Un exemple d'attaque NTP par réflexion

Stéphane Bortzmeyer
<stephane+blog@bortzmeyer.org>

Première rédaction de cet article le 27 février 2014

https://www.bortzmeyer.org/attaque-ntp-en-vrai.html

J'ai déjà écrit ici à propos des attaques par déni de service utilisant NTP et la réflexion <https://www.bortzmeyer.org/ntp-reflexion.html>. Une caractéristique peu connue des attaques sur l'Internet est qu'elles sont rarement étudiées en détail et de manière publique. Chacun garde jalousement ses traces, il y a parfois une étude faite par le service informatique local ou par une organisation extérieure mais il est rare que les détails soient publiés, surtout lorsqu'une procédure judiciaire a été lancée. Résultat, la science ne progresse pas, puisqu'on ne peut pas tenir compte de l'expérience des autres, et on trouve sur les forums publics des tas d'affirmations non sourcées et souvent fausses. Voici donc l'examen sommaire d'une vraie attaque NTP.

Relisez mon article précédent <https://www.bortzmeyer.org/ntp-reflexion.html> pour le contexte et notamment sur le fait que ces attaques par réflexion sont un jeu à trois, l'Attaquant, la Victime et le Réflecteur. Dans le cas de cette attaque, observée hier (et merci au donateur pour le pcap), le Réflecteur était un routeur Juniper (oui, c'est un réflecteur NTP par défaut <http://www.gossamer-threads.com/lists/nsp/juniper/49151>). Il avait bien les ACL nécessaires mais, apparemment suite à une bogue de JunOS, elles n'étaient pas appliquées (mettre à jour JunOS a guéri le problème). Son adresse IP est 192.0.2.67 (et, comme disent les articles dans la presse, les adresses IP ont été changées pour protéger les témoins).

L'attaque a été détectée par la quantité de données pompées par la machine. Une fois détectée, l'administrateur du routeur a pu capturer 24 secondes de trafic pour l'analyser. Premier problème, les fichiers pcap n'étaient pas normaux. Par exemple, cette commande tcpdump marchait bien :

```
reading from file attack-ntp-bis-february-2014.pcap, link-type JUNIPER_ETHER (Juniper Ethernet)
14:22:05.732503 IP 192.0.2.67.22 > 198.51.100.1.3004: Flags [P.], seq 3663360956:3663361164, ack 954084525, win
14:22:05.746028 IP 198.51.100.1.3004 > 192.0.2.67.22: Flags [.], ack 0, win 1267, options [nop,nop,TS val 101630
14:22:05.746176 IP 198.51.100.1.3004 > 192.0.2.67.22: Flags [.], ack 208, win 1267, options [nop,nop,TS val 1016
...
```

Mais une commande très proche mais ne voulant garder que les paquets NTP échoue :

```
% tcpdump -n -r attack-ntp-bis-february-2014.pcap udp and port 123
reading from file attack-ntp-bis-february-2014.pcap, link-type JUNIPER_ETHER (Juniper Ethernet)
tcpdump: pcap_loop: bogus savefile header
```

Bon, fichier bizarre ou corrompu. Certains programmes s'en tiraient mieux que d'autres (tshark y arrivait) mais le mieux est d'essayer de réparer, avec l'excellent pcapfix <http://f00l.de/pcapfix>. Comme le dit sa documentation, « *"pcapfix tries to repair your broken pcap files, fixing the global header and recovering the packets by searching and guessing the packet headers"* ». Une fois pcapfix appliqué (`pcapfix -v attack-ntp-february-2014.pcap`, qui produit un fichier `fixed_attack-ntp-february-2014.pcap`), cela se passe un peu mieux (sans être parfait, mais je n'avais pas le temps de me plonger dans les détails du format pcap et de ses variations). On voit bien l'attaque, un seul paquet de requête générant quatre paquets de réponse, pour un facteur d'amplification NTP de 193 (le facteur réel est bien plus bas, car il y a les en-têtes UDP, IP et Ethernet) :

```
14:22:16.653701 IP 203.0.113.220.8080 > 192.0.2.67.123: NTPv2, Reserved, length 8
14:22:16.653912 IP 192.0.2.67.123 > 203.0.113.220.8080: NTPv2, Reserved, length 440
14:22:16.653984 IP 192.0.2.67.123 > 203.0.113.220.8080: NTPv2, Reserved, length 440
14:22:16.654069 IP 192.0.2.67.123 > 203.0.113.220.8080: NTPv2, Reserved, length 440
14:22:16.654152 IP 192.0.2.67.123 > 203.0.113.220.8080: NTPv2, Reserved, length 224
```

L'examen avec Wireshark des paquets montre qu'il y a bien utilisation de la fameuse commande `monlist` (`MON_GETLIST_1` dans le paquet). Ici, `203.0.113.220` est la Victime, qui va se recevoir tout le trafic amplifié dans la figure. Le premier paquet a une adresse IP source usurpée, il est en fait envoyé par l'Attaquant au Réflecteur, prétendant venir de la Victime.

Avec un petit coup de tshark, on peut trouver les victimes, au nombre de quinze pendant cette période :

```
% tshark -n -r attack-ntp-bis-february-2014.pcap  -T fields -e ip.dst udp.srcport eq 123 \
     | sort |uniq |wc -l
15
```

(Pour ceux qui ne connaissent pas bien tshark : `-e ip.dst` lui dit de n'afficher que l'adresse IP de destination et `udp.srcport eq 123` lui dit de n'afficher que les paquets UDP envoyés par le réflecteur à la victime.) Rappelons que c'était en seulement 24 secondes. L'attaquant change donc de cible souvent, ou bien attaque plusieurs victimes en même temps. L'examen des adresses IP avec whois montre de tout : une banque japonaise, un gros pétrolier, un FAI, un gros éditeur de logiciels, etc.

Quels sont les ports utilisés ? Demandons à awk un coup de main :

```
% tshark -n -r attack-ntp-bis-february-2014.pcap  -T fields  -e udp.dstport udp.srcport eq 123 \
     | awk '{ ports[$1] = ports[$1] + 1 } END {for(p in ports) { print p, "=", ports[p]}}'
...
80 = 19711
8080 = 5380
123 = 1
43594 = 1848
```

Traditionnellement, NTP utilisait le port 123, aussi bien en source qu'en destination. Cela peut expliquer l'unique paquet ayant ce port de destination (qui était le port source du paquet envoyé au réflecteur, rappelez-vous qu'on analyse ici les paquets sortants du réflecteur). Plus étonnant est la place de 80. La première réaction est évidemment que l'attaquant visait un serveur HTTP et voulait être sûr de l'atteindre. Mais HTTP fonctionne sur TCP, pas sur UDP (sauf des cas non-standards comme les serveurs de Call of Duty). Donc, si l'attaquant craignait qu'un pare-feu ne lui bloque le trajet, se servir du port 80 ne va pas l'aider, le port 80/UDP est souvent fermé. Il y a plusieurs hypothèses : attaquant incompétent, choix d'un port un peu au pifomètre, hypothèse par l'attaquant que les pare-feux seront mal configurés, autorisant 80 aussi bien en UDP qu'en TCP, désir d'être moins visible sur les données NetFlow, etc.