Faire tourner sshd sur un autre port que 22

Stéphane Bortzmeyer
<stephane+blog@bortzmeyer.org>

Première rédaction de cet article le 17 décembre 2010. Dernière mise à jour le 20 décembre 2010

https://www.bortzmeyer.org/sshd-port-alternatif.html

La plupart des serveurs et routeurs connectés à l'Internet ont un serveur SSH qui écoute sur le port 22 pour permettre l'accès à distance et l'administration de la machine. Très souvent, des attaques automatiques sont lancées contre ces machines. Même si elles échouent, elles remplissent les journaux et déclenchent des alarmes inutiles. Je recommande personnellement de ne jamais faire tourner le serveur SSH sur le port 22.

Voici un exemple d'une attaque réelle (je n'ai pas modifié l'adresse IP source de l'attaquant car, comme d'habitude, abuse n'a jamais répondu à mon signalement <https://www.bortzmeyer.org/abuse-ne-repond-pas.html>). Il s'agit apparemment d'une attaque par dictionnaire classique, où l'assaillant essaie plusieurs mots de passe classiques pour des comptes courants dans les pays anglo-saxons (john, adam, kevin) :

```
Dec  9 05:35:10 mon-serveur sshd[28839]: Invalid user john from 173.45.74.230
Dec  9 05:35:10 mon-serveur sshd[28839]: pam_unix(sshd:auth): check pass; user unknown
Dec  9 05:35:10 mon-serveur sshd[28839]: pam_unix(sshd:auth): authentication failure; logname= uid=0 euid=0 tty=
Dec  9 05:35:11 mon-serveur sshd[28839]: Failed password for invalid user john from 173.45.74.230 port 40514 ssh
Dec  9 05:35:12 mon-serveur sshd[28841]: Invalid user john from 173.45.74.230
Dec  9 05:35:12 mon-serveur sshd[28841]: pam_unix(sshd:auth): check pass; user unknown
Dec  9 05:35:12 mon-serveur sshd[28841]: pam_unix(sshd:auth): authentication failure; logname= uid=0 euid=0 tty=
Dec  9 05:35:14 mon-serveur sshd[28841]: Failed password for invalid user john from 173.45.74.230 port 41395 ssh
Dec  9 05:35:16 mon-serveur sshd[28843]: Invalid user kevin from 173.45.74.230
Dec  9 05:35:16 mon-serveur sshd[28843]: pam_unix(sshd:auth): check pass; user unknown
Dec  9 05:35:16 mon-serveur sshd[28843]: pam_unix(sshd:auth): authentication failure; logname= uid=0 euid=0 tty=
Dec  9 05:35:18 mon-serveur sshd[28843]: Failed password for invalid user kevin from 173.45.74.230 port 42402 ssh
Dec  9 05:35:19 mon-serveur sshd[28845]: Invalid user kevin from 173.45.74.230
Dec  9 05:35:19 mon-serveur sshd[28845]: pam_unix(sshd:auth): check pass; user unknown
Dec  9 05:35:19 mon-serveur sshd[28845]: pam_unix(sshd:auth): authentication failure; logname= uid=0 euid=0 tty=
Dec  9 05:35:20 mon-serveur sshd[28845]: Failed password for invalid user kevin from 173.45.74.230 port 43071 ssh
Dec  9 05:35:21 mon-serveur sshd[28847]: Invalid user adam from 173.45.74.230
Dec  9 05:35:21 mon-serveur sshd[28847]: pam_unix(sshd:auth): check pass; user unknown
Dec  9 05:35:21 mon-serveur sshd[28847]: pam_unix(sshd:auth): authentication failure; logname= uid=0 euid=0 tty=
Dec  9 05:35:23 mon-serveur sshd[28847]: Failed password for invalid user adam from 173.45.74.230 port 43842 ssh
Dec  9 05:35:24 mon-serveur sshd[28849]: Invalid user adam from 173.45.74.230
Dec  9 05:35:24 mon-serveur sshd[28849]: pam_unix(sshd:auth): check pass; user unknown
Dec  9 05:35:24 mon-serveur sshd[28849]: pam_unix(sshd:auth): authentication failure; logname= uid=0 euid=0 tty=
Dec  9 05:35:27 mon-serveur sshd[28849]: Failed password for invalid user adam from 173.45.74.230 port 44743 ssh
```

Ici, l'attaque a apparemment échoué. Mais, même si le serveur SSH a un accès restreint (par exemple avec la directive `AllowUsers` de OpenSSH), c'est ennuyeux d'avoir ses journaux encombrés par de telles attaques, qui sont très courantes. Un script PHP bogué, une prise de contrôle à distance, même sans passer root et hop, tout serveur dédié n'importe où sur la planète commence un balayage systématique des serveurs et routeurs, pour le compte du craqueur masqué derrière lui.

Ma méthode préférée pour garder mes journaux courts et pour embêter un peu les pirates est de faire tourner le serveur sur un autre port. Avec OpenSSH, c'est :

```
Port 42666
```

dans le fichier de configuration. Et, là, plus d'attaques.

J'insiste bien sur le fait que le but principal est d'éviter l'encombrement des journaux. Changer de port ralentit les craqueurs mais n'est pas réellement un gros avantage en matière de sécurité (croire cela serait croire au STO). Un craqueur compétent pourrait faire un nmap sur le serveur et découvrir le port SSH par la bannière envoyée :

```
% telnet mon-serveur 42666
Trying 2001:db8:37a::d946:bee8:232...
Connected to mon-serveur.
Escape character is '^]'.
SSH-2.0-OpenSSH_5.1p1 Debian-5
...
```

Mais, en pratique, la plupart des attaques sont bêtes et massives. Pas de subtilité, juste tester le port 22. Sur les serveurs qui écoutent sur un autre port, on ne voit jamais, à l'heure actuelle, d'attaques par dictionnaire.

Certaines personnes pensent que changer de port va leur compliquer la vie à eux aussi, en les obligeant à indiquer le numéro de port à chaque commande SSH, par exemple à taper `ssh -p 42666 mon-serveur` au lieu de `ssh mon-serveur`. Mais non ; OpenSSH permet de mettre le numéro de port une fois pour toutes dans le fichier `~/.ssh/config` :

```
Host mon-serveur
  Port 42666
```

et c'est tout, il n'y aura plus rien à taper (si on travaille sur plusieurs machines, ce qui est mon cas, on peut synchroniser son répertoire <https://www.bortzmeyer.org/home-in-subversion.html>). Même chose avec un client SSH comme ConnectBot <http://code.google.com/p/connectbot/> sur Android, qui permet d'indiquer le numéro de port lors de l'enregistrement d'un serveur.

Et que faire si on ne contrôle pas le pare-feu et que les ports alternatifs sont bloqués ? La bonne solution est d'utiliser le port 443 (celui de HTTPS) qui est rarement bloqué. Et si on a déjà un serveur Web sur ce port ? Dans ce cas, il y a sslh <http://www.rutschle.net/tech/sslh.shtml>.

Existe-t-il d'autres méthodes pour contrarier ce genre d'attaquants ? Oui, bien sûr, on peut restreindre l'accès à SSH par adresse IP source (voir un exemple sur Linux <http://www.debian-administration.

org/articles/87>). Cela se fait souvent sur les routeurs, qu'on n'administre que depuis le réseau interne, mais ce n'est pas toujours possible pour les serveurs, il faut bien pouvoir se connecter à distance. Il y a aussi la possibilité de faire du « toquage à la porte » avec un logiciel comme knockd (cf. un bon article en français <http://souvenirfromlife.fr/blog/post/2008/12/08/Comment-avoir-un-pont-levis-sous ou avec une solution plus simple <http://www.debian-administration.org/articles/268>. Encore une autre solution est de détecter automatiquement les clients SSH qui abusent et de les filtrer. C'est ce que fait fail2ban mais je ne l'ai personnellement pas encore tenté. On peut aussi faire la même chose avec DenyHosts (qui utilise un "*TCP Wrapper*" et pas Netfilter), avec iptables <http://www.debian-administration.org/articles/187> et le module "recent" <http://www.snowman.net/projects/ipt_recent/> (attention, cette solution a des limites <http://lists.debian.org/debian-user-french/2010/12/msg00307.html>). Ces techniques sont toutes plus sûres que le simple changement de port mais sont également plus compliquées à mettre en œuvre.

Merci à Pascal Hambourg, Sébastien Rodriguez, Cyril Bouthors, Clochix et aux utilisateurs de Twitter dont j'ai oublié de noter le nom.

Je recommande un très bon article sur l'état des attaques SSH par force brute : « "*Observations from two weeks of SSH brute force attacks*" <http://www.lightbluetouchpaper.org/2012/01/25/observations-from-two-weeks-of-ssh-brute-force-attacks/> ». Et il faut se rappeler qu'il n'y a pas que les méchants qui balaient l'Internet sur le port 22. les chercheurs en sécurité le font aussi <http://blog.erratasec.com/2013/09/we-scanned-internet-for-port-22.html> (et trouvent plein de machines).