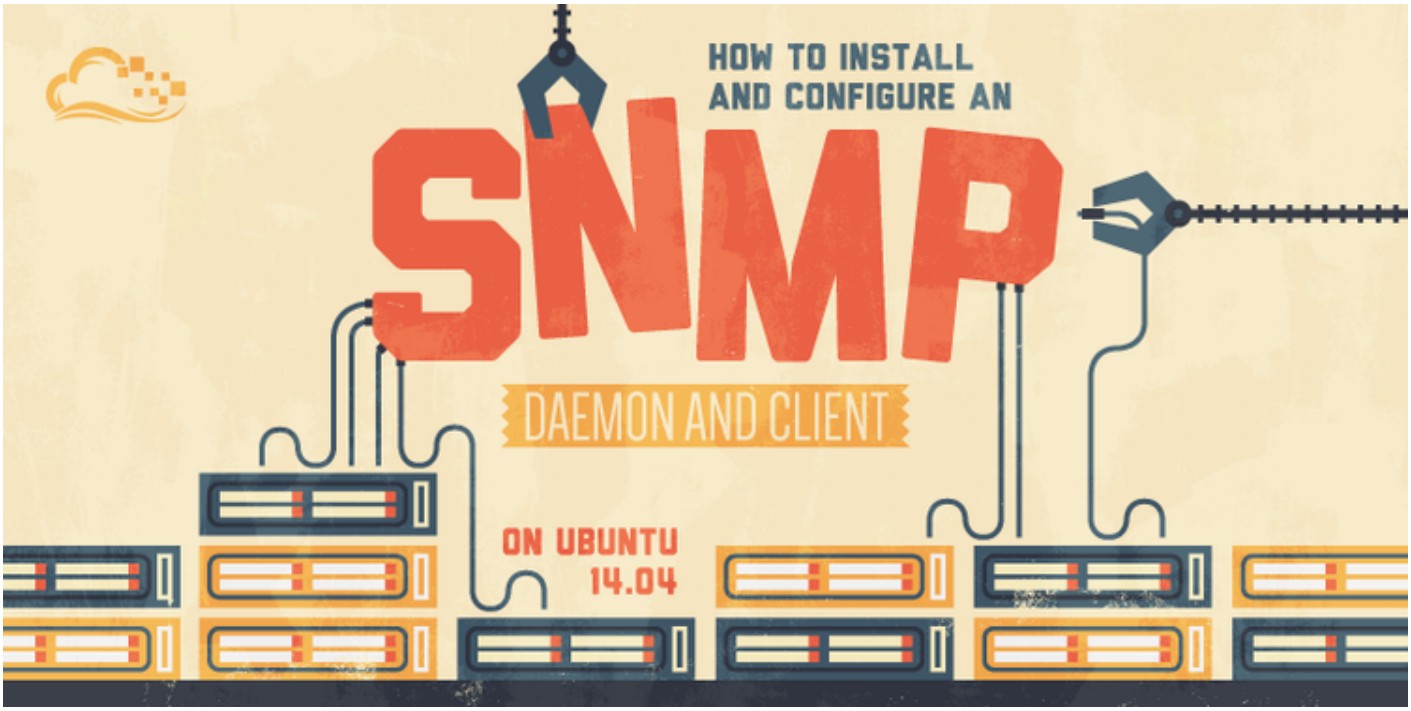
By: Justin Ellingwood



How To Install and Configure an SNMP Daemon and Client on Ubuntu 14.04

Posted Aug 18, 2014 228.1k Networking System Tools Ubuntu

Tutorial Series

This tutorial is part 2 of 3 in the series: Monitoring and Managing your Network with SNMP

Introduction

A large part of being a system administrator is collecting accurate information about your servers and infrastructure. There are a number of tools and options for gathering and processing this type of information. Many of them are built upon a technology called **SNMP**.

SNMP stands for simple network management protocol. It is a way that servers can share information about their current state, and also a channel through which an administer can modify pre-defined values.

While the protocol itself is very simple, the structure of programs that implement SNMP can be very complex.

In our last guide, we learned about the basics of the SNMP protocol. In this guide, we will begin to demonstrate how to setup the tools to communicate using SNMP. We will be using two Ubuntu 14.04 servers to demonstrate, but most other systems should be able to follow along with a few modifications.

Install the SNMP Daemon and Utilities

We can begin to explore how SNMP can be implemented on a system by installing the daemon and tools on some Ubuntu systems.

We will use two servers, one will contain the manager portion, while the other server will have the agent. We could choose to install the agent on the manager machine as well, but keeping them separate makes it easier to demonstrate what functionality is provided by each component.

On the first server, update the apt database and install the manager component. Along with this, we will also download another package called `snmp-mibs-downloader` which contains some proprietary information about standard MIBs that allow us to access most of the MIB tree by name:

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

On our second server, the one that we will be interacting with that will run the daemon, we can install the necessary components by typing:

```
sudo apt-get update
sudo apt-get install snmpd
```

Now that you have installed these components, we need to configure our setup.

Configuring the SNMP Manager

As we mentioned above, most of the bulk of the work happens in the agent component, so our configuration is actually pretty easy on this machine. We just need to modify one file to make sure that our client can use the extra MIB data we installed.

Open the `/etc/snmp/snmp.conf` file in your text editor with sudo privileges:

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

In this file, there are a few comments and a single un-commented line. To allow the manager to import the MIB files, we simply need to comment out the `mibs :` line:

```
#mibs :
```

Save and close the file when you are finished.

We are now finished configuring the manager portion, but we will still need to use this server to help us configure our agent computer.

Configuring the SNMP Agent Machine

As a true client-server system, the agent computer does not have any of the external tools needed to configure its own SNMP setup. We can modify some configuration files to make some changes, but most of the changes we need to make will be done by connecting to our agent server from our management server.

To get started, on our agent computer, we need to open the daemon's configuration file with sudo privileges:

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

Inside, we will have to make a few changes. These will mainly be used to bootstrap our configuration so that we can manage it from our other server.

First, we need to change the `agentAddress` directive. Currently, it is set to only allow connections originating from the local computer. We need to comment out the current line, and uncomment the line underneath, which allows all connections (we will be locking this down soon):

```
#  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
```

Next, we will need to temporarily insert a `createUser` line. These directives are not normally kept in this file, but we will be removing it again in a moment, so it doesn't matter too much.

The user we are creating will be called `bootstrap` and will be used as a template in which to create our first "real" user. The SNMP packages do this through a process of cloning the user's properties.

When defining a new user, you must specify the authentication type (MD5 or SHA) as well as supply a passphrase that **must** be at least 8 characters. If you plan on using encryption for the transfer, like we are, you also must specify the privacy protocol (DES or AES) and optionally a privacy protocol passphrase. If no privacy protocol passphrase is supplied, the authentication passphrase will be used for the privacy protocol as well.

Our user creation line will look like this:

```
createUser bootstrap MD5 temp_password DES
```

Now that we have a new user specified, we need to set up the level of access that this user will have. We will set this up for our `bootstrap` user, and also for the new user we will be creating, called `demo`. We will allow them read and write access by using the `rwuser` directive (the alternative is `rouser` for read-only access).

We will enforce the use of encryption by specifying `priv` after our user. If we wanted to restrict the user to a specific part of the MIB, we could specify the highest-level OID that the user should have access to at the end of the line.

For our purposes, both of our lines will be fairly simple:

```
rwuser bootstrap priv
rwuser demo priv
```

When you are finished making these changes, save and close the file.

To implement these changes, restart the `snmpd` service:

```
sudo service snmpd restart
```

Now, from the machine that you installed the management software on, we can connect to our agent server to create our regular user.

We will do this using the `snmpusm` tool, which is used for user management. You will need to know the IP address of your agent server for this to function correctly.

Before we begin, we will talk a bit about the general structure of sending an SNMP command.

The General Structure of SNMP Commands

When using the suite of tools included in the `snmp` package (the `net-snmp` software suite), you will notice a few patterns in the way you must call the commands.

The first thing you must do is authenticate with the SNMP daemon that you wish to communicate with. This usually involves supplying quite a few pieces of information. The common ones are below:

- **-v VERSION**: This flag is used to specify the version of the SNMP protocol that you would like to use. We will be using v3 in this guide.
- **-c COMMUNITY**: This flag is used if you are using SNMP v1 or v2-style community strings for authentication. Since we are using v3-style user-based authentication, we will not be needing this.
- **-u USER-NAME**: This parameter is used to specify the username that you wish to authenticate as. To read or modify anything using SNMP, you must authenticate with a known username.
- **-l LEVEL**: This is used to specify the security level that you are connecting with. The possible values are `noAuthNoPriv` for no authentication and no encryption, `authNoPriv` for authentication but no encryption, and `authPriv` for authentication and encryption. The username that you are using must be configured to operate at the security level you specify, or else the authentication will not succeed.
- **-a PROTOCOL**: This parameter is used to specify the *authentication* protocol that is used. The possible values are `MD5` or `SHA`. This must match the information that was specified when the user was created.
- **-x PROTOCOL**: This parameter is used to specify the *encryption* protocol that is used. The possible values are `DES` or `AES`. This must match the information that was specified when the user was created. This is necessary whenever the user's privilege specification has `priv` after it, making encryption mandatory.
- **-A PASSPHRASE**: This is used to give the authentication passphrase that was specified when the user was created.

- **-X PASSPHRASE**: This is the encryption passphrase that was specified when the user was created. If none was specified but an encryption algorithm was given, the authentication passphrase will be used. This is required when the `-x` parameter is given or whenever a user's privilege specification has a `priv` after it, requiring encryption.

Using this information, we can begin to construct our commands. Given how we set up our bootstrap user, the commands we will be using with that account will look like this:

```
snmp_command -u bootstrap -l authPriv -a MD5 -x DES -A temp_password -X temp_password remote_host
```

For instance, from your management server, you can test to make sure your bootstrap account is available by typing:

```
snmpget -u bootstrap -l authPriv -a MD5 -x DES -A temp_password -X temp_password remote_host 1.3.6
```

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

The `1.3.6.1.2.1.1.1.0` string is the OID that is responsible for displaying system information. It will basically return the output of `uname -a` on the remote system.

Now that we have verified that we can correctly authenticate to the server running the SNMP daemon, we can continue on to create our regular user account.

Set Up the Regular User Account

Although we have specified the privileges for the `demo` user account in our `snmpd.conf` file, we haven't actually created this user yet. We are going to use the `bootstrap` user as a template for our new user.

On the management server, we can create the user from the template using the `snmpusm` tool and the following general syntax:

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

So, using what we know about the authentication flags we need to pass, and leveraging the user account we already have (`bootstrap`), we can make a user that fits the user privileges we have already defined (`demo`).

The command will look like this:

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

You should receive the following message:

```
User successfully created.
```

We now have a fully functioning user called `demo` on our remote server. However, it is still using the same authentication information as the `bootstrap` account. We should change the password to something else. This time, we will use the `demo` account to authenticate. Remember, passwords **must** be at least 8 characters long:

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

You should receive the following message back:

```
SNMPv3 Key(s) successfully changed.
```

We can test our new credentials and password by asking our remote server how long the SNMP service has been running. We will use the `snmpget` command to get a single value from the other machine.

This time, we will take advantage of the extra MIB definitions we downloaded. We can use these to ask for the value by name instead of the OID numeric ID.

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

You should get back a value that represents the last time that the remote SNMP daemon was restarted:

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

Command Flag	Description	Translated `snmp.conf` directive

Creating a Client Configuration File

You have probably noticed by this point that the authentication details for all of your SNMP commands will be fairly static with each request. Rather than typing these in each time, we can create a client-side configuration file that will contain the credentials we are connecting with.

The client configuration file can be placed in two different locations depending on how wide-spread you wish to share it.

If you want to share your login credentials with any valid user on your management machine, you can place your configuration details into the global `snmp.conf` file. You would need to open that file with sudo privileges:

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

If, however, you want to define the authentication credentials for your user alone, you can create a hidden `.snmp` directory within your user's home directory, and create the file there:

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

Regardless of your decision on where to place your configuration, the contents will be the same.

The commands that we are using to authenticate are in the table below. In the right-hand column, you can see the directive names that should be used to set those configuration details within the `snmp.conf` file:

Command Flag	Description	Translated `snmp.conf` directive
-u USERNAME	The SNMPv3 username to authenticate as.	defSecurityName USERNAME
-l authPriv	The security level to authenticate with.	defSecurityLevel authPriv

Command Flag	Description	Translated `snmp.conf` directive
-a MD5	The authentication protocol to use.	defAuthType MD5
-x DES	The privacy (encryption) protocol to use.	defPrivType DES
-A PASSPHRASE	The authentication passphrase for the supplied username.	defAuthPassphrase PASSPHRASE
-X PASSPHRASE	The privacy passphrase fro the supplied username.	defPrivPassphrase PASSPHRASE

Using this information, you can construct an appropriate `snmp.conf` file. For our guide, this will look like this:

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

When you are finished, save and close the file.

Now, you can issue commands without supplying the authentication details. You will only need the SNMP command, the host, and the command arguments.

Instead of typing:

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

We can simply type:

```
snmpget remote_host sysUpTime.0
```

As you can see, this significantly reduces the amount of information we need to supply in each request.

Removing the Bootstrap Account

Now that your regular account is configured correctly, we can remove the `bootstrap` account, since it is fairly insecure.

On your agent server, open the `/etc/snmp/snmpd.conf` file again with sudo privileges.

Find and comment out (or remove) both of the lines that we previously added that reference the `bootstrap` user:

```
#createUser bootstrap MD5 temp_password DES
#rwuser bootstrap priv
```

Save and close the file.

Now, restart the SNMP daemon:

```
sudo service snmpd restart
```

This will fulfill the recommendation of not having `createUser` directives in the normal `snmpd.conf` file. It will also remove privileges from that temporary user.

If you want to completely remove the `bootstrap` user from the usmUserTable, you can do so by issuing this command from the management server:

```
snmpusm remote_host delete bootstrap
```

You will receive the following response:

```
User successfully deleted.
```

Conclusion

At this point, you should have a fully configured client-server setup what can communicate securely using the SNMP protocol. You can easily add additional daemons on other hosts and configure account access across your entire infrastructure.

In the next guide, we'll go over some of the basic usage of the net-snmp tools that we have been working with. We will demonstrate how to retrieve values one-by-one or by bulk and how to modify data.

By: Justin Ellingwood

Tutorial Series

Monitoring and Managing your Network with SNMP

SNMP, or simple network management protocol, is a well-established way of monitoring and managing diverse sets of networked systems. In this series, we will introduce you to the basics of the protocol, teach you how to install the agent and manager components on several hosts, and demonstrate how to use the net-snmp suite of utilities to gather information and modify the configuration of remote hosts.

Related Tutorials

How To Use Netcat to Establish and Test TCP and UDP Connections on a VPS

How To Use LVM To Manage Storage Devices on Ubuntu 16.04

An Introduction to LVM Concepts, Terminology, and Operations

How To Install and Configure Zabbix to Securely Monitor Remote Servers on CentOS 7

How To Set Up an NFS Mount on Ubuntu 16.04

5 Comments

1

brodock *August 26, 2014*

Why are all the examples using MD5 and DES when both are considered now extreme insecure?

People who are reading this tutorial should be advised to stick to SHA and AES.
While both MD5 and SHA can be used to obfuscate the plain password string, it's well known that both can be easily beated by a number of attacks like comparing pre-generated hashes using a rainbow tree, or using the power of dedicated graphics card to brute-force them, so consider it as unsafe as storing the

plaintext, which means, that anyway who can access the agent machine will be able to recover that password.

That leads to another recommendation to use unique password on every agent machine.

0

marcin170288 *November 20, 2014*

Thanks for this great tutorial:)
by the way I find small bug in "Removing the Bootstrap Account" section , should be `#rwuser bootstrap priv` instead `#rwuser demo priv`

0

jellingwood MOD *November 20, 2014*

@marcin170288: You're absolutely right. Great catch! Should be good to go now.

Thanks!

0

scotterk *April 1, 2015*

Never set this up before, I have two questions:
Under Install SNMP Daemmon and Utilities, it says to install the manager component software. Where do I find this software?

SCROLL TO TOP

Also, where it says I will need the IP of the agent machine to use snpusm, where do I specify or enter that IP address?

Thanks a bunch, sorry I'm really new to this and my boss asked me to set it up.

0

tknerr *October 9, 2015*

Hi, nice tutorial!

One thing I didn't get though: why not creating the "demo" directly via the "createUser" directive in /etc/snmp/snmpd.conf?

Why the indirect way using a temporary bootstrap user?