command **SnmpWalk.exe**. However, you don't need to type the **.exe** part.

Parameters and Options for snmpwalk in Windows

The **SnmpWalk** command runs in a Command Prompt window and it has its own parameter invoking format. The takes the form of a dash, an identifier, a colon, and then a value. For example, -r:192.168.2.1 gives the IP address of a device to query. Some variables are switches and don't need a value.

Here is a list of the variables for SnmpWalk:

Parameter	Description	Default
-q	Quiet mode (suppress header; print variable values only)	
-r:host	Name or network address (IPv4/IPv6) of remote host	
-p:port	SNMP port number on remote host	161
-t:timeout	SNMP timeout in seconds (1-600)	5
-v:version	SNMP version. Supported version: 1, 2c or 3	1
-c:community	SNMP community string for SNMP v1/v2c	public
-ei:engine_id	Engine ID. Format: hexadecimal string. (SNMPv3)	
-sn:sec_name	SNMP security name for SNMPv3	
-ap:auth_proto	Authentication protocol. Supported: MD5, SHA (SNMPv3)	

WHAT'S IN THIS ARTICLE?

Parameter	Description	Default
-aw:auth_passwd	Authentication password (SNMPv3)	
-pp:priv_proto	Privacy protocol. Supported: DES, IDEA, AES128, AES192, AES256, 3DES (SNMPv3)	
-pw:priv_passwd	Privacy password (SNMPv3)	
-cn:cont_name	Context name. (SNMPv3)	
-ce:cont_engine	Context engine. Format: hexadecimal string (SNMPv3)	
-os:start_oid	Object ID (OID) of first SNMP variable to walk	0.1
-op:stop_oid	Object ID (OID) of last SNMP variable to walk	Walk to the very last variable
-csv	Output in CSV (Comma Separated Values) format	

Some of the parameters can only be used with SNMP version 3. This requirement is noted in the entries for the relevant parameters in the above table.

Installing snmpwalk in Linux

On Linux devices, snmpwalk is available as a package for you to install. The process of doing this depends on your Linux distribution. However, you can install Linux by entering the following commands:

- Redhat/Fedora/CentOs:

```
yum install net-snmp-utils
```

- Ubuntu:

```
apt-get install snmp
```

Parameters and Options for snmpwalk in Linux

The snmpwalk command has has a range of different parameters that you can use. These include the following:

hostname	The SNMP agent name
community	The type of read community
object_id	Specify an object ID to return all SNMP objects below it. If NULL then the root of the SNMP object is taken as the object_id
timeout	Number of microseconds before the first timeout
retries	How many times to retry the connection in the event of a timeout
-Os	Shows the last symbolic element of an OID
-c	Sets a community string
-v	Specifies the SNMP version you want to use

How to Minimize the Information You See With snmpwalk

One of the first things you'll notice when running snmpwalk is that there are too many results for you to read. You can easily end up with thousands of different results. To get around this vendors help you by providing you with a **MIB file**. **The MIB file is used to specify which OIDs are available on a device**. Many vendors provide you with a MIB file for each device you have. Having a MIB file allows you to run a query that is specific to that file, rather than walking through everything.

See also: SMNP tools

A note about the SNMP MIB structure

The MIB has a tree structure and there is no set width of branching at each node. The OID is a dot-notation system that shows the trail from the root node to the current node. Each node is represented by a number that has a meaning but you need a reference document to work out what each value means. The OID for a node includes the numbers of all the parent nodes above it. So, the OID shows a path down to a specific point on the MIB tree.

Not every node will be present because in some cases, a node has a series of possible child nodes, each of which represents a value option, for example, 1=Yes and 2=No, so that level of the tree wouldn't have both the 1 and 2 nodes present because they are mutually exclusive.

How snmpwalk operates

The snmpwalk function is like a crawler that steps down each branch of the tree as far as values are available. When it reaches the end of the line, it looks up one level to see whether there are other nodes available apart from the one that was just visited. Each of the nodes can have a variable extent – it is possible that one leads down several levels, another has only one node below it, and another is the end node of a line.

The root node has no number and all of the useful SNMP values are found below just one of the three nodes beneath root. This is the **iso** node, which is numbered **1**. There is only one node beneath iso, which is the **org** node and it is numbered **3**. Beneath the org node is only one node, which is **dod** (**6**). There is only one node beneath dod, which is called **internet** and it is numbered **1**. The next level has four nodes, but if you are working with a LAN, all of your meaningful data will appear under **private**, which is number **4**. Below this, you will find only one node: **enterprise** (number **1**).

The top part of the MIB tree means that all useful MIB OIDs start with **.1.3.6.1.4.1**. If you are a regular investigator of SNMP data, you will remember .1.3.6.1.4.1 like your bank card PIN.

There is one more factor that you need to know about the numbers in the OID. This relates to the manufacturer of the device that hosts the SNMP agent. The SNMP agent is pre-installed on all network devices. This courtesy has become an industry standard. So, before shipping each device, the producer installs firmware and a number of utilities, including the SNMP agent. This provides the manufacturer with an opportunity to mark its territory

because the enterprise node can be followed by an identifier that denotes the supplier.

So, the industry has got together and assigned a number to each manufacturer. These are called Private Enterprise Numbers PENs) and the list is maintained by the Internet Assigned Numbers Authority (IANA). You can search the List of Private Enterprise Numbers at the IANA website.

So, if you are looking for data in a MIB from a switch created by Cisco Systems, the OIDs you will be looking at will all start with .1.3.6.1.4.1.9. For a device from APC, the OIDs with interesting information will all start with .1.3.6.1.4.1.318.

Using snmpwalk

The exact syntax for the snmpwalk command depends on the implementation that you are using. So, with the examples that you were advised to download above, the format of the command would be:

For Linux –

```
$ snmpwalk –v1 –c public 192.168.8.1
```

This uses SNMP v1 with the community string “public” and looks at the device with the IP address 192.168.8.1.

For Windows, that same query would be written as –

```
C:\Users\Laptop> snmpwalk –v:1 –r:192.168.8.1 –
c:”public”
```

In that Windows version, you wouldn’t need to put in the -v:1 or -c:”public” parameters because these give

the default values. So, if you missed those out, the command would use those values anyway.

Knowing that you are only going to get interesting information from a specific branch of the MIB tree, you can cut the scan short to skip directly to that part of the MIB.

On Linux, try:

```
$ snmpwalk -v1 -c public 192.168.8.1
.1.3.6.1.4.1.318
```

On Windows, you would type:

```
C:\Users\Laptop> snmpwalk -r:192.168.8.1 -
os:.1.3.6.1.4.1.318
```

These two examples are for querying an APC device that has the IP address 192.168.8.1, so alter those numbers for your network.

Save Time with snmpwalk

The concept of **snmpwalk** seems complicated, but it isn't in reality. The snmpwalk command is simply **a shorthand way to use multiple GETNEXT requests without having to type lots of different commands**. With one snmpwalk command, you can fire off lots of different GETNEXT and view your infrastructure health.

If you're using the snmpwalk command then it's important to remember to cut down the amount of information you receive as much as possible. Specifying a MIB will allow you to see specific information rather than data collected from an entire database of object IDs!