notes

Frédéric Perrin

10 novembre 2010

Contents

1 Dailymotion inaccessible

En utilisant hutch comme proxy pour Firefox (proxy SOCKS monté avec `ssh -D 8081 fperrin@hutch`), dailymotion.com n'est pas navigable : on obtient seulement une page blanche. FF n'indique pas de timeout DNS ni de connexion refusée, juste une page blanche. En utilisant lynx depuis hutch, la page d'accueil de dailymotion.com est affichée correctement.

Avec tcpdump, nous capturons deux traces : capture-B4.dump et capture-AFTR.dump, qui sont obtenues en visitant successivement :

- dailymotion.com avec Firefox utilisant hutch comme proxy SOCKS ;
- dailymotion.com avec lynx tournant sur hutch ;
- lwn.net avec Firefox utilisant hutch comme proxy SOCKS.

2 Lecture des traces

Sur l'AFTR, la capture se fait avec `tcpdump -A -s 1500 -i eth0`. Nous voyons cependant des paquets à destination de 192.168.1.133 (l'adresse de hutch derrière le NAT du B4) : il me semble que ces paquets ne devraient être vus que sur eth1, et encapsulés dans de l'ip6, non ? —Non, enfin si : ces paquets sont bien du IP4 dans IP6 (c'est Wireshark qui les fait apparaître comme de l'IP4 tout court dans le résumé), il n'empêche qu'ils ne devraient pas apparaître sur eth0. Bref. Ou alors on tape sur eth0 depuis le début. Bref. —En fait, B4 -> AFTR tombe sur eth1, mais AFTR -> B4 repart par eth0.

2.1 lwn.net avec FF *via* hutch

Poignée de main TCP OK.

La requête `GET /` passe dans un seul paquet TCP. La réponse arrive dans des paquets IP de 1 380 octets.

2.2 dailymotion.com avec FF *via* proxy SOCKS

La poignée de main se passe bien.

2.2.1 Vu depuis hutch

La requête `GET /` ne tient pas en un seul paquet IP (on envoie un très gros cookie). Cette requête fait 1560 octets. Le premier paquet envoyé fait 1380 octets (octets 1–1328 du flux TCP). Ce paquet n'est pas arrivé jusqu'à l'AFTR, il a été rejeté par le B4 avec un message ICMP *destination unreachable* : *fragmentation needed*. Le deuxième paquet, bien plus petit, est correctement envoyé, puis acquitté de façon sélective par dailymotion.com (octets 1329–1560). hutch retransmet alors le premier segment TCP, cette fois dans deux paquets IP avec un MTU diminué de 100 octets. On reçoit bien un acquittement sélectif des octets 1229–1560 de la part de dailymotion.com. Les 1228 premiers octets de cette requêtes sont régulièrement retransmits par hutch, avec des intervalles de plus en plus longs, et 10 secondes plus tard, dailymotion.com abandonne en envoyant un FIN.

2.2.2 Vu depuis l'AFTR

Le deuxième segment TCP de cette requête est bien arrivé jusqu'à l'AFTR (et est marqué par Wireshark par *Previous segment lost*). Ce deuxième

paquet a été envoyé à dailymotion.com, qui répond avec un acquittement sélectif de ce deuxième segment TCP.

Lorsque hutch retransmet le premier segment TCP, cette fois découpé en deux paquets IP, le deuxième paquet IP (octets 1229–1328) est bien reçu, mais il n'y a pas trace du premier paquet IP (octets 1–1228), qui n'est donc jamais reçu par l'AFTR.

2.3 Analyse

Les paquets IP de 1380 octets dans le sens AFTR -> hutch sont correctement traités, et arrivent jusqu'à hutch. Les paquets IP de 1380 octets dans le sens hutch -> AFTR sont rejetés par le B4, avec un ICMP *Fragmentation needed*. Lorsque hutch retransmet ces paquets en diminuant la MTU de 100 octets, ces paquets sont droppés silencieusement quelque part entre hutch et l'AFTR.

Un tcpdump rapide sur pessac168 (la passerelle de sortie par défaut du VLAN utilisateurs du ResEl) montre que :

- le premier paquet IP envoyé par hutch a été fragmenté par le B4 en deux parties avant d'être encapsulé dans le tunnel ip6 (malgré le bit *don't fragment* mis à 1 par hutch, et le fait que le B4 a envoyé un ICMP *destination unreachable*) ;
- lors de la retransmission du premier segment TCP 1–1328 en deux segments (1–1228 et 1229–1328), le premier paquet IP a quand même dû être fragmenté par le B4 ;
- la fragmentation faite par le B4 produit toujours un paquet IP de 1280 octets + le reste dans un autre paquet ;
- pessac168 a bien faire suivre au hop suivant les retransmissions du premier segment ;
- le hop suivant de pessac168 est pessac, un routeur Cisco, sur lequel je n'ai pas encore eu le courage de mettre un port en monitor pour regarder ce qui se passe dessus.

Il me semble qu'on peut en déduire que *quelque chose* droppe silencieusement les paquets de taille 1280 entre pessac168 et l'AFTR. Pourtant :

```
fperrin@pessac168:~$ ping6 -c1 -M do -s 1453
      2001:660:7301:1:20c:29ff:fe7c:85d2
```

```
      | grep -i from
From 2001:660:7301:3168::168 icmp_seq=1
      Packet too big: mtu=1500

fperrin@pessac168:~$ ping6 -c1 -M do -s 1452
      2001:660:7301:1:20c:29ff:fe7c:85d2
      | grep -i from
1460 bytes from 2001:660:7301:1:20c:29ff:fe7c:85d2:
      icmp_seq=1 ttl=61 time=1.99 ms
```

Il y a de la marge...

Ah ! en tcpdump'ant sur eth1 sur l'AFTR (mais pourquoi ne l'ai-je pas fait plus tôt ?) on voit bien la retransmission du premier segment (1–1238) arriver sur l'AFTR. C'est donc l'AFTR qui ne réassemble pas ces paquets IP... Dans la sortie de debug de `./aftr -g`,