On ne peut pas analyser tous les protocoles avec Netfilter

Stéphane Bortzmeyer
<stephane+blog@bortzmeyer.org>

Première rédaction de cet article le 4 avril 2012. Dernière mise à jour le 27 juin 2012

https://www.bortzmeyer.org/dns-netfilter-u32.html

Le système de traitement des paquets Netfilter est intégré dans le noyau Linux depuis des années. Il permet de créer des pare-feux, de faire de la limitation de trafic <https://www.bortzmeyer.org/rate-limiting-dos.html> et bien d'autres choses encore. Il permet de n'appliquer une règle qu'à certains paquets, selon le protocole de transport utilisé, selon les adresses IP ou les ports source ou destination, etc. Pour les autres cas, le module `u32` permet de chercher des correspondance dans n'importe quelle partie du paquet. Peut-on analyser tous les protocoles avec `u32` ? (Attention, il va y avoir du binaire dans l'article.)

Prenons l'exemple du DNS. Certains champs du paquet DNS sont accessibles via un écart fixe par rapport au début du paquet. Ceux-là sont faciles à chercher avec `u32`. La syntaxe de ce dernier n'est pas triviale, mais il existe un excellent tutoriel <http://www.stearns.org/doc/iptables-u32.current.html>. `0>>22&0x3C@` fait sauter l'en-tête IPv4. La longueur de l'en-tête (qui n'est pas constante) se trouve dans quatre bits du premier octet du paquet IP. On décale de 22 bits pour avoir ces bits et pour les multiplier par quatre (l'unité de longueur est le mot de quatre octets, cf. RFC 791 [1], section 3.1), on masque avec `0x3c` pour n'extraire que la longueur et on se déplace de cette quantité dans le paquet (c'est le rôle du signe @). Ensuite, on indique le déplacement après l'en-tête, pour UDP (le protocole le plus courant pour le DNS), c'est huit octets d'en-tête (RFC 768). Donc, `0>>22&0x3C@8` nous fait sauter les en-têtes IPv4 et UDP et arriver au début du paquet DNS.

Imaginons qu'on veuille sélectionner uniquement les requêtes, pas les réponses. Le RFC 1035, section 4.1.1, nous apprend que le bit permettant de différencier requêtes et réponses est le premier du troisième octet du paquet. Il est à zéro lorsque le paquet est une requête. On peut le sélectionner avec `10&0x80000000=0x0` (se déplacer de 10 octets, les huit d'UDP et les deux du premier champ DNS, le "*Query ID*", puis masquer avec les quatre octets `0x80000000` - un bit à un - et tester). En combinant tout cela, on peut utiliser `iptables` pour faire une règle qui ne s'appliquera qu'aux requêtes DNS :

1. Pour voir le RFC de numéro NNN, `https://www.ietf.org/rfc/rfcNNN.txt`, par exemple `https://www.ietf.org/rfc/rfc791.txt`

```
iptables --append ${CHAIN} --destination ${DEST} \
    --protocol udp --destination-port 53 \
    --match u32 --u32 "0>>22&0x3C@10&0x80000000=0x0" \
    --jump ${ACTION}
```

Pour n'avoir que les réponses, on utiliserait `0x80000000=0x80000000` pour tester si le bit est à un. Une alternative est d'utiliser l'opérateur de décalage de `u32`, le `>>` pour ne garder que le bit QR avant de le tester : `8>>15&0x01=0` au lieu de `10&0x4000=0x0000`. (Il existe une application pratique <http://doc.powerdns.com/powerdns-advisory-2012-01.html>, avec exemple, au filtrage des réponses.)

Maintenant, supposons qu'on veuille mettre en place une limitation de trafic par exemple pour faire face à une attaque par déni de service. J'avais déjà écrit un article à ce sujet <https://www.bortzmeyer.org/rate-limiting-dns-open-resolver.html> où on limitait **toutes** les requêtes DNS. Maintenant, si je fais face à une attaque spécifique, par exemple utilisant les requêtes de type `ANY` en raison de leur fort taux d'amplification (la réponse est bien plus grosse que la requête) : puis-je appliquer une règle Netfilter seulement aux paquets où la requête est de type `ANY` ?

Le problème est que, si les premiers champs du paquet DNS (comme `Flags`) sont à une distance fixe du début du paquet, ce n'est plus le cas par la suite. `u32` sait traiter les cas (comme l'en-tête IPv4) où la distance est indiquée dans le paquet mais ne sait pas faire une analyse complète (en termes pédants, `u32` n'est pas un langage de Turing).

Or, dans un paquet DNS (cf. RFC 1035, section 4.1.2), le type de requête (champ `QTYPE` pour "*Query Type*") est situé **après** une structure compliquée, le nom demandé (champ `QNAME` pour "*Query Name*"). Le langage d'`u32` ne connait que décalage et masquage, alors qu'il nous faudrait des tests et des boucles (ou bien la récursion), pour pouvoir suivre les différents composants ("*labels*") du `QNAME` (qui peuvent être en nombre quelconque). Si `u32` n'a pas cela, d'ailleurs, ce n'est pas par paresse ou ignorance de son auteur, mais parce que ce langage est conçu pour des analyses ultra-rapides, et dont la durée est prévisible (ce qui n'est plus le cas dès qu'on met des boucles).

Donc, limiter uniquement les requêtes `ANY` ne semble **pas** possible avec Netfilter. C'est pour cela que les exemples DNS qu'on trouve sur le Web ne traitent que des cas simples comme les attaques dites NS . <https://www.bortzmeyer.org/dns-attaque-ns-point.html> ("*Upward Referrals*" <https://www.dns-oarc.net/oarc/articles/upward-referrals-considered-harmful>) où le `QNAME` est toujours le même (en l'occurrence un simple point, la racine du DNS). On peut alors filtrer <http://gcolpart.evolix.net/blog21/ferme-ton-bind/> avec `0>>22&0x3C@20>>24=0` (sauter 20 octets pour arriver au `QNAME` qui doit être uniquement composé d'un octet longueur valant zéro).

Une solution astucieuse est de commencer par la **fin** du paquet et non pas son début, pour tomber sur le `QTYPE` avant le `QNAME`. Mais cela ne marche pas non plus car les requêtes DNS modernes ont en général à la fin une section additionnelle, également de longueur variable (conséquence du RFC 6891).

Damien Wyart me signale qu'un module DNS pour Netfilter, qui résoudrait ce problème, est en cours de développement <http://software.klolik.org/xt_dns/>. Je ne l'ai pas encore testé, mais, à la lecture, ce code a le même défaut que pas mal de règles iptables ou tcpdump qu'on trouve sur le Web. Il analyse le paquet DNS en partant de la fin, pour éviter d'avoir à décoder le `QNAME`. Résultat, il ne marche pas du tout avec EDNS, qui ajoute une section additionnelle à la fin.

Une solution partielle, qui ne résoudrait pas le cas général, serait de faire une règle spécifique à un nom de domaine, si l'attaque utilise toujours le même (car on connait alors la longueur du `QNAME` et on peut le sauter). C'est limité mais mieux que rien. Comme l'attaquant peut changer facilement le `QNAME` utilisé, le mieux est d'utiliser un petit programme qui prenne en paramètre un nom et fabrique le code `u32`. C'est le cas du script (en ligne sur `https://www.bortzmeyer.org/files/generate-netfilter-u32-dns-rule.py`) :

```
% python generate-netfilter-u32-dns-rule.py --qname isc.org
0>>22&0x3C@20&0xFFFFFFFF=0x03697363&&0>>22&0x3C@24&0xFFFFFFFF=0x036f7267&&0>>22&0x3C@28&0xFF000000=0x00000000
```

Le programme peut aussi s'utiliser ainsi dans un script shell :

```
# L'action donnée ici est juste un exemple
action="LOG --log-prefix DNS-ANY-query-$domain"
rule=$(python generate-netfilter-u32-dns-rule.py --qname $domain --qtype ANY)
iptables --append ${CHAIN} --destination ${DEST} --protocol udp --destination-port 53 \
    --match u32 --u32 "$rule" \
    --jump ${action}
```

Autre approche, voici (dû à Frédéric Bricout) les commandes `tc` pour faire une limitation de trafic à 2 kb/s, exclusivement sur les requêtes `ripe.net` (0472697065 = ripe - avec la longueur en premier octet, 036e6574 = net) :

```
tc qdisc add dev eth1 root handle 1: htb default 30
tc filter add dev eth1 parent 1: protocol ip prio 10 u32 \
      match u32 0x04726970 0xffffffff at 40 match u32 0x65036e65 0xffffffff at 44 \
      match u32 0x740000ff 0xffffffff at 48 police rate 2kbit buffer 10k drop flowid :1
```

Une autre idée, que j'emprunte à Kim Minh Kaplan, consiste à jouer sur la longueur des noms demandés :

```
iptables -N DNSratelimit
# Y ajouter les règles de limitations de débit

iptables -N DNSrules

# Empêche les requêtes ANY pour .
iptables -A DNSrules -j DNSratelimit -m u32 --u32 "0>>22&0x3C@20&0xFFFFFF00=0x0000FF00"

# Empêche les requêtes ANY pour un TLD
# 21 = 20 pour IP+UDP+DNS + 1 pour l'octet de longueur
iptables -A DNSrules -j DNSratelimit -m u32 --u32 "0>>22&0x3C@20>>24&0xFF@21&0xFFFFFF00=0x0000FF00"

# Envoyer les requêtes DNS à la bonne chaîne
iptables -A INPUT -p udp --dport 53 -j DNSrules --match u32 --u32 "0>>22&0x3C@10&0x80000000=0"
```

Il faudrait aussi y ajouter des règles pour éviter certains traitements erronnés, etc. Malheureusement cette technique se heurte à une autre limitation de u32 :

```
# Empêche les requêtes pour domaine directement sous un TLD
# 22 is 20 for IP+UDP+DNS and 2 length byte
iptables -A DNSrules -m u32 --u32 "0>>22&0x3C@20>>24&0xFF@21>>24&0xFF@22&0xFFFFFF00=0x0000FF00"
```

Ce code échoue :

```
iptables v1.4.12.2: u32: at char 48: too many operators
Try 'iptables -h' or 'iptables --help' for more information.
```

Et, effectivement, la documentation stipule « "*no more than 10 numbers (and 9 operators) per location*" ». Donc, cette voie ne semble pas idéale.

Maintenant, la question philosophique : les concepteurs du format des paquets DNS auraient-ils dû faire un format plus amical aux analyseurs ?

Merci à Kim Minh Kaplan pour sa relecture attentive du code binaire et pour la bogue détectée.