Firewalls

mandantenfähig, redundant, deklarativ

IBH IT-SERVICE

Thomas Liske <liske@ibh.de>

Leiter NOC

Chemnitzer Linux-Tage 2025

whoami

- Dresden, Germany
- IBH IT-Service GmbH
- @liske
- @liske@ibh.social

Tagsüber

- Leiter NOC
- RZ- & ISP-Infrastruktur

Nachts

- Alpine Linux
  Contributer
- Codeberg e.V.
  Infrastruktur
- DD-IX e.V.
  Vorstand & Technik

Software-Projekte

- apt-dater
- ifstate
- imvirt
- needrestart
- python-apds9960

...

IBH IT-SERVICE

Deine Karriere in der IT beginnt bei uns!

IBH-POWER-UPs:

- ✓ Open-Source-Denken
- ✓ eigenes Rechenzentrum
- ✓ DU-Mentalität
- ✓ flexible Arbeitszeiten
- ✓ u.v.m.

WWW.IBH.DE/JOBS

Firewall

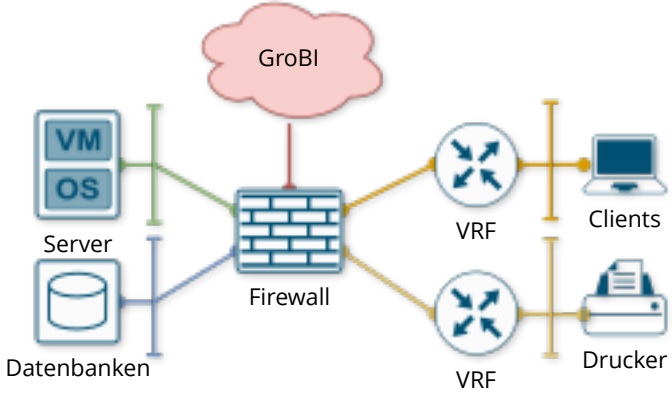
- Netzwerkgerät zur gegenseitigen Abschottung von Netzbereichen
- Regelwerk (Policy) legt fest, welche Pakete (nicht) passieren dürfen

Einbindung

- Layer 2: Bridge
- Layer 3: Routing Hop

Overlay Netze vs. zentrale Firewalls

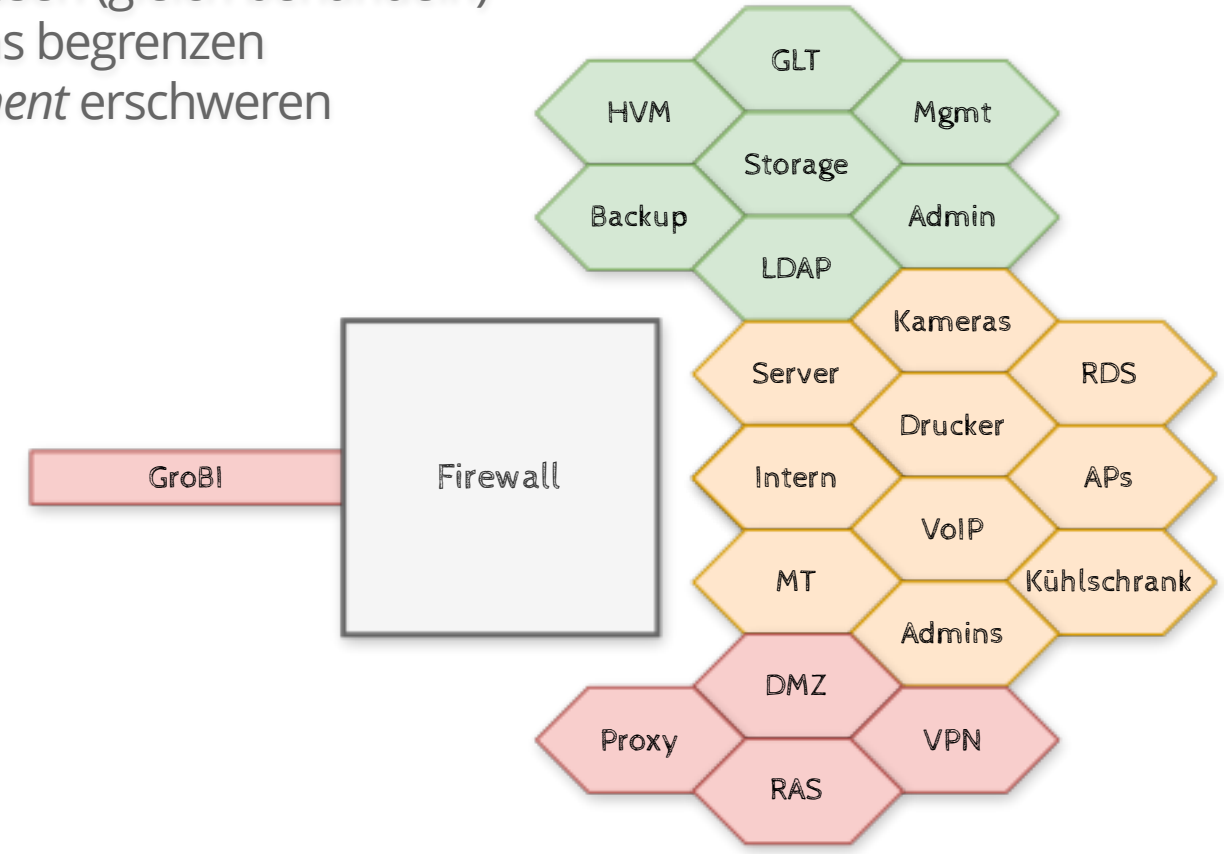
- Layer 2: Access bis zur Firewall
- Layer 3: via VRFs zur Firewall



Netzwerk (Security) Design

Separierung

- Systeme mit ähnlicher Gefährdung und Berechtigung zusammenfassen (gleich behandeln)
- Fehlerdomains begrenzen
- *Lateral Movement* erschweren

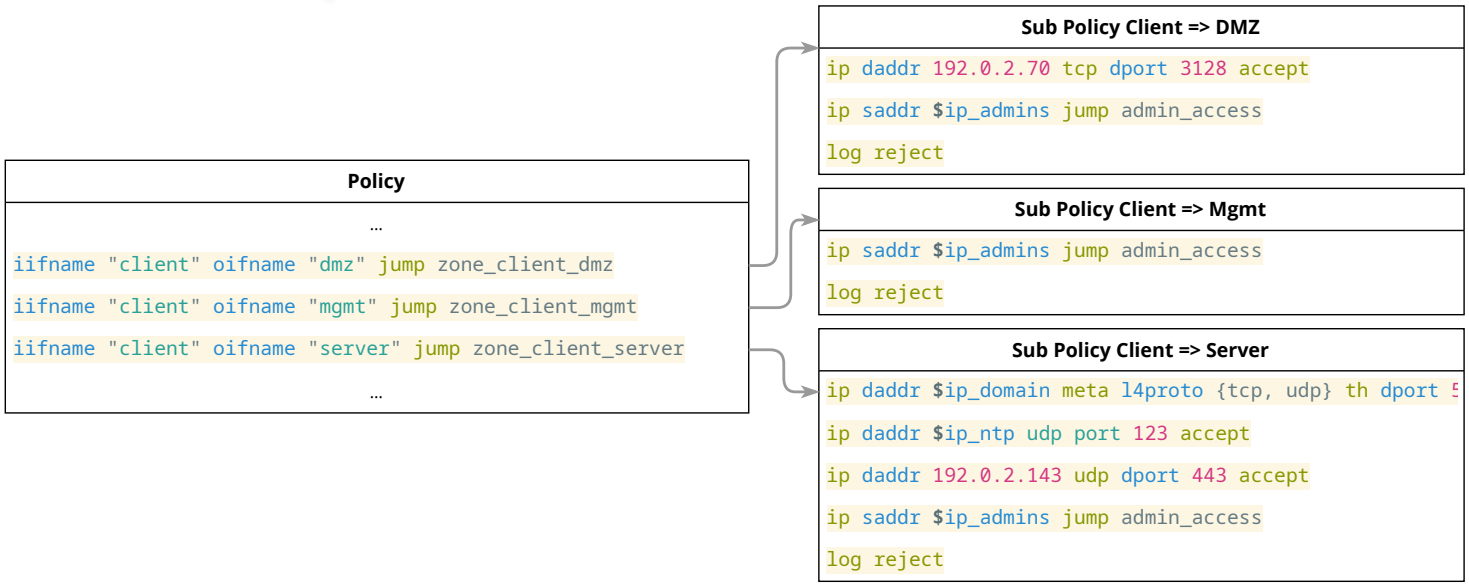


Regelwerke schreiben

- lineare Listen sind doof
  - unübersichtlich je mehr Regeln benötigt werden
  - Semantik nicht einfach zu begreifen → gefährlich

Vorgehen

- Interfaces zu Zonen zuordnen (n:1)
- eine *Sub Policy* pro Quell-/Zielzonenkombination
- Ansteuerung in *Policy* sortiert nach Quellzone → Zielzone

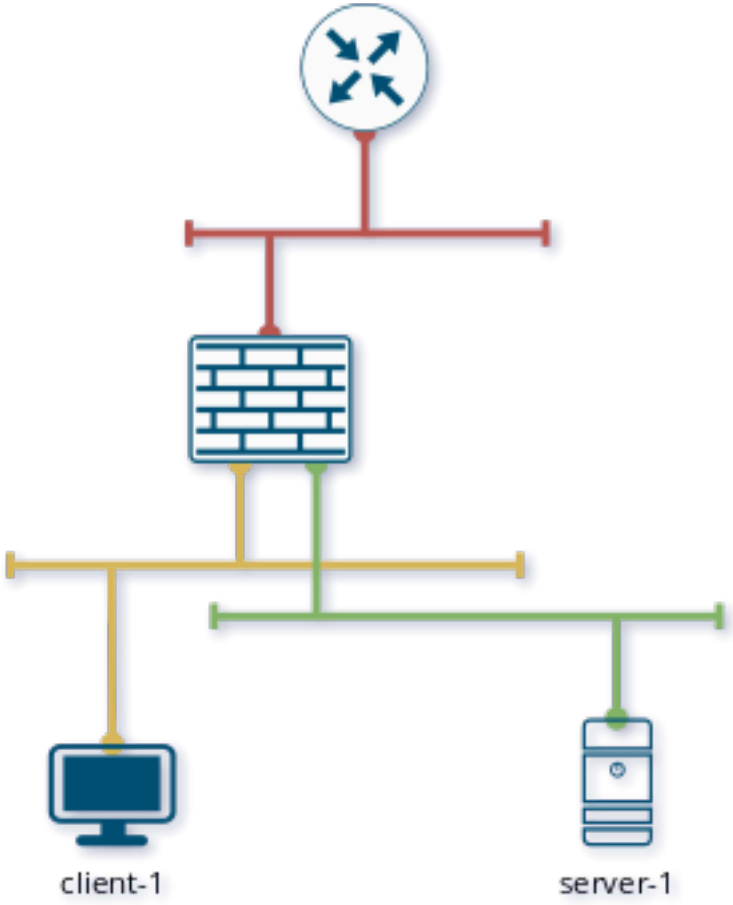


Demo #1

Einzelne Firewall

Technologie Stack

- Ansible Deployment
- Firewall
  - Alpine Linux
  - nftables



Redundanz

Stateful Firewalls

Kommunikation ist (meist) bidirektional

```
# permit request…
ip6 saddr 2001:db8:7357::21 ip6 daddr 2001:db8:15372::/48 tcp dport 22 accept
ip6 saddr 2001:db8:7357::7 ip6 daddr 2001:db8:c3d2::/48 tcp dport {80, 443} accept
…

# …and responses 🤯
ip6 saddr 2001:db8:15372::/48 tcp sport 22 ip6 daddr 2001:db8:7357::21 accept
ip6 saddr 2001:db8:c3d2::/48 tcp sport {80, 443} ip6 daddr 2001:db8:7357::7 accept
…
```

- wird schnell unübersichtlich
- erlaubt Antworten ohne vorherige Anfragen
- Protokolle mit unterschiedlichen Ports (FTP, SIP, TFTP, …)?
- (Legacy NAT?)

Connection Tracking

1. Verbindungs**aufbau** explizit beregeln (NEW)
2. Antwortpaket vormerken
   - Destination IP → Source IP
   - Source IP → Destination IP
   - Protocol = Protocol
   - Destination Port → Source Port
   - Source Port → Destination Port
3. Empfang der Antwort (ESTABLISHED)
4. ggf. Vormerken protokollspezifischer Ports (RELATED)

   ftp, sip, tftp, …
5. Verbindungsinformation vorhalten bis…
   - Verbindungsabbau
   - **Timeout**

Conntrack Sysctl

```
# sysctl -a|grep -E nf_conntrack_.+_timeout
…
net.netfilter.nf_conntrack_dccp_timeout_timewait = 240
net.netfilter.nf_conntrack_frag6_timeout = 60
net.netfilter.nf_conntrack_generic_timeout = 600
net.netfilter.nf_conntrack_gre_timeout = 30
net.netfilter.nf_conntrack_gre_timeout_stream = 180
net.netfilter.nf_conntrack_icmp_timeout = 30
net.netfilter.nf_conntrack_icmpv6_timeout = 30
net.netfilter.nf_conntrack_sctp_timeout_closed = 10
net.netfilter.nf_conntrack_sctp_timeout_cookie_echoed = 3
net.netfilter.nf_conntrack_sctp_timeout_cookie_wait = 3
net.netfilter.nf_conntrack_sctp_timeout_established = 210
net.netfilter.nf_conntrack_sctp_timeout_heartbeat_sent = 30
net.netfilter.nf_conntrack_sctp_timeout_shutdown_ack_sent = 3
net.netfilter.nf_conntrack_sctp_timeout_shutdown_recd = 3
net.netfilter.nf_conntrack_sctp_timeout_shutdown_sent = 3
net.netfilter.nf_conntrack_tcp_timeout_close = 10
net.netfilter.nf_conntrack_tcp_timeout_close_wait = 60
net.netfilter.nf_conntrack_tcp_timeout_established = 432000
net.netfilter.nf_conntrack_tcp_timeout_fin_wait = 120
net.netfilter.nf_conntrack_tcp_timeout_last_ack = 30
net.netfilter.nf_conntrack_tcp_timeout_max_retrans = 300
net.netfilter.nf_conntrack_tcp_timeout_syn_recv = 60
net.netfilter.nf_conntrack_tcp_timeout_syn_sent = 120
net.netfilter.nf_conntrack_tcp_timeout_time_wait = 120
net.netfilter.nf_conntrack_tcp_timeout_unacknowledged = 300
net.netfilter.nf_conntrack_udp_timeout = 30
net.netfilter.nf_conntrack_udp_timeout_stream = 120
```

Connection Table Size

- nf_conntrack_buckets
- nf_conntrack_max

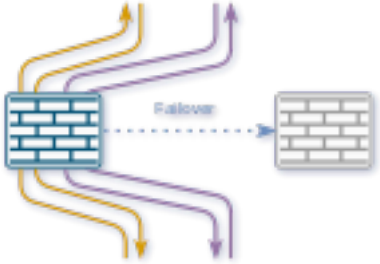
Kernel Doku:
nf_conntrack-sysctl.txt

Stateful Firewall

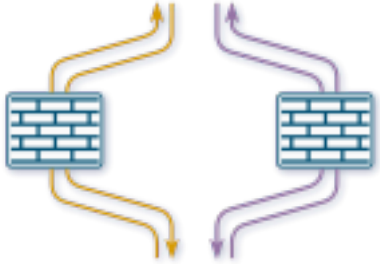
Definiert durch

- Netzwerkkonfiguration
- Regelwerk (Policy)
- Connection Tracking (volatil)

single node



active/standby

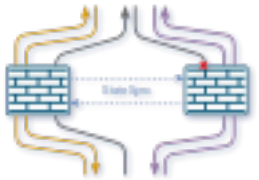


active/active

HA Modus

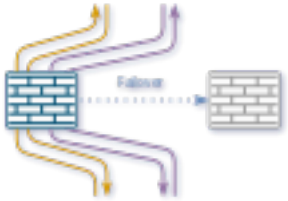
active/active

- synchroner Abgleich vom Connection Tracking schwierig (Race Condition)
- echte active/active Lösungen aufwendig



active/standby

- Netzwerkkonfiguration + Policy replizieren
- kontinuierlichen Abgleich des Connection Trackings



conntrack-tools

conntrack

- Admin CLI statt /proc/net/nf_conntrack
- anzeigen & bearbeiten der Kernel Conntrack Tabellen

```
fw-1 [~]# conntrack -L
icmp     1 29 src=10.42.21.7 dst=10.255.11.7 type=8 code=0 id=7454 [UNREPLIED] src=10.255.11.7 dst=192.168.42.254 type=0 code=0 id=7454 mark=0 use=1
icmp     1 10 src=10.42.21.7 dst=10.42.22.7 type=8 code=0 id=7448 src=10.42.22.7 dst=10.42.21.7 type=0 code=0 id=7448 mark=0 use=1
tcp      6 432000 ESTABLISHED src=10.42.255.1 dst=10.42.255.11 sport=42050 dport=22 src=10.42.255.11 dst=10.42.255.1 sport=22 dport=42050 [ASSURED] mark=0 use=1
unknown  112 599 src=10.42.11.2 dst=224.0.0.18 [UNREPLIED] src=224.0.0.18 dst=10.42.11.2 mark=0 use=1
conntrack v1.4.8 (conntrack-tools): 4 flow entries have been shown.
fw-1 [~]#
```

conntrackd

- Abgleich der Conntrack Tables zwischen Hosts (HA)

```
fw-1 [~]# conntrackd -s
cache internal:
current active connections:           132485
connections created:                  449883    failed:           0
connections updated:                  264748    failed:           0
connections destroyed:                317398    failed:           0

external inject:
connections created:                     635    failed:           0
connections updated:                    2262    failed:           0
connections destroyed:                   202    failed:           0

traffic processed:
                  0 Bytes                        0 Pckts

TCP traffic (active device=svc-ha) server=connected client=connected:
            59269232 Bytes sent              313932 Bytes recv
              534714 Pckts sent                5927 Pckts recv
                   0 Error send                   0 Error recv

message tracking:
                   0 Malformed msgs                   0 Lost msgs

fw-1 [~]#
```

Failover Routing

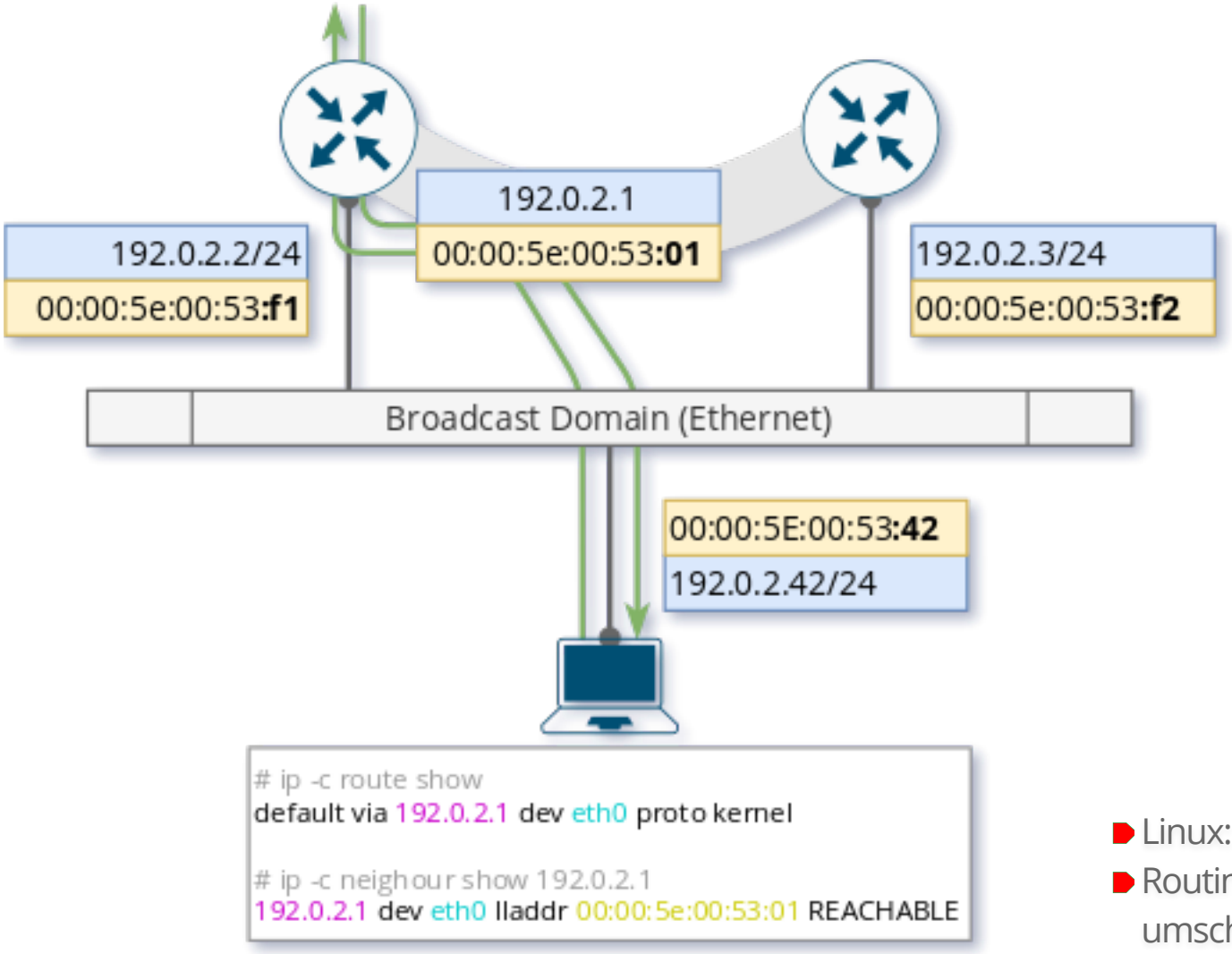
- Firewall wirkt auf Layer 3
- *Default Gateway* für Clients & Server
  → **kein** dynamisches Routing

First-Hop Redundancy Protocol

- Protokollfamilie für hochverfügbare *Gateways*
- typische Vertreter:
  - HSRP
  - VRRP
  - CARP

keepalived

- L4 Loadbalancer (IPVS)
- **VRRP** Implementation
- **BFD** Support
- Hooks und Tracker an vielen Stellen nutzbar

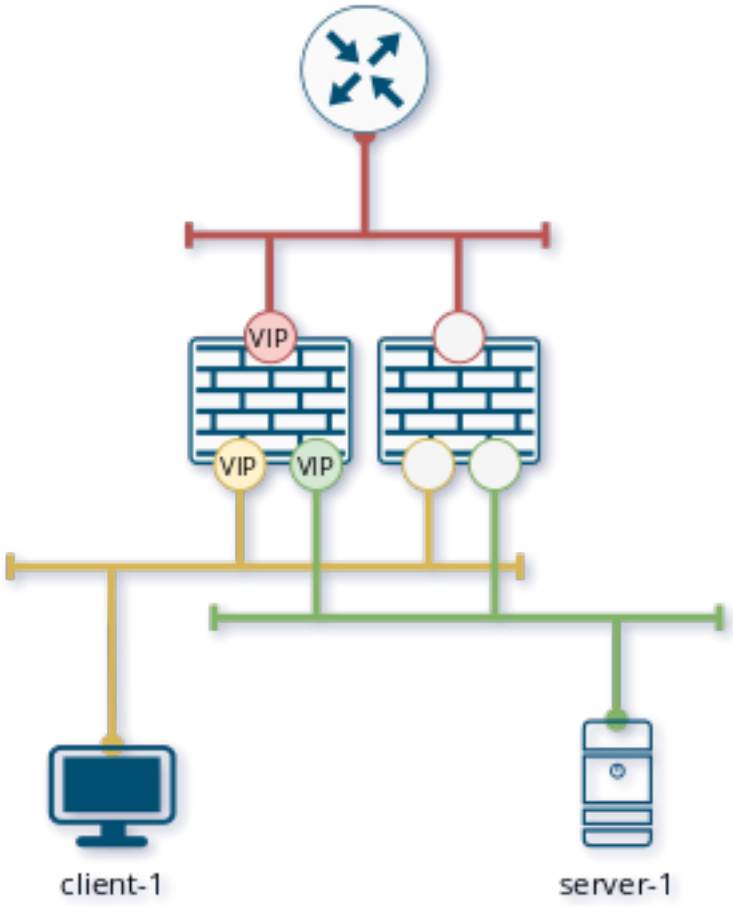


- Linux: Interfaces haben **genau** eine MAC
- Routing: alle VIPs müssen synchron umschalten

Demo #2 + #3 – redundante Firewalls

Technologie Stack

- Ansible Deployment
- Firewall
  - Alpine Linux
  - nftables
  - conntrackd
  - keepalived



Herausforderungen

- für **V**MACs werden eigene Interfaces benötigt
  - assymetrisches Routing inbound vs. outbound
  - kein Reverse Path Filter
- FHRP in unsicheren* Netzen?

Ansatz

Idee

- (virtueller) dedizierter Failover Link
- IP-Funktion der Standby Firewall abschalten
- nur **eine** IP je Netz benötigt

Umsetzung

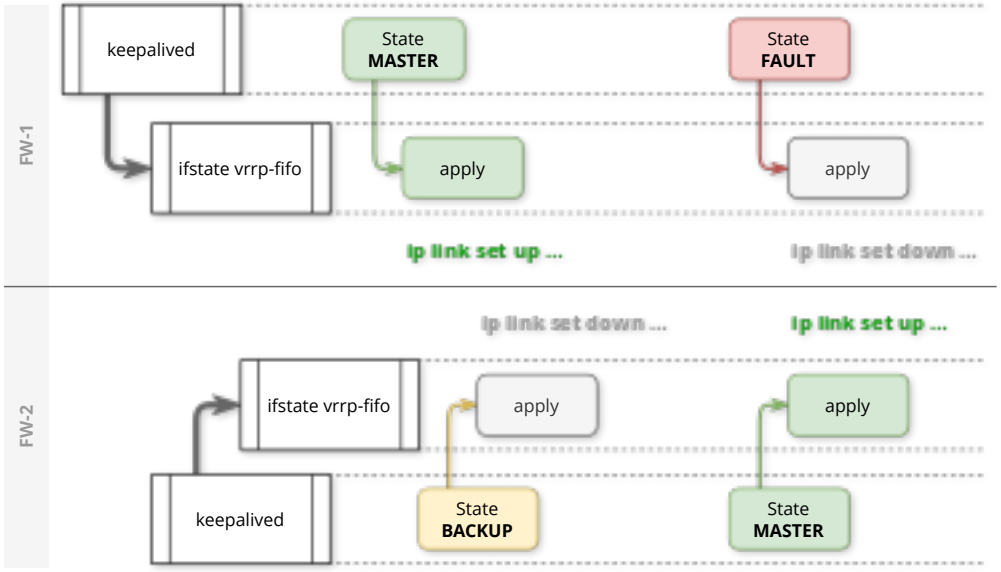
- keepalived steuert ifstate (VRRP FIFO)
- ifstate aktiviert Interfaces je nach VRRP State

keepalived.conf

```
global_defs {
  # vrrp notify fifo (ifstate)
  vrrp_notify_fifo /run/vrrp-ifstate.fifo
  vrrp_notify_fifo_script "/usr/bin/ifstatecli vrrp-fifo"
}

vrrp_instance CLT {
  # start-up default state
  state BACKUP

  # VRRP interface
  interface server

  # VRRP w/o VIP (requires keepalived 2.2.8+)
  no_virtual_ipaddress
…
```

ifstate

```
interfaces:
  # …
  - name: server-1
    addresses:
      - 10.42.11.1/24
    link:
      state: up
      kind: vlan
      vlan_id: 1011
      link: mgmt
      address: "42:12:1d:10:11:01"
    vrrp: &vrrp-clt
      name: CLT
      type: instance
      states:
      - master
```



Network Namespaces

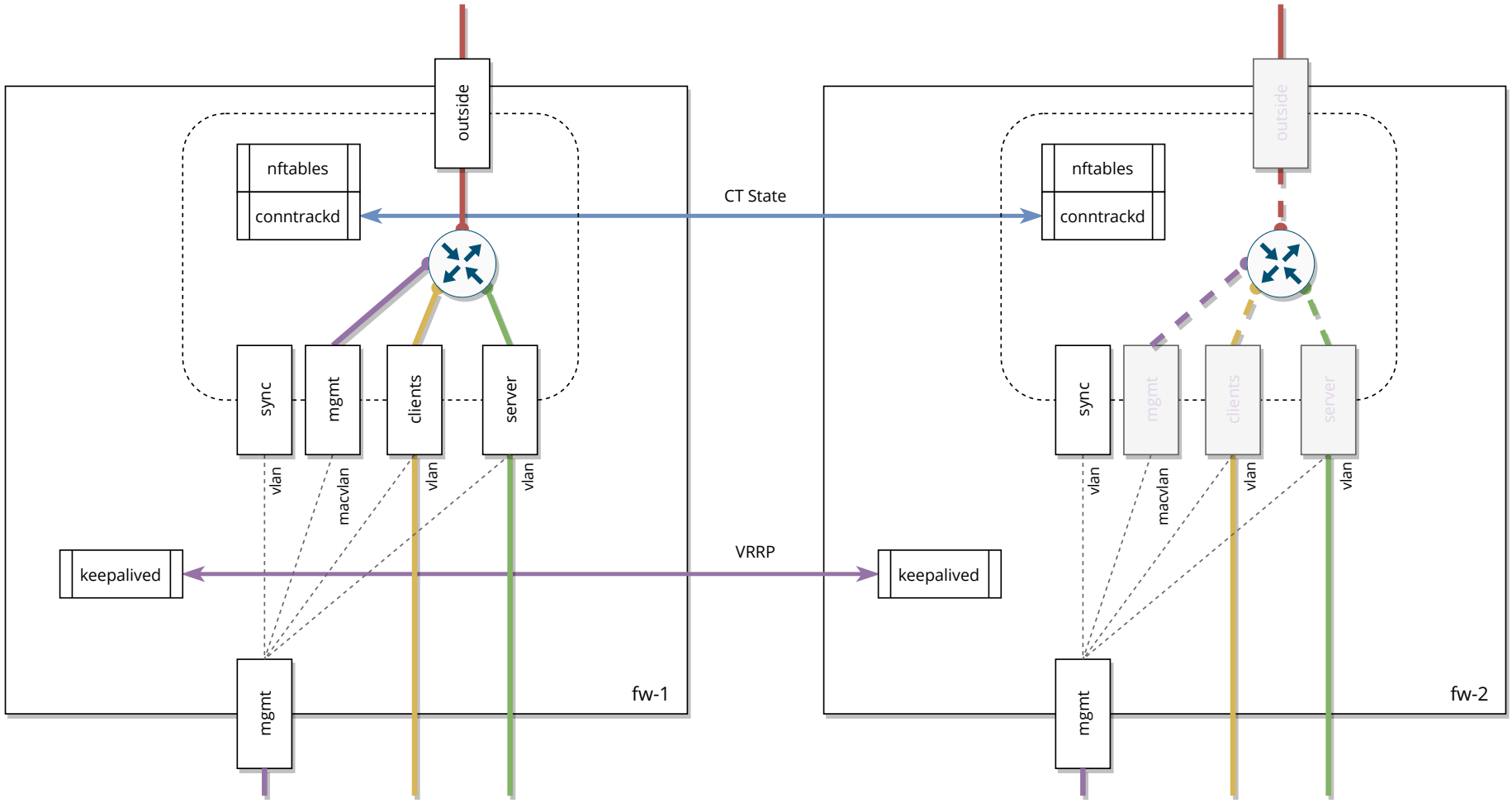
Virtualisierungsfunktion für den Linux IP Stack

Motivation

- Trennung Management-Zugriff Firewall und Produktivnetz
- Beregelung Firewall-Knoten wie normale Hosts
- Failover-Link vom Produktivnetz trennen

Idee

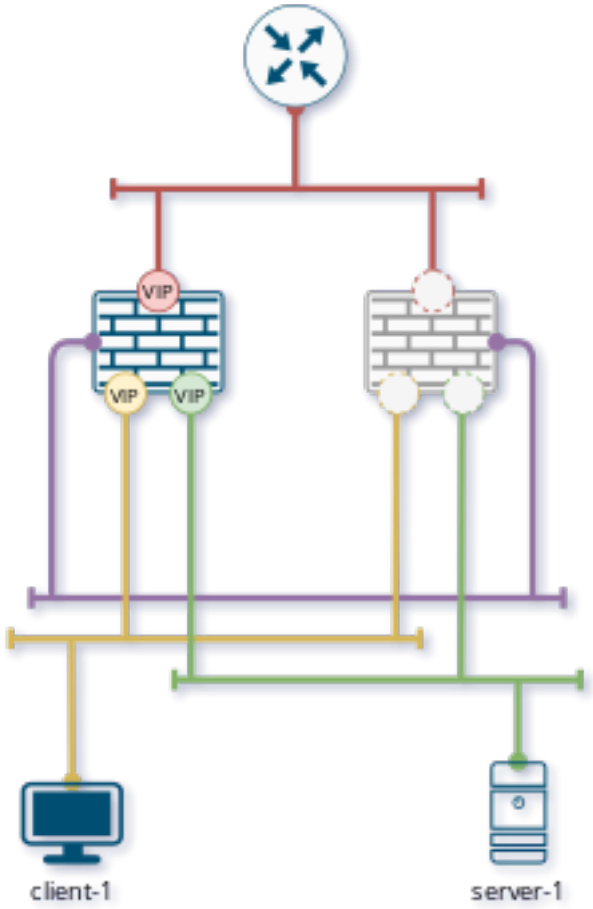
- Produktiv-Traffic in NetNS einsperren
- nur benötigte Dienste im NetNS betreiben
  (conntrackd, dnsmasq, ...)
- Duplizierung von Interfaces möglich (Typ macvlan)


outside
nftables
conntrackd
CT State
sync
mgmt
clients
server
vlan
macvlan
vlan
vlan
keepalived
VRRP
keepalived
mgmt
fw-1
outside
nftables
conntrackd
sync
mgmt
clients
server
vlan
macvlan
vlan
vlan
mgmt
fw-2

Demo #4 + #5 – VRRP Fifo

Technologie Stack

- Ansible Deployment
- Firewall
  - Alpine Linux
  - nftables
  - conntrackd
  - keepalived
  - ifstate vrrp-fifo
  - netns (#5)



Mandantenfähigkeit

dediziert pro Mandant

- Namespace für Produktiv-Traffic
- nftables Regelwerk
- conntrackd Instanz mit Sync Tunnel
- VRRP-Instanz im globalen keepalived
- ggf. weitere Dienste (dnsmasq)

Lastverteilung

- aktive Tenants auf beide Knoten verteilen
- Isolation bei Debugging, (D)DoS, … möglich

sysctl Tuning

```
# https://www.kernel.org/doc/Documentation/sysctl/net.txt
# https://docs.kernel.org/admin-guide/sysctl/net.html

## IPv4 interface
net.ipv4.conf.*.accept_redirects = 0       # ignore ICMP redirects
net.ipv4.conf.*.accept_source_route = 0    # ignore SRR options
net.ipv4.conf.*.arp_ignore = 2             # strict ARP handling
net.ipv4.conf.*.arp_notify = 1             # send gratuitous ARPs
net.ipv4.conf.*.forwarding = 1             # enable forwarding
net.ipv4.conf.*.rp_filter = 1              # strict reverse path filtering RFC3704
net.ipv4.conf.*.send_redirects = 0         # never send ICMP redirects

## IPv6 interface
net.ipv6.conf.*.accept_ra = 0              # don't accept RA
net.ipv6.conf.*.accept_redirects = 0       # ignore ICMPv3 redirects
net.ipv6.conf.*.autoconf = 0               # disable auto address config
net.ipv6.conf.*.forwarding = 1             # enable forwarding
net.ipv6.conf.*.ndisc_notify = 1           # send gratuitous NAs
net.ipv6.conf.*.optimistic_dad = 1         # assume IPv6 DAD to succeed RFC4429

## conntrack table size
net.netfilter.nf_conntrack_buckets = 2097152
net.netfilter.nf_conntrack_max = 2097152
```

Zusammenfassung

Take Aways

- Strukturierung Firewall Policy
- Linux Firewall Cluster
- Tooling im Userspace
- Sysctl Tunings

liske@ibh.de
@liske@ibh.social

https://codeberg.org/liske/clt2025-liske-firewalls

Demo: Mind the gaps!

- gleichzeitiges Deployment beider Firewalls
- Dienste beim Booten starten
- Logging, Monitoring, Notifications
- Test bei schlechtem Wetter
- auf Debugging vorbereitet sein

Die Präsi enthält...

- animate.css
- highlight.js
- fork-awesome
- impress.js
- impress.js-progress
- impress-console