Netfilter High Availability

Krisztián Kovács
<hidden@balabit.hu>

BalaBit IT Ltd.

Netfilter Workshop 2005
2005.10.3.

Overview

Overview

Overview

Overview

Overview

Redundancy in General

Why do we need redundancy?

- Packet filters are critical entities of the network:
  - security;
  - availability (SPOF)[1].
- The same holds for routers, too.

Redundancy solutions

LAN	redundant cabling, redundant switches (Spanning Tree Protocol, trunking)
gateway	multiple gateway routers, virtuális routers (VRRP and friends)
routing	redundant paths and dynamic routing

[1] Single Point Of Failure

Virtual Router Redundancy Protocol

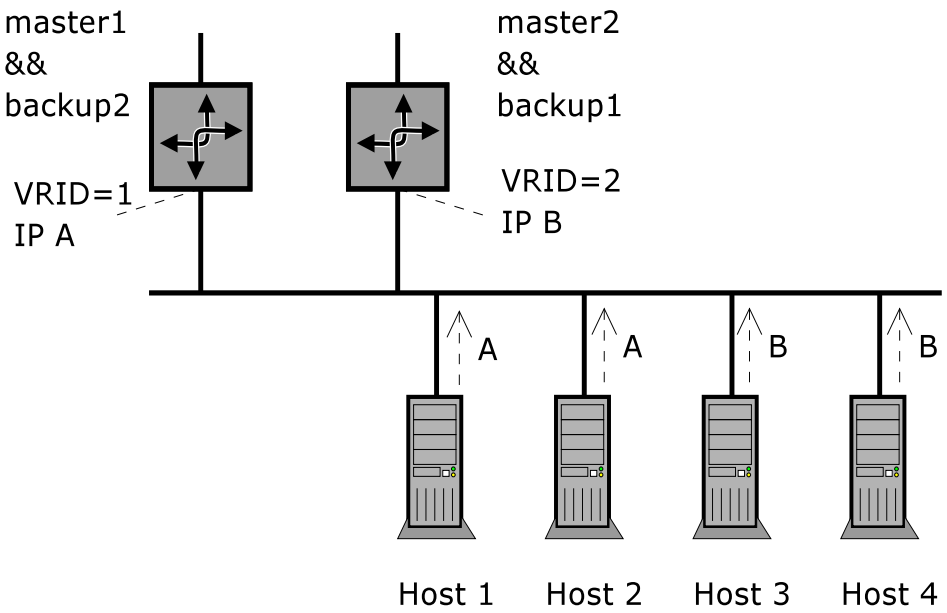
Virtual routers

- Network nodes communicate through *virtual routers*.
- Every virtual router has a unique IP and MAC address.
- Each virtual router is backed by multiple real routers.
- Participants of a virtual router group elect a *master* which configures the virtual MAC and IP address onto itself.

VRRP

- Takes care of the election process.
- Utilizes periodical advertisement messages.
- Priorities make it easy to configure preferred order.

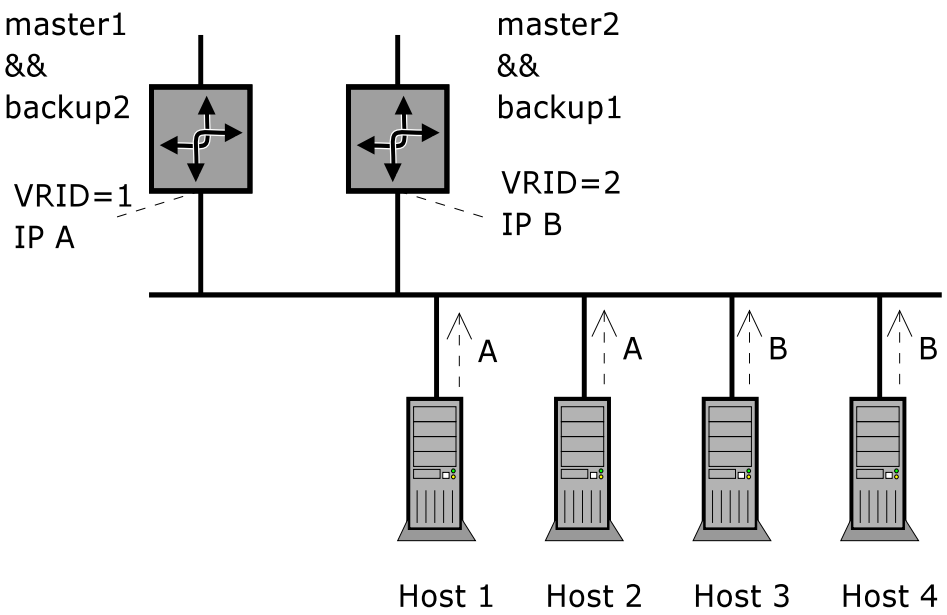
Improving Utilization



Typical VRRP arcitecture

- Two virtual routers (A, B); half of the clients use A, others use B.
- Both routers participate in both virtual routers; one of them is the master of A, the other is the master of B.

Improving Utilization



Typical VRRP arcitecture

- Two virtual routers (A, B); half of the clients use A, others use B.
- Both routers participate in both virtual routers; one of them is the master of A, the other is the master of B.

Redundancy and Packet Filters

Stateless packet filters

- No internal state: decisions are based on the ruleset exclusively.
- "Picky router"
- The same problem as in case of routers: VRRP is sufficient.

Stateful packet filters

- Maintain internal state: past events can have consequences on decisions.
- Taking over all network addresses is not enough: we also need the internal state table.
- VRRP is not satisfactory, we need a system handling the state tables as well.

Redundancy and Packet Filters

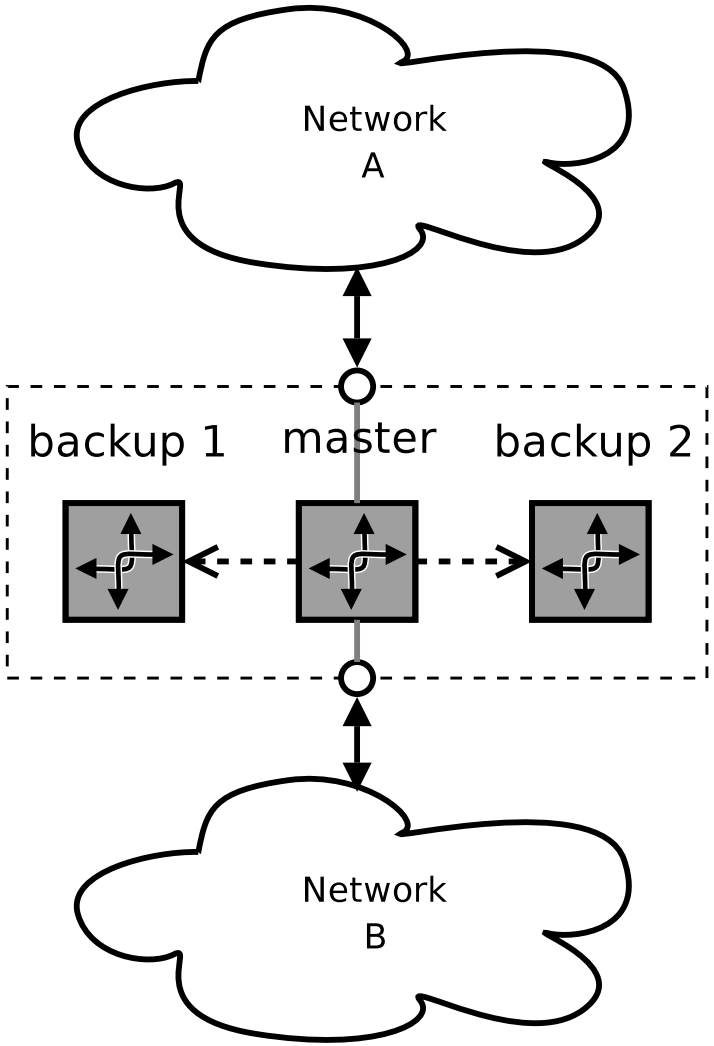
Stateless packet filters

- No internal state: decisions are based on the ruleset exclusively.
- "Picky router"
- The same problem as in case of routers: VRRP is sufficient.

Stateful packet filters

- Maintain internal state: past events can have consequences on decisions.
- Taking over all network addresses is not enough: we also need the internal state table.
- VRRP is not satisfactory, we need a system handling the state tables as well.

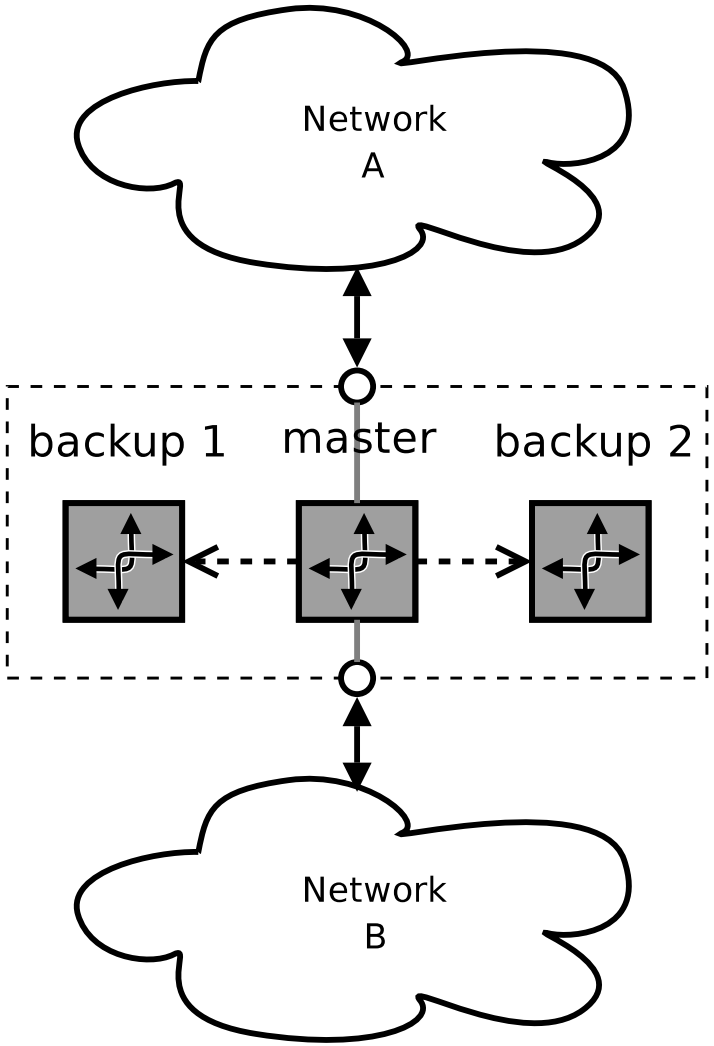
State Replication



State replication

- master processes and forwards the incoming packets
- sends state update messages through the replication network
- backup nodes update their state tables based on these messages

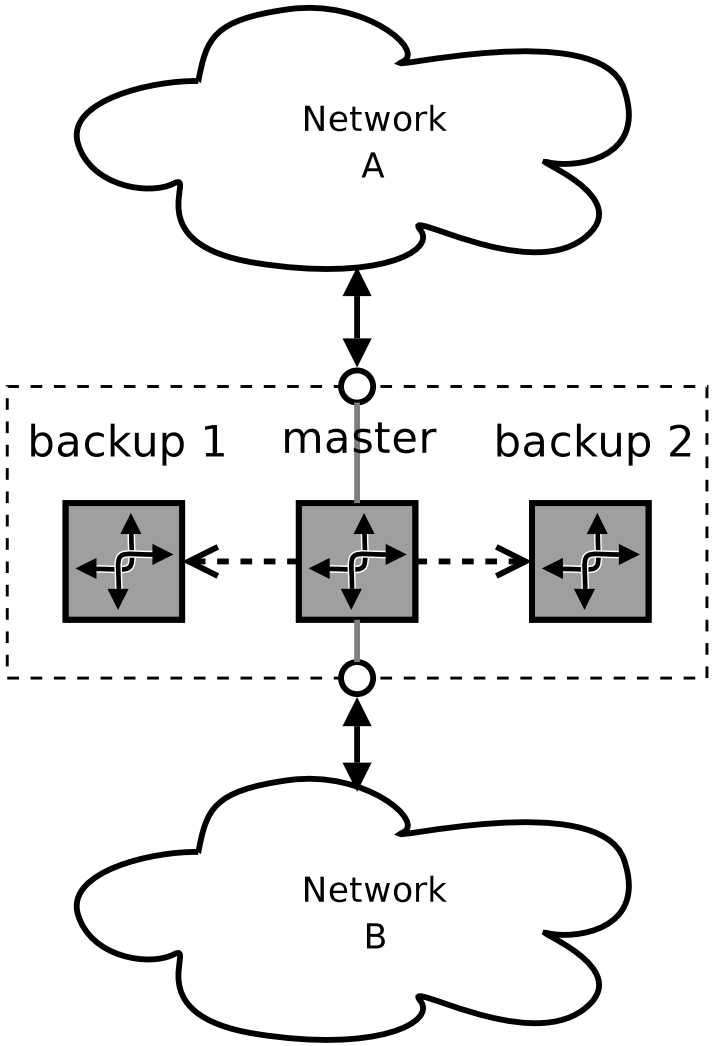
State Replication



State replication

- master processes and forwards the incoming packets
- sends state update messages through the replication network
- backup nodes update their state tables based on these messages

State Replication



State replication

- master processes and forwards the incoming packets
- sends state update messages through the replication network
- backup nodes update their state tables based on these messages

Overview

Netfilter Connection Tracking

Its responsibilities:

- Maintaining connection state.
- For each incoming packet:
  - associates a connection with the packet;
  - determines the relation between the connection and the packet.

note Relation can be one of NEW, ESTABLISHED, RELATED or INVALID.

What kind of data does the state table contain?

conntrack entry : describes a logical connection (endpoints, current state, NAT transformations, accounting).

expectation : "expected connection", describes the endpoints of a connection which will have a special relation to an already existing connection when it arrives (example: FTP data channel).

Overview

State Replication for Netfilter

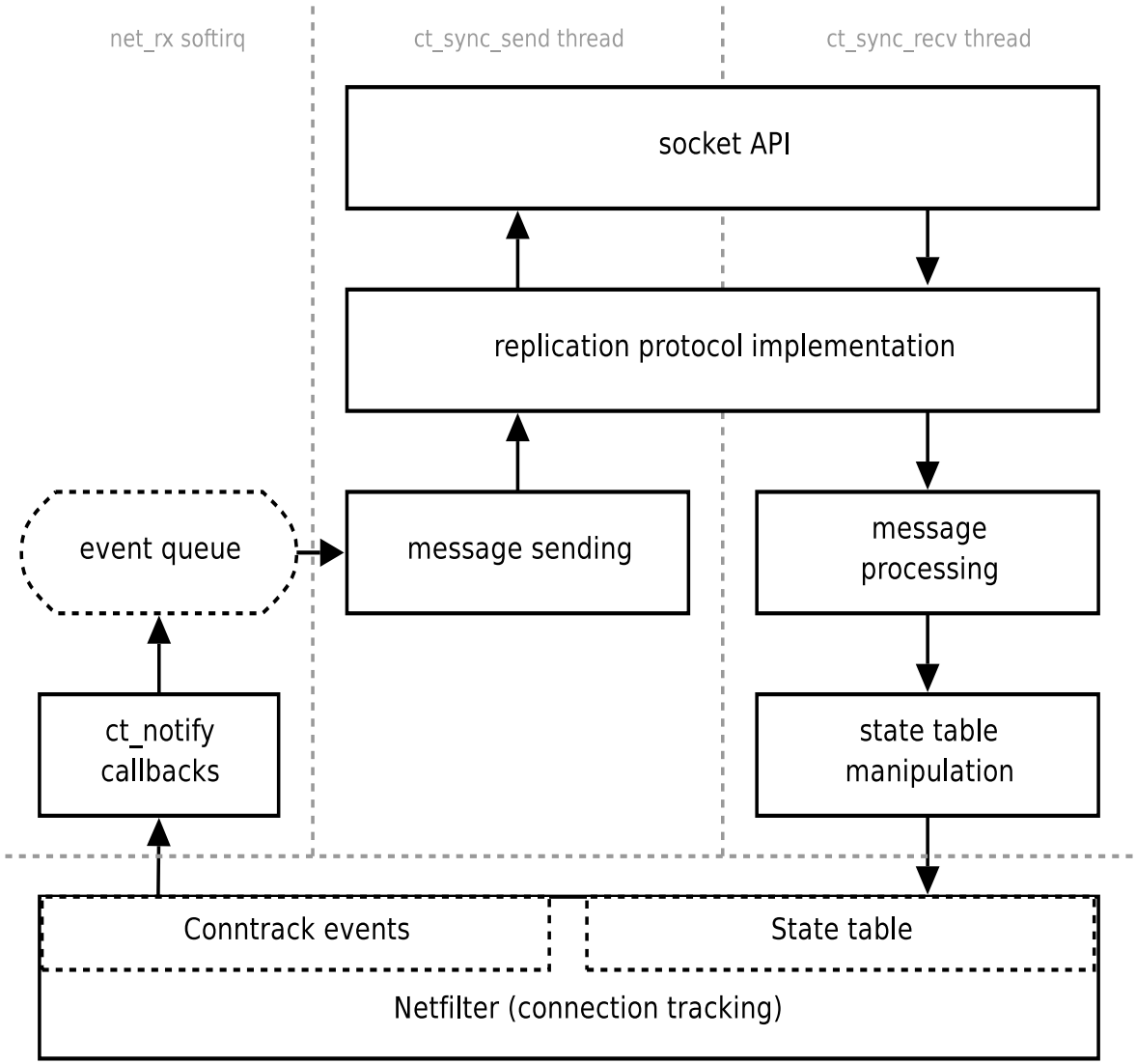
ct_sync

- State table replication solution for Netfilter
- Integrates tightly with the connection tracking system, does not modify other parts of Netfilter

Prerequisites

- Working IP failover solution (*keepalived*, *heartbeat*).
- Dedicated, preferably physically separated network for replication.
- A dedicated network interface on each node.

ct_sync Internals



Major components

- connection tracking events
- event queue
- sender thread
- receiver thread
- conntrack table manipulation

The Underlying Network Protocol

Assumptions about the environment

- Dedicated 100Mbit/1Gbit Ethernet (for two nodes: crossover Ethernet cable).
- High load on the firewalls, we have to minimize the overhead as much as possible.
- Optimize for the number of packets.

Dummy multicast protocol

- Multicast UDP based.
- Negative acknowledgement (NACK) based error detection.
- Capable of batching events and sending multiple update messages in one packet.

Error Detection and Recovery

Packet loss: detection

- Master sends out numbered packets and has a backlog with the last q packets sent.
- Slaves compare the sequence number of each packet with that of the previous received packet.
- If a slave detects a gap in sequence numbers it requests recovery in a NACK message (contains the seqno of the last message whose seqno was OK).

. . . and recovery

- If the master has all missing packets in its backlog: resend missing packets.
- Otherwise full re-synchronization follows. . .

Full re-synchronization

When is it necessary?

- If the master does not have all the missing bits in its backlog to recover a slave.
- A new node has been added to our cluster, or one of the nodes rebooted.

How can we do that?

- Master sends an update message for each element of its state table.
- These messages are sent in a scheduled manner (only n updates per second), real update messages are interleaved with these.

Additional Features I.

Conntrack exemptions

- Replication protocol packets can cause state changes by themselves, so new packets are sent describing these changes...
- We should exempt these packets from connection tracking (for example by using the NOTRACK target).

Built-in NOTRACK functionality

If enabled, the traffic through the dedicated interface is not processed by connection tracking.

Additional Features I.

Conntrack exemptions

- Replication protocol packets can cause state changes by themselves, so new packets are sent describing these changes...
- We should exempt these packets from connection tracking (for example by using the NOTRACK target).

Built-in NOTRACK functionality

If enabled, the traffic through the dedicated interface is not processed by connection tracking.

Additional Features II.

Services utilizing the virtual IP address

- Occasionally we would like to run other services on the firewall nodes utilizing the virtual IP (HTTP, etc.).
- Starting these daemons requires the virtual IP to be set up.
- It's convenient if we can do this at system start-up.

Layer 2 drop

Slave nodes drop *all* Ethernet frames except those arriving on the dedicated interface.

Additional Features II.

Services utilizing the virtual IP address

- Occasionally we would like to run other services on the firewall nodes utilizing the virtual IP (HTTP, etc.).
- Starting these daemons requires the virtual IP to be set up.
- It's convenient if we can do this at system start-up.

Layer 2 drop

Slave nodes drop *all* Ethernet frames except those arriving on the dedicated interface.

Additional Features III.

Partial synchronization

- In some cases we don't want to synchronize the complete state table.
- Could help lowering the synchronization overhead.

Using CONNMARK and ct_sync's cmarkbit parameter

If enabled, ct_sync only replicates conntrack entries with a given bit set in their connmark field.

Additional Features III.

Partial synchronization

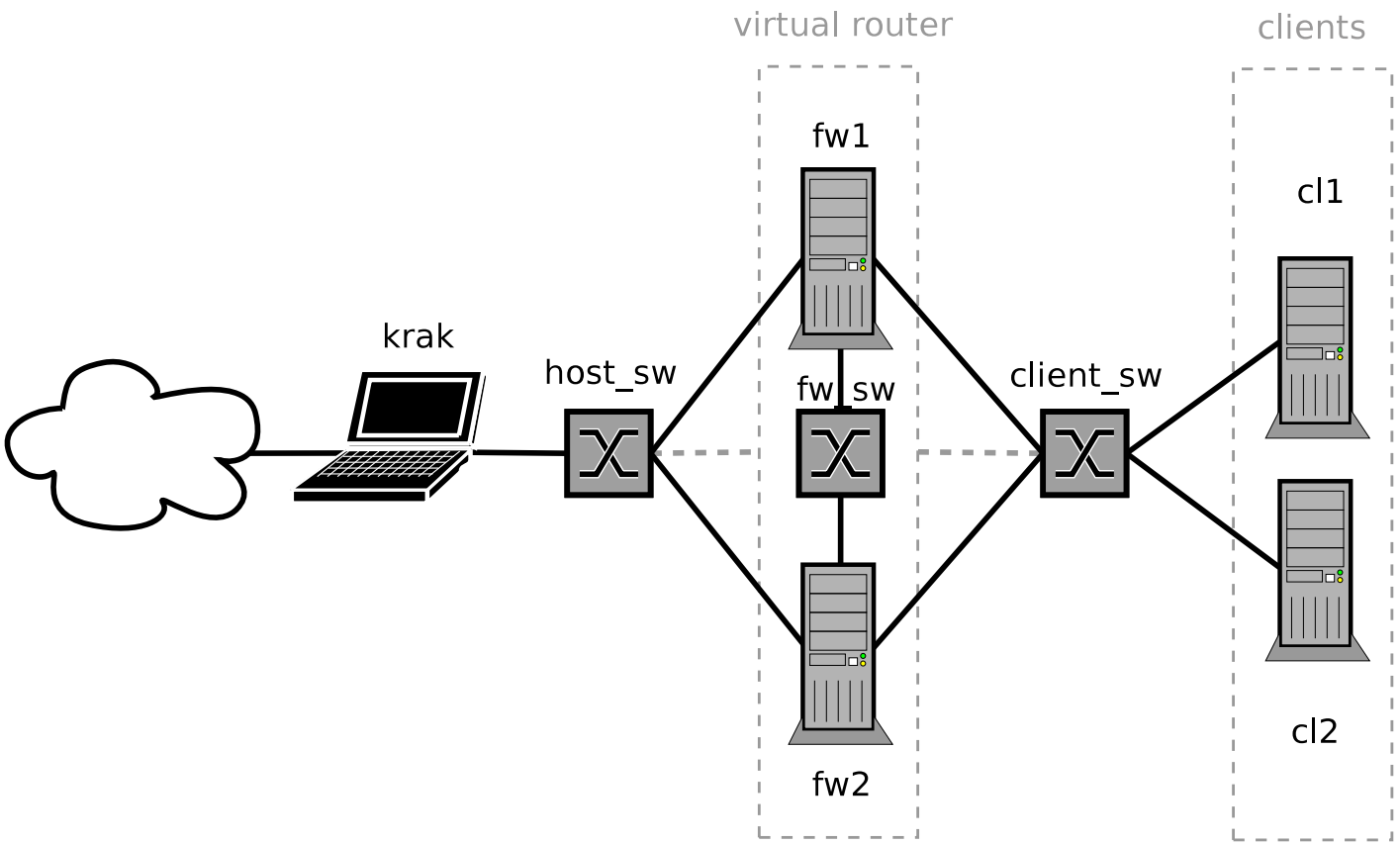
- In some cases we don't want to synchronize the complete state table.
- Could help lowering the synchronization overhead.

Using CONNMARK and ct_sync's cmarkbit parameter

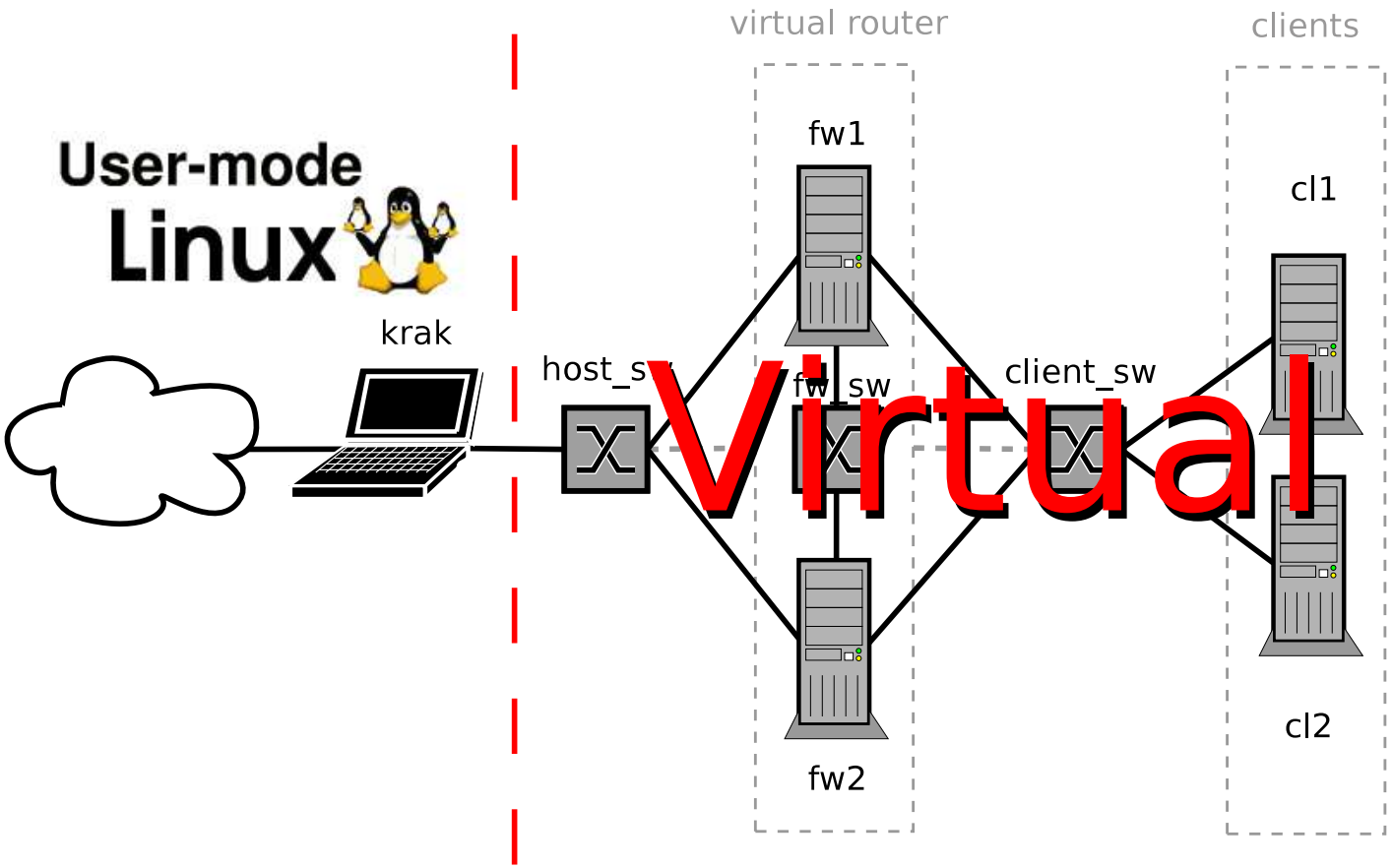
If enabled, ct_sync only replicates conntrack entries with a given bit set in their connmark field.

Overview

Example system



Example system



What Do We Have To Set Up?

IP failover

- Define VRRP groups.
- We want these synchronized, having the virtual IP on one interface and not on the other does not make sense.
- Set up hooks to notify ct_sync of the state transitions.

ct_sync

- Disable TCP window tracking.
- Load the ct_sync kernel module (tricky!).

What Do We Have To Set Up?

IP failover

- Define VRRP groups.
- We want these synchronized, having the virtual IP on one interface and not on the other does not make sense.
- Set up hooks to notify ct_sync of the state transitions.

ct_sync

- Disable TCP window tracking.
- Load the ct_sync kernel module (tricky!).

Defining the VRRP Instances

VRRP instance VI_1 on fw1

```
vrrp_instance VI_1 {
    interface eth0
    state MASTER
    virtual_router_id 61
    priority 80
    authentication {
      auth_type PASS
      auth_pass secret
    }
    virtual_ipaddress {
        192.168.1.254
    }
}
```

Defining the VRRP Sync Group

VRRP sync group on fw1

```
vrrp_sync_group G1 {
  group {
    VI_1
    VI_2
  }
  notify_master /root/script_master.sh
}
```

ct_sync Configuration

Kernel module

- ct_sync comes as a loadable kernel module.
- Mandatory parameter: syncdev.

Example

```
fw1# modprobe ct_sync syncdev=eth2 l2drop=1
```

Connecting ct_sync and Keepalived

What happens when failing over?

- keepalived detects that the old master does not send announcement messages anymore.
- A new master is elected and it configures the virtual addresses onto itself.
- keepalived running on the new master notifies ct_sync about the state transition.

keepalived.conf

```
notify_master /root/script_master.sh
```

/root/script_master.sh

```
echo 1 > /proc/sys/net/ipv4/netfilter/ct_sync/state
```

ct_sync Limitations and Future Plans

Current limitations

- Each node can participate in at most one VRRP group.
- TCP window tracking incompatibility.
- Does not replicate expectations.
- Bugs, bugs, bugs.

Planned features

- Multi-group capability, and thus sane VRRP setups.
- Active-active operation.

Summary

Not a silver bullet

- The target is reasonable operation using cheap hardware, not a perfect system.

Still under development

- Although progress is very slow due to lack of developers...

Further Information

Documentation

- Harald Welte, „ct_sync: state replication of ip_conntrack”, *Proceedings of the Ottawa Linux Symposium*, 2004, pp 537-545

  `http://www.finux.org/Reprints/Reprint-Welte-OLS2004.pdf`

Software

- Keepalived

  `http://keepalived.sf.net`

- `ct_sync`: Netfilter Subversion repository:

  `http://svn.netfilter.org/cgi-bin/viewcvs.cgi/branches/netfilter-ha/`

E-Mail

- Netfilter-failover mailing list:

  `https://lists.netfilter.org/mailman/listinfo/netfilter-failover/`