Migration vers nftables

Capitoul 20 avril 2023

Laurent GUERBY
IMT Mines Albi

Contexte

- Pare-feu Stormshield SN900 hors support, non redondé
- Passage progressif du reste du SI IMT Mines Albi en virtualisé sous proxmox+ceph
- Mise en place prévue PCA/PRA

Plus et moins

- Avantage du virtualisé : uniformité du SI, backups, OS standard avec outils git, mise à jours logicielles et matérielles aisées
- Pour PCA/PRA si appliance alors il faut en acheter pour l’autre site et développer la synchronisation
- Inconvénient du virtualisé : performances pas garanties par un ASIC ou FPGA, DDoS ?
- Note : le monde de l’appliance pare-feu a pas mal d’activité SSI CVE / supply chain ces derniers temps ...

Choix

- En virtualisé : dérivé pf BSD, iptables ou nftables Linux, 6wind DPDK
- Pas de familiarité dans l’équipe avec BSD ni 6wind
- Nftables semble abordable pour du DIY
- Nftables déjà utilisé par des outils comme crowdsec et firewalld

Préparation de la migration

- Switch avec VLAN en interception devant le stormshield pour tous les liens - au lieu de liens arrivant directement sur un des ports du Stormshield.
- Bascule Stormshield => Machine virtuelle nftables en faisant shut sur les ports du switch vers stormshield et en allumant l'interface trunk de la VM sous proxmox
- Retour en eteignant l'interface trunk de la VM puis "no shut" sur switch
- Juste un petit temps pour mise a jour table MAC et ARP (arping -U au besoin)
- Moulinette python3 maison pour analyser et convertir les définitions Stormshield (fichiers objet,objectsgroup,network,networkgroup, servicegroup, Filter/NN)
- ~ 400 lignes python3, ~200 règles, ~700 définitions post garbage collect

Iptables => nftables

- https://en.wikipedia.org/wiki/Nftables
- Since Linux kernel 3.13 released on 19 January 2014
- iptables -A OUTPUT -d 1.2.3.4 -j DROP
- nft add rule ip filter output ip daddr 1.2.3.4 drop

Outils iptables => nftables

- Lien https://wiki.nftables.org/wiki-nftables/index.php/Moving_from_iptables_to_nftables
- % iptables-translate -A INPUT -p tcp --dport 22 -m conntrack --ctstate NEW -j ACCEPT

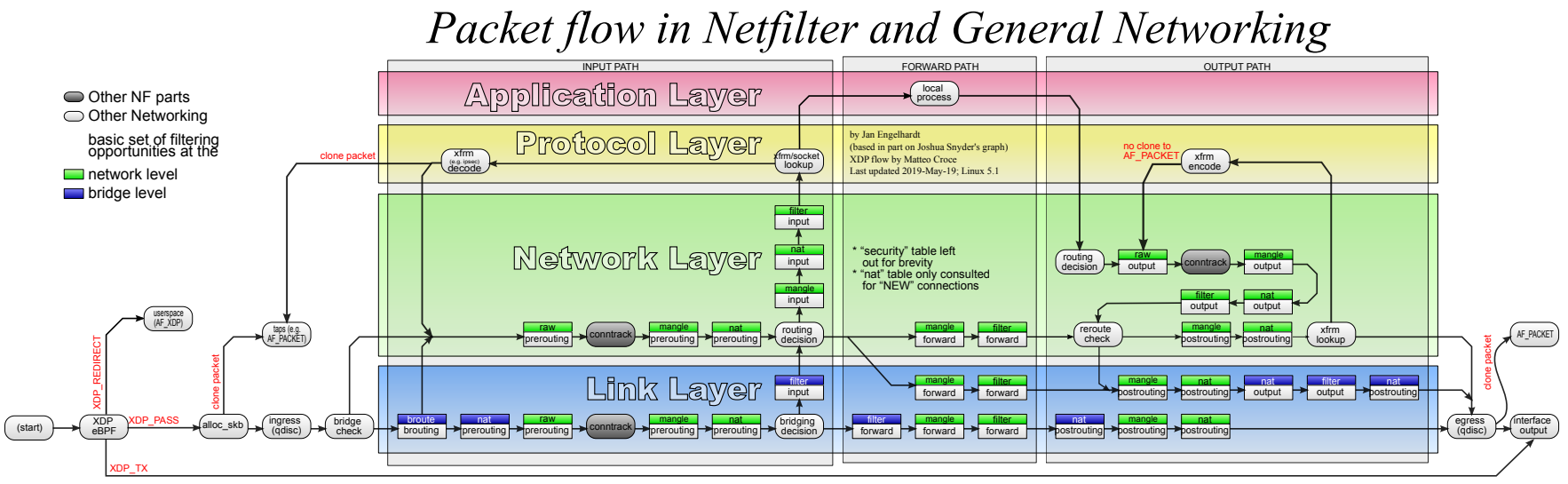
  nft add rule ip filter INPUT tcp dport 22 ct state new counter accept
- % iptables-save > iptables.txt
- % iptables-nft-restore < iptables.txt
- (Outils pas utilisés pour la migration IMT Mines Albi)

Nouveautés nftables

- Remplacement atomique de ruleset
- Tables et chaines plus génériques
- Ajout, modification et retrait de règles plus simples
- Et surtout sets, maps, vmaps, meters
- Lien https://wiki.nftables.org/wiki-nftables/index.php/Main_differences_with_iptables

Netfilter comme avant

- Lien https://upload.wikimedia.org/wikipedia/commons/3/37/Netfilter-packet-flow.svg



/etc/nftables.conf

- ```
  #!/usr/sbin/nft -f

  flush ruleset

  define prive-172_16 =  172.16.0.0/16

  define my_host = 192.168.1.100

  define netbios-ns = {  tcp . 137 , udp . 137 }

  define all_udp_above_1024 = { udp . 1025-65535 }
  ```
  ```

define

- ```
  define grp-abus = { $host1, $host2 }

  define grp-services-Eduroam = { $domain_udp,$ftp,$http,
  $https,$ssh,$imap,$imaps,$pop3,$pops,$smtp,$smtp-
  587,$domain,$ntp,$openVPN,$openVPN-udp,$http_proxy }

  define grp-reseau = { $prive-172_16, $prive-10 }
  ```
  ```

define if

- ```
  define if_rt_site2 = eth0.38

  define if_ariane = eth0.39

  define if_alize = eth0.40


  define if_internet = { $if_rt_site2 , $if_ariane }

  define if_nat = { $if_rt_site2 }
  ```
  ```

Table raw

- table my_raw {

```
    chain my_notrack_in {
      type filter hook prerouting priority raw;
      iif eth1 notrack;
    }

    chain my_notrack_out {
      type filter hook output priority raw;
      oif eth0.16 ip daddr A.B.C.D udp dport 514 notrack;
      oif eth1 notrack;
    }

  }
```

Table inet

```
table inet my_filter {
  chain my_forward {
    type filter hook forward priority filter;
    ip saddr $grp-abus log prefix "DP100 " flags all counter drop;
    ip daddr $grp-abus log prefix "DP110 " flags all counter drop;
    ct state established counter accept;
    ip protocol icmp ct state related log prefix "AT240 " flags all counter accept;
    ct state related log prefix "DP250 " flags all counter drop;
```

SSI related

- L'utilisation de ct state (connection tracking state) permet d'eviter une fois la connection établie de repasser toute les règles a chaque paquet donc d'optimiser les performances et diminuer le bruit dans les logs.
- A noter le ct state related spécifique a ICMP : il permet a un client TCP a l'intérieur du réseau de l'école de bien recevoir les paquets ICMP pour les "unreachable" & cie - sinon un client reste juste coincé jusqu'a timeout.
- Le related est limité a l'ICMP car il peut être problématique sur d'autres protocoles.
- https://gist.github.com/azlux/6a70bd38bb7c525ab26efe7e3a7ea8ac

  Problème d'ouvertue de port non désirée sur une configuration IPtables
- https://github.com/rtsisyk/linux-iptables-contrack-exploit
- https://github.com/regit/secure-conntrack-helpers/blob/master/secure-conntrack-helpers.rst
- https://www.synacktiv.com/en/publications/icmp-reachable.html

Table inet 2

- ip saddr $grp_prive oif $if_internet meta l4proto . th dport $grp_port_http log prefix "AT349 " flags all counter accept;

  iif $if_internet ip daddr $grp_srv_dns meta l4proto . th dport $grp_port_dns log prefix "AT464 " flags all counter accept;

  ip saddr $prive-10 ip daddr $prive-172_16 meta l4proto . th dport {$https,$ssh} log prefix "AT446 " flags all counter accept;

  log prefix "DROP999 " flags all counter drop;

SNAT

```
table ip my_nat {
  map my_map_nat {
    type ipv4_addr : ipv4_addr ;
    flags interval;
    elements = {
       192.168.100.0/24 : $nat_pour_192.168.100,
       192.168.101.0/24 : $nat_pour_192.168.101
    }
  }
  chain my_postrouting {
    type nat hook postrouting priority srcnat;
    oif $if_nat snat ip prefix to ip saddr map @my_map_nat;
  }
}
```

Tools

- Executer /etc/nftables.conf
- Pas d’impact si erreur de syntaxe
- nft list ruleset # -j pour json
- nft add element my_nat my_set_nat { 10.0.0.0/24 : 1.2.3.4 };
- Python (pas regardé)

Configuration

- `apt-get install isc-dhcp-relay arping mtr-tiny nftables conntrack rsyslog`
- `# cat /etc/rsyslog.d/50-local.conf`

  ```
  *.* @A.B.C.D
  ```

- `# cat /etc/rc.local`

  ```
  #!/bin/bash

  /usr/bin/screen -dmS conntrack-logger bash -c 'conntrack -E -o
  extended,timestamp,id -b 16777216 | logger --rfc5424=notime,nohost -t CT
  -d -n A.B.C.D'
  ```

Network (ifupdown2)

- cat /etc/network/interfaces

```
auto eth0.50
iface eth0.50 inet static
    address 172.16.50.254/24
    up ip route add 172.20.0.0/16 via 172.16.50.1
    up ip route add 172.21.0.0/16 via 172.16.50.1
iface eth0.60
   address 192.168.60.2/24
   gateway 192.168.60.1
```

Sysctl

- net.ipv4.ip_forward=1
- net.ipv6.conf.all.forwarding=1
- net.netfilter.nf_conntrack_acct=1
- net.netfilter.nf_conntrack_timestamp=1
- net.netfilter.nf_conntrack_max=16777216
- kernel.printk = 3 4 1 3

Sysctl 2

- net.ipv4.neigh.default.gc_thresh3=24456
- net.ipv4.neigh.default.gc_thresh2=12228
- net.ipv4.neigh.default.gc_thresh1=8192
- net.ipv4.neigh.default.gc_interval=3600
- net.ipv4.neigh.default.gc_stale_time=3600
- net.ipv6.neigh.default.gc_thresh1=8192
- net.ipv6.neigh.default.gc_thresh2=12228
- net.ipv6.neigh.default.gc_thresh3=24456
- net.ipv6.neigh.default.gc_interval=3600
- net.ipv6.neigh.default.gc_stale_time=3600

Sysctl 3

- net.ipv4.conf.all.accept_redirects=0
- net.ipv4.conf.all.send_redirects=0
- net.ipv6.conf.all.accept_ra=0
- net.ipv6.conf.all.autoconf=0
- net.ipv6.conf.all.accept_redirects=0

Conclusion

- En production depuis mardi soir
- Trois essais cause diverses boulettes
- Suite : automatisation, IPv6, suricata/snort
- conntrackd pour redondance ?
- Questions ?