Netfilter updates: NetDev 2.1

<pablo@netfilter.org>
Pablo Neira Ayuso

What does this presentation cover?

- Not a tutorial... but incremental updates on Netfilter and nf_tables.
- For those new to nftables:
  - See http://people.netfilter.org/pablo/nft-tutorial.pdf
  - https://wiki.nftables.org
  - man nft(8)
- nf_tables replacement for {ip,ip6,eb,arp}_tables
- Heavy use of maps to reduces number of rule inspection
- nftables 0.7 (Dec 20th, 2016)

nf_tables performance numbers

- Dropping packets, with 4.11.0-rc+patch
- iptables from prerouting/raw:
  - iptables -I PREROUTING -t raw -p udp –dport 9 -j DROP
    6076928pps 2916Mb/sec
- nftables from ingress:
  - nft add rule netdev ingress udp dport 9 drop
    11855461pps 5690Mb/sec
- So nft was almost twice as fast as iptables! Cool!

New nf_tables extensions: **fib**

- Forward Internet Base (FIB) lookups
  - **Syntax:** fib *key data operator* expression
  - *key*: saddr, daddr, mark, iif, oif

    tuple represented through concatenation, eg. saddr . iif
  - *data*: oif, oifname, address type

    oif: output interface index

    oifname: output interface name

    address type:
    - unicast, local, broadcast, anycast, multicast, blackhole, unreachable, prohibit
  - *operator*: eq, neq, vmap, map

New nf_tables extensions: **fib** (2)

- Drop if reverse lookup fails (reverse path filter)
  - nft add rule filter prerouting fib saddr . iif oif missing drop
- Drop if there is not destination route for this packet
  - nft add rule filter prerouting fib daddr oif missing drop
- Drop packets to an address not configured on interface
  - nft add rule filter prerouting \
    fib daddr . iif type != { local, broadcast, multicast } drop
- Verdict map to perform action on address type:
  - nft filter prerouting meta mark set 0xdead \
    fib daddr . mark type vmap { \
    blackhole : drop, prohibit : jump prohibited, unreachable : drop }

New nf_tables extensions: **fib** (3)

- Integrates well with existing infrastructure and userspace, eg. Quagga
  - Remotely triggered black hole (RTBH) through BGP
    - drops unwanted traffic before entering protected network
- No ingress support yet
  - … but it should be very easy to add.

New nf_tables extensions: **rt**

- Access packet routing metainformation
  - **Syntax:** rt *key operator* expression
  - key: classid, nexthop

    nexthop: IPv4/IPv6 address

    classid: routing realm
    - Realm allows you to group routes via iproute2
    - in /etc/iproute2/rt_realms
  - *operator*: eq, neq, gt, lt, gte, lte, vmap, map

New nf_tables extensions: **rt** (2)

- Drop any traffic to 192.168.1.0/24 that is not routed via 192.168.0.1

  ```
  nft add rule filter postrouting \
      ip daddr 192.168.1.0/24 rt nexthop != 192.168.0.1 drop
  ```

- Count outgoing traffic per nexthop, times out after 10 minutes.

  ```
  nft add rule filter postrouting \
      flow table nh { rt nexthop timeout 600s counter }
  ```

- Dump content
  - nft list flow table filter nh

```
table ip filter {
        flow table nh {
                type ipv4_addr
                elements = { 142.154.64.1: counter packets 1026 bytes 332076,
                             24.19.12.1: counter packets 3405 bytes 212434 }
        }
}
```

New nf_tables extensions: **notrack**

- Explicitly disable connection tracking
  - **Syntax:** notrack
    - Needs to happen before the Connection Tracking
      - Hint: Before priority -300
- Traffic going to tcp/80 skips conntrack

```
nft add table raw
nft add chain raw prerouting { \
    type filter hook prerouting priority -300\; }
nft add rule raw prerouting tcp dport  80 notrack
```

New nf_tables extensions: **quota**

- Support for byte based quota
  - **Syntax:** quota {*over*} *value unit*
  - over: Optional, inverts matching criteria
  - value
  - unit: bytes, mbytes
- Enforce quota per flow

```
nft add rule raw prerouting \
    flow table http { \
        ip saddr timeout 60s quota over 50 mbytes } drop
```

- Packet-based quota should be easy to add too...

Updated nf_tables extensions: **payload**

- Update layer 4 checksum if field belongs to pseudoheader, eg. saddr, daddr
  - **Syntax:** ip {saddr,daddr} set expression
- Stateless NAT 1:1 for load balancing

```
nft add rule netdev filter ingress \
    ip saddr set numgen inc mod 2 map { \
        0 : 192.168.10.10, \
        1 : 192.168.10.11 }
```

Netfilter logging

- Required minimal changes to reuse the generic nf_log infrastructure from ingress.
- Print packet in human readable format to the kernel log buffer via pr_*() folks.
- Log some packets reaching the last rule in the policy
  - nft add rule netdev filter ingress \
    limit rate 2/second log prefix \"packet drop \" drop
  - *packet drop IN=wlan0 OUT= MAC=b1:24:a0:c6:96:a8:00:10:18:f3:57:44:08:00 SRC=8.8.8.8 DST=172.20.1.180 LEN=84 TOS=0x00 PREC=0x00 TTL=55 ID=40364 PROTO=ICMP TYPE=0 CODE=0 ID=1414 SEQ=108*
- New nf_log_all_netns sysctl.
  - Enables logging for all existing netns.
    - pernet syslog seems tricky and it's been discontinued...

Connection Tracking updates

- Two skbuff fields, on different cache lines:
  - skb->nfctinfo, only 3 bits
    - New, established, related + reply
  - skb->nfct, pointer to conntrack object
- … solution:
  - Rename skb->nfctinfo to skb->_ct
  - Store skb->nfctinfo (3 bits) stored in skb->_ct
  - Force mm to allocate objects aligned at 8 bytes for skb->_ct
- Remove timer per conntrack, use garbage collector
  - Get rid of struct timer
  - Add workqueue-based garbage collector
  - Remove central spinlock in NAT byaddr hashtable via rhashtable rhlist
- Results: Better performance, half less CPU consumption!

Connection Tracking updates (2)

- On-demand hook per-namespace registration nf_conntrack and defrag
  - Avoid hook cost if not needed according to policy
- UDPlite merged into UDP
  - Remove copy & paste code 8)
- SCTP is now built-in by default into conntrack
  - Problems with generic connection tracker and missing modprobe
  - Complainins on breaking SCTP from SOHO Linux-based routers

nf_tables named objects

- Provide replacement for iptables extended accounting infrastructure (nfacct)
  - Add named counters

    nfacct add http-traffic
  - Listing existing counters

    nfacct list
  - Atomic dump-and-reset

    nfacc list reset
- From iptables:

  iptables -A PREROUTING -t raw -p tcp –dport 80 \
  -m nfacct –nfacct-name http-traffic
- Extended later on to support quotas by Linaro
  - Including event notification on quota exceeded

nf_tables named objects (2)

- Reuse nfacct from nf_tables?
  - Not easy to do
    - No 2-commit phase protocol for atomic incremental updates
- nfacct was grown code:
  - Limited to counters, then quotas
  - Other stateful objects such as limit rates?
- Scalability problems: one rule per counter

nf_tables named objects (3)

- New nf_tables infrastructure to accomodate named objects
  - New NFT_MSG_{NEW,DEL,GET}OBJ commands
  - nft_register_obj() and nft_unregister_obj()
  - struct nft_object_type represents the object
    - netlink interface and attributes
    - eval function to access the object from the packet path
- Currently supported:
  - Counter
  - Quota
  - Ratelimit? Not yet, easy to add.

nf_tables named objects (4)

- Add new named counter
  nft add counter filter http-traffic
- Add new quota
  nft add quota filter http-traffic 25 mbytes
- nft add rule filter output \
  tcp dport https counter name http-traffic
- nft add rule filter output counter name tcp dport map { \
  443 : "https-traffic", \
  80 : "http-traffic", \
  22 : “ssh-traffic”, \
  25 : "smtp-traffic", \
  }

nf_tables named objects (5)

- Add map

```
nft add map filter badguys { \
        type ipv4_addr : counter \; }
```

- Reference it from rule

```
nft add rule filter input counter name \
        ip saddr map @badguys
```

- Add new counter objects to map

```
nft add counter filter badguy1
nft add counter filter badguy2
nft add element filter badguys { \
        192.168.2.3 : "badguy1" }
nft add element filter badguys { \
        192.168.2.4 : "badguy2" }
```

nf_tables named objects (6)

- List existing counters
  nft list counters table filter
- List existing quota
  nft list quotas
- Atomic dump and reset
  nft reset counter filter http-traffic
  table ip filter {
  counter http-traffic {
  packets 3134 bytes 12684312
  }
  }
- Same for quotas:
  nft list quota filter https-quota
  table ip filter {
  quota https-quota {
  25 mbytes used 2048 bytes
  }
  }

nf_tables ct helpers

- No automatic assignment of helpers anymore
  - Read “Secure use of iptables and connection tracking helpers”
- Helper lookup from packet path (now obsolete):
  - Conntrack helpers enabled via modprobe
  - look up for helper
  - Attach it to conntrack object
- Now wxplicit helper configuration
  - iptables -I PREROUTING -t raw -p tcp --dport 21 \
    -j CT --helper ftp

nf_tables ct helpers (2)

- New ct helper named object, eg.

```
helper “sip-5060” { \
    type sip protocol ip l4proto udp\; }
```

- From rules:

```
nft add rule x y udp dport 5060 \
     ct helper set “sip-5060”
```

- One single rule using dictionary:

```
nft add rule x y ct helper set udp dport map { \
    69 : “tftp-69”, \
    5060: “sip-5060” }
```

Migrating from iptables to nft

- Facilitate migration from iptables to nftables
  - iptables-translate
  - iptables-restore-translate
- 61 translations available (of 107 extension)
  - Some missing kernel code to be mapped
  - Some of them will not be translated: Obsolete
  - Missing code in the kernel
  - More details at wiki.nftables.org
- Test infrastructure available for translations

Migrating from iptables to nft (2)

- Let's make a quick demo...

nf_tables VM description

- Need a way to publicize to userspace a description of available capabilities (to be implemented)
- nf_tables VM is behind the Netlink interface curtain
  - Instructions available in the nf_tables VM
  - Netlink command types
  - Describe Netlink attributes in TLV format
    - Attributes that you can use with this instruction
    - Range of acceptable values for these attributes
- Why this?
  - Generate more optimized bytecode that runs faster based on VM capabilities
  - Deprecate things we don't want anymore

nf_tables VM description (2)

- … This is also useful for hardware offloads too
  - The bytecode must describe the rule in a simple way
  - No bytecode optimizations
    - eg. skip payload merge
  - *meta l4proto* special semantics
    - Hardware should use *ip6 nexthdr*
- If driver gets out of sync with software representation.

nf_tables VM description (2)

- New netlink NFT_MSG_GETVMDESC command
- Add nf_tables_desc.c file with descriptions
- Define structure that describes instruction
  - Compile breakage macro if description is missing
    - Developers don't forget to add these bits
- Transparent to the user
  - Implemented by nft command line tool userspace
  - Likely a new command to show VM capabilities for user reference in human readable format
    - eg. nft describe vm
  - … and in json for robots

nf_tables sets

- Set backend representation depends on:
  - Set implementation big O notation describes scalability in terms of performance and memory
  - estimated size in elements
  - Other useful description information, eg. Interval
- Good to hides set backend details behind the curtain
  - We can deprecate obsolete set representations
- Existing set backends.
  - Hashtable, via rhashtable.
  - Rbtree, for ranges. Replacement?
  - Bitmap, for key lengths <= 16 bits.
    - Several million packets faster than hashtable!

nf_tables sets (2)

- New description to represent subsets (not implemented):
  - nft add set x y { type ipv4_addr; \
    range 192.168.0.1-192.168.0.255 }
- Implementation details:
  - Add new flag NFT_SET_PARTIAL
  - Meta information key needs to be network byte order for memcmp()
  - Basic bignum to substraction to calculate offset
    - eg. 128 bits IPv6 address
- Thus we can select better representation, eg. bitmap

nf_tables sets (3)

- Catch-all element for maps:
  - Default action on no element found, eg.

```
nft add rule filter prerouting \
    ip saddr vmap { 1.2.3.4 : accept, \
                    1.2.3.5 : accept, \
                    * : drop }
```

- Add more optimized backend implementations
  - Constant sets don't need a resizable hashtable...
  - Worth a hashtable for 2 or 3 elements?
    - Add very specialized silly sets, eg. List or array

nf_tables sets (4)

- Add set if it doesn't exist, do nothing if exists

  nft add set x y { type ipv4_addr\; }

- Create command, bails out if set exists, eg.

  nft create set x y { type ipv4_addr\; }
  <cmdline>:1:1-35: Error: Could not process rule: File exists
  create set x y { type ipv4_addr; }
  ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^

- Flush set elements

  nft flush set x y

- Inverted set lookups

  nft add rule x y tcp dport != { 80, 443 } drop

nf_tables sets (5)

- People like names

  nft add element filter badguys { evil-target.com, bad-guy.com }

- Problem: DNS names not realiable for policies
- Solution: Use variables
  define bad_guy_org = 1.2..3.4
  define evil_target_com = 4.3.2.1
  define bad_people = { $bad_guy_org, $evil_target_com }
- Include it from master policy file
  include “bad-guys.nft”
  add set filter bad-people { type ipv4_addr; }
  add element filter bad-people $bad_people
- Good points:
  - Useful to improve ruleset maintainability
  - Robots can autogenerate this

Rule deletion by description

- Just like in iptables, ie.

  nft delete rule filter prerouting tcp dport 80 notrack

- So you don't need to:

  nft list ruleset -a # list rules with unique handle number

  nft delete rule filter prerouting handle 85

- Need to deal with anonymous sets,
  - eg. Ip saddr { 1.2.3.4, 1.2.3.5, 1.2.3.6 }
  - Sort elements and perform full comparison
  - Look up for rule handle
- Feature ready, but patchset still incomplete in userspace
  - Fix assymmetries between linearize and delinearize path

Improve error reporting

- # nft add rule filter prerouting udp dport 53 ip daddr 8.8.8.8 drop
  <cmdline>:1:1-62: Error: Could not process rule: No such file or directory
  add rule filter prerouting udp dport 53 ip daddr 8.8.8.8 drop
  ^^
- Too sparse grain, better would be...
- # nft add rule filter prerouting udp dport 53 ip daddr 8.8.8.8 drop
  <cmdline>:1:1-62: Error: Could not process rule: No such file or directory
  add rule filter prerouting udp dport 53 ip daddr 8.8.8.8 drop
  ^^^^
- Got patches already to kernel to support this:
  - Missing nfnetlink, libmnl and userspace code

Netfilter updates: NetDev 2.1
Questions?

<pablo@netfilter.org>
Pablo Neira Ayuso