

# nftables

## budoucnost linuxového firewallu

Petr Krčmář



18. listopadu 2014



Uvedené dílo (s výjimkou obrázků) podléhá licenci Creative Commons Uvedte autora 3.0 Česko.

[www.petrkrcmar.cz](http://www.petrkrcmar.cz)

# Historie síťového filtru v Linuxu

- ipfwadm
  - 1995 - 1999
  - jádro 1.2.1 - 2.2.0
  - bezstavový filtr, IPv4 only, NAT vedle
  - každý paket se posuzoval zvlášť
  - méně bezpečné = nutno otevírat všem nebo nikomu
- ipchains
  - 1999 - 2001
  - jádro 2.2.0 - 2.4.0
  - podpora IPv6, více protokolů
  - stále bezstavový filtr
  - stále stejné nevýhody bezstavovosti

# Současnost síťového filtru v Linuxu

- iptables/netfilter
  - 2001 - ?
  - jádro 2.4.0 - současná
  - univerzální netfilter (hooky), conntrack a NAT
  - + iptables implementující obecné tabulky s pravidly
  - výsledkem plně stavový filtr (umí i bezstavový)
- nftables
  - leden 2014 - budoucnost
  - jádro 3.12 - současná
  - projekt ale běží od roku 2008
  - zatím stále ve vývojové fázi
  - nenahrazuje netfilter, jen iptables
  - vyvíjí netfilter core team, není to akce proti iptables

# Milníky v historii nftables

„...the biggest change to Linux firewalling since the introduction of iptables in 2001“ – Patrick McHardy

- projekt představen na Netfilter Workshop 2008
- první implementace v březnu 2009
- poté projekt vypadal mrtvě, zmizel i web
- v březnu 2010 přechod do beta verze
- říjen 2012 rozhraní kompatibilní s iptables
- říjen 2013 pull request do jádra
- v lednu 2014 zařazeno do jádra 3.13
- stále probíhá intenzivní vývoj (zdaleka není hotovo)

# Proč nový firewall?

- IPtables jsou neefektivní, přerostlé a málo dynamické
- duplikují kód: iptables, ip6tables, arptables, ebtables
- obsahují spoustu kódu pro konkrétní protokoly
- např. kód pro získání portu z UDP a TCP je jiný
- rozšíření funkčnosti = tvorba jaderného modulu
- např. xt\_dns pro zkoumání typů DNS provozu
- v jádře jsou zhruba dvě stovky modulů k IPtables
- spousta věcí dodrátovaných dodatečně – ipset
- přidání či změna pravidla jsou výkonnostní problém
  - vše se pošle do userspace, upraví a nahraje zpět
  - často je výhodnější save a restore

# Jak to řeší nftables?

- neobsahují **žádný** kód, který by rozuměl protokolům
- implementují jednoduchý virtuální stroj
- ten dostává „program“ od uživatelské utility
- instrukcí je minimum, pokrývají všechny potřeby
- snadno lze přidat další
- rozšíření funkčnosti = úprava uživatelské utility nft
- není třeba měnit jádro (!)
- např. podpora icmpv6 = 100řádkový patch
- desetina kódu v jádře
  - IPtables: 70K řádek v jádře, 50K v userspace
  - nftables: **7K řádek** v jádře, 50K v userspace

# Univerzálnost nftables

- nftables neobsahují kód pro hledání IP adres
- nic jako „porovnej IP adresu paketu se 192.168.1.1“
- místo toho se vloží univerzální kód

```
payload load 4 offset network header + 16 => reg 1  
compare reg 1 192.168.1.1
```

- první řádek skočí v hlavičce o 16 bytů
- pak načte čtyři byty do reg 1
- druhý řádek porovná obsah reg 1
- stroj nftables používá opcode (bytecode)
- vychází z Berkeley Packet Filter



# Mnoho možností

- takový přístup je velmi mocný
- jazyk umožňuje mnoho věcí

```
payload load 4 offset network header + 16 => reg 1
set lookup reg 1 load result in verdict register
{ "192.168.1.1" : jump chain1,
  "192.168.1.2" : drop,
  "192.168.1.3" : jump chain2 }
```

- vyhledání adresy ve slovníkové tabulce
- nahrazuje ipset
- předání paketů do dalších řetězců

# Možnost slučování akcí

- použití více slovníků či map
- jeden paket může být testován na více vlastností
- i **více akcí** v řetězci: započítat, logovat, zahodit
- výsledek = není třeba opakovat testy vlastností
- nftables dostávají od network stacku metadata
- data z conntracku, metadata – délku paketu, protokol, adresy a další

# Co je jinak proti IPtables

- syntaxe je podobná iproute2
- parametry se zapisují bez pomlček ve volném pořadí
- změny jsou atomické a rychlé
- je možné bez problémů za běhu měnit pravidla
  - jediná transakce v Netlinku
  - minimum komunikace z userspace do jádra
- nftables nemají žádné vestavěné řetězce
- počítadla je třeba explicitně zapínat

# Co budete potřebovat

- jádro > 3.13, lépe > 3.14, nejlépe co nejnovější
- utilitu nft (v Jessie balík nftables)
- není třeba psát programy ručně, dělá to utilita
- má i možnost dekompilace = vytvoří čitelný výstup

- tabulka (table) – kontejner pro řetězce a sety
  - ip
  - ip6
  - arp
  - bridge
  - inet
- neexistují žádné předdefinované tabulky
- výchozí tabulka je ip

- řetězec (chain) – kontejner pro pravidla
- leží vždy uvnitř tabulky
  - filter
  - route
  - nat
- může mít hook, typ a prioritu
- neexistují žádné předdefinované řetězce

- hook – jaderný výstup z netfilteru
- je na něj možné zavěsit callback
- pak se provoz posílá do řetězce
- hook ale **není nutný**
  - prerouting – vše vstupující do počítače
  - forward – pakety pro jiné počítače vyžadující forward
  - input – pakety pro lokální počítače
  - output – pakety pocházející z lokálního počítače
  - postrouting – pakety opouštějící počítač

# Terminologie – pravidlo

- pravidla – výrazy sledující parametry paketů
  - ip – IP protokol
  - ip6 – IPv6 protokol
  - tcp – TCP protokol
  - udp – UDP protokol
  - udplite – UDP-lite protokol
  - sctp – SCTP protokol
  - dccp – DCCP protokol
  - ah – IPsec AH režim
  - esp – IPsec ESP režim
  - ipcomp – IPcomp hlavičky
  - icmp – icmp protokol
  - icmpv6 – icmpv6 protokol
  - ct – connection tracking
    - meta – metadata jako síťová rozhraní
- každý má své parametry



- možné řetězit bez opakování testů
- je možné vykonat následující akce
  - accept – pušť paket a dál ho nezkoumej
  - drop – zahod' paket a dál ho nezkoumej
  - reject – zahod' paket a pošli o tom ICMP zprávu
  - queue – předej paket do userspace (libnetfilter\_queue)
  - snat – zařid' source NAT
  - dnat – zařid' destination NAT
  - jump – skoč do jiného řetězce
  - return – ukonči současný řetězec a vrať se
  - goto – jako jump, ale bez návratu
  - counter – započítej paket
  - log – zaloguj aktivitu

## Praktické použití

# Založení tabulky

- založíme si tabulku pro ipv4-filter

```
# nft -f /etc/nftables/ipv4-filter
```

- podíváme se, co se založilo

```
# nft list table filter
```

# Přidání pravidel

- přidáme si pravidla

```
# nft add rule ip filter output ip\  
daddr 77.78.107.135 reject
```

- podíváme se, co se přidalo

```
# nft list table filter
```

# Vložení pravidel

- pravidla je možné strkat před/mezi předchozí
- pomocí `-a` si zjistíme číslo pravidla
- pomocí `add` vložíme **za** pravidlo
- pomocí `insert` vložíme **před** pravidlo

```
# nft nft list table filter -n -a
# nft add rule filter output position 10 \
ip daddr 1.2.3.4 reject
# nft insert rule filter output position 10 \
ip daddr 5.6.7.8 reject
```

- přidáme si počítadlo

```
# nft add rule filter output ip daddr 8.8.8.8 counter
```

- podíváme se na stav
- parametr -nn vypne host lookup

```
# ping 8.8.8.8  
# nft list -nn table filter
```

# Limity a logování

- možné limitovat různé druhy provozu
- příklad s limitem pingů

```
# nft add rule filter input icmp type echo-request \  
  limit rate 10/second accept
```

- pro logování je třeba modul ipt\_LOG
- informace se sypou do syslogu

```
# nft add rule filter input tcp dport 22 ct state new \  
  log prefix "Pripojeni pres SSH: \  
  " accept
```

# Výrazy, rozsahy a skoky

- k dispozici !=, >, <, >=, <=
- pozor, v shellu nutno escapovat
- možné použít i rozsahy

```
# nft add rule filter output tcp dport != 22 counter  
# nft add rule filter input tcp dport 1-1024 counter reject
```

- skoky do jiných řetězců
- jump, je možné se vrátit pomocí return
- goto skok bez návratu
- řetězce **nemusí mít** hooky

```
# nft add chain ip filter sshcko  
# nft add rule filter input tcp dport 22 jump sshcko  
# nft add rule ip filter sshcko counter  
# nft add rule ip filter sshcko return
```



- NAT má zvláštní pravidla
- pouze první paket nového flow projde do řetězce NAT
- zavede se nová vazba, případně upraví paket
- poté ostatní pakety upravují podle vzniklé vazby
- řetězec prerouting musíte vytvořit, i když nemá pravidla – aktivuje odnatování paketů

```
# nft add table nat
# nft add chain nat prerouting { type nat \
hook prerouting priority 0 \; }
# nft add chain nat postrouting { type nat \
hook postrouting priority 0 \; }
# nft add rule nat postrouting ip saddr \
192.168.1.0/24 oif eth0 snat 195.18.52.55
```

- nftables má zabudovanou podporu setů
- není potřeba žádná nadstavba typu ipset
- lze použít sadu **libovolných** selektorů
- připravené jsou slovníky a mapy (další slide)
- jednoduché anonymní sety nebo pojmenované

```
# nft add rule filter output tcp dport { 22, 23 } counter

# nft add set filter zlobivaci { type ipv4_addr\;}
# nft add element filter zlobivaci { 192.168.1.4 }
# nft add element filter zlobivaci \
  { 192.168.2.8, 192.168.2.15 }
# nft add rule ip input ip saddr @zlobivaci reject
```

# Slovníky a mapy

- slovníky umožňují propojení elementu a akce
- ohromné zjednodušení, nic takového iptables nemají

```
# nft add rule ip filter input ip protocol vmap \  
{ tcp : jump tcp-chain, udp : jump udp-chain, \  
  icmp : jump icmp-chain}  
# nft add rule ip filter input counter drop
```

- mapy přiřazují výstup ke vstupu (interně set)
- například různou NAT adresu podle portů

```
# nft add rule ip filter input dnat set tcp dport \  
map { 80 : 192.168.1.100, 8888 : 192.168.1.101 }
```

# Ukládání a obnova

- nastavené tabulky je možné uložit
- poté triviálně obnovit

```
# nft list table filter -n > ulozeno  
# nft -f ulozeno
```

- lze použít i XML nebo JSON

```
# nft export xml  
# nft export json
```

# Monitoring událostí

- je možné monitorovat události v nftables
- sledovat objekty (tables, chains, rules, sets, elements)
- nebo jejich změny (new, destroy)
- výstup v XML nebo JSON

```
# nft monitor
# nft monitor rules
# nft monitor new
# nft monitor new rules xml
```

# Mazání všeho možného

- řetězec lze smazat, pokud v něm nejsou pravidla
- tabulku lze smazat, pokud v ní nejsou řetězce
- pro mazání pravidel je třeba si vypsat jejich čísla
- mazání jako v iptables není možné, je v plánu

```
# nft list table filter -a
# nft delete rule filter input handle 11
```

- kompletní vyprázdnění tabulky
- vymazání řetězců
- vymazání tabulky

```
# nft flush table filter
# nft delete chain ip filter input
# nft delete chain ip filter output
# nft delete chain ip filter forward
# nft delete table filter
```

# Kompatibilita s IPtables

- pro hladký přechod existuje vrstva kompatibility
- je možné přímo použít pravidla pro IPtables
- hodně vývojové stádium, je v tom dost zmatek
- <https://git.netfilter.org/iptables/>
- utilita iptables-compat
- v plánu je konverzní utilita

- autor už v roce 2008 dělal benchmarky
- výsledky srovnatelné s IPtables
- prostor pro optimalizace
- používají se binární stromy
- optimalizované datové struktury
- mnoho věcí je možné sloučit
- testy se neopakují, vícenásobné akce
- nftables mají desetkrát méně kódu
- teoreticky to může být i výkonnější



## Otázky?

Petr Krčmář  
petr.krcmar@iinfo.cz