

Implementación del protocolo SNMP

Administración de servicios en Red

4CV2

Hernández León Francisco Alejandro

Ochoa Rivera Luis Gonzalo

Vargas Gamboa Ricardo Alan

Vega Bocanegra Carlos

Manual de Usuario

Manual para servidor

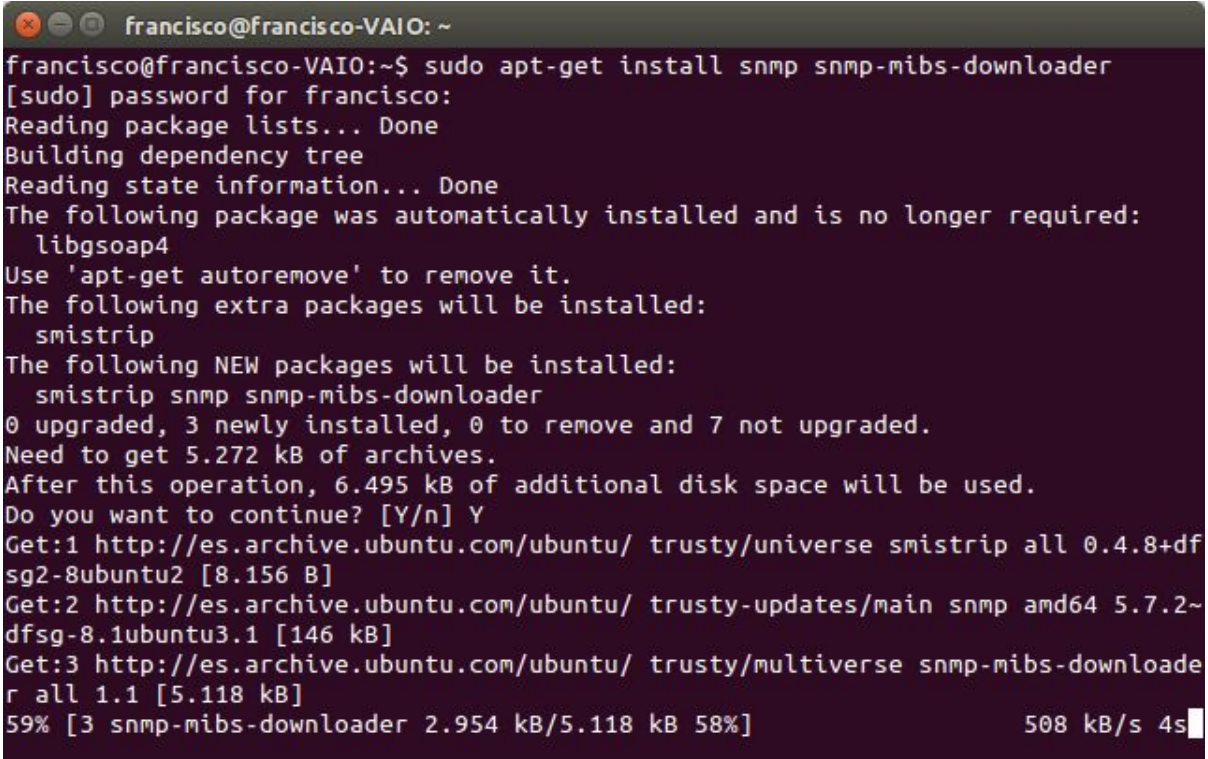
Manual para clientes

Manual para servidor

Para este manual se hizo la instalación, configuración y administración usando un sistema operativo Ubuntu 14.04 LTS de 64 bits. Los comandos utilizados para las diferentes acciones del manual podrían variar dependiendo la distribución utilizada.

Es necesario instalar primero el servidor SMTP, para lo cual tenemos que realizar el siguiente comando:

```
sudo apt-get install snmp snmp-mibs-downloader
```



```
francisco@francisco-VAIO: ~  
francisco@francisco-VAIO:~$ sudo apt-get install snmp snmp-mibs-downloader  
[sudo] password for francisco:  
Reading package lists... Done  
Building dependency tree  
Reading state information... Done  
The following package was automatically installed and is no longer required:  
  libgsoap4  
Use 'apt-get autoremove' to remove it.  
The following extra packages will be installed:  
  smstrip  
The following NEW packages will be installed:  
  smstrip snmp snmp-mibs-downloader  
0 upgraded, 3 newly installed, 0 to remove and 7 not upgraded.  
Need to get 5.272 kB of archives.  
After this operation, 6.495 kB of additional disk space will be used.  
Do you want to continue? [Y/n] Y  
Get:1 http://es.archive.ubuntu.com/ubuntu/ trusty/universe smstrip all 0.4.8+dfsg2-8ubuntu2 [8.156 B]  
Get:2 http://es.archive.ubuntu.com/ubuntu/ trusty-updates/main snmp amd64 5.7.2~dfsg-8.1ubuntu3.1 [146 kB]  
Get:3 http://es.archive.ubuntu.com/ubuntu/ trusty/multiverse snmp-mibs-downloader all 1.1 [5.118 kB]  
59% [3 snmp-mibs-downloader 2.954 kB/5.118 kB 58%] 508 kB/s 4s
```

Se utilizarán dos servidores, uno contendrá la porción administradora y otro servidor tendrá el agente. Podemos elegir instalar el agente en la computadora del administrado, pero mantenerlos separados nos proporciona sencillez para demostrar la funcionalidad que provee cada componente.

```
francisco@francisco-VAIO: ~  
francisco@francisco-VAIO:~$ sudo apt-get install snmpd  
[sudo] password for francisco:  
Reading package lists... Done  
Building dependency tree  
Reading state information... Done  
The following package was automatically installed and is no longer required:  
  libgsoap4  
Use 'apt-get autoremove' to remove it.  
The following NEW packages will be installed:  
  snmpd  
0 upgraded, 1 newly installed, 0 to remove and 7 not upgraded.  
Need to get 73,2 kB of archives.  
After this operation, 232 kB of additional disk space will be used.  
Get:1 http://es.archive.ubuntu.com/ubuntu/ trusty-updates/main snmpd amd64 5.7.2  
~dfsg-8.1ubuntu3.1 [73,2 kB]  
Fetched 73,2 kB in 0s (113 kB/s)  
Preconfiguring packages ...  
Selecting previously unselected package snmpd.  
(Reading database ... 80%
```

Configurando el Administrador SNMP

La gran mayoría del trabajo ocurre en el agente, nuestra configuración es muy sencilla en la máquina, solamente es necesario modificar un archivo para asegurarnos de que nuestro cliente puede utilizar datos MIB extra que instalamos.

Abrir el archivo en /etc/snmp/snmp.conf con privilegios de administrador.

```
francisco@francisco-VAIO:~$ sudo gedit /etc/snmp/snmp.conf
```

En este archivo, existen pocos comentarios y una línea comentada, para permitir al administrador importar los archivos MIB, se necesita comentar la línea "mibs:".

Ahora, es necesario configurar la máquina de los agentes SNMP.

Para ello debemos abrir el archivo localizado en /etc/snmp/snmpd.conf

```
root@francisco-VAIO:/home/francisco# gedit /etc/snmp/snmpd.conf
```

Dentro del archivo realizaremos algunos cambios, estos se utilizarán principalmente para inicializar la configuración que administramos desde nuestro otro servidor.

Primero, se requiere realizar el cambio de la directiva agentAdress, esta directiva sirve para aceptar las diferentes conexiones originadas desde cierta computadora, por defecto está configurada para aceptar conexiones sólo de la computadora local.

Se comentará la línea indicada y se descomentará la de abajo la cual permitirá todas las conexiones a ese servidor. (Puede ser cambiada para que solo acepte a un solo cliente)

```
# Listen for connections from the local system only
#agentAddress udp:127.0.0.1:161
# Listen for connections on all interfaces (both IPv4 *and* IPv6)
agentAddress udp:161,udp6:[::1]:161
```

A continuación, insertamos temporalmente una línea en dicho archivo de configuración para un usuario. Estas directivas no permanecen normalmente en este archivo, pero facilita la demostración del añadido de usuarios.

El usuario que vamos a crear, se llamará redes3 y será la base sobre la cual crearemos nuestro primer usuario real. Los paquetes SNMP hacen esto a través de un proceso de clonado de las propiedades del usuario anterior.

Cuando se define un nuevo usuario, se debe especificar el tiempo de autenticación (MD5 o SHA) así como proporcionar una contraseña que debe ser de al menos 8 caracteres, si se tiene planeado utilizar cifrado para la transferencia, tal como lo haremos, se debe especificar el protocolo de privacidad (DES o AES) y opcionalmente una contraseña del protocolo de privacidad. Si no se proporciona una

contraseña para el protocolo de privacidad se utilizará la contraseña de autenticación como la del protocolo de igual forma.

Nuestra sentencia para la creación del usuario de prueba será:

```
createUser redes3 MD5 redesdale DES
```

Ahora que se ha creado un nuevo usuario, es necesario establecer el nivel de acceso que el usuario tendrá. Lo estableceremos para nuestro usuario, lo haremos para nuestro usuario redes3 y también para el nuevo usuario que se creará que se llamará demo. Permitiremos acceso de lectura y escritura mediante utilizar la directiva rwuser (rouser sería para solo lectura), añadiremos la utilización de cifrado mediante añadir priv después de nuestro usuario.

Si queremos restringir al usuario a una parte específica del MIB, podemos especificar el nivel maximo de OID al que el usuario debería tener acceso al final de la línea.

Las lineas se declaran como sigue:

```
rwuser bootstrap priv
```

```
rwuser demo priv
```

Una vez realizados estos cambios, salva y cierra el archivo.

Para implementar estos cambios, se debe reiniciar el servicio snmpd.

```
sudo service snmpd restart
```

```
francisco@francisco-VAIO:~$ sudo service snmpd restart
[sudo] password for francisco:
* Restarting network management services:
```

Ahora, es momento de conectar un servidor agente para crear nuestro usuario regular.

Haremos esto utilizando la herramienta snmpsm, la cual es utilizada para administración de usuarios. Es necesario saber la dirección IP del servidor agente para que funcione correctamente.

Antes de empezar, hablaremos un poco acerca de la estructura general para enviar un comando SNMP.

La estructura general de los comandos

SNMP

Cuando se utiliza la suite de herramientas que están incluidas en el paquete snmp, se pueden encontrar algunos patrones en la forma en que deben ser utilizados o llamados los comandos.

La primera cosa que se debe realizar para comunicarse exitosamente mediante SNMP, es autenticarnos con el demonio SNMP que nos queremos comunicar.

Los comandos comunes se enlistan a continuación:

- -v VERSION: Esta bandera se utiliza para especificar la versión del protocolo SNMP, que deseamos utilizar, utilizamos la v3 en este manual.
- -c COMMUNITY: Esta bandera se utiliza si estamos utilizando SNMP v1 o v2. No es necesaria en la v3.
- -u USER-NAME: Este parámetro es utilizado para especificar el usuario con el que deseamos autenticarnos. Para hacer lectura o modificación de cualquiera utilizando SNMP, debemos autenticarnos con un nombre de usuario conocido.
- -l LEVEL: Este es utilizado para especificar el nivel de seguridad con el que nos estamos conectando. Los valores posibles son noAuthNoPriv para conectarnos sin autenticación y sin cifrado, authNoPriv para autenticación pero no cifrado y authPriv para autenticación y cifrado. El nombre de usuario que estamos utilizando debe ser configurado para operar en el nivel de seguridad que deseemos, de otra manera la autenticación no procederá.
- -a PROTOCOL: Este parámetro es utilizado para especificar el protocolo de autenticación que será usado. Los valores posibles son MD5 o SHA. Este debe coincidir con la información del protocolo cuando el usuario fue creado.
- -x PROTOCOL: Este parámetro es utilizado para especificar el protocolo de cifrado que será utilizado. Los valores posibles son DES o AES. Este debe coincidir con la información del protocolo cuando el usuario fue creado. Esto es necesario siempre que el la especificación de privilegios del usuario tenga un priv después de él, volviendo el cifrado obligatorio.

- -A PASSPHRASE: Este es utilizado para proporcionar la contraseña de autenticación que fue establecida cuando el usuario fue creado.
- -X PASSPHRASE: Este es la contraseña de cifrado que fue especificada cuando el usuario fue creado. Si ninguna es especificada pero el algoritmo de cifrado se especifica se utilizará la misma contraseña que la del usuario. Esta es requerida cuando el parámetro x es dado o cuando una especificación de privilegios de usuario tiene un priv después de ella, solicitando cifrado.

Utilizando esta información, podemos comenzar a construir nuestros comandos, de acuerdo a como configuramos nuestro usuario redes3, los comandos que estaremos utilizando en esa cuenta se verán de la siguiente forma:

comando_snmp -u redes3 -l authPriv -a MD5 -x DES -A redesdale -X redesdale
host_remoto snmp_sub_comando_o_opciones

Por ejemplo, para nuestro servidor de administración, debemos estar seguros que la cuenta redes3 se encuentra disponible mediante escribir:

snmpget -u redes3 -l authPriv -a MD5 -x DES -A redesdale -X redesdale
remote_host 1.3.6.1.2.1.1.1.0

```
iso.3.6.1.2.1.1.1.0 = STRING: "Linux francisco-VAIO 3.16.0-50-generic #67~14.04.1-Ubuntu SMP Fri Oct 2 22:07:51 UTC 2015 x86_64"
```

La cadena 1.3.6.1.2.1.1.0 es el OID que es responsable por mostrar la información del sistema. Este básicamente regresará la salida de **uname-a** en el sistema remoto.

Ahora, que hemos verificado que podemos correctamente autenticarnos al server corriendo el demonio SNMP, ahora podemos continuar a crear nuestra cuenta regular de usuario.

Configurar la cuenta regular de usuario.

A pesar de que hemos especificado los privilegios para la cuenta de usuario en nuestro archivo **snmpd.conf**, nosotros no hemos actualmente creado ese usuario aún. Nosotros vamos a utilizar el usuario `redes3` como template para nuestro nuevo usuario.

En el servidor de administración, nosotros podemos crear el usuario para el template, utilizando la herramienta `snmpusm` y siguiendo su sintaxis general.

```
snmpusm authentication_info remote_host create new_user existing_user
```

Así, usando las banderas de autenticación necesarias y aprovechando la cuenta de usuario que ya se tiene (`bootstrap`), se puede hacer un usuario que se acomode a los privilegios de usuario que se han definido (`demo`).

Este comando se verá así:

```
snmpusm -u bootstrap -l authPriv -a MD5 -x DES -A temp_password -X temp_password  
remote_host create demo bootstrap
```

Se debería obtener la siguiente respuesta:

User successfully created.

Ahora se tiene un usuario completamente funcional llamado **demo** en nuestro servidor remoto. Sin embargo, aún está usando la misma información que la cuenta

bootstrap. Se debería cambiar la contraseña. Esta vez, se usará al usuario demo para autenticarse. Hay que recordar que las contraseñas deben ser de al menos 8 caracteres:

```
snmpusm -u demo -l authPriv -a MD5 -x DES -A temp_password -X temp_password remote_host  
passwd temp_password my_new_password
```

Se debe obtener una respuesta como la siguiente:

SNMPv3 Key(s) successfully changed.

Se pueden probar las nuevas credenciales preguntando a nuestro servidor cuánto tiempo ha estado activo. Se usará un solo comando snmpget para obtener el valor de la otra máquina.

Esta vez, se aprovecharán las definiciones MIB que se descargaron para pedir el valor por nombre en vez de usar el OID numérico.

```
snmpget -u demo -l authPriv -a MD5 -x DES -A my_new_password -X  
my_new_password remote_host sysUpTime.0
```

Se obtendrá el siguiente valor que representa la última vez que el SNMP *daemon* remoto fue reiniciado.

DISMAN-EVENT-MIB::sysUpTimeInstance = Timeticks: (383018) 1:03:50.18

Creando un Archivo de Configuración para el Cliente.

En este punto es probable que se note que los detalles de autenticación para todos los comandos SNMP son por lo regular estáticos en cada solicitud. En vez de

escribirlos estos cada vez, es posible crear un archivo de configuración del lado del cliente el cual contendrá las credenciales con las cuales con las cuales nos conectaremos.

El archivo de configuración del cliente puede ser colocado en dos diferentes lugares dependiendo de qué tan amplio deseamos compartirlo.

Si deseamos compartir las credenciales de entrada con cualquier usuario válido en la máquina de administración, podemos poner los detalles de configuración dentro de un archivo global `snmp.conf`, es necesario abrir tal archivo con privilegios de super usuario.

```
sudo gedit /etc/snmp/snmp.conf
```

Si, se desea definir las credenciales de autenticación para tu usuario solo, es necesario crear un directorio oculto `snmp` con el directorio `home` del usuario y crear el archivo ahí.

```
mkdir ~/.snmp
cd ~/.snmp
gedit snmp.conf
```

Sin importar la decisión de dónde poner la configuración, el contenido será el mismo.

Los comandos que se utilizarán para autenticar están en la siguiente tabla. En la columna derecha, los nombres de las directivas que serían usadas para definir tales detalles de configuración dentro del archivo **`snmp.conf`**:

Opciones del Comando	Descripción	Translated <code>snmp.conf</code> directive

-u USERNA ME	El usuario SNMPv3 para autenticarnos.	defSecurityName USERNA ME
-l authPriv	El nivel de seguridad para autenticarnos.	defSecurityLevel authPriv
-a MD5	El protocolo de autenticación a utilizar	defAuthType MD5
-x DES	El protocolo de cifrado a emplear	defPrivType DES
-A PASSPH RASE	La contraseña de autenticación para el usuario	defAuthPassphrase PASSP HRASE
-X PASSPH RASE	La contraseña para el usuario suministrado	defPrivPassphrase PASSP HRASE

Utilizando esta información, es posible crear un archivo snmp.conf apropiado. For

Debe verse de la siguiente forma:

```
defSecurityName demo
defSecurityLevel authPriv
defAuthType MD5
defPrivType DES
defAuthPassphrase demo
```

defPrivPassphrase **demo**

Después de esto se guarda y se cierra el archivo.

Ahora se puede ejecutar comandos sin poner los detalles de autenticación.

Solamente se necesitará el comando snmp, el host y los argumentos del comando.

En lugar de poner:

```
snmpget -u demo -l authPriv -a MD5 -x DES -A my_new_password -X  
my_new_password remote_host sysUpTime.0
```

Se puede simplemente poner:

```
snmpget remote_host sysUpTime.0
```

Como se puede ver, esto reduce significativamente la cantidad de información que se debe proveer en cada solicitud.

Removiendo la cuenta original

Ahora que la cuenta regular está configurada correctamente, se puede remover la cuenta **bootstrap**, ya que es un poco insegura.

En nuestro servidor agente, abrir el archivo **/etc/snmp/snmpd.conf** con privilegios de super usuario.

Encontrar y comentar (o remover) las dos líneas creadas previamente añadidas que referencian al usuario bootstrap:

```
#createUser bootstrap MD5 temp_password DES
```

```
#rwuser bootstrap priv
```

Guardar y cerrar el archivo.

Ahora, hay que reiniciar el *SNMP daemon*:

```
sudo service snmpd restart
```

Esto cumplirá la recomendación de no tener directivas de *createUser* en el archivo de configuración normal **snmpd.conf**. También removerá los privilegios de ese usuario temporal.

Si se quiere remover el usuario bootstrap de manera definitiva de la tabla *usmUserTable*, se puede hacer al ejecutar este comando desde el servidor de administración.

```
snmpusm remote_host delete bootstrap
```

Se recibirá la siguiente respuesta:

```
User successfully deleted.
```

Conclusiones

En este punto se tiene completa la configuración cliente-servidor que se puede comunicar con seguridad utilizando el protocolo SNMP. Se pueden agregar fácilmente *daemons* en otros hosts y configurar la cuenta de acceso a través de toda la infraestructura. En la siguiente sección revisaremos el uso básico de las herramientas net-snmp con las que se han trabajado. Se demostrará cómo recuperar valores uno por uno o por grupos y cómo modificar la información.

Cómo usar las herramientas snmp para administrar y monitorear servidores

Introducción

Gran parte de ser administrador de un sistema es juntar información precisa acerca de sus servidores e infraestructura. Hay varias herramientas y opciones para obtener y procesar este tipo de información. Muchas de ellas están construidas sobre la tecnología SNMP.

Ahora se explicará cómo usar la configuración que se hizo previamente para obtener información y manipular hosts remotos.

Usando comando de clientes SNMP

El conjunto de herramientas que se ha usado contiene unas cuantas utilidades para preguntar o establecer valores OID en hosts remotos. Afortunadamente, la mayoría de esas herramientas usa un conjunto de sintaxis compartidas y tienen patrones de uso similares. Revisaremos el uso básico de algunos de los más populares.

A partir de este punto nos referiremos a toda la información de autenticación que se le necesita dar a los comando snmp, como *authentication_info*.

Si se tiene una configuración con el archivo snmp.conf se puede remover esta sección del comando ya que los detalles de autenticación serán leídos desde dicho archivo.

Si no se tiene el archivo se deberá sustituir la información de autenticación en cada comando con la información necesaria para conectarse al *daemon* remoto. Como ejemplo la cuenta *demo*, con lo valores asignados previamente quedaría:

```
-u demo -l authPriv -a MD5 -x DES -A my_new_password -X my_new_password
```

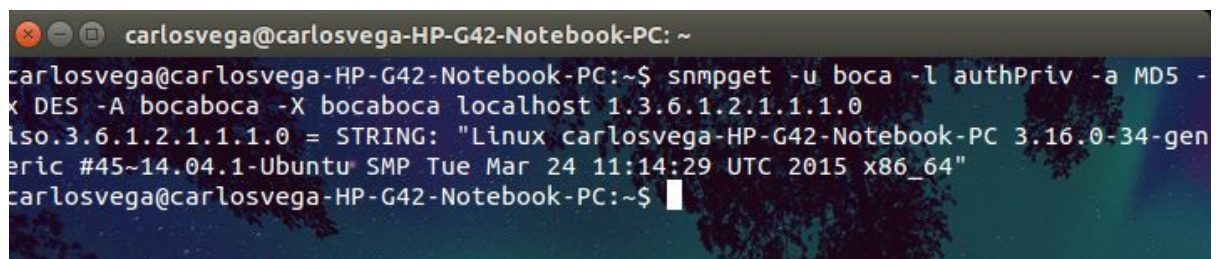
Ahora que tenemos la información de autenticación, veremos algunos de los comandos disponibles.

Obteniendo valores de un solo OID con snmpget

Esta es probablemente el comando más básico para preguntar información usando SNMP. Usando las banderas básicas de autenticación, el comando `snmpget` puede ser usado para leer el valor de cualquier OID al que el usuario tenga acceso.

El uso básico es especificar un OID numérico conocido. Por ejemplo, podemos recuperar la descripción del sistema introduciendo:

```
snmpget authentication_info host 1.3.6.1.2.1.1.1.0
```

A terminal window screenshot showing the command `snmpget -u boca -l authPriv -a MD5 -x DES -A bocaboca -X bocaboca localhost 1.3.6.1.2.1.1.1.0` being executed. The output is: `iso.3.6.1.2.1.1.1.0 = STRING: "Linux carlosvega-HP-G42-Notebook-PC 3.16.0-34-generic #45~14.04.1-Ubuntu SMP Tue Mar 24 11:14:29 UTC 2015 x86_64"`. The terminal title bar shows the user `carlosvega` on a machine named `carlosvega-HP-G42-Notebook-PC`.

```
carlosvega@carlosvega-HP-G42-Notebook-PC: ~  
carlosvega@carlosvega-HP-G42-Notebook-PC:~$ snmpget -u boca -l authPriv -a MD5 -  
x DES -A bocaboca -X bocaboca localhost 1.3.6.1.2.1.1.1.0  
iso.3.6.1.2.1.1.1.0 = STRING: "Linux carlosvega-HP-G42-Notebook-PC 3.16.0-34-gen  
eric #45~14.04.1-Ubuntu SMP Tue Mar 24 11:14:29 UTC 2015 x86_64"  
carlosvega@carlosvega-HP-G42-Notebook-PC:~$
```

Como hemos instalado el paquete *snmp-mibs-downloader* en nuestro equipo administrador, también podemos referenciar OID's comunes por su nombre. Por ejemplo, podemos obtener la misma información introduciendo:

```
snmpget authentication_info host sysDescr.0
```



```
carlosvega@carlosvega-HP-G42-Notebook-PC: ~  
carlosvega@carlosvega-HP-G42-Notebook-PC:~$ snmpget -u boca -l authPriv -a MD5 -  
x DES -A bocaboca -X bocaboca localhost -Oqv SNMPv2-MIB::sysDescr.0  
Linux carlosvega-HP-G42-Notebook-PC 3.16.0-34-generic #45~14.04.1-Ubuntu SMP Tue  
Mar 24 11:14:29 UTC 2015 x86_64  
carlosvega@carlosvega-HP-G42-Notebook-PC:~$
```

Obteniendo el valor del siguiente OID disponible con `snmpgetnext`

Este comando es usado para obtener el valor del OID que sigue de uno dado. Como la base de datos MIB es una jerarquía escalable, sus valores pueden ser recuperados secuencialmente. Al usar esta propiedad se puede encontrar el valor (y número OID) para el siguiente objeto de cualquiera en el árbol.

Por ejemplo, se vió cómo obtener la descripción del sistema. Para encontrar cuál es el siguiente OID y su valor se puede llamar al mismo comando, pero esta vez con el comando `snmpgetnext`.

`snmpgetnext authentication_info host sysDescr.0`

`SNMPv2-MIB::sysObjectID.0 = OID: NET-SNMP-MIB::netSnmpAgentOIDs.10`

```
carlosvega@carlosvega-HP-G42-Notebook-PC:~$ snmpgetnext localhost sysDescr.0  
SNMPv2-MIB::sysObjectID.0 = OID: NET-SNMP-MIB::netSnmpAgentOIDs.10
```

Esto regresa el ObjectID del sistema, el cual es el siguiente objeto secuencial en el árbol. Podemos repetir este paso una y otra vez usando el OID regresado para obtener cada objeto secuencial:

```
snmpgetnext authentication_info host sysObjectID.0
```

```
DISMAN-EVENT-MIB::sysUpTimeInstance = Timeticks: (114216) 0:19:02.16
```

Este comando puede usar OID's numéricos o de cadenas tal como la anterior.

Usando snmpwalk para recuperar un sección de la jerarquía MIB

Para obtener todos los OIDs debajo de alguno específico se puede usar el comando snmpwalk. Este regresa el árbol entero que existe debajo de un punto específico.

Por ejemplo, podemos obtener todos los valores en la porción de sistema del árbol introduciendo:

```
snmpwalk authentication_info host system
```

```
SNMPv2-MIB::sysDescr.0 = STRING: Linux target 3.13.0-24-generic #46-Ubuntu SMP Thu  
Apr 10 19:11:08 UTC 2014 x86_64
```

```
SNMPv2-MIB::sysObjectID.0 = OID: NET-SNMP-MIB::netSmpAgentOIDs.10
```

```
DISMAN-EVENT-MIB::sysUpTimeInstance = Timeticks: (126926) 0:21:09.26
```

```
SNMPv2-MIB::sysContact.0 = STRING: admin@example.com
```

```
SNMPv2-MIB::sysName.0 = STRING: target
```

```
SNMPv2-MIB::sysLocation.0 = STRING: Sitting on the Dock of the Bay
```

```
SNMPv2-MIB::sysServices.0 = INTEGER: 72
```

```
. . .
```

```
Archivo Editar Ver Buscar Terminal Ayuda
-x DES -A bocaboca -X bocaboca localhost -Oqv SNMPv2-MIB::system
Linux carlosvega-HP-G42-Notebook-PC 3.16.0-34-generic #45~14.04.1-Ubuntu SMP Tue
Mar 24 11:14:29 UTC 2015 x86_64
SNMPv2-SMI::enterprises.8072.3.2.10
0:1:58:40.95
Me <me@example.org>
carlosvega-HP-G42-Notebook-PC
Sitting on the Dock of the Bay
72
0:0:00:00.25
SNMPv2-SMI::snmpModules.11.3.1.1
SNMPv2-SMI::snmpModules.15.2.1.1
SNMPv2-SMI::snmpModules.10.3.1.1
SNMPv2-MIB::snmpMIB
SNMPv2-SMI::mib-2.49
SNMPv2-SMI::mib-2.4
SNMPv2-SMI::mib-2.50
SNMPv2-SMI::snmpModules.16.2.2.1
SNMPv2-SMI::snmpModules.13.3.1.3
SNMPv2-SMI::mib-2.92
The MIB for Message Processing and Dispatching.
The management information definitions for the SNMP User-based Security Model.
The SNMP Management Architecture MIB.
The MIB module for SNMPv2 entities
The MIB module for managing TCP implementations
The MIB module for managing IP and ICMP implementations
The MIB module for managing UDP implementations
View-based Access Control Model for SNMP.
The MIB modules for managing SNMP Notification, plus filtering.
The MIB module for logging SNMP Notifications.
0:0:00:00.24
0:0:00:00.24
0:0:00:00.24
0:0:00:00.25
0:0:00:00.25
0:0:00:00.25
0:0:00:00.25
0:0:00:00.25
0:0:00:00.25
0:0:00:00.25
0:0:00:00.25
carlosvega@carlosvega-HP-G42-Notebook-PC:~$
```

Este comando va ejecutando comando *snmpgetnext* al host hasta que puede construir el árbol completo debajo del valor solicitado.

Si se desea recuperar el árbol MIB entero, se puede ejecutar el comando en la raíz:

snmpwalk authentication_info host .

Esto regresará el árbol entero que es accesible al usuario.

Este puede ser usado, junto una búsqueda de nombre OID específicos con *grep*.

Por ejemplo, se puede saber que el OID *sysUpTime.0* regresa la cantidad de tiempo

que el *daemon* SNMP ha estado en operación en hosts remotos, pero tal vez se quiera saber cuánto ha estado en línea el servidor.

Se puede usar el comando `snmpwalk` para obtener la jerarquía entera de OIDs y luego filtrarla con `grep` para buscar cualquiera que tenga uptime en su nombre. Se usará la bandera `-i` para apagar la sensibilidad mayúsculas/minúsculas en la búsqueda:

```
snmpwalk authentication_info host . | grep -i uptime
```

Se obtendrá una respuesta que se verá de la siguiente manera:

```
DISMAN-EVENT-MIB::sysUpTimeInstance = Timeticks: (113856) 0:18:58.56
SNMPv2-MIB::sysORUpTime.1 = Timeticks: (0) 0:00:00.00
SNMPv2-MIB::sysORUpTime.2 = Timeticks: (0) 0:00:00.00
SNMPv2-MIB::sysORUpTime.3 = Timeticks: (0) 0:00:00.00
SNMPv2-MIB::sysORUpTime.4 = Timeticks: (0) 0:00:00.00
SNMPv2-MIB::sysORUpTime.5 = Timeticks: (0) 0:00:00.00
SNMPv2-MIB::sysORUpTime.6 = Timeticks: (0) 0:00:00.00
SNMPv2-MIB::sysORUpTime.7 = Timeticks: (0) 0:00:00.00
SNMPv2-MIB::sysORUpTime.8 = Timeticks: (0) 0:00:00.00
SNMPv2-MIB::sysORUpTime.9 = Timeticks: (0) 0:00:00.00
SNMPv2-MIB::sysORUpTime.10 = Timeticks: (0) 0:00:00.00
HOST-RESOURCES-MIB::hrSystemUptime.0 = Timeticks: (9741455) 1 day, 3:03:34.55
NOTIFICATION-LOG-MIB::nlmLogVariableID."default".1.1 = OID:
DISMAN-EVENT-MIB::sysUpTimeInstance
NET-SNMP-AGENT-MIB::nsModuleName."" .8.1.3.6.1.2.1.1.3.127 = STRING: mibII/sysUpTime
```

Después de intentar algunos de estos valores, se puede encontrar que el OID `hrSystemUptime.0` contiene el valor correcto de tiempo activo. Ahora, cada vez que

se quiera saber por cuánto tiempo ha pasado desde que la máquina inicio, podemos usar el OID:

```
snmpget authentication_info host hrSystemUptime.0
```

```
HOST-RESOURCES-MIB::hrSystemUptime.0 = Timeticks: (9795352) 1 day, 3:12:33.52
```

Como se puede observar, snmpwalk puede ser muy útil para descubrir los OIDs correctos para cierto valores.

Traducción entre OIDs numéricos y de texto con snmptranslate

Una de los comandos más útiles ni siquiera se necesita conectar al host remoto, sino simplemente ayuda a descubrir información acerca de la jerarquía del MIB.

Al usar snmptranslate, se puede convertir fácilmente resultados numéricos a su representación en texto.

```
snmptranslate 1.3.6.1.2.1.1.1.0
```

```
SNMPv2-MIB::sysDescr.0
```

Esto regresa el módulo MIB que define el nombre textual así como el nombre del OID.

También se puede usar la misma herramienta para el proceso inverso. Cuando se encontró el MIB textual para saber el tiempo activo del sistema (hrSystemUptime.0), se pudo haber tenido curiosidad por dónde estaba definido en el árbol. Se puede pasar el bandera -On para obtener la dirección numérica.

Hay que recordar incluir el módulo MIB que es dado cada que se recibe información acerca de un OID:

snmptranslate -On HOST-RESOURCES-MIB::hrSystemUptime.0

.1.3.6.1.2.1.25.1.1.0

También se puede usar esta herramienta para obtener muchos otros detalles acerca de cualquier punto. Por ejemplo, con la bandera -Td, se puede obtener una descripción completa con su camino en el árbol al final de la misma.

snmptranslate -Tp 1.3.6.1.2.1.1.1.0

SNMPv2-MIB::sysDescr.0

sysDescr OBJECT-TYPE

-- FROM SNMPv2-MIB

-- TEXTUAL CONVENTION DisplayString

SYNTAX OCTET STRING (0..255)

DISPLAY-HINT "255a"

MAX-ACCESS read-only

STATUS current

DESCRIPTION "A textual description of the entity. This value should include the full name and version identification of the system's hardware type, software operating-system, and networking software."

::= { iso(1) org(3) dod(6) internet(1) mgmt(2) mib-2(1) system(1) sysDescr(1) 0 }

Se puede modificar la salida mostrada con un parámetro -O_, donde el "_" es reemplazado por un formato de salida. Se puede ver la lista completa en la sección "OUTPUT OPTIONS" en la página man del comando snmpcmd, pero algunas de las opciones más comunes son:

Bandera	Descripción	Ejemplo
de Salida		

-Oa	Mostrar en cadenas ASCII	SNMPv2-MIB::sysDescr.0
-Of	Mostrar camino textual completo del OID	.iso.org.dod.internet.mgmt.mib-2.system.sysDescr.0
-On	Mostrar camino numérico del OID	.1.3.6.1.2.1.1.1.0
-Os	Mostrar sólo la representación textual final del OID	sysDescr.0

Hay que notar que las opciones de formato mencionadas también pueden ser aplicadas a la mayoría de las demás herramientas de esta suite para darle formato a la salida como sea requerido.

Obtener salida formateada y tabular usando snmptable

Parte de la información almacenada dentro de SNMP es tabular. A pesar de que snmpwalk tiene la habilidad de mostrar todos los datos relevantes, el formato no es ideal para ciertos usos.

Por ejemplo, si se usa snmpwalk en el OID udpTable:

snmpwalk authentication_info host udpTable

Se obtendría lo siguiente:

UDP-MIB::udpLocalAddress.0.0.0.0.161 = IPAddress: 0.0.0.0

UDP-MIB::udpLocalAddress.0.0.0.0.35679 = IPAddress: 0.0.0.0

UDP-MIB::udpLocalPort.0.0.0.0.161 = INTEGER: 161

UDP-MIB::udpLocalPort.0.0.0.0.35679 = INTEGER: 35679

Sin embargo, si hacemos la misma petición con snmptable:


```
snmptable authentication_info host udpTable
```

Se obtendría una tabla con un buen formato como la siguiente:

```
udpLocalAddress  udpLocalPort  
  
    0.0.0.0          161  
  
    0.0.0.0          35679
```

Este es un mejor formato y más fácil de leer para una persona.

Modificando valores con snmpset

Este comando es usado para escribir el valor a un OID. Mientras que hasta ahora, otros comandos han sido usados para obtener información, este comando es usado para modificar datos en el host.

Aunque el comando snmpset hereda la mayoría de su sintaxis de otros comandos, requiere de información adicional para establecer los valores. La sintaxis básica luce de la siguiente manera:

```
snmpset authentication_info host OID_to_modify data_type new_value
```

La mayoría de los campos son autodescriptivos. Sin embargo, los tipos de dato requieren de más explicación. Cada tipo es representado por un solo caracter. La lista de posibles tipos se muestra a continuación.

- **i**: Integer
- **u**: Unsigned integer
- **s**: String
- **x**: Hexadecimal string
- **d**: Decimal string
- **n**: Null object

- **o**: Object ID
- **t**: Time ticks
- **a**: IP Address
- **b**: Bits

Como se ha descargado el paquete *snmp-mibs-downloader*, la mayoría del tiempo se puede introducir solamente '=' en lugar de uno de los tipos de identificadores.

Para demostrar este comando, se puede comentar uno de los valores establecidos en el archivo *snmpd.conf* en la computadora agente. El especificar valores en el archivo de configuración esencialmente lo hace a prueba de comandos, previniendo su modificación usando comandos snmp.

En la computadora agente, se abre el archivo */etc/snmp/snmpd.conf*:

```
sudo nano /etc/snmp/snmpd.conf
```

Comentar la directiva *sysLocation*:

```
#sysLocation Sitting on the Dock of the Bay
```

Guardar y cerrar el archivo. Y reiniciar el servicio.

```
sudo service snmpd restart
```

Ahora, desde la computadora administradora, se puede establecer la *sysLocation* como "Earth" introduciendo lo siguiente. Tomar en cuenta que la "s" especifica que el tipo de dato es un String.

```
snmpset authentication_info host sysLocation.0 s "Earth"
```

```
SNMPv2-MIB::sysLocation.0 = STRING: Earth
```

Para probar el especificador de tipo '=' introducimos:

```
snmpset authentication_info host sysLocation.0 = "New York City"
```

```
SNMPv2-MIB::sysLocation.0 = STRING: New York City
```

Así interpreta correctamente el valor como una cadena de texto.

Solicitudes eficientes con snmpbulkget y snmpbulwalk

El ejecutar solicitudes snmpget y snmpwalk puede crear mucho tráfico de red cuando son usadas repetidamente.

Los comandos snmpbulkget y snmpbulkwalk fueron creados para reducir este problema. Estos empaquetarán todos los valores regresados dentro de una sola transacción en vez de una transacción por cada valor de OID regresado. También se puede pasar más de un OID a la vez.

Para usar snmpbulkget, se pasa uno o más OIDs o ramas y se obtendrán tantos valores de OIDs adicionales como quepan en el paquete.

```
snmpbulkget authentication_info host system
```

```
SNMPv2-MIB::sysDescr.0 = STRING: Linux target 3.13.0-24-generic #46-Ubuntu
```

```
SMP Thu Apr 10 19:11:08 UTC 2014 x86_64
```

```
SNMPv2-MIB::sysObjectID.0 = OID: NET-SNMP-MIB::netSnmpAgentOIDs.10
```

```
DISMAN-EVENT-MIB::sysUpTimeInstance = Timeticks: (429891) 1:11:38.91
```

```
SNMPv2-MIB::sysContact.0 = STRING: call now
```

```
SNMPv2-MIB::sysName.0 = STRING: target
```

```
SNMPv2-MIB::sysLocation.0 = STRING: New York City
```

```
SNMPv2-MIB::sysServices.0 = INTEGER: 72
```

```
SNMPv2-MIB::sysORLastChange.0 = Timeticks: (0) 0:00:00.00
```

```
SNMPv2-MIB::sysORID.1 = OID: SNMP-MPD-MIB::snmpMPDCompliance
```

```
SNMPv2-MIB::sysORID.2 = OID:
```

SNMP-USER-BASED-SM-MIB::usmMIBCompliance

Una cosa que hay que notar es que snmpbulkget opera como un comando snmpgetnext, ya que deja al objeto dado como un argumento. En el ejemplo de arriba, en lugar de darle un objeto específico al argumento, se le ha dado una rama. Se puede pensar al comando snmpbulkget como una llamada a snmpwalk pero sus resultados serán empaquetados.

El comando snmpbulkwalk opera de manera similar pero continuará haciendo comandos bulkget hasta que el subárbol entero sea recuperado.

Conclusión

Como se puede ver, al usar la suite net-snmp se puede recuperar y manipular datos de una gran variedad de formas. Al hacer un script con estas acciones o al aprovechar estas utilidades en otras aplicaciones se puede construir ambientes complejos de monitoreo y administración.

