Load Balancing with nftables

by Laura García (Zen Load Balancer Team)
Netdev 1.1

Prototype of

Load Balancing with nftables

Goal:

High Performance Load Balancer

Load Balancing Solutions

Load Balancing Solutions

Linux Virtual Server

iptables

nftables

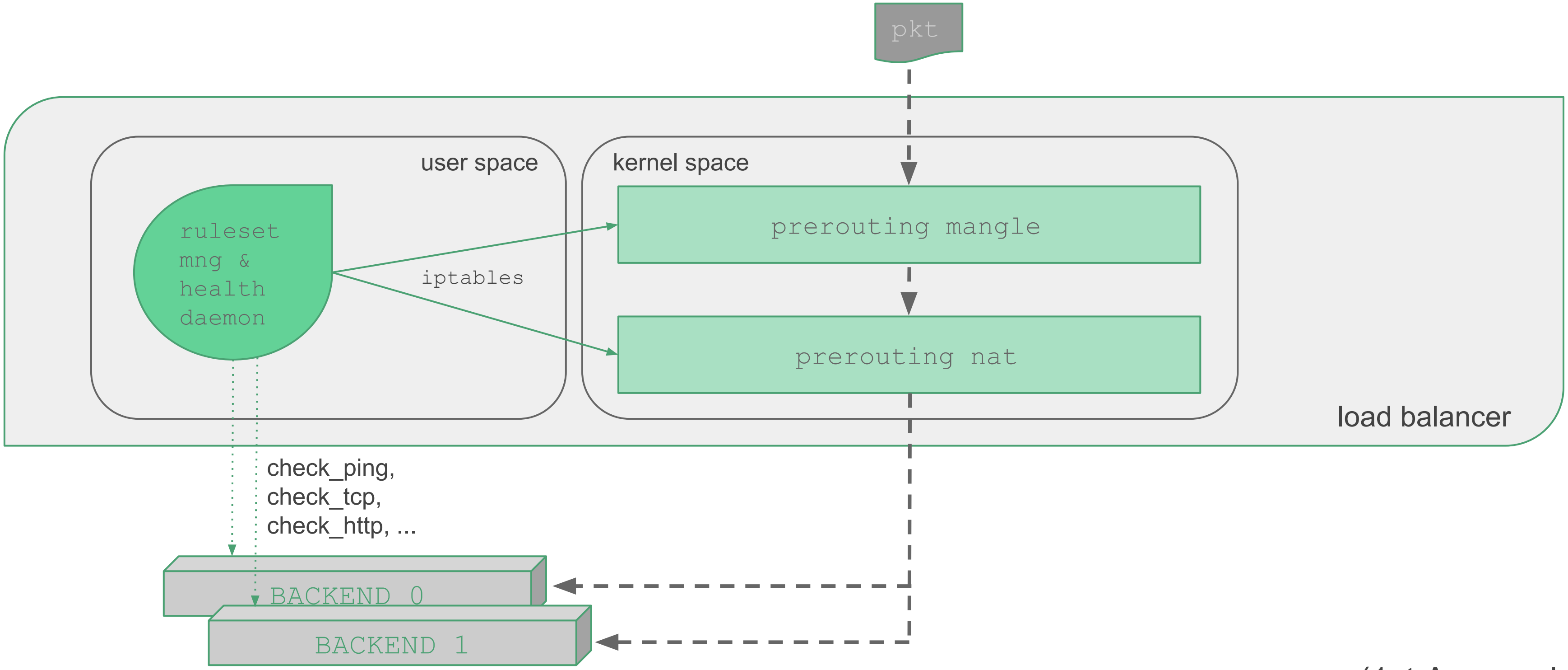
Load Balancing Solutions - LVS

- Feature complete & versatile schedulers
- Several forwarding methods
- Integrated health checks
- Built on top of netfilter
- Mostly kernel code base

Load Balancing Solutions - iptables

- Schedulers based on xtables extensions
- SNAT and DNAT as forwarding methods
- Mark packets and forwarding
- Backend health checks from user space

Load Balancing Solutions - iptables

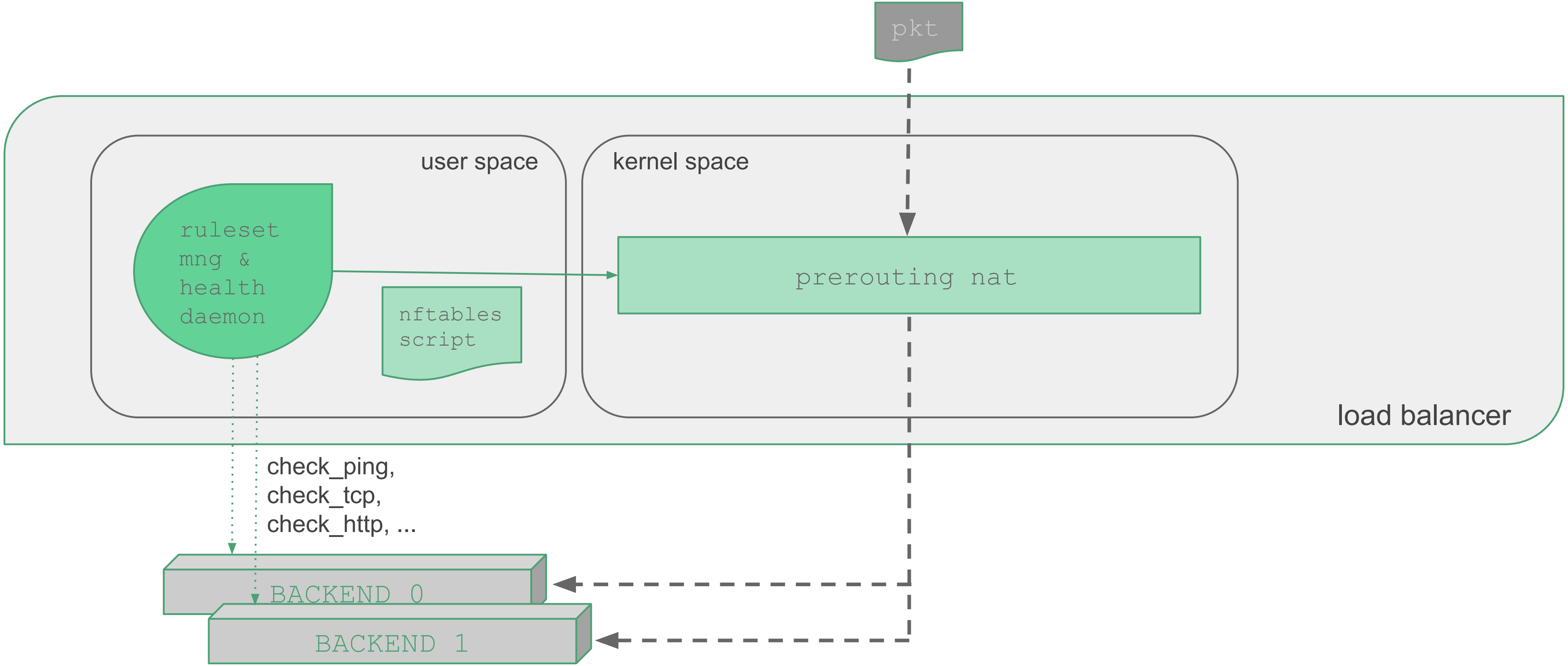


(1st Approach)

Load Balancing Solutions - nftables

- Using nftables infrastructure
    - nft libraries
    - nftables VM & its instructions
- Dynamic and atomic rules
- No marking packets needed
- Several forwarding methods

Load Balancing Solutions - nftables



Features to accomplish

Features to accomplish

Schedulers

round robin, weight, least connections

Features to accomplish

Persistence

Source IP

Features to accomplish

Forwarding methods

SNAT, DNAT

Features to accomplish

Health checks

Backend monitoring in user space at different levels

Features to accomplish

Good Integration

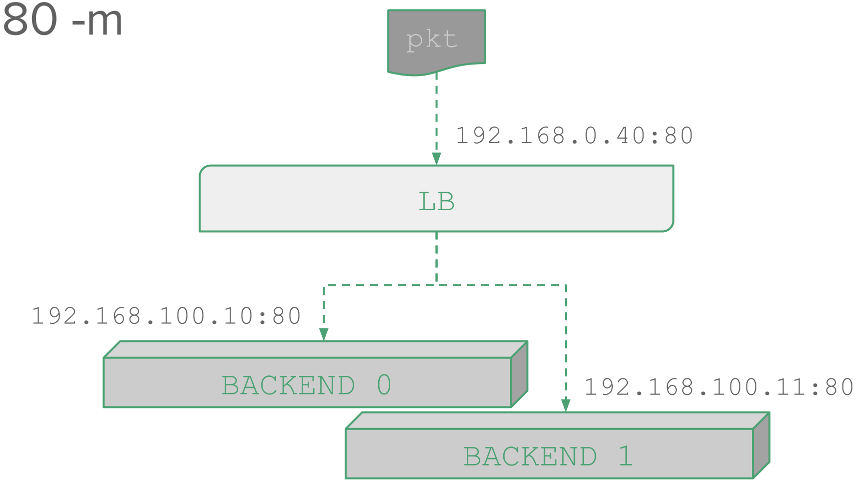
QoS, filtering

Use Cases

Use Cases

Round Robin Load Balancing with LVS

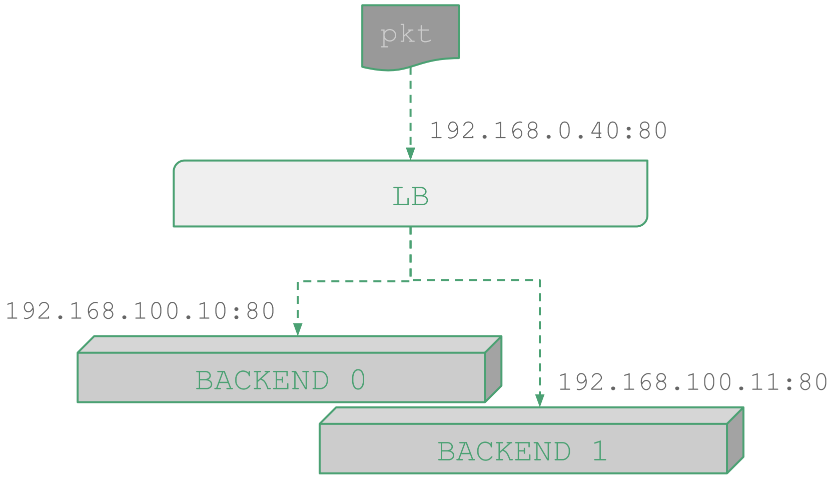
```
ipvsadm -A -t 192.168.0.40:80 -s rr
ipvsadm -a -t 192.168.0.40:80 -r 192.168.100.10:80 -m
ipvsadm -a -t 192.168.0.40:80 -r 192.168.100.11:80 -m
```



Use Cases

Round Robin Load Balancing with IPT

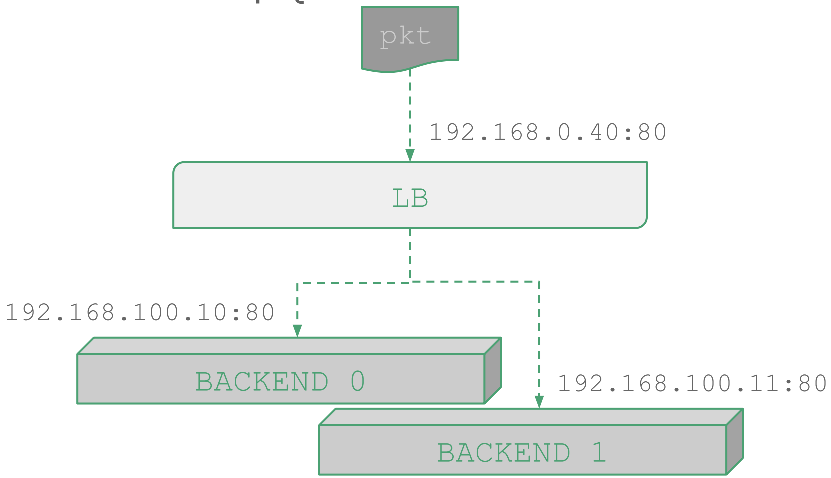
```
iptables -t nat -A PREROUTING -m statistic --mode nth --every 2 --packet 0 -d
192.168.0.40 -p tcp --dport 80 -j DNAT --to-destination 192.168.100.10:80

iptables -t nat -A PREROUTING -m statistic --mode nth --every 2 --packet 1 -d
192.168.0.40 -p tcp --dport 80 -j DNAT --to-destination 192.168.100.11:80
```



Use Cases

Round Robin Load Balancing with NFT

```
table ip lb {
        chain prerouting {
                type nat hook prerouting priority 0; policy accept;
                ip daddr 192.168.0.40 tcp dport http dnat nth 2 map {
                        0: 192.168.100.10,
                        1: 192.168.100.11
                }
        }
}
```



Use Cases

Weight Load Balancing with LVS

```
ipvsadm -A -t 192.168.0.40:80 -s wrr
ipvsadm -a -t 192.168.0.40:80 -r 192.168.100.10:80 -m -w 100
ipvsadm -a -t 192.168.0.40:80 -r 192.168.100.11:80 -m -w 50
```

Use Cases

Weight Load Balancing with IPT

```
iptables -t nat -A PREROUTING -m statistic --mode random --probability 1 \
        -d 192.168.0.40 -p tcp --dport 80 -j DNAT --to-destination 192.168.100.10:80
iptables -t nat -A PREROUTING -m statistic --mode random --probability 0.33 \
        -d 192.168.0.40 -p tcp --dport 80 -j DNAT --to-destination 192.168.100.11:80
```

Use Cases

Weight Load Balancing with NFT

```
table ip lb {
        chain prerouting {
                type nat hook prerouting priority 0; policy accept;
                ip daddr 192.168.0.40 tcp dport http dnat random upto 100 map {
                        0-66: 192.168.100.10,
                        67-99: 192.168.100.11
                }
        }
}
```

Use Cases

Weight Load Balancing Multiport with LVS

```
iptables -A PREROUTING -t mangle -d 192.168.0.40 -p tcp -m multiport \
        --dports 80,443 -j MARK --set-mark 1

ipvsadm -A -f 1 -s wrr
ipvsadm -a -f 1 -r 192.168.100.10:0 -m -w 100
ipvsadm -a -f 1 -r 192.168.100.11:0 -m -w 50
```

Use Cases

Weight Load Balancing Multiport with IPT

```
iptables -t nat -A PREROUTING -m statistic --mode random --probability 1 \
        -d 192.168.0.40 -p tcp -m multiport --dports 80,443 -j DNAT \
        --to-destination 192.168.100.10

iptables -t nat -A PREROUTING -m statistic --mode random --probability 0.33 \
        -d 192.168.0.40 -p tcp -m multiport --dports 80,443 -j DNAT \
        --to-destination 192.168.100.11
```

Use Cases

Weight Load Balancing Multiport with NFT

```
table ip lb {
        chain prerouting {
                type nat hook prerouting priority 0; policy accept;
                ip daddr 192.168.0.40 tcp dport { http,https } dnat random upto 100 map {
                        0-66: 192.168.100.10,
                        67-99: 192.168.100.11
                }
        }
}
```

Use Cases

Weight LB IP persistence with LVS

```
ipvsadm -A -t 192.168.0.40:80 -s wrr -p 300
ipvsadm -a -t 192.168.0.40:80 -r 192.168.100.10:80 -m -w 100
ipvsadm -a -t 192.168.0.40:80 -r 192.168.100.11:80 -m -w 50
```

Use Cases

Weight LB IP persistence with IPT

```
iptables -t mangle -A PREROUTING -j CONNMARK --restore-mark
iptables -t mangle -A PREROUTING -m statistic --mode random --probability 1 \
        -d 192.168.0.40 -p tcp --dport 80 -j MARK --set-xmark 1
iptables -t mangle -A PREROUTING -m statistic --mode random --probability 0.33 \
        -d 192.168.0.40 -p tcp --dport 80 -j MARK --set-xmark 2
iptables -t mangle -A PREROUTING -m recent --name "mark1_list" --rcheck --seconds 120 \
        -d 192.168.0.40 -p tcp --dport 80 -j MARK --set-xmark 1
iptables -t mangle -A PREROUTING -m recent --name "mark2_list" --rcheck --seconds 120 \
        -d 192.168.0.40 -p tcp --dport 80 -j MARK --set-xmark 2
iptables -t mangle -A PREROUTING -m state --state NEW -j CONNMARK --save-mark

iptables -t nat -A PREROUTING -m mark --mark 1 -j DNAT -p tcp \
        --to-destination 192.168.100.10:80 -m recent --name "mark1_list" --set
iptables -t nat -A PREROUTING -m mark --mark 2 -j DNAT -p tcp \
        --to-destination 192.168.100.11:80 -m recent --name "mark2_list" --set
```

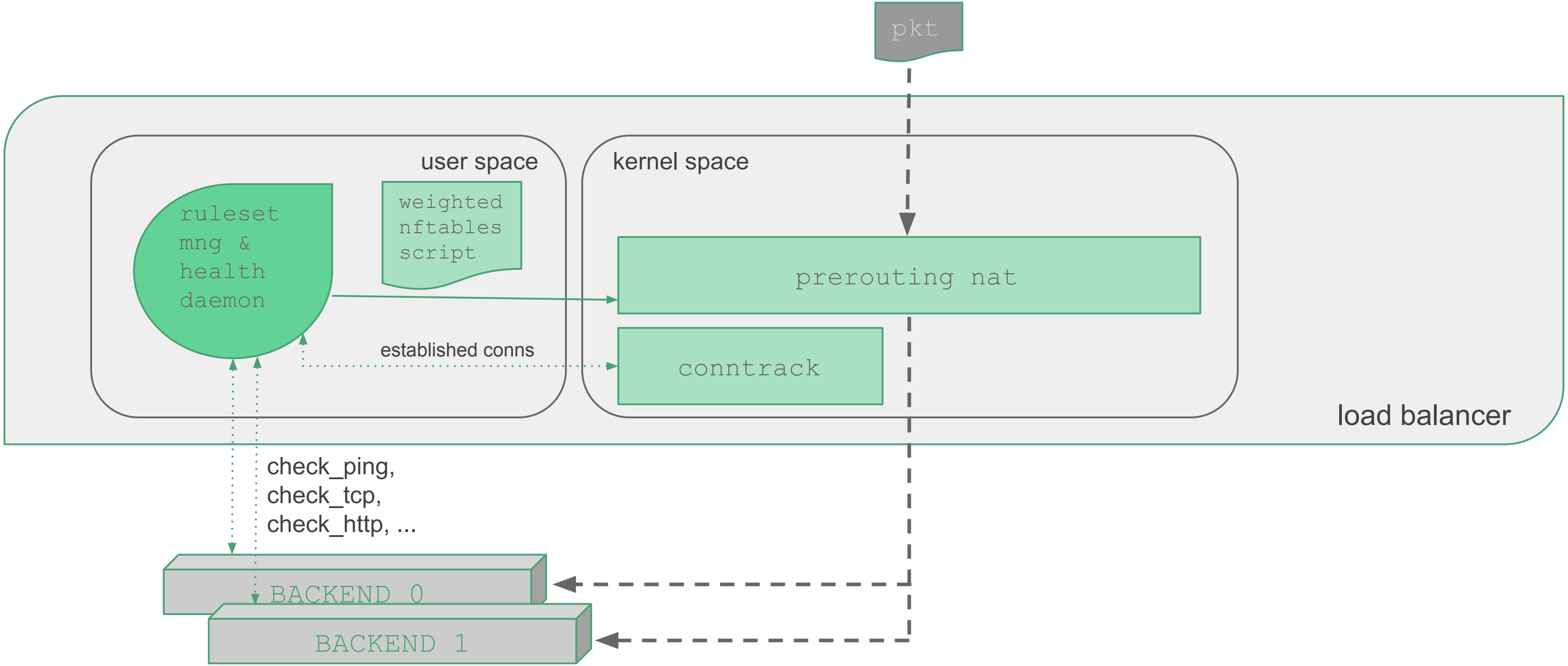
Use Cases

Weight LB IP persistence with NFT

```
table ip lb {
    map dnat-cache { type ipv4_addr : ipv4_addr; timeout 120s; }
    chain cache-done { dnat ip saddr map @dnat-cache }
    chain prerouting {
        type nat hook prerouting priority 0; policy accept;
        ip saddr @dnat-cache goto cache-done
        ip daddr 192.168.0.40 tcp dport http dnat random upto 100 map {
            0-66: 192.168.100.10,
            67-99: 192.168.100.11    }
        map dnat-cache add { ip saddr : ip daddr }
    }
}
```

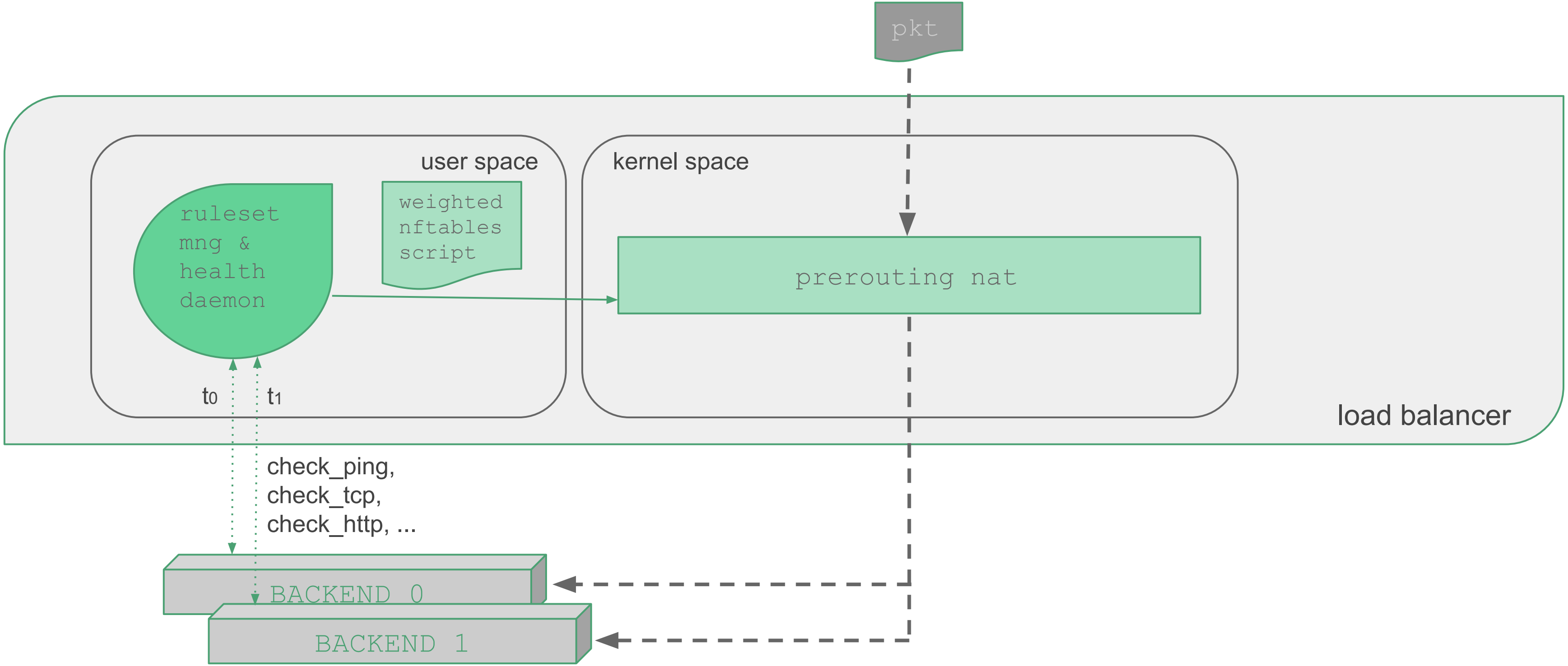
Use Cases

Weighted Least Connections with NFT



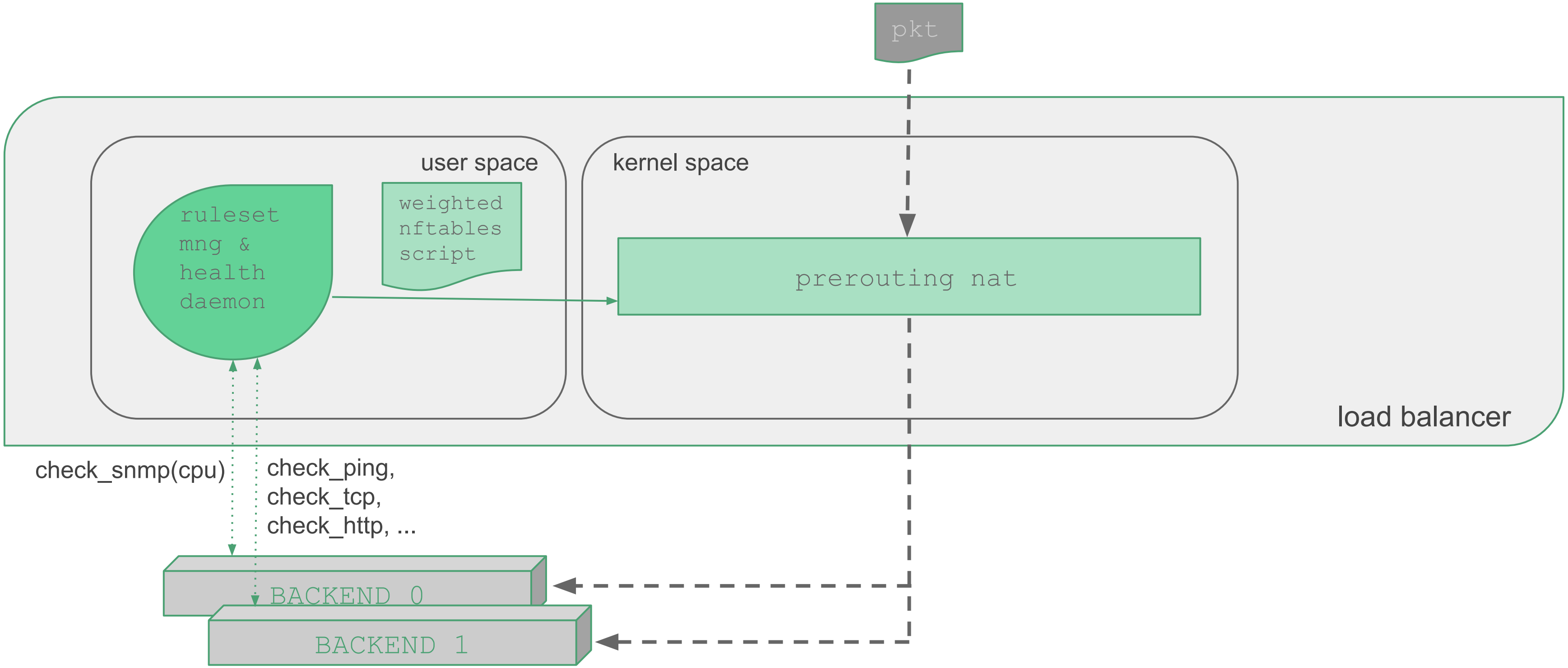
Use Cases

Weighted Least Response with NFT



Use Cases

Weighted Least CPU Load with NFT



Work to do

Work to do

Implement some native functions in nftables

random, nth, maps enhancements

Work to do

Daemon nft-lbd

health checks support, dynamic weight (least connections, least response, etc.)

Conclusions

Conclusions

Simplify kernel infrastructure

Move complexity to User Space

Conclusions

Consolidate kernel development

Avoid duplicated work, better maintenance, native LB support

Conclusions

Unique API for networking handling

nftables

Questions? Thank you!

Load Balancing with nftables

Laura García (Zen Load Balancer Team)
lauragl@sofintel.net