SNMP v3 sur Unix, pour Cacti et Icinga

Stéphane Bortzmeyer
<stephane+blog@bortzmeyer.org>

Première rédaction de cet article le 25 septembre 2013

https://www.bortzmeyer.org/snmp-v3-inciga-cacti.html

Le protocole SNMP d'accès distant à l'information d'une machine est utilisé depuis bientôt un quart de siècle. Bien que la version actuelle de SNMP, la v3, ait été normalisée il y a quinze ans, il semble que les antiques versions 1 et 2 soient toujours largement utilisées. Voici des exemples de configuration d'un serveur Unix avec SNMP v3, et on interrogation par le logiciel de supervision Icinga et le logiciel de statistiques Cacti.

C'est que SNMP v3 est plus complexe, aussi bien lorsqu'on veut lire les RFC qui le normalisent (RFC 3414 [1], RFC 3415, etc) que lorsqu'on regarde la configuration des logiciels clients et serveurs. Mais il présente un avantage décisif en matière de sécurité. En SNMP v1 (RFC 1157), il n'y avait aucune confidentialité : tout écoutant du réseau pouvait savoir ce qu'on faisait. Et aucune authentification sérieuse non plus : le mot de passe (baptisé « communauté » pour faire croire qu'il n'était pas aussi sensible qu'un mot de passe) circulait en clair et était donc interceptable. SNMP v3, au contraire, fournit un authentification fiable et, si on le veut, de la confidentialité.

Sur un serveur Unix, le logiciel Net-SNMP <http://net-snmp.sourceforge.net/> fournit un serveur capable de parler le v3. Sa configuration est baroque mais on trouve d'innombrables documentations en ligne comme celle de Tom Clegg <http://tomclegg.net/snmpv3-cacti> ou celle de Nathan Drier <http://www.redspin.com/blog/2009/07/01/simplenetwork-management-protocol-snmpv3/>, toutes les deux pour Debian, ou bien celle du Wiki officiel <https://wiki.archlinux.org/index.php/Snmpd> pour Arch Linux.

Par exemple, sur une Debian, je fais :

```
% sudo aptitude install snmpd
```

1. Pour voir le RFC de numéro NNN, https://www.ietf.org/rfc/rfcNNN.txt, par exemple https://www.ietf.org/rfc/rfc3414.txt

Puis j'édite `/etc/snmp/snmpd.conf` pour lui dire d'écouter sur toutes les adresses, IPv4 et IPv6. La ligne pertinente est :

```
agentAddress udp:161,udp6:[::]:161
```

(Oui, SNMP tourne sur UDP, port 161.) Pendant qu'on édite `snmpd.conf`, on modifie aussi les variables `sysLocation` et `sysContact` pour avoir des informations plus précises. On teste avec lsof que le serveur écoute bien :

```
% sudo lsof -i:snmp
COMMAND   PID USER   FD   TYPE DEVICE SIZE/OFF NODE NAME
snmpd   19240 snmp    8u  IPv4 227237      0t0  UDP *:snmp
snmpd   19240 snmp   11u  IPv6 227238      0t0  UDP *:snmp
```

Ici, tout est bon.

Il reste à configurer un utilisateur, avec son mot de passe (qui ne sera pas transmis en clair). Les administrateurs système ordinaires vont choisir un nom « parlant » comme `admin`. Les paranoïaques vont tirer ce nom au hasard, pour augmenter la difficulté d'une attaque :

```
% dd if=/dev/random bs=16 count=1 | md5sum
1+0 records in
1+0 records out
16 bytes (16 B) copied, 4.2661e-05 s, 375 kB/s
b5e9b185d879ee20b27be80195e9522a  -
```

Et on choisirait `b5e9b185d879ee20b27be80195e9522a` comme nom d'utilisateur... Pour le reste de cet article, je vais être petit-bras et prendre un nom et un mot de passe lisible, pour faciliter la lecture (le nom d'utilisateur est de toute façon en clair, même si on utilise l'option de chiffrement, qui ne chiffre que les données). Le nom sera `clinique` et le mot de passe `cahuzac` (une très mauvaise combinaison dans la réalité, puisque trop facile à trouver).

Le moyen le plus simple de créer un utilisateur est sans doute :

```
# Oui, il faut arrêter le serveur pour cela...
% /etc/init.d/snmpd stop

% sudo net-snmp-config --create-snmpv3-user -a SHA -x AES
Enter a SNMPv3 user name to create:
clinique
Enter authentication pass-phrase:
cahuzac
Enter encryption pass-phrase:
  [press return to reuse the authentication pass-phrase]
```

Ici, on a donné à la commande `net-snmp-config` les paramètres `-a SHA` (l'algorithme de condensation utilisé pour l'authentification sera donc SHA) et `-x AES` (l'algorithme de chiffrement utilisé pour la confidentialité sera AES). Le but est d'utiliser systématiquement l'option de chiffrement. On utilise le même mot de passe pour les deux services. Après cette commande, vous devriez avoir un `rouser clinique` (ou `rouser authOnlyUser`) dans votre `snmpd.conf` (`rouser` = *"Read-Only User"*).

Une autre méthode consiste à modifier `/var/lib/snmp/snmpd.conf` (qui n'est **pas** le `/etc/snmp/snmpd.con` pour y mettre :

```
createUser clinique SHA "cahuzac" AES "cahuzac"
```

Lorsqu'on relancera le démon (c'est pour cela qu'il **doit** être arrêté), le fichier `/var/lib/snmp/snmpd.conf` sera réécrit et les mots de passe en clair disparaîtront. Une troisième méthode de création (merci à Vincent Bernat pour le rappel) d'un compte utilise SNMP lui-même, en écrivant dans la table `snmpUsmUserTable` (par exemple via le programme `snmpusm`).

On redémarre le serveur et, maintenant, il est temps de tester. Commençons par des essais avec un client SNMP en ligne de commande, `snmpwalk`, issu du même paquetage Net-SNMP, en supposant que le nom de la machine visée est dans la variable `MYSERVER` :

```
% snmpwalk -v 3 -u clinique -a SHA -A  cahuzac -x AES -X cahuzac \
    -l authPriv $MYSERVER
```

(Préciser `-a SHA` est obligatoire car, sinon, MD5 est utilisé par défaut. Quant à `-l authPriv`, "*Authentication & Privacy*", cela active le chiffrement, afin de garantir la confidentialité des données. Voir la section 2.2 du RFC 3415.) Vous devez obtenir quelque chose comme :

```
SNMPv2-MIB::sysDescr.0 = STRING: Linux foobar.example.net 3.9.3-x86_64-linode33 #1 SMP Mon May 20 10:22:57 EDT 2
SNMPv2-MIB::sysObjectID.0 = OID: NET-SNMP-MIB::netSnmpAgentOIDs.10
DISMAN-EVENT-MIB::sysUpTimeInstance = Timeticks: (3442047) 9:33:40.47
SNMPv2-MIB::sysContact.0 = STRING: root@example.net
SNMPv2-MIB::sysLocation.0 = STRING: Floor 3, Row 12, Rack 5
...
IF-MIB::ifOutOctets.1 = Counter32: 791502
IF-MIB::ifOutOctets.2 = Counter32: 0
IF-MIB::ifOutOctets.3 = Counter32: 142039121
```

Si ce n'est pas le cas, à part une mauvaise saisie du mot de passe, vérifiez les points suivants :

— Tester depuis le serveur, en faisant un `snmpwalk ... 127.0.0.1`. Si cela marche, mais que cela échoue à distance, vous avez peut-être un pare-feu zélé sur le trajet, ou bien le démon n'écoute que sur les adresses locales (cf. lsof plus haut et, sur une Debian, la variable `SNMPDOPTS` dans `/etc/default/snmpd`).

— Question sécurité, vous avez peut-être installé TCP wrapper. Auquel cas, vérifiez qu'ils autorisent SNMP. Par exemple, si tous les accès se feront depuis les machines `192.0.2.3` et `2001:db8:ba27::3`, mettez dans `/etc/hosts.allow` quelque chose comme `snmpd: 192.0.2.3,[2001:db8:ba27::3]`.

Si cela marche, parfait, on va pouvoir passer à des clients SNMP plus riches. Ah, au passage, la même commande avec une adresse IPv6 (qu'il faut mettre entre crochets) :

```
% snmpwalk -v 3 -u clinique -a SHA -A  cahuzac -x AES -X cahuzac \
    -l authPriv "udp6:[2001:db8:dc0::1:cafe]"
```

Configurons maintenant Cacti pour produire de jolis graphes grâces à SNMP. Dans le menu "*Devices*" de la console de Cacti, on sélectionne "*Add*" (en haut à droite), on va indiquer "*SNMP Version*" = 3, le nom, le mot de passe, les algorithmes de cryptographie utilisés. On laisse le champ `Context` vide. Cacti met apparemment automatiquement le niveau de sécurité à `authPriv`. Lorsqu'on enregistre cette configuration, Cacti affiche le résultat d'une requête SNMP (description de la machine, contact, localisation, etc). S'il affiche à la place "*SNMP error*", c'est qu'il y a une erreur dans la configuration (vous avez bien testé avec `snmpwalk` avant ?) Si tout marche, on peut alors créer les graphes via Cacti.

Et pour Icinga <https://www.bortzmeyer.org/icinga.html> ? D'abord, quel est l'intérêt de SNMP pour un logiciel de supervision ? C'est qu'Icinga utilise les mêmes tests que Nagios et qu'un des tests Nagios, `check_snmp` a une option très pratique pour surveiller le trafic réseau. Les MIB habituelles de SNMP ne fournissent que des valeurs instantanées. On souhaiterait souvent avoir des taux, par exemple des bits/seconde et pas juste un total de bits, afin de pouvoir déclencher une alarme, par exemple en cas d'une attaque par déni de service. Certaines MIB (celle de Juniper <https://www.bortzmeyer.org/rate-mib-juniper.html>) fournissent des variables pour cela, mais tout le monde n'a pas un Juniper. Alors, on va utiliser une option très pratique de `check_snmp`, `--rate`, qui enregistre les mesures sur disque pour pouvoir calculer un taux. (Attention donc à ce qu'Icinga ait le droit d'écrire dans le répertoire en question.) Créons une commande :

```
define command{
        command_name    check_snmp_myserver
        command_line    $USER1$/check_snmp -H $HOSTADDRESS$ -P 3 -a SHA -x AES -U clinique -A cahuzac -X cah
    }
```

Et utilisons là dans la configuration :

```
define service{
       use                             generic-service
       host_name                       myserver
       service_description             OUT_PACKET_RATE
       check_command                   check_snmp_myserver!ifOutUcastPkts.3!1000!4000!
   }
```

Ici, on utilise la variable SNMP `ifOutUcastPkts` (nombre de paquets sortants) sur la troisième interface réseau. Si on voit plus de 1 000 p/s, Icinga lève un avertissement, si on voit plus de 3 000 p/s, une alerte. On verra dans le journal :

```
[1380120148] SERVICE ALERT: myserver;OUT_PACKET_RATE;CRITICAL;SOFT;2;SNMP RATE CRITICAL - *5370*
```

À noter qu'une autre solution est suggérée dans la documentation d'Icinga <http://docs.icinga.org/latest/en/monitoring-routers.html>, en récupérant les données de MRTG.