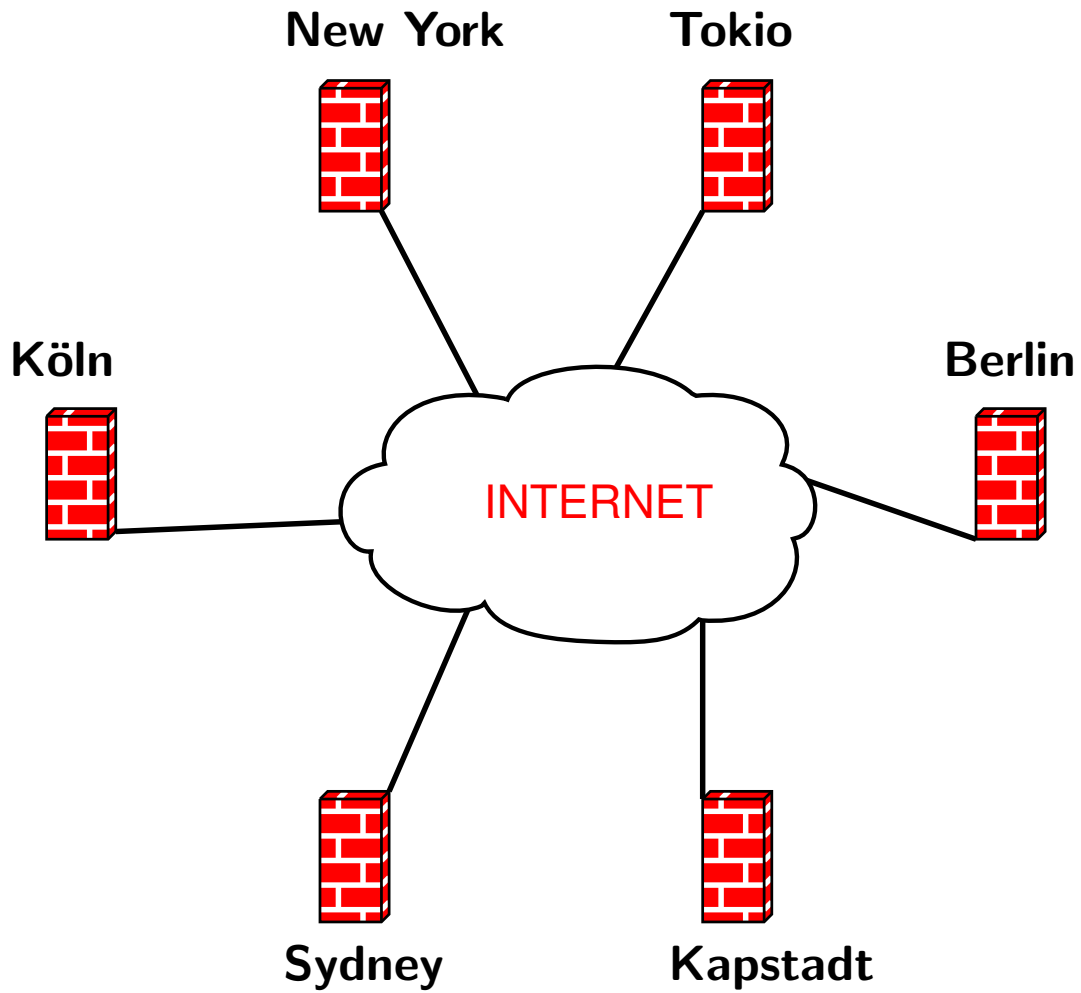
- Bash und Perl sichern einfache Nachvollziehbarkeit
- Verschlüsselung: IPSec und ssh, anerkannte, offene kryptographische Sicherheit
- Freie Software: Quellen mit überprüfbarer Sicherheit
- Freie Software: dauerhafte und zuverlässige KnowHow-Quelle

Übersicht: Lizenz

simple security policy editor
ist freie Software und unterliegt der
GNU General Public License

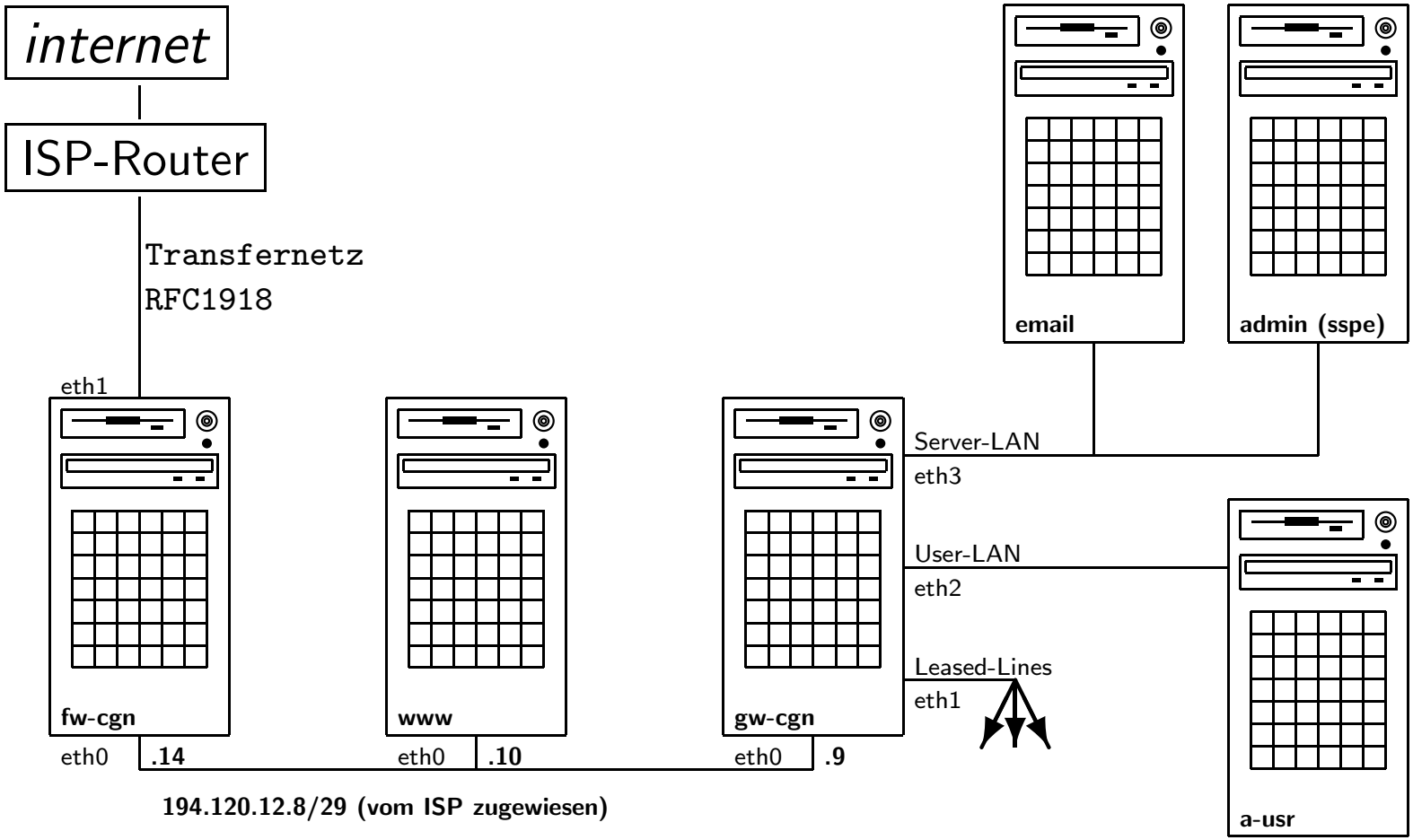
http://sspe.sourceforge.net

Übersicht: das Firmennetzwerk



6 Standorte an beliebigen Internet-Providern

Übersicht: ein typischer Firmenstandort



Der Standort des Admin-PC spielt keine Rolle.

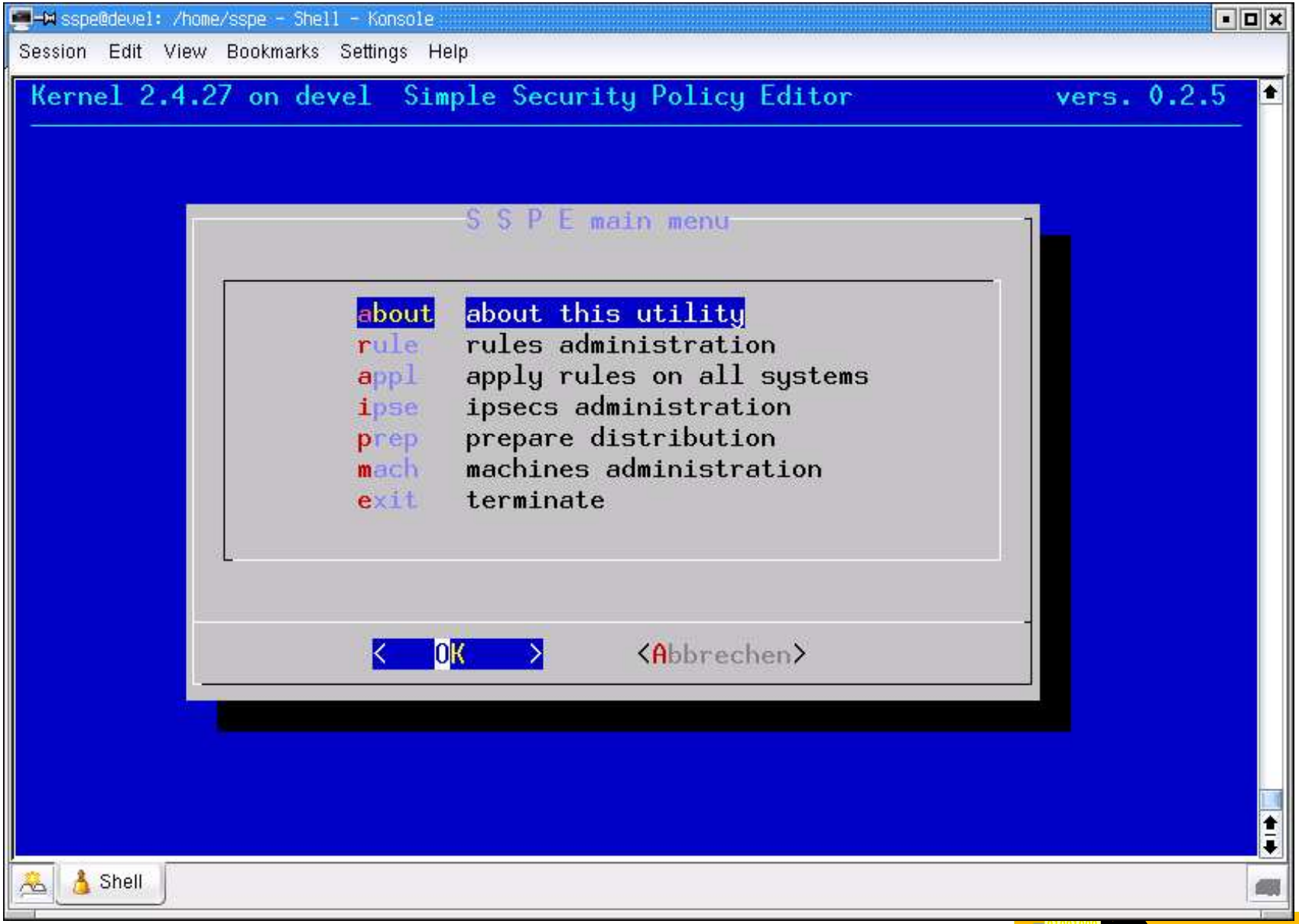
Firewall: Voraussetzungen

- Minimalsystem aus debian/stable und debfoster
- monolithischer Kernel
- root, sonst keine Benutzer
- keine unnötigen Services, nur ssh

Firewall: Entwurfskriterien

- Admins Traum: sowenig Arbeit wie möglich $\Longleftrightarrow$ Faulheit stärkt die Glieder
- zentrale Administration $\Rightarrow$ Konsistenz
- Fehler führen nicht zum Abbruch $\Rightarrow$ Verfügbarkeit
- Top-Down Softwareentwurf
- Inselumgebung für die ersten Versuche
- Linux^{TM}und Cisco^{TM}als erste Plattformen
- Dialog als Rahmen

Firewall: dialog



Hauptmenü

Firewall: hostnet

Definitionen in CIDR-Notation:

```
# File: hostnet
# Name       Address              # Comment
#
any          0.0.0.0/0            # the whole        internet
many         0.0.0.0/1            # lower half       internet
many         128.0.0.0/1          # upper half       internet
#
a-usr        192.168.1.126/32     # Alice            user-LAN
a-usr        192.168.1.125/32     # Bob              user-LAN
admin        192.168.1.193/32     # sspe-home        server-LAN
gw-cgn       192.168.1.222/32     # gateway cologne  server-LAN
gw-cgn-e     194.120.12.9/32      # gateway cologne  external
cgn-e        194.120.12.8/29      # cologne net      external
fw-cgn       194.120.12.14/32     # firewall cologne external
user-cgn     192.168.1.0/25       # users            user-LAN
cgn-net      192.168.1.0/24       # cgn completely   internal
```

Gruppierung erfolgt durch Namensgleichheit

Firewall: rules

```
# File: rules.admin
# Src      Dst       Dir Prot Port Action Options
#
a-usr      admin     1   tcp  ssh  accept INSEC
many       admin     1   tcp  ssh  deny
admin      gw-cgn    1   tcp  ssh  accept
#
```

```
Dir    =  [ 1 | 2 ]
Prot   =  [ ip | icmp | tcp | udp | esp | 0 ... 255 ]
Port   =  [ name | num = 0 ... 65535 | :num | num: | num1:num2 ]
Action =  [ accept | reject | deny ]
```

Firewall: Abhängigkeiten

Inhaltliche Abhängigkeiten der generierten Kommandos

- Host-, Netzdefinitionen
- Firewall Regelsatz
- Interfaces, Routingtabelle
- nathosts, privates
- Paketmangling-Dateien

Zeitliche Abhängigkeiten während der Generierung

- apply-options (sleep, wait)

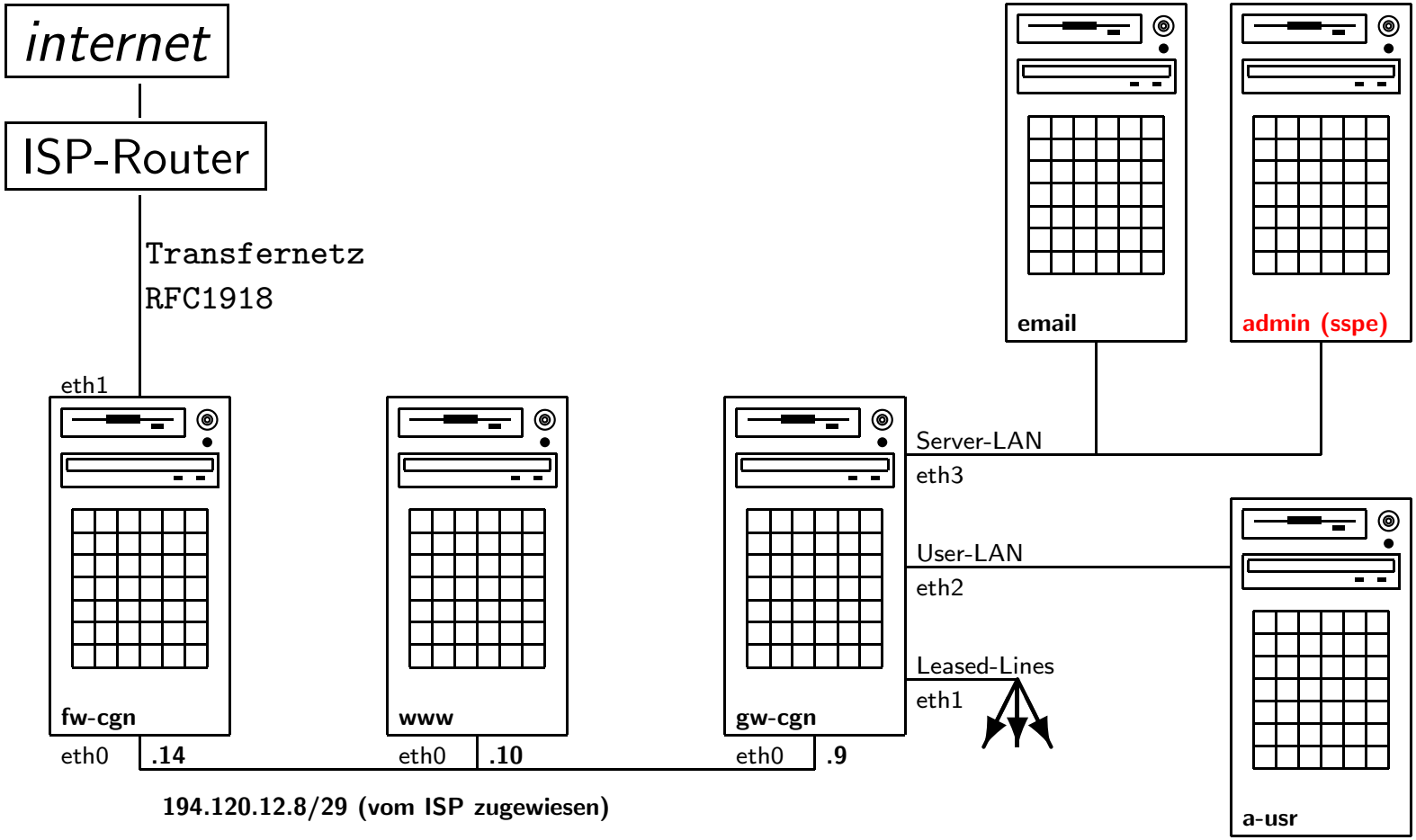
Firewall: Prolog

```
/sbin/iptables  -P INPUT   ACCEPT
/sbin/iptables  -P OUTPUT  ACCEPT
/sbin/iptables  -P FORWARD ACCEPT
/sbin/iptables  -F  >/dev/null 2>/dev/null
/sbin/iptables  -t nat -F  >/dev/null 2>/dev/null
/sbin/iptables  -F tcp__tab  >/dev/null 2>/dev/null
/sbin/iptables  -X tcp__tab  >/dev/null 2>/dev/null
/sbin/iptables  -F udp__tab  >/dev/null 2>/dev/null
/sbin/iptables  -X udp__tab  >/dev/null 2>/dev/null
/sbin/iptables  -F icmp_tab  >/dev/null 2>/dev/null
/sbin/iptables  -X icmp_tab  >/dev/null 2>/dev/null
/sbin/iptables  -F IPSEC  >/dev/null 2>/dev/null
/sbin/iptables  -X IPSEC  >/dev/null 2>/dev/null
/sbin/iptables  -F logdrop  >/dev/null 2>/dev/null
/sbin/iptables  -X logdrop  >/dev/null 2>/dev/null
/sbin/iptables  -N logdrop
/sbin/iptables  -A INPUT   -i lo -j ACCEPT
/sbin/iptables  -A OUTPUT  -o lo -j ACCEPT
/sbin/iptables  -A INPUT   -s 127.0.0.1/8 -j logdrop
/sbin/iptables  -A FORWARD -s 127.0.0.1/8 -j logdrop
/sbin/iptables  -N IPSEC
/sbin/iptables  -A FORWARD -p esp     -j IPSEC
/sbin/iptables  -A FORWARD -p ah      -j IPSEC
/sbin/iptables  -A FORWARD -p ipencap -j IPSEC
/sbin/iptables  -N tcp__tab
/sbin/iptables  -A FORWARD -p tcp  -j tcp__tab
/sbin/iptables  -N udp__tab
/sbin/iptables  -A FORWARD -p udp  -j udp__tab
/sbin/iptables  -N icmp_tab
/sbin/iptables  -A FORWARD -p icmp -j icmp_tab
```

Firewall: Epilog

```
/sbin/iptables  -A INPUT    -j logdrop
/sbin/iptables  -A OUTPUT   -j logdrop
/sbin/iptables  -A FORWARD  -j logdrop
/sbin/iptables  -A logdrop -j LOG --log-tcp-options --log-ip-options \
                   --log-level 7 --log-prefix "gw-cgn-dropped: "     \
                   -m limit --limit 3/second --limit-burst 6
/sbin/iptables  -P INPUT   DROP
/sbin/iptables  -P OUTPUT  DROP
/sbin/iptables  -P FORWARD DROP
```

Firewall: Generierung für Admin



firewall: iptables für admin

File: iptables-rules

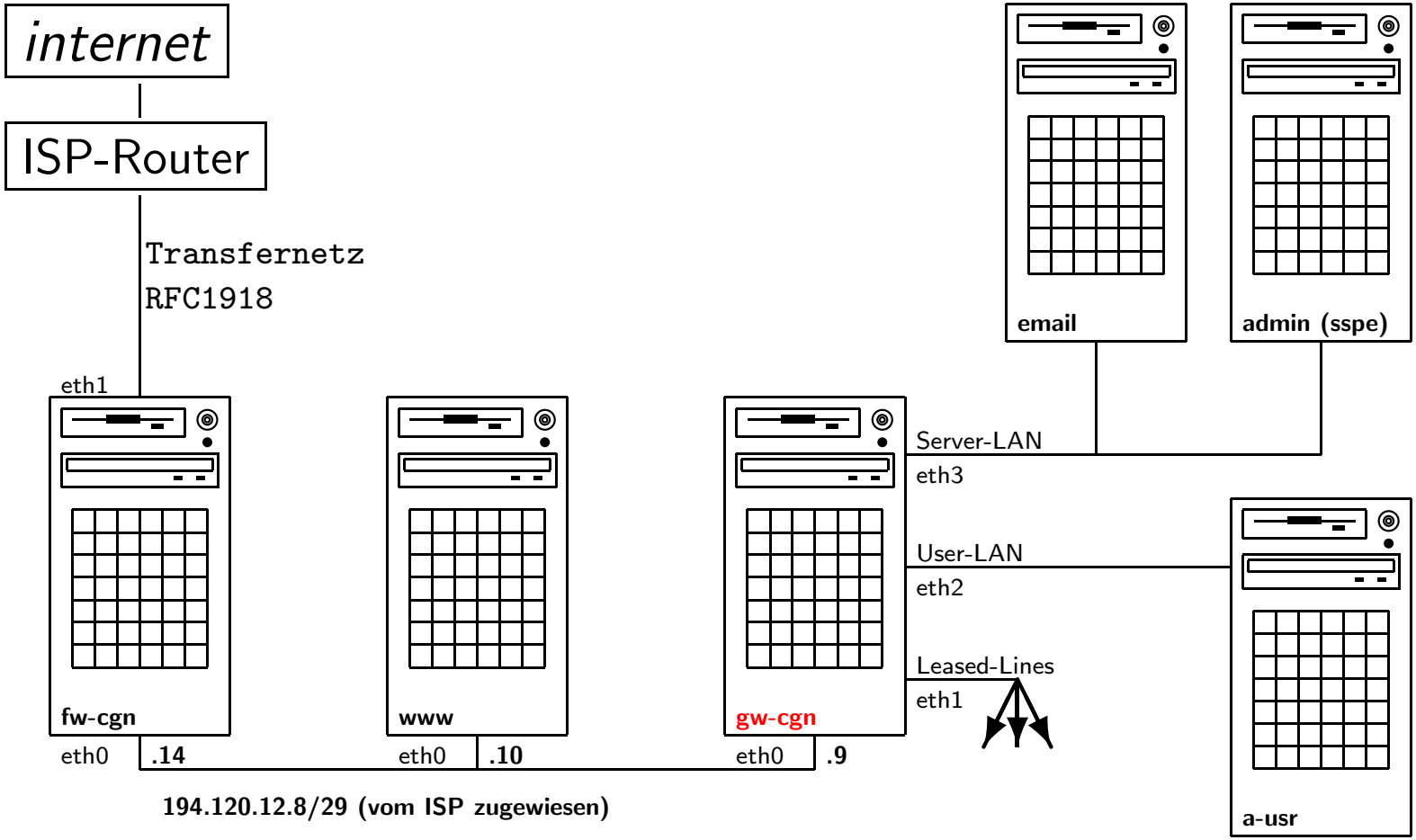
```
# File: iptables-rules for admin
/sbin/iptables -A INPUT  -i eth0                \
     -s 192.168.1.126/32 -d 192.168.1.193/32  \
     -p tcp --sport 0: --dport ssh            \
     -m state --state NEW,ESTABLISHED,RELATED \
     -j  ACCEPT

/sbin/iptables -A OUTPUT -o eth0                \
     -s 192.168.1.193/32 -d 192.168.1.126/32  \
     -p tcp --sport ssh --dport  0:           \
     -m state --state     ESTABLISHED,RELATED \
     -j  ACCEPT
```

Regel:

```
a-usr     admin   1  tcp  ssh  accept
```

Firewall: Regeln für gw-cgn



firewall: iptables für gw-cgn

File: iptables-rules

```
# File: iptables-rules for gw-cgn
/sbin/iptables  -A tcp__tab                          \
     -s 192.168.1.126/32 -d 192.168.1.193/32  \
     -p tcp --sport 0: --dport ssh                   \
     -m state --state NEW,ESTABLISHED,RELATED \
     -j  ACCEPT

/sbin/iptables  -A tcp__tab                          \
     -s 192.168.1.193/32 -d 192.168.1.126/32  \
     -p tcp                                          \
     -m state --state     ESTABLISHED,RELATED \
     -j  ACCEPT
```

Regel:

```
a-usr     admin   1  tcp  ssh  accept
```

Firewall: privates, nathosts

Private Netze untereinander: nie NAT!

```
# File: privates
172.16.0.0/16   # Berlin
172.17.0.0/16   # Cologne
172.21.0.0/16   # Tokio
```

Wer macht NAT für wen wo?

```
# File: nathosts
172.16.0.0/16          194.120.12.9          # Berlin
172.17.0.0/16          192.168.111.1         # Cologne
172.21.0.0/16          192.168.119.1         # Tokio
```

Firewall: packet-mangling

packet-mangling:

```
#
# experimental
#
# FILE: mangle-start for gw-cgn
#
# force icmp to minimize-delay
#
/sbin/iptables -t mangle -F
#
# 0x10 = minimize Delay!
/sbin/iptables -t mangle -A PREROUTING -p icmp -j TOS --set-tos 0x10
#
```

Jedes Shellkommando kann benutzt werden!
Wesentliche Einschränkung: ausschließlich nach dem Prolog bzw. vor dem Epilog einzubinden.

Firewall: konzeptionelles

- Gemeinsame Regeln verschiedener Maschinen: symbolic Links
- Regelsuche: ausschließlich im lokalen Verzeichnis
- Splittung des Regelsatzes: admin, head, ipsec, local, users, tail
- Regeldatei kann, muß aber nicht vorhanden sein
- Optionen in Regeln: LOG, NONAT, NOIF, DNS, FTP, SYSL, NTP, IPSEC, VNC, ...

Firewall: Einschränkungen

- IPSec ausschließlich an Gateways auf eth0
- Fehler werden erst angezeigt, wenn alle Ziele fertig sind

Firewall: Logging

- syslog oder syslog-ng lokal oder remote
- Drop-Regeln vermindern Logaufkommen
- Logging auf Accept-Regel hilft entstören

Firewall: VPN - Implikationen

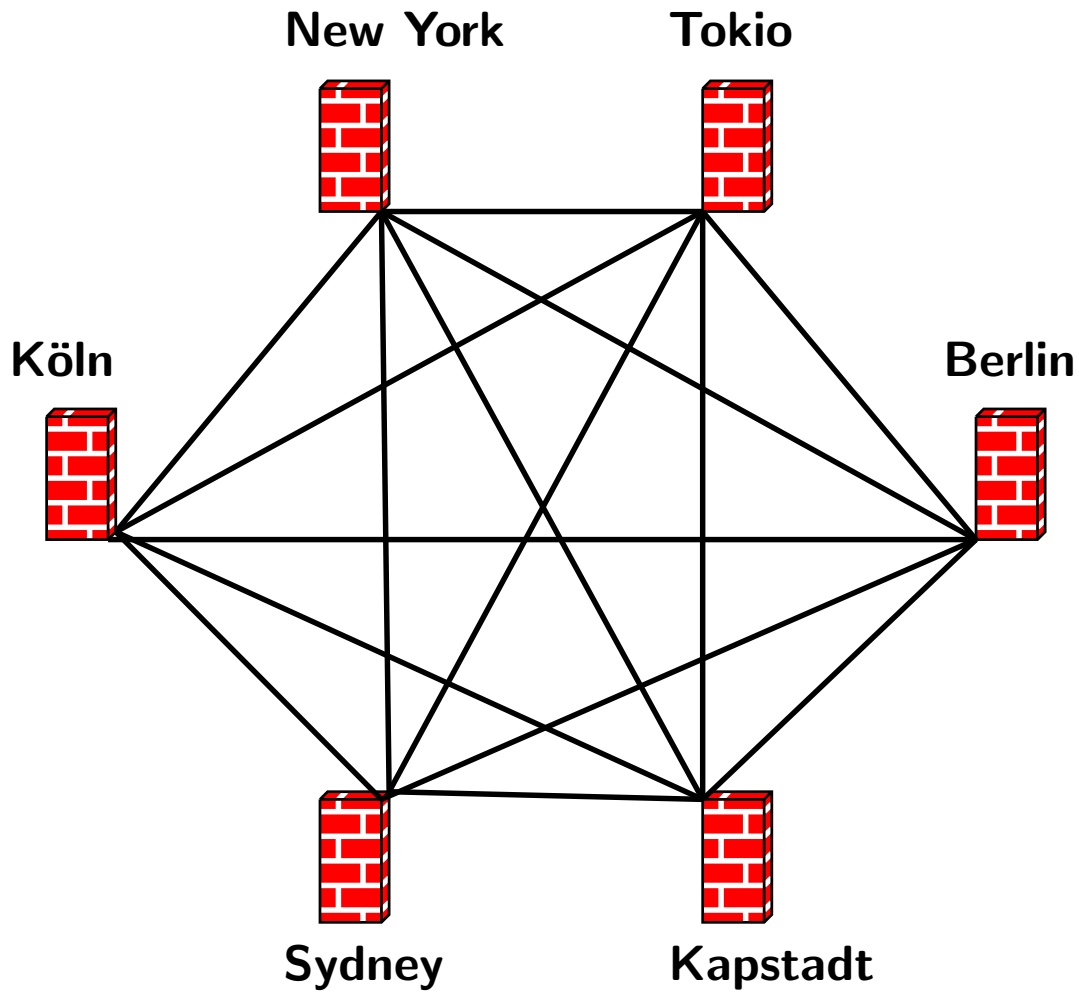
ssh und IPSec

- ausschliesslich ssh zur Administration
- IPSec und ssh nicht wechselseitig abhängig
- ssh durch IPSec nur zu internen Maschinen ohne IPSec
- IPSec verändert Routing, hat also Einfluß auf Generierung!

Erkenntnis:

Paranoid zu sein bedeutet nicht, daß keiner hinter einem her wäre!

VPN: das Firmennetzwerk



6 Standorte an beliebigen Internet-Providern
per IPSec voll vermascht mit $S * (S - 1) = 30\,Tunneln$

VPN: erster Denkansatz

- Gleiche ipsec.conf an allen Standorten, d.h.
  pluto wählt die passenden connections aus
- Voraussetzung: alle sind gleichzeitig erreichbar
- Zeitsteuerung manuell, Neuladen per cron und ntp synchron sinnvoll
- Overhead für Änderungen ist erträglich,
  30 Sekunden downtime bei der Neukonfiguration
- Konfiguration und PreSharedKeys aus sspe-konfig: ipsecs
- voll vermaschtes Netz, singuläre Standort-Anbindung zusätzlich möglich
- Verteilung mit scp: ipsec.conf.new
- supervisor-script prüft und aktiviert Konfiguration

VPN: ipsec-supervisor mit Vermaschung

Script am vernetzten Standort:

```
#!/bin/bash
if [ -f /etc/ipsec.secrets.new ] ; then
        if [ -f /etc/ipsec.conf.const ] ; then
                cat /etc/ipsec.conf.const >> /etc/ipsec.conf
        fi
        mv /etc/ipsec.secrets.new /etc/ipsec.secrets
        /etc/init.d/ipsec restart
fi
```

ipsec.conf und ipsec.secrets.new werden gemeinsam übertragen
ipsec.conf.const enthält die Konfiguration für singuläre Anbindungen und wird manuell einmal erstellt und auf die beiden Endpunkte verteilt

crontab:

```
* * * * * /root/bin/ipsec-supervisor >/dev/null 2>/dev/null
```

VPN: ipsec-supervisor ohne Vermaschung

Script am Standort mit singulärer Anbindung:

```
#!/bin/bash
if [ -f /etc/ipsec.secrets.new ] ; then
        if [ -f /etc/ipsec.conf.const ] ; then
                cat /etc/ipsec.conf.const  > /etc/ipsec.conf
        fi
        mv /etc/ipsec.secrets.new /etc/ipsec.secrets
        /etc/init.d/ipsec restart
fi
```

ipsec.conf und ipsec.secrets.new werden gemeinsam übertragen
ipsec.conf.const enthält die Konfiguration für singuläre Anbindungen und wird manuell einmal erstellt und auf die beiden Endpunkte verteilt

crontab:

```
* * * * * /root/bin/ipsec-supervisor >/dev/null 2>/dev/null
```

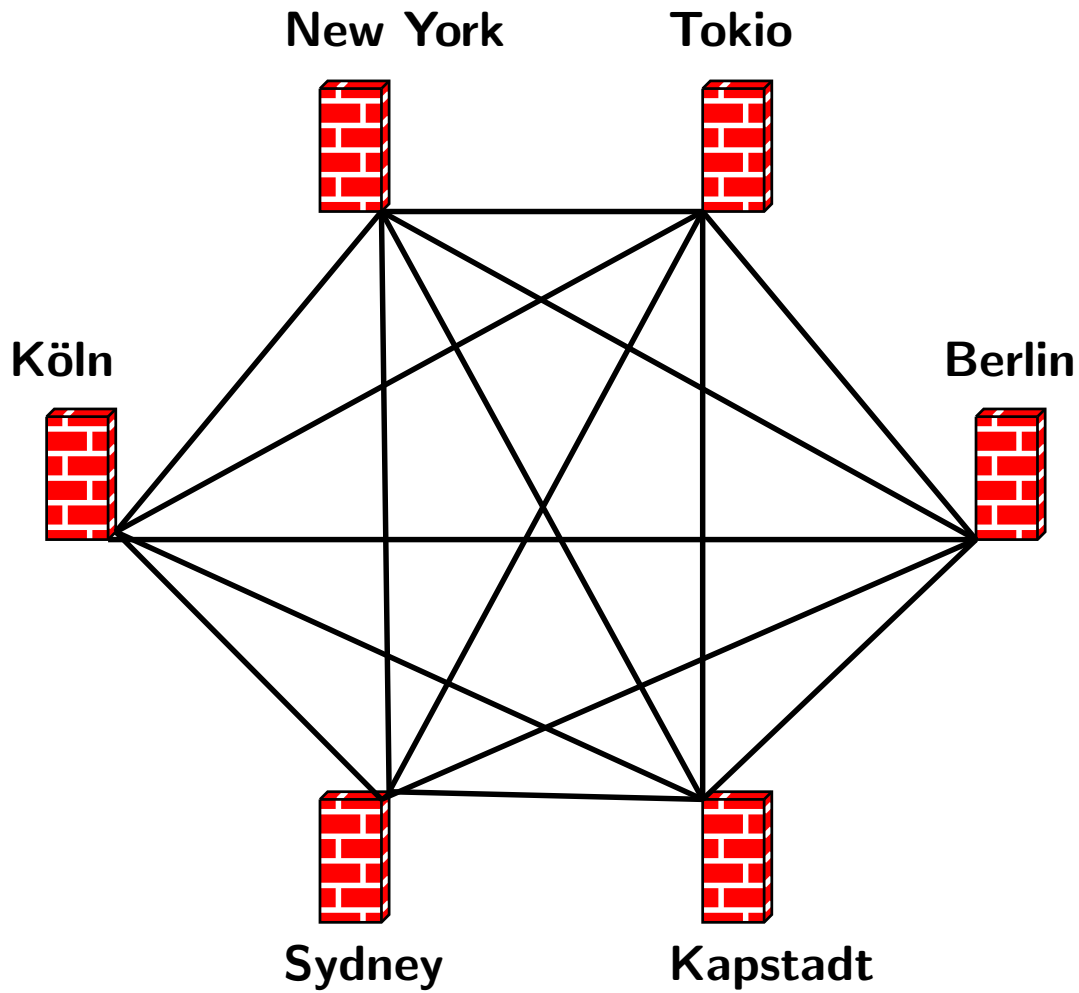
VPN: ipsecs Konfigurationsdatei

keep it simple:

```
# loc.   gateway          next-Hop         subnet
bln      172.22.0.41      172.22.0.46      10.11.48.0/21
cgn      172.22.0.25      172.22.0.30      10.11.40.0/21
nyc      172.22.0.65      172.22.0.70      10.11.4.0/22
sdy      172.22.0.17      172.22.0.22      10.0.0.0/8
kap      172.22.0.9       172.22.0.14      10.11.56.0/21
tok      172.22.0.1       172.22.0.6       10.11.16.0/21
to2      172.22.0.1       172.22.0.6       10.11.80.0/21
```

Hieraus werden alle ipsec.conf und ipsec.secrets generiert

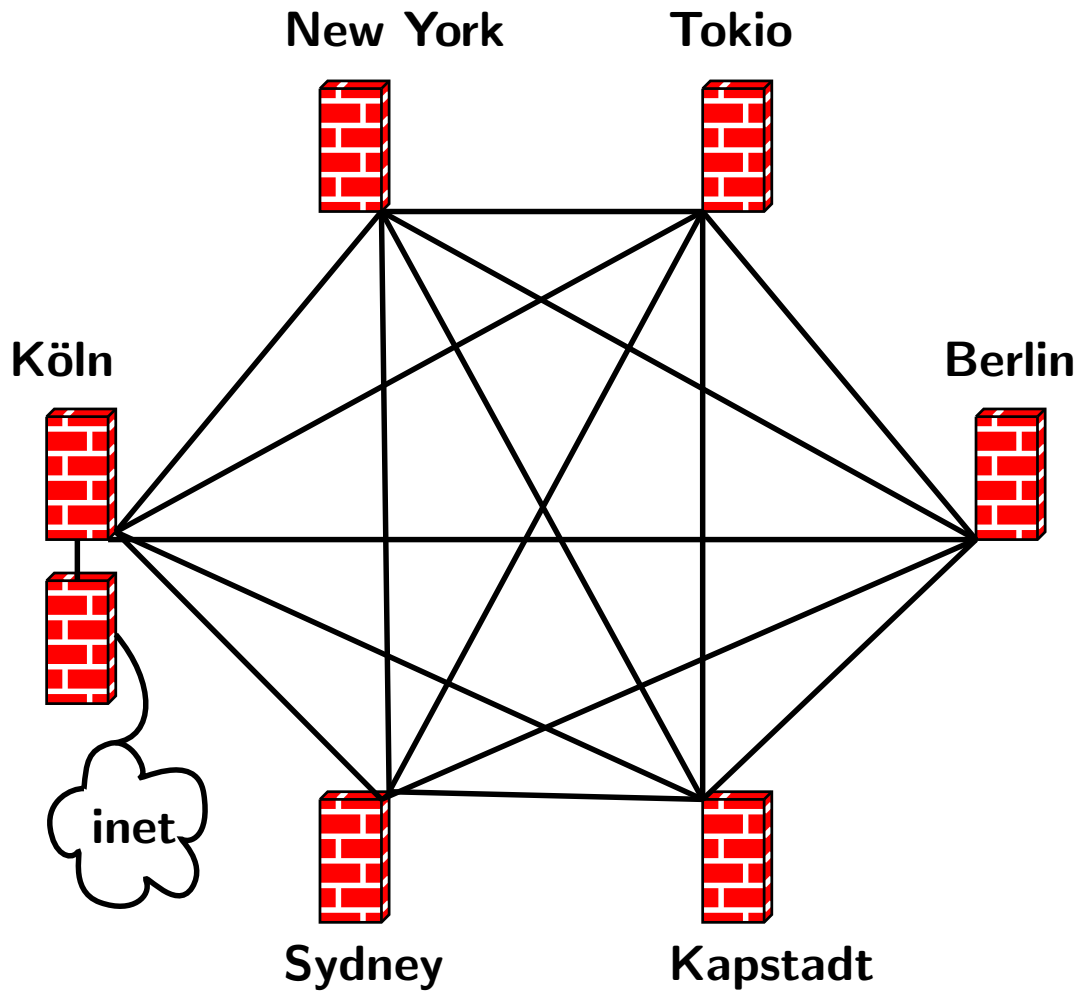
VPN: das Firmennetzwerk vor dem Umbau



6 Standorte an beliebigen Internet-Providern

IPSec voll vermascht mit $S * (S - 1) = 30\,Tunneln$

VPN: das Firmennetzwerk nach dem Umbau



1 ISP + 6 Standorte an einem ISP-MPLS-VPN,
IPSec voll vermascht mit $S * (S - 1) + 5 * 12 = 90$ *Tunneln*

VPN: zweiter Denkansatz

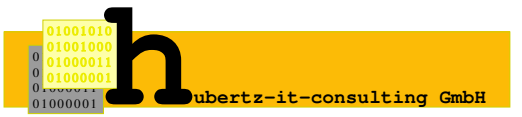
- jedes VPN-GW ist anders
- exakte Konfiguration pro Gateway erzeugen
  pluto muß nicht mehr auswählen
- Routen des Internet per IPSec möglich
- Routinglücke für ssh zur Administration
- Overhead für Änderungen bleibt erträglich,
  36 Sekunden downtime trotz mehrfacher Anzahl Tunnel

VPN: ipsecs Konfigurationsdatei

keep it simple:

```
# loc.  gateway          next-Hop         subnet
bln     172.22.0.41      172.22.0.46      10.11.48.0/21
cgn     172.22.0.25      172.22.0.30      10.11.40.0/21
nyc     172.22.0.65      172.22.0.70      10.11.4.0/22
sdy     172.22.0.17      172.22.0.22      10.0.0.0/8
kap     172.22.0.9       172.22.0.14      10.11.56.0/21
tok     172.22.0.1       172.22.0.6       10.11.16.0/21
to2     172.22.0.1       172.22.0.6       10.11.80.0/21
I01     172.22.0.25      172.22.0.30      0.0.0.0/1
I02     172.22.0.25      172.22.0.30      128.0.0.0/3
I03     172.22.0.25      172.22.0.30      160.0.0.0/5
I04     172.22.0.25      172.22.0.30      168.0.0.0/6
I05     172.22.0.25      172.22.0.30      172.0.0.0/12
### !!! never open next line or gateways will be lost !!!!
### !!!Ixx 172.22.0.25   172.22.0.30      172.16.0.0/12 !!!
I06     172.22.0.25      172.22.0.30      172.32.0.0/11
I07     172.22.0.25      172.22.0.30      172.64.0.0/10
I08     172.22.0.25      172.22.0.30      172.128.0.0/9
I09     172.22.0.25      172.22.0.30      173.0.0.0/8
I10     172.22.0.25      172.22.0.30      174.0.0.0/7
I11     172.22.0.25      172.22.0.30      176.0.0.0/4
I12     172.22.0.25      172.22.0.30      192.0.0.0/3
```

Hieraus werden alle ipsec.conf und ipsec.secrets generiert



VPN: weitere Möglichkeiten

- Handlungsreisende (roadwarrior) mit X.509-Authentisierung
- `vpndialer.sf.net` für IPSec vom beliebigen M$-PC
  (freie Software von Thomas Kriener)
- Sperrliste für einzelne Clients: CRL der PKI
- L2TP (durch vpndialer initiiert) durch IPSec zur Änderung des Routings im PC

Betrieb: Erfahrungen

- Produktionseinsatz ab April 2002
  Verbeserungen sind im Changelog gelistet
- Mehrerer Kunden und interner Bedarf gedeckt
- ca. 50 Maschinen mit iptables gesichert
- einige hundert Anwender-PC geschützt
- Kosten drastisch minimiert gegenüber kommerzieller Firewall-Lösung

Betrieb: Erfahrungen

- Debian macht security-fixes einfach
- RedHat funktioniert auch, SuSe vermutlich ebenso
- Scriptänderungen einfach machbar
- Erweiterungen
- z.B. HA, dyn.Routing, ...

Dokumentation

Eine Sicherheitsarchitektur ist nur so gut wie ihre Dokumentation
Bei sspe soll sie mit LaTeX aus der laufenden Konfiguration erzeugt werden:

- Übersicht der Netzwerkarchitektur
  einmalig zu zeichnendes Bild(dia),
  pstricks mit Referenzen (Seitenzahlen der Geräte-Seiten)
- Konfiguration der einzelnen Maschinen
  Interfaces, Routing, ...
- Firewall Definitionen und Regeln
- VPN Konfiguration
- Server-configs, z.B. bind, apache, squid, ...
- Zertifizierungsstelle, Zertifikate
- Geplant ist eine weitgehende Vollständigkeit, d.h.
  alles sollte aus der Dokumentation wiederherstellbar sein

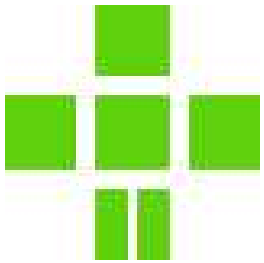
Andere Plattform

IP-Filterung sollte in sspe nicht auf iptables beschränkt bleiben:

- Cisco
- OpenBSD
- Solaris
- Ideen, Vorschläge und weitere Entwickler erwünscht!

Danke

Ich bedanke mich für die Aufmerksamkeit. Sie finden mich gleich am Stand der



Free Software Foundation Europe fellows

Frohes Schaffen

Johannes Hubertz