

Moderní linuxový firewall s nftables

Petr Krčmář



7. listopadu 2021



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

www.petrkrcmar.cz

- linuxák od roku 1998
- správce serverů
- lektor a konzultant
- šéfredaktor [Root.cz](#)
- člen [vpsFree.cz](#)
- organizátor [LinuxDays](#)
- můj web je [petrkrcmar.cz](#)



Historie síťového filtru v Linuxu

Historie síťového filtru v Linuxu

- ipfw
 - 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

Historie síťového filtru v Linuxu

- ipfw
 - 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 bezstavosti

Současnost síťového filtru v Linuxu

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)

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
 - poprvé v jádře 3.13
 - projekt ale běží od roku 2008
 - 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
- prvotní vydání, poté další vývoj

Proč nový firewall?

- IPtables jsou neefektivní 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
 - délka paketu, protokol, adresy a další

Co je jinak proti IPtables

- syntaxe je podobná firewallu **pf** ze světa BSD
- nebo jako u iproute2 (ip) či tcpdump
- 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, doporučeno 4.10, nejlépe co nejnovější
- modul `nf_tables`, vyflushované `iptables`
- utilitu `nft` (v Debianu balík `nftables`)
- není třeba psát programy ručně, dělá to utilita
- má i možnost dekompilace = vytvoří čitelný výstup
- umožňuje uložit a načíst stav

Terminologie – tabulka

- tabulka (table) – kontejner pro řetězce a sety
 - ip
 - ip6
 - inet (IPv4 + IPv6)
 - arp
 - bridge
 - netdev
- neexistují žádné předdefinované tabulky
- výchozí tabulka je ip

Terminologie – řetězec

- 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

Hooky k tabulkám

- hooky pro ip, ip6 a inet
 - prerouting, input, forward, output, postrouting
- hooky pro arp
 - input, output
- bridge
 - sleduje data tekoucí rozhraním
- hook pro netdev
 - ingress (data přicházející na rozhraní)

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

Terminologie – akce

- možné řetězit bez opakování testů
- je možné vykonat následující akce
 - accept – pusť 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.conf
```

- podíváme se, co se založilo

```
# nft list table inet filter
```

Přidání pravidel

- přidáme si pravidla

```
# nft add rule inet filter output ip\  
daddr 77.78.107.135 reject  
# nft add rule inet filter output ip\  
daddr www.linuxdays.cz reject  
# nft add rule inet filter output ip6\  
daddr 2a01:430:17:1::ffff:613 reject
```

- podíváme se, co se přidalo

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

- přidáme si počítadlo

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

- podíváme se na stav

```
# ping 8.8.8.8  
# nft list table inet filter
```

Limity a logování

- možné limitovat různé druhy provozu
- příklad s limitem SSH spojení
- možno omezovat i toky (od jádra 4.3)

```
# nft add rule inet filter input tcp dport ssh \  
  limit rate 15/minute counter accept  
# nft add rule filter input limit rate \  
  10 mbytes/second burst 9000 kbytes accept
```

- pro logování je třeba modul ipt_LOG
- informace se sypou do syslogu
- čili dnes journalctl -f

```
# nft add rule inet filter output 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 proměnné a rozsahy

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

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

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

- NAT má zvláštní pravidla
- pouze první paket flow projde do řetězce NAT
- zavede se nová vazba, případně upraví paket
- poté ostatní pakety upravují podle vzniklé vazby
- jádro <4.18 bylo třeba založit (prázdný) prerouting

```
# 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 inet filter output tcp dport { 22, 23 } \  
counter  
  
# nft add set inet filter zlobivaci { type ipv4_addr\  
# nft add element inet filter zlobivaci { 192.168.1.4 }  
# nft add element inet filter zlobivaci \  
{ 192.168.2.8, 192.168.2.15 }  
# nft add rule inet filter input ip saddr @zlobivaci \  
reject
```


Slovníky a mapy

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

```
# nft add rule inet filter input ip protocol vmap \  
  { 22 : accept, 23 : drop, 25 : jump posta }  
# nft add rule inet filter input counter drop
```

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

```
# nft add rule ip filter prerouting 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 inet filter > ulozeno  
# nft -f ulozeno
```

- lze použít export do JSON

```
# nft -j list ruleset
```

Skriptování

- není potřeba používat shellové skripty
- vše možné naskriptovat s nftables
- include, proměnné, sety...

```
#!/usr/sbin/nft

flush ruleset # všechno vylejt
include "pravidla-pro-nat.ruleset" # vložit soubor
define google_dns = 8.8.8.8 # definice proměnné a setu
define ntp_servers = { 84.77.40.132, 176.31.53.99, 81.19.96.148 }

add table inet filter
add chain inet filter input { type filter hook input priority 0; \
    policy drop; }
add rule inet filter input ct state established,related counter accept

add rule inet filter input ip saddr $google_dns counter
```

Monitoring událostí

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

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

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

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

- kompletní vyprázdnění tabulky
- vymazání řetězců
- vymazání tabulky
- ultimátní vymazání všeho

```
# nft flush table filter
# nft delete chain inet filter input
# nft delete table filter
# nft flush ruleset
```

Kompatibilita s IPtables

- balíček iptables (v Debianu) obsahuje nové utility
 - iptables-translate pro překlad z ipt do nft
 - iptables-nft pro načtení staré syntaxe do nf_tables
 - iptables-legacy je původní IPtables
 - plus všechny varianty jako ebtables-, ip6tables-, ...
- Debian dává pomocí alternatives prioritu nftables
- rozhodně **nemíchejte** použití x_tables a nf_tables
 - může to vést k „neočekávaným výsledkům“

```
# iptables-save > iptables.txt
# iptables-nft-restore < iptables.txt
```

Jednoduchý firewall s nftables

```
table inet firewall {
  chain input {
    type filter hook input priority 0; policy drop;
    iifname "lo" accept
    ct state established,related accept
    meta l4proto icmp meta nfproto ipv4 \
        limit rate 10/second accept
    meta l4proto ipv6-icmp meta nfproto ipv6 \
        limit rate 10/second accept
    tcp dport { ssh, https } accept
    udp dport { 33434-33499 } reject
  }
}
```

- Firewall s nftables na Root.cz
- wiki.nftables.org
- `man nft`
- Phil Sutter, Red Hat: [Benchmarking nftables](#)

Otázky?



Petr Krčmář
petr.krcmar@iinfo.cz