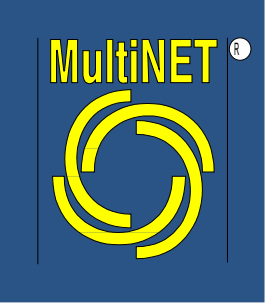


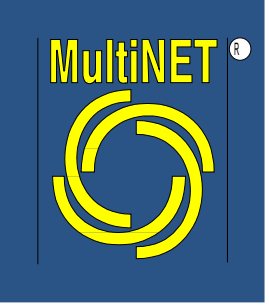
Eine hochverfügbare Firewall mit `iptables` und `fwbuilder`

Secure Linux Administration Conference, 11. Dec 2008

Dr. Michael Schwartzkopff

Eine einfache Firewall

- Eine einfache Firewall mit Linux ist schnell eingerichtet
  - 3 Schnittstellen (extern, intern, DMZ)
  - Prozessorleistung, Speicher und Festplatten sind heute fast ohne Bedeutung
- Probleme:
  - Hochverfügbarkeit: aktiv / passiv Cluster mit `heartbeat`
  - Synchronisieren des State Tables → kein Abreißen der Verbindungen im Fehlerfall
  - einfache Administration als Mittelweg zwischen Webinterface und eigenen Scripten



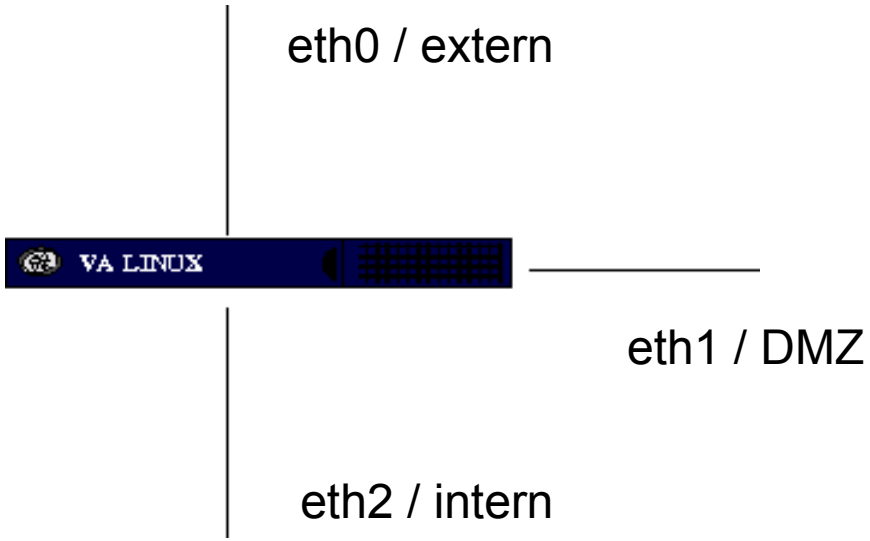
Administration mit fwbuilder

- Erzeugung eines Regelsatzes über ein Webinterface oder distributionseigene Werkzeuge zu unflexibel.
- Die Scripte selbst zu schreiben ist auf die Dauer zu umständlich. Besonders bei vielen Systemen.
- Lösung: `fwbuilder`
  - flexibel (!)
  - skaliert angemessen
  - GUI bietet Überblick

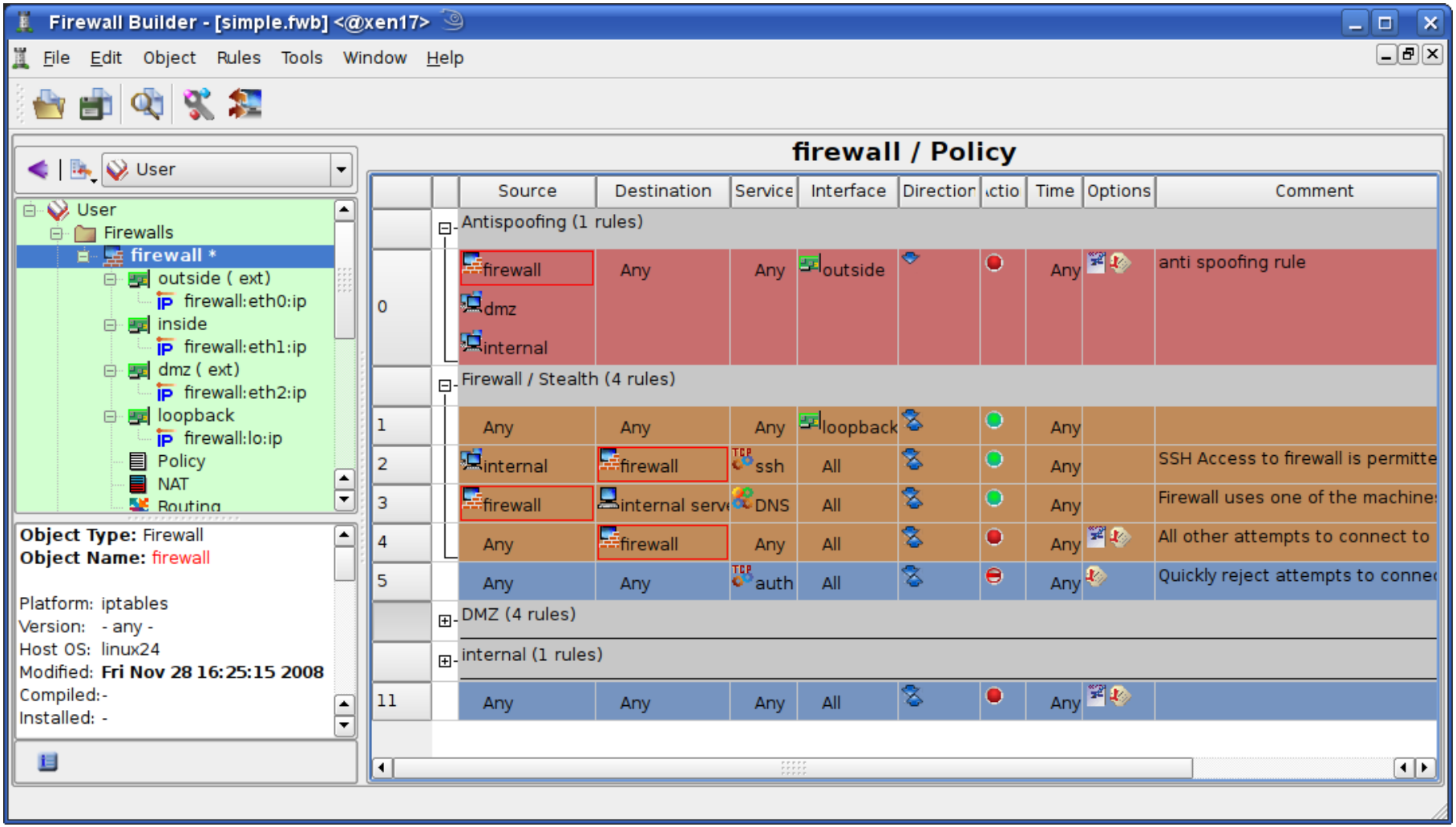
`fwbuilder`: Fähigkeiten

- Gut nutzbare GUI, die hinter kommerziellen Lösungen nicht zurückbleibt.
- Skaliert durch 2-Tier Architektur, d.h. Erzeugung des Regelsatzes und Durchsetzung sind getrennt.
- Die grafische Darstellung des Regelsatzes bietet einen bessern Überblick als Scripte.
- Automatische Entdeckung von Regelkonflikten.
- viel mehr, besonders in Version 3.

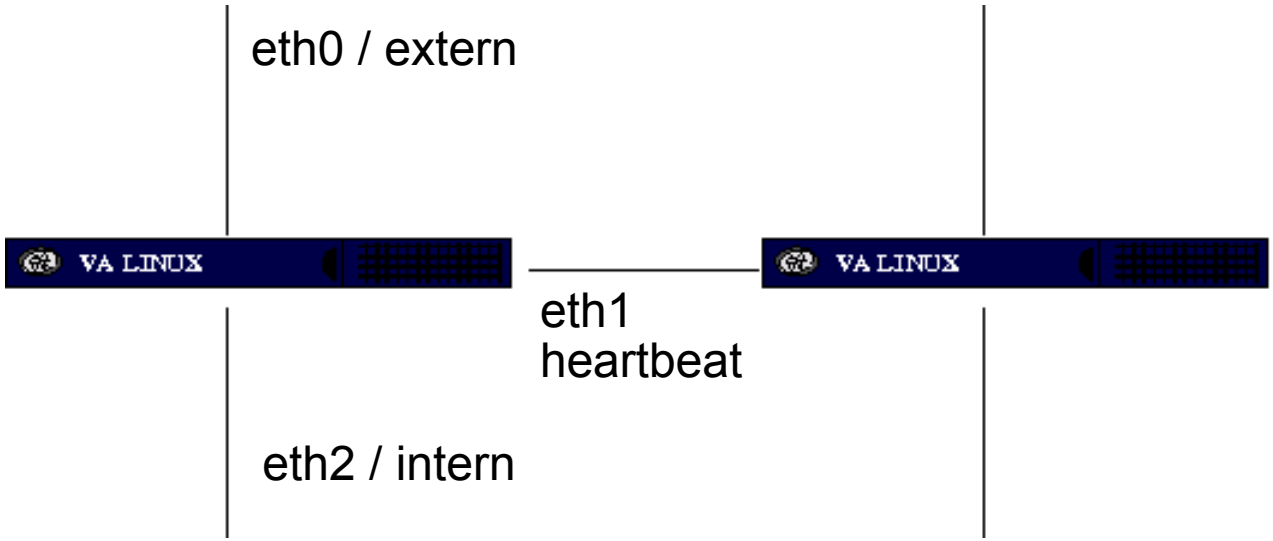
einfache Firewall: Schema



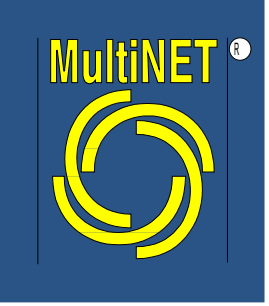
Regelsatz in fwbuilder



bessere Firewall: HA



DMZ optional



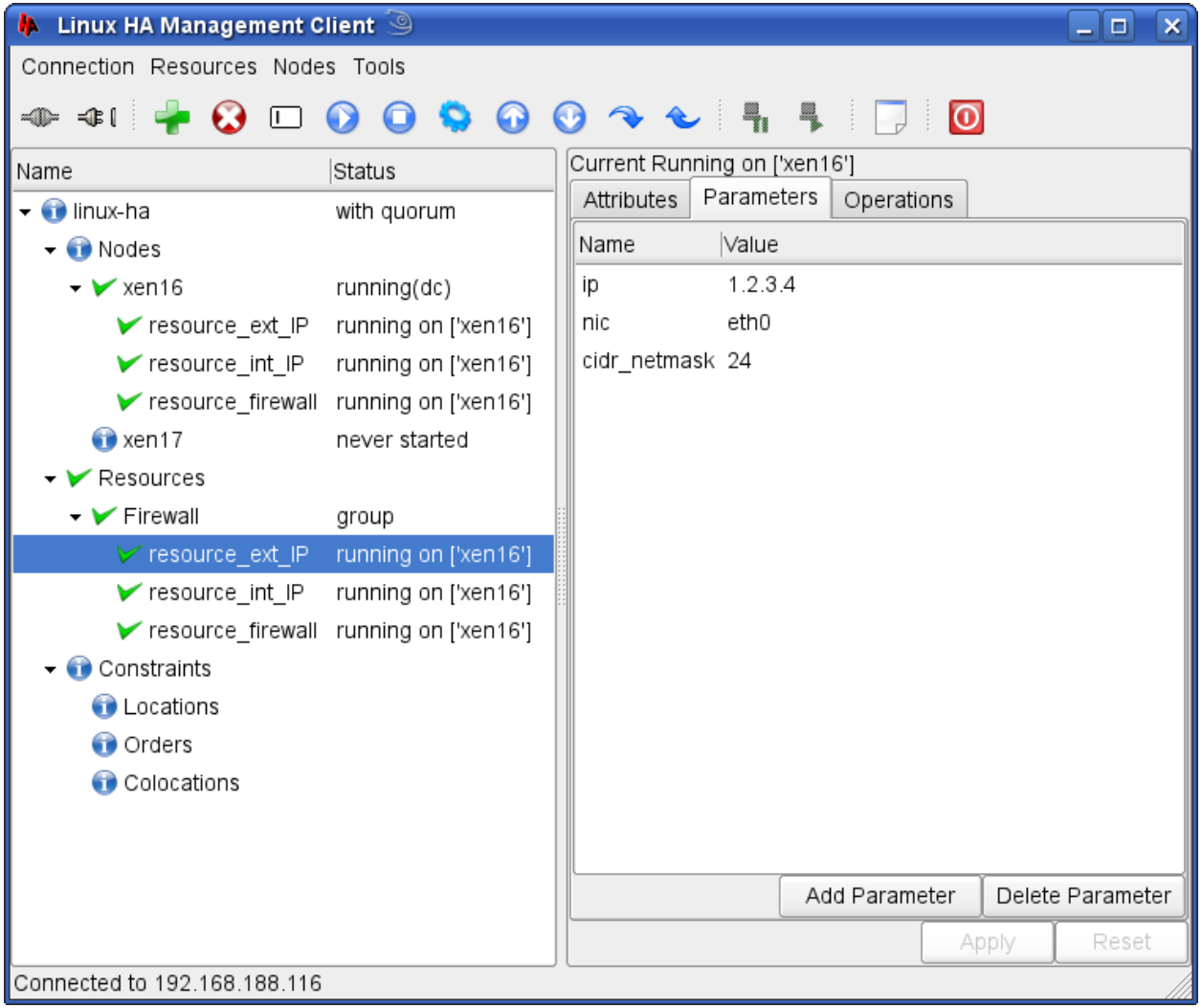
Aufbau einer HA Firewall

- Bei Ausfall der aktiven Firewall übernimmt das passive System.
- Die Firewall als Gesamtsystem ist sofort wieder einsatzbereit.
- `heartbeat` *verwaltet virtuelle Cluster-Adressen*. Über diese läuft das Routing.
- `heartbeat` *kann* auch `ip_forward` verwalten. Muss aber nicht, weil keine Pakete an die passive Firewall weitergeleitet werden.

Ressourcen in `heartbeat`

- IP Adressen als `heartbeat`-eigene (OCF) Ressourcen.
  - Gute Integration in `heartbeat`
  - Ressource: `IPaddr2`
- Firewall als init-Script:
  - verwaltet `ip_forward`
  - ruft beim Start das Firewallscript auf
  - `status` überprüft den Zustand von `ip_forward`.
- Gruppierung der Ressourcen:
  - Co-Lokation auf einem Knoten und evtl. Anordnung

HA Firewall in der GUI von heartbeat



`heartbeat` Kommunikation zulassen!

```
iptables -I INPUT -i eth1 --dport 694 -j ACCEPT

iptables -I OUTPUT -o eth1 --dport 694 -j ACCEPT
```

- Merken und später in den Regelsatz einbauen!

Synchronisation des Status beim Failover

- Beim Failover reißen alle Verbindungen ab, da der Connection Table zwischen den Knoten des Clusters nicht abgeglichen wird.
- Frühere Lösung `ct_sync` suboptimal.
- Jetzt: `conntrackd`
- Ist z.B. in debian/testing (aka lenny) enthalten.

```
# apt-get install conntrackd
```

conntrackd

- conntrackd verwaltet zwei Zwischenspeicher (einfacher Modus):
  - intern: gibt den connection table des Kernel wieder. Dieser wird per Multicast veröffentlicht.
  - extern: Meldungen von anderen Knoten werden hier zwischengespeichert, aber noch nicht an den Kernel weitergegeben
- Beim Umschalten wird der der Inhalt des externen Caches in die Tabelle des Kernels synchronisiert.
- Es gibt auch einen Modus, der direkt synchronisiert. Damit sind asymmetrische Systeme möglich.

/etc/conntrackd.conf (I)

```
Sync {
        Mode FTFW { ... }
        Multicast {
                IPv4_address 225.0.0.50
                IPv4_interface 192.168.10.2
                Interface eth1
                Group 3780
        }
}
(...)
IgnoreTrafficFor {
        IPv4_address 127.0.0.1
        (...)       # all dedicated and cluster IP addresses
}
IgnoreProtocol {
        UDP
        ICMP
        IGMP
        VRRP
}
```

Kommunikation für `conntrackd`

```
iptables -I INPUT -i eth1 -d 225.0.0.50 -j ACCEPT

iptables -I OUTPUT -o eth1 -d 225.0.0.50 -j ACCEPT
```

- Merken und später in den Regelsatz einbauen!

/etc/init.d/firewall anpassen

- Bisher:
  - *start* / *stop* lädt bzw. löscht Regelsatz.
  - ip_forward wird mit *start* / *stop* gesetzt bzw. gelöscht.
  - *status* fragt ip_forward ab.

- Zusätzlich:
  - conntrackd Befehle (-c, -f, -R) bei *start*
  - conntrackd Befehl -n bei *stop*.

System fertig!

- Alle Voraussetzungen für eine HA Firewall fertig!
- Im Betrieb wird der connection table der aktiven Firewall immer in den Cache des passiven Systems übertragen.
- Download geht weiter, auch wenn die aktuell aktive Firewall abstürzt:
  - `heartbeat` erkennt den Fehler.
  - Die virtuellen Adressen werden auf den bisher passiven Knoten verschoben.
  - Dort wird auch der Regelsatz aktiviert und der connection table übernommen.

Fehlt: Integration in `fwbuilder`

- `fwbuilder` ist aktuell nicht für Cluster ausgelegt.
- Version 4 soll das auch können.
- Gesuchte Lösung soll:
  - Elegant, symmetrisch und simpel sein.
  - Einfach zu bedienen sein, d.h. Änderungen des Regelsatzes sollen nur einmal eingetragen werden müssen.

  → *Keine* zwei komplett unterschiedlichen Regelsätze für beide Knoten.

Zwei Varianten

- Beide Knoten sind unterschiedlich (dedizierte IP Adressen)!
- Mir sind zwei Varianten der Darstellung des Regelsatzes in `fwbuilder` eingefallen:

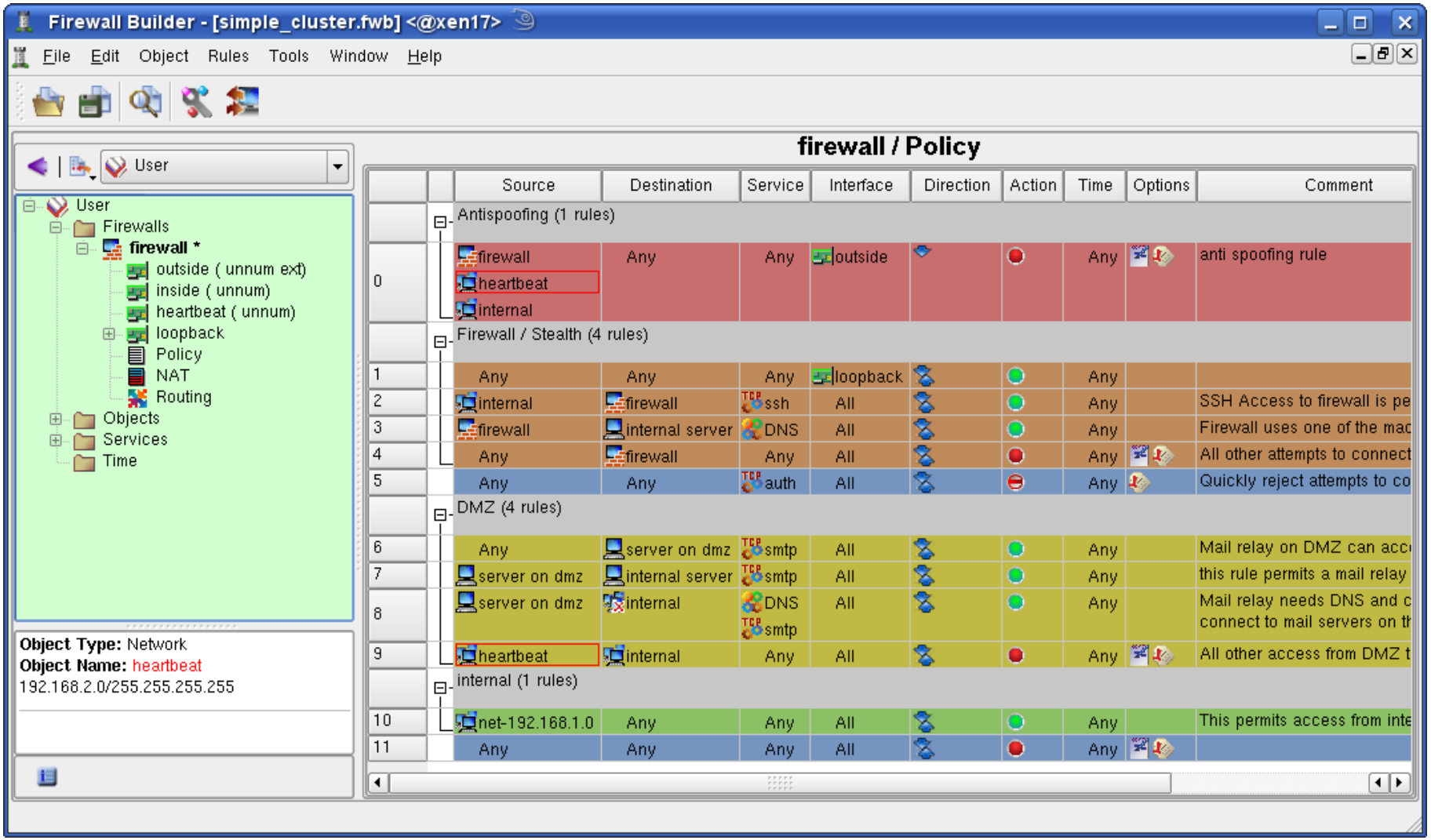
1) Cluster als eine Firewall, ohne IP Adressen der Schnittstellen.

2) Cluster als zwei Firewalls mit Referenz auf einen gemeinsamen Regelsatz.

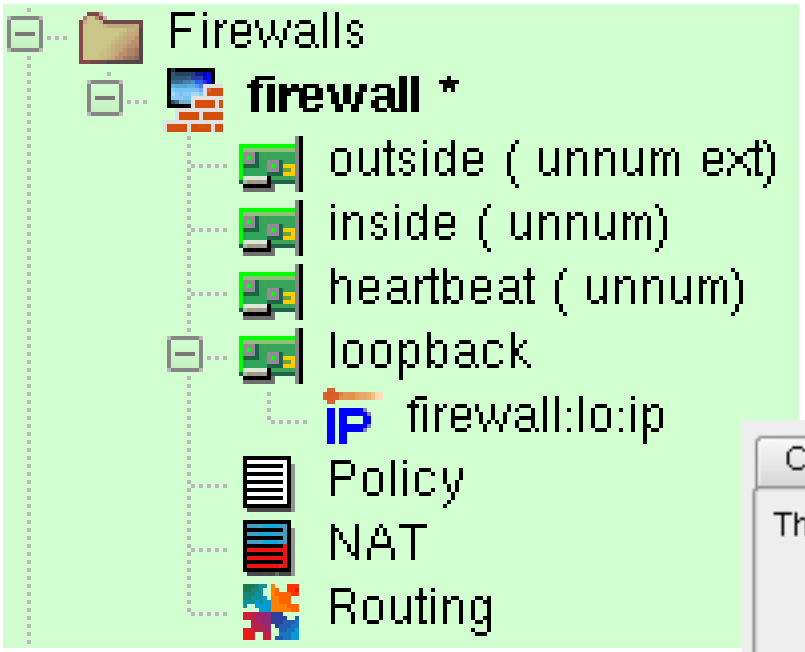
Lösung: Eine Firewall

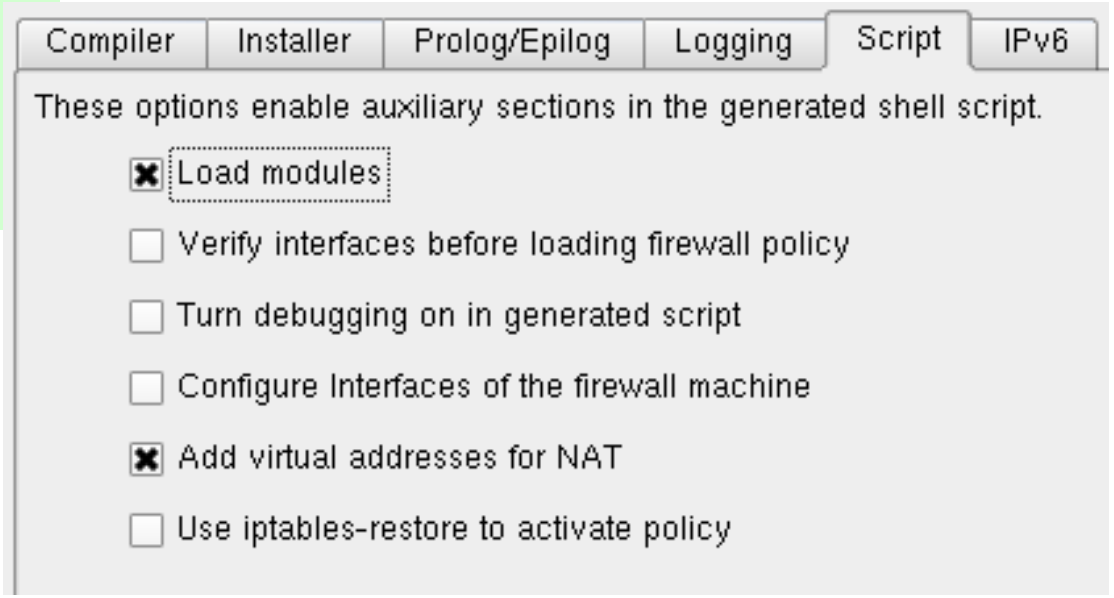
- Cluster wird als eine Firewall definiert.
- Interface erhalten *keine* IP Adressen in `fwbuilder`. Adressen werden durch OS bzw. `heartbeat` verwaltet.
- Antispoofing Regeln durch Objekte, Schnittstellen und In- bzw. Out – Regeln.
- Installation des Regelsatzes auf die Clusteradresse.
- Verteilung des Regelsatzes durch cronjob:
  `rsync -tu -e ssh /etc/firewall.fw root@node:/etc/`

Darstellung in `fwbuilder`



Details der Einzelfirewall



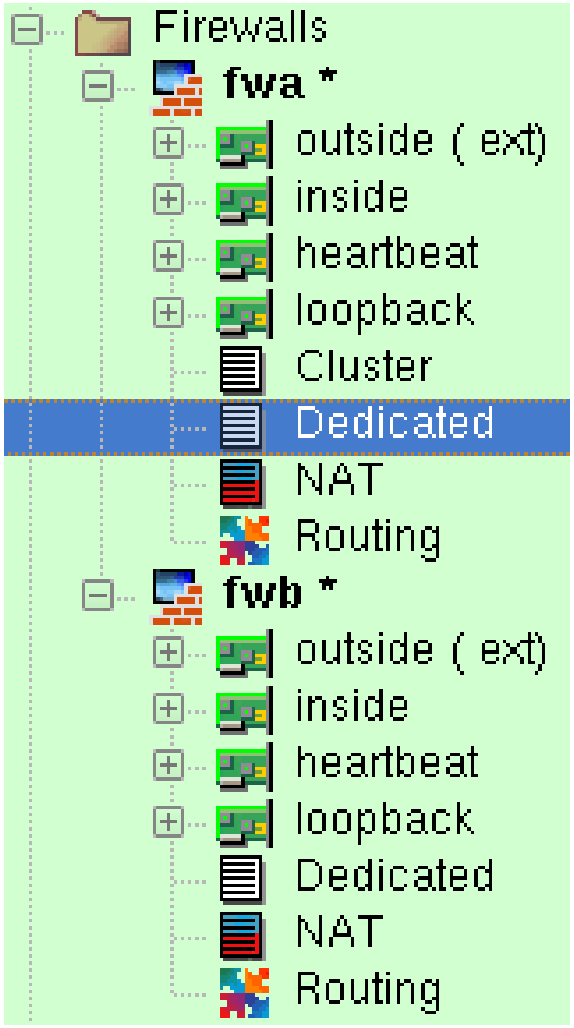


Lösung: Zwei Firewalls

- Beide Knoten sind in `fwbuilder` repräsentiert.
- Beide Knoten haben alle Schnittstellen, eigene dedizierte *und* IP – Adressen des Clusters definiert.
- Jede Firewall hat den Antispoofing- und Stealth Teil in einer lokalen Policy als „top“ – Regelsatz.
- Bei *einer* Firewall ist die Policy für den restlichen Regelsatz definiert. Von jeder „top“ – Policy wird auf diese verwiesen.
- Kein Verweis auf die Firewall selbst in der Cluster – Policy!

Darstellung in `fwbuilder`

Details


Firewalls
fwa *
outside (ext)
inside
heartbeat
loopback
Cluster
Dedicated
NAT
Routing
fwb *
outside (ext)
inside
heartbeat
loopback
Dedicated
NAT
Routing

Detail (II)

		Source	Destination	Service	Interface	irectic	Actior	Time	Option	Comment
0		fwa heartbeat internal	Any	Any	outside			An		anti spoofing rule
1		Any	Any	Any	loopback			An		
2		internal	fwa	TCP ssh	All			An		SSH Access to firewa
3		fwa	internal se	DNS	All			An		Firewall uses one of
4		fwb	fwa	UDP heartbeat UDP conntrackd	heartbeat			An		HB and conntrackd ta
5		fwa	fwb	UDP heartbeat UDP conntrackd	heartbeat			An		
6		Any	fwa	Any	All			An		All other attempts to d
7		Any	Any	Any	All		Clu	An		

Danke für die Aufmerksamkeit

Fragen?