Global Information Assurance Certification Paper

GCIH version 2.1a – Option 2
Support for the Cyber Defence Initiative
SNMP – Hiding in Plain Sight

Jim Sproule

Abstract

Some say the road less travelled is often less travelled for a reason. Ignoring this, I requested and was granted permission to deviate slightly from the prescribed format of the GCIH certification report and chose ports 161 and 162 (SNMP) for my contribution to the Cyber Defence Initiative.

The Simple Network Management Protocol (SNMP) has two types of vulnerabilities. Those inherent to SNMP versions 1 and 2; and those that are the result of vendor implementations. A vendor implementation fault in SNMP on Microsoft Windows 2000 systems results in a memory leak vulnerability that is exploitable with any SNMP management software or utility. Microsoft fixed the problem in a service pack release but legacy systems remain exploitable.

An attack against a vulnerable system consumes memory at approximately 30 megabytes per SNMP request. Continuous SNMP requests will consume all available memory and the system's performance will degrade. The impact of the exploit is a denial of service of the functions provided by the victim system.

The report contains specific mitigation strategies for vulnerable Windows 2000 systems. Also, since many vendors provide SNMP functionality turned on by default using well-known "passwords" this report contains general mitigation strategies to protect environments from SNMP vulnerabilities.

I hope that you will find some interest in my submission.

Table of Contents

List of Figures

List of Tables

Part 1 – Targeted Ports

Targeted Service

This report is about the Simple Network Management Protocol (SNMP).

The SANS/FBI Top20[1] vulnerability list has a section titled the "Top Vulnerabilities to UNIX Systems[2]". Its fourth entry is the Simple Network Management Protocol (SNMP). This entry documents some of the vulnerabilities of SNMP version 1.

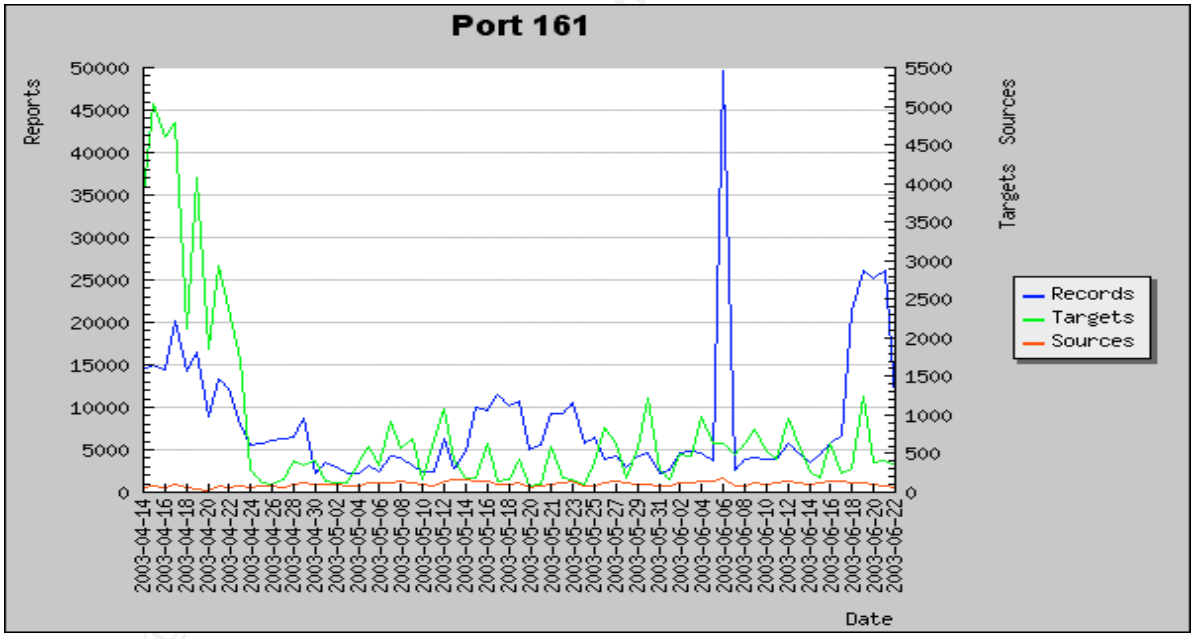
The Internet Assigned Numbers Authority[3] (IANA) documents the following:

- Port 161 both TCP and UDP are assigned to Simple Network Management Protocol (SNMP)
- Port 162 both TCP and UDP are assigned to (Simple Network Management Protocol TRAPS (SNMPTRAP)

The "requests for comments" (RFC) number 1700[4] confirms this port assignment.

Participating system and firewall administrators submit logs from Internet firewalls to the Internet Storm Center[5] (ISC). The ISC tabulates and provides an interface to this data. This service generated the following graphs for ports 161 and 162.

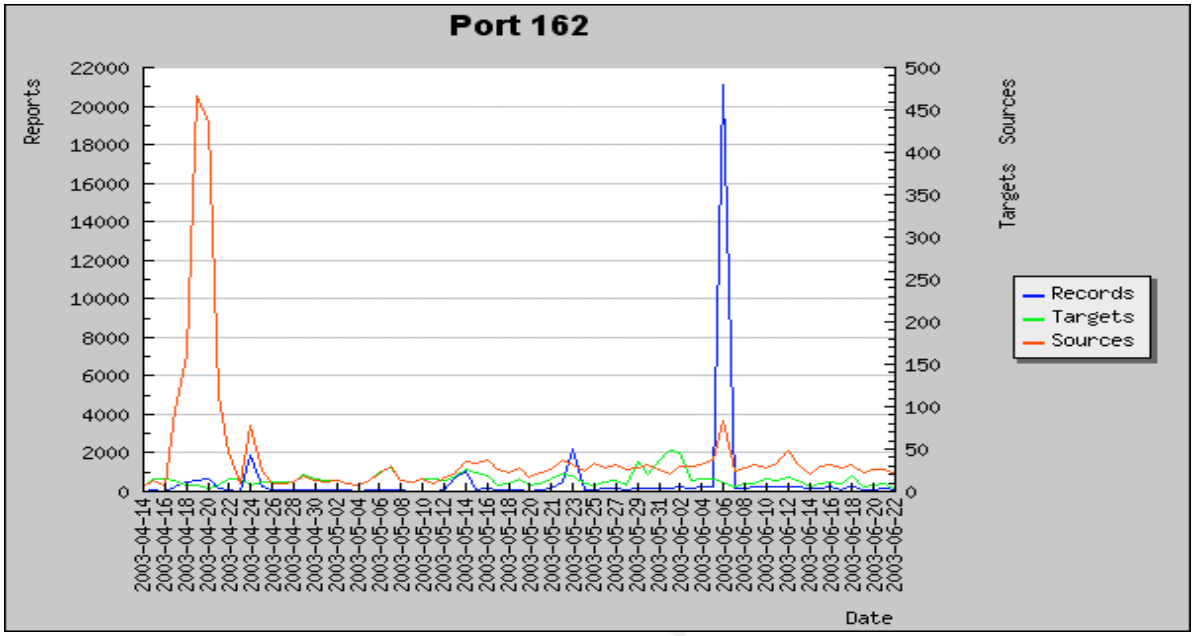
Port 161 - 70 Days of History[6]



• Figure 1: 70 Day History Port 161

[1] "TOP 20 LIST" July 5 2003 URL: http://www.sans.org/top20/
[2] "Simple Network Management Protocol" July 3 2003 URL: http://www.sans.org/top20/#U4
[3] PORT NUMBERS" June 30, 2003 URL: http://www.iana.org/assignments/port-numbers
[4] "ASSIGNED NUMBERS" June 30, 2003 URL: http://www.cis.ohio-state.edu/cgi-bin/rfc/rfc1700.html
[5] "Trends" June 22, 2003 URL: http://isc.incidents.org/trends.html
[6] "Port Reports" June 22, 2003 URL:
http://isc.incidents.org/port_details.html?port=161&repax=1&tarax=2&srcax=2&percent=N&days=70&Redraw=Submit+Query

Port 162 - 70 Days of History[7]



• Figure 2: 70 Day History Port 162

These charts show the number of source IP addresses (Sources), target IP addresses (Targets), and records reported (Reports) for SNMP traffic on the Internet as reported by the ISC participants. The charts indicate the relative amount of port scanning activity over the chart period for SNMP (port 161) and SNMPTRAP (port 162).

When compared to other "popular" ports the numbers of sources and targets reported in these charts is very small. For example, for port 80 the ISC reports source and target counts measured in the hundreds of thousands. However, the charts do contain some interesting information.

During the period April 14th through April 22nd, the number of SNMP (port 161) target addresses averages between 3000 and 5000 whereas the average over the remaining period is less than 1000. This represents an approximate 300% increase.

During the same period, there is a corresponding increase in source addresses of SNMPTRAPS (port 162). Though trivial in absolute terms with source and target addresses measured in the hundreds, this period experienced a large relative increase. This is not evidence to suggest an attack using SNMP - if it was an attack it was very limited - but something happened on ports 161 and 162 between April 14th and April 22nd.

[7] "Port Reports" June 30, 2003 URL: http://isc.incidents.org/port_details.html?port=162&repax=1&tarax=2&srcax=2&percent=N&days=70&Redraw=Submit+Query

Description of SNMP

Overview of Network Management and SNMP

The Simple Network Management Protocol (SNMP) provides a communication mechanism for managing Internet Protocol (IP) connected objects. Originally intended for monitoring and controlling network devices, manageable entities now include any IP connected object that complies with the SNMP standards. Printers, servers, routers, firewalls, switches, telephony equipment, HVAC systems, production lines, power control systems, and applications are all examples of SNMP manageable entities.

Stevens[8] provides a concise description of IP network management using SNMP:

> TCP/IP network management consists of three pieces.
>
> 1. A Management Information Base (MIB) that specifies what variables the network elements maintain (the information can be queried and set by the manager). RFC 1213 [McCloghrie and Rose 1991] defines the second version of this, called MIB-II.
>
> 2. A set of common structures and an identification scheme used to reference the variables in the MIB. This is called the Structure of Management Information (SMI) and is specified in RFC 1155 [Rose and McGloghrie 1990]. For example, the SMI specifies that counter is a nonnegative integer that counts from 0 through 4,294,967,295 and then wraps around to 0.
>
> 3. The protocol between the manager and the element, called Simple Network Management Protocol (SNMP). RFC 1157 [Case et al. 1990] specifies the protocol. This details the format of the packets exchanged. Although a wide variety of transport protocols could be used, UDP is normally used with SNMP.

The communication stream of SNMP version 1 (SNMPv1) has only five commands and responses. The data retrieved and manipulated by these commands has a standardized format and indexing scheme. This standardized data, common indexing scheme, and limited command set provide the common basis for managing a diverse population of entities.

The Simple Network Management Protocol (SNMP): RFCs and Versions

The following is a summary of the information contained in some of the requests for comments (RFCs) associated with the components of SNMP. A web search resulted in the following RFC counts[9].

[8] "Chapter 25: SNMP" TCP/IP Illustrated, Volume1 The Protocols: W. Richard Stevens
[9] "Category/Subject or Protocol at URL" June 29, 2003 URL: http://www.garlic.com/~lynn/rfcterms.htm

Category/Subject or Protocol	Acronym	Number of RFCs
Network Management	NMS	337
Simple Network Management Protocol version 1	SNMPv1	96
Simple Network Management Protocol version 2	SNMPv2	25
Simple Network Management Protocol version 3	SNMPv3	26
Structure of Management Information	SMI	4
Structure of Management Information version 2	SMIv2	15
Management Information Base	MIB	239
Management Information Base version 2	MIB-2	44

• Table 1: RFC Counts by SNMP Associated Category

There are currently three versions of SNMP.

Version 1 (SNMPv1), detailed in RFC-1157[10], currently has a status of "Historic"[11]. This RFC documents the basic workings of SNMPv1 and the basic architectural model. SNMPv1 applications reside on "network management stations" and these manage "network elements". A network management station can monitor and control many network element end-points in a typical hub and spoke fashion.

The RFC specifies that the management data presentation and communication encoding must use a subset of the Basic Encoding Rules[12] (BER) and a sub-set of the definitions within the Abstract Syntax Notation One[13] (ASN.1). The RFC defines the five mandatory commands and responses:

- GetRequest — retrieve a particular data object value
- GetNextRequest — retrieve the next data object value
- GetResponse — return a data object value
- SetRequest — set a data object value
- Trap — notify that a data object value has changed

Within the specifications are provisions for defining "communities" which connect a managed end-point to an application residing on a network management station. Communities define areas of management and are not a security password.

SNMPv1 has no security requirements for data encryption, end-point authentication, or for the prevention of data modification during transmission. There are provisions for defining "views" which are subsets of the available data objects and for defining the levels of access to management data objects as:

[10] "Simple Network Management Protocol (SNMP)" June 29 2003 URL: http://www.isi.edu/in-notes/rfc1157.txt
[11] "RFC Index" June 29 2003 URL: http://www.ietf.org/iesg/1rfc_index.txt
[12] "X.690 – ASN.1 Encoding Rules" June 30 2003 URL: http://www.itu.int/ITU-T/studygroups/com17/languages/X.690-0207.pdf
[13] "X.683 – OSI Networking and System Aspects – ASN.1" June 30 2003 URL: http://www.itu.int/ITU-T/studygroups/com17/languages/X.683-0207.pdf

- Read-only
- Read-write
- Write-only
- None

Although the specification provides for segregating data (via views) and controlling how and who accesses data (via communities) the whole application is inherently insecure. The lack of end-point authentication, data encryption, and data authentication are serious deficiencies of SNMPv1. In addition, the community feature of SNMP is not equivalent to a password, a common misconception, and does not provide end-point authentication even in a trusted network.

SNMP version 2 (SNMPv2), defined in RFC 3416[14] (status "Standard") makes some minor changes. The specification added the following operations to the 5 in SNMPv1:

- GetBulkRequest retrieve many data object values
- InformRequest notify that a data object value has changed
- SNMPv2-Trap SNMPv2 form of "Trap" or "Inform"
- Report added but not defined within the RFC

The SNMPv2 "InformRequest" operation replaces the SNMPv1 "Trap" operation.

With respect to security, SNMPv2 makes no changes or additions to SNMPv1. It recommends that implementers move to the user and view security mechanisms specified in SNMP version 3 (SNMPv3).

SNMPv3 introduces many security requirements to SNMP. RFC 3411[15] defines the architecture for SNMP frameworks and definitions for associated SNMP RFCs. This RFC specifically comments on the security deficiencies of versions 1 and 2 and defines the following threats that should be addressed by any SNMP implementation:

- Information modification changing datagrams in-flight
- Element masquerading spoofing sources or datagrams
- Message stream modification communication interference
- Disclosure eavesdropping

RFC 3414[16] (status "Standard") provides the specifications for user based security and specifically addresses all four of the above security deficiencies. The specification requires that an SNMPv3 implementation must support the "HMAC-MD5-96"[17] protocol should support the "HMAC-SHA-96"[18] protocol. This provides

[14] "Version 2 of the Protocol Operations for the SNMP" June 29 2003 URL: http://www.isi.edu/in-notes/rfc3416.txt
[15] "Architecture for Describing SNMP Frameworks" June 29 2003 URL: http://www.isi.edu/in-notes/rfc3411.txt
[16] "User-based security Model (USM) for version 3 of the SNMP" URL: http://www.isi.edu/in-notes/rfc3414.txt
[17] "HMAC-MD5-96" is a 96 bit key hashing (with MD5) protocol that ensures data integrity and data source authentication. See RFC-1321 for MD5 and RFC-2104 for HMAC from URL: http://community.roxen.com/developers/idocs/rfc/rfc2403.html

username plus secret key authentication of the communications to ensure both authentication of the source and data integrity.

SNMPv3 implementations must also use the "CBC-DES" Symmetric Encryption[19] protocol to ensure data privacy. Lastly, implementations must provide data fields used to ensure the prevention of tampering with the message stream. The encryption and hashing provided above protects these fields from modification.

The Structure of Management Information and Object Identifiers

The Structure of Management Information (SMI) is the set of rules associated with the definitions, characteristics, and names of management information. RFC 2578[20] defines the Structure of Management Information version 2 (SMIv2) and documents the following:

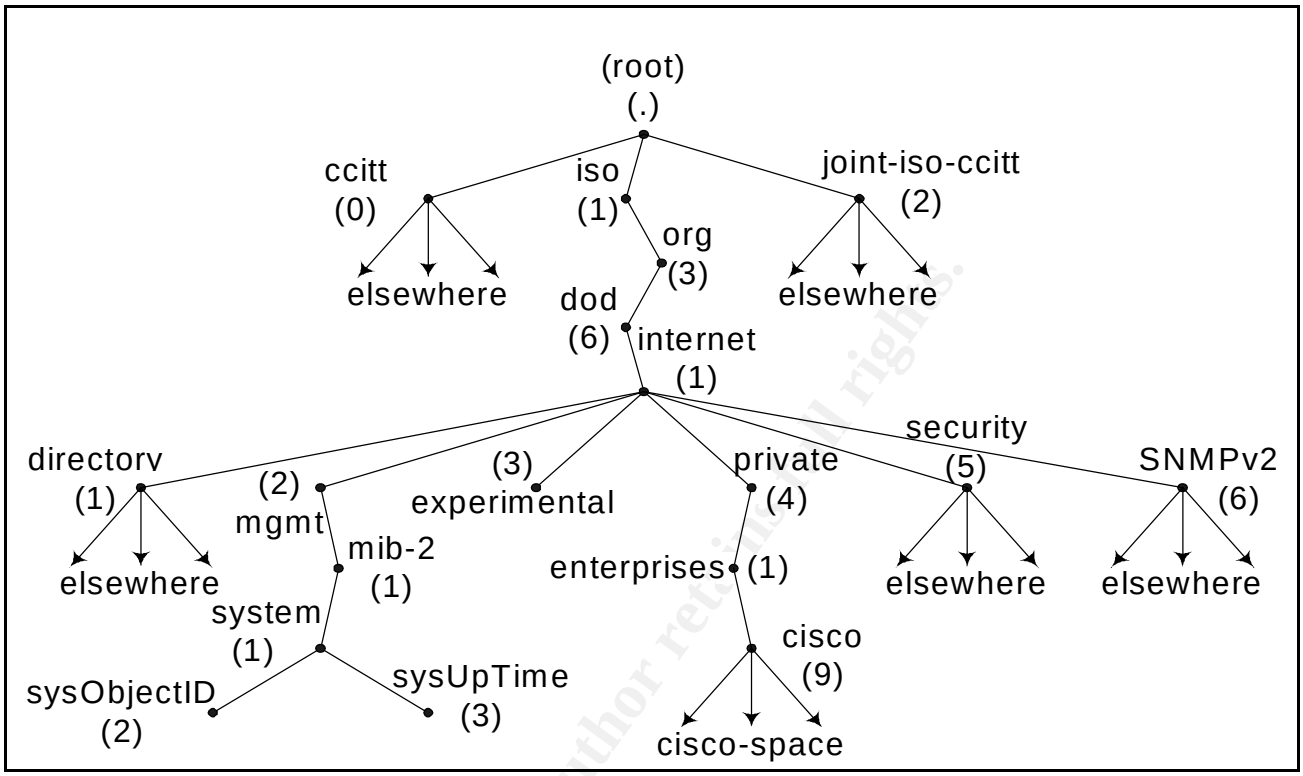
- A set of rules specifying how data objects are defined
- A set of data object types and their characteristics
- A basic set of data objects
- A naming structure that uniquely identifies every data object

The naming structure defined in the SMIv2 is the Object Identifier (OID). A unique OID labels every data object. The map of the OID space looks like an inverted tree and can be written either as a series of numbers or words. Each number or word in the OID is a node.

The purpose of the OID is to uniquely identify every piece of management data. The OID naming structure also provides a mechanism for locating any data object.

[18] "HMAC-SHA-96" is similar to "HMAC-MD5-96" but uses SHA rather than MD5, for SHA see URL: http://www.faqs.org/rfcs/rfc3174.html
[19] "CBC-DES Symmetric Encryption" provides data encryption see URL: http://www.zvon.org/tmRFC/RFC3414/Output/chapter8.html
[20] "Structure of management Information Version 2 (SMIv2)" July 4 2003, URL: http://www.isi.edu/in-notes/rfc2578.txt

The basic part of the OID tree is below:



• Figure 3" Partial View of OID Name Space

For example, the diagram contains an object called “sysObjectID” in the bottom left hand corner. It has an object identifier (OID) of “1.3.6.1.2.1.1.2” in numbers and “iso.org.dod.internet.mgmt.mib-2.system.sysObjectID” in words.

“Walking” the OID space locates the “sysObjectID” object. Starting at the top, the search initially traverses the “ccitt” branch of the tree by walking down each of its sub-branches. Since “sysObjectID” is not in the “ccitt” branch, it would then search down the “iso” branch. Eventually getting to the “internet” node where the search would traverse the “directory” branch and again fail to find “sysObjectID”. The search then moves on to the “mgmt” branch finally to “mib-2”, then “system”, then to “sysObjectID”. All higher nodes contained in the path to “sysObjectID” make up its OID.

The MIB and A MIB: an Example

Summarizing, the SNMP is the command set and communication stream between a management station and managed end-points; the SMI defines the format and naming structure of the management data objects.

The actual collection of data objects is the Management Information Base (MIB). Querying and manipulating this collection of data is how SNMP applications manage their end-points. An SNMP software agent resides on each managed end-point and communicates with the end-point via the MIB.

The definition of "the MIB" is the entire collection of management data. The data values of "the MIB" exist on all of the managed devices. The identification of a unique data element in "the MIB" is the combination of its OID and the end-point's address.

The term "a MIB" describes a specific collection of information pertinent to a particular type of end-point or function. For example, Cisco™ has 566 SNMPv1 MIBs and 568 SNMPv2 MIBs[21]. These MIBs define Cisco's specific entries in "the MIB".

All SNMP manageable entities must support a MIB called MIB-II[22]. RFC 1213, which has a status of "Standard", defines MIB-II. It is an update of MIB version 1 defined in RFC 1155. MIB-II imports the object identifier (OID) nodes "internet" (1.3.6.1) and "mgmt" (1.3.6.1.2) and defines the following next levels of OID nodes:

OID (Num)	OID (Name)	# of Next Levels
1.3.6.1.2.1	iso.org.dod.internet.mgmt.mib-2	11
1.3.6.1.2.1.1	iso.org.dod.internet.mgmt.mib-2.system	7
1.3.6.1.2.1.2	iso.org.dod.internet.mgmt.mib-2.interfaces	2
1.3.6.1.2.1.3	iso.org.dod.internet.mgmt.mib-2.at	1
1.3.6.1.2.1.4	iso.org.dod.internet.mgmt.mib-2.ip	23
1.3.6.1.2.1.5	iso.org.dod.internet.mgmt.mib-2.icmp	26
1.3.6.1.2.1.6	iso.org.dod.internet.mgmt.mib-2.tcp	15
1.3.6.1.2.1.7	iso.org.dod.internet.mgmt.mib-2.udp	5
1.3.6.1.2.1.8	iso.org.dod.internet.mgmt.mib-2.egp	6
1.3.6.1.2.1.9	iso.org.dod.internet.mgmt.mib-2.cmot	0
1.3.6.1.2.1.10	iso.org.dod.internet.mgmt.mib-2.transmission	0
1.3.6.1.2.1.11	iso.org.dod.internet.mgmt.mib-2.snmp	30

• Table 2: OID Nodes Defined in MIB-II

Some of these nodes, for example "system" (1.3.6.1.2.1.1), define more nodes and levels of the OID space defined in MIB-II. MIB-II defines the basic "network" like data objects associated with an end-point.

Vendors can request a unique "enterprise" (1.3.6.1.4.1) identifier. This provides a mechanism for satisfying their particular devices' management requirements. Enterprise identifiers extend the "private" (1.3.6.1.4) branch in the OID space and have an OID value of 1.3.6.1.4.1.xxx where xxx is the assigned vendor number. IANA assigns the number to a vendor; there are currently 17,630 enterprise identifier assignments[23].

[21] "MIBs downloaded and counted" June 23 2003: ftp.cisco.com/pub/mibs/v1 and ftp.cisco.com/pub/mibs/v2
[22] "RCF-1213 MIB-II" July 02 2003 URL: http://www.mg-soft.si/RFC1213-MIB.html
[23] "Enterprise Numbers" July 18 2003 URL: http://www.iana.org/assignments/enterprise-numbers

For example, Cisco™ has an enterprise identifier of nine, and 1.3.6.1.4.1.9 is Cisco's branch in the OID space. Cisco can create Cisco specific management data and functions that use this branch of the OID tree.

The Cisco™ MIB called "OLD-CISCO-SYSTEM-MIB.my"[24] contains the following statements:

```
***** Beginning of MIB left out for Simplicity *****
     IMPORTS
            local
                 FROM CISCO-SMI;
     lsystem             OBJECT IDENTIFIER ::= { local 1 }

***** Some information removed for simplicity *****

     writeNet OBJECT-TYPE
         SYNTAX  DisplayString
         ACCESS  write-only
         STATUS  mandatory
         DESCRIPTION
                 "Write configuration to host using TFTP."
         ::= { lsystem 55 }

***** End of MIB removed for simplicity *****
```

This MIB imports an object called "local" from the MIB called "Cisco-SMI MIB"[25]. The OID for "local" is "iso.org.dod.internet.private.enterprise.cisco.local" in words or "1.3.6.1.4.1.9.2" in numbers. Within "local" is "lsystem" (1.3.6.1.4.1.9.2.1) and within "lsystem" is "writeNet" (1.3.6.1.4.1.9.2.1.55).

Writing an IP address in the "writeNet" data object on a Cisco device will cause the device to tftp its configuration to that IP address. The SNMP application CiscoWorks™ manipulates this OID to backup Cisco device configurations. CiscoWorks™ uses the SNMP operation SetRequest to perform this task.

The 566 Cisco SNMPv1 MIBs that make up the Cisco MIB space contain over 8000 modifiable variables. Modifying any of these variables on a Cisco device can potentially change the operation of the device. This is the power of SNMP.

[24] "Cisco FTP Site" June 29 2003 URL: ftp.cisco.com\pub\mibs\v1\OLD-CISCO-SYSTEM-MIB.my
[25] "Cisco FTP Site" June 29 2003 URL: ftp.cisco.com\pub\mibs\v1\CISCO-SMI-MIB.my

An Analogy for SNMP, SMI, and MIB

The format of the following dictionary[26] entry probably looks familiar:

> aard vark \ ärd – värk \ *n* [obs. Afrik, fr. Afrik *aard* earth + *vark* pig] : a large burrowing nocturnal African mammal (*Orycteropus afer* of the order Tubulidentata) that has an extensile tongue, powerful claws, large ears, and heavy tail and feeds on ants and termites

• Figure 4: Dictionary Definition Analogy

The rules used to create the dictionary entry are analogous to the rules defined in the Structure of Management Information (SMI). For example, the italicized "*n*" means that the word is a noun. All entries in the dictionary with an italicized "*n*" are also nouns.

In this analogy, the entire dictionary is equivalent to the MIB. The entry or word (aardvark) is analogous to the object identifier (OID). The alphabetic order that the words have in the dictionary provides the mechanism for locating a particular word.

Each node in the dictionary entry is a letter in the word and spelling the word locates it in the dictionary. This is analogous to traversing the OID space to get to a particular data object entry.

The definition of the word contains the data values for the entry. The analogy to the communication flows of SNMP would be how the user reads or modifies the dictionary.

The only component missing from the analogy is the "trap" operation. Unlike SNMP managed end-points, dictionaries tend not to wake up and tell you when an entry for a word has changed.

Basic Working of SNMP and an Example

An SNMP manager initiates three of the five SNMPv1 operations:

- GetRequest
- GetNextRequest
- SetRequest

The managed SNMP agent responds to these three with a single operation type:

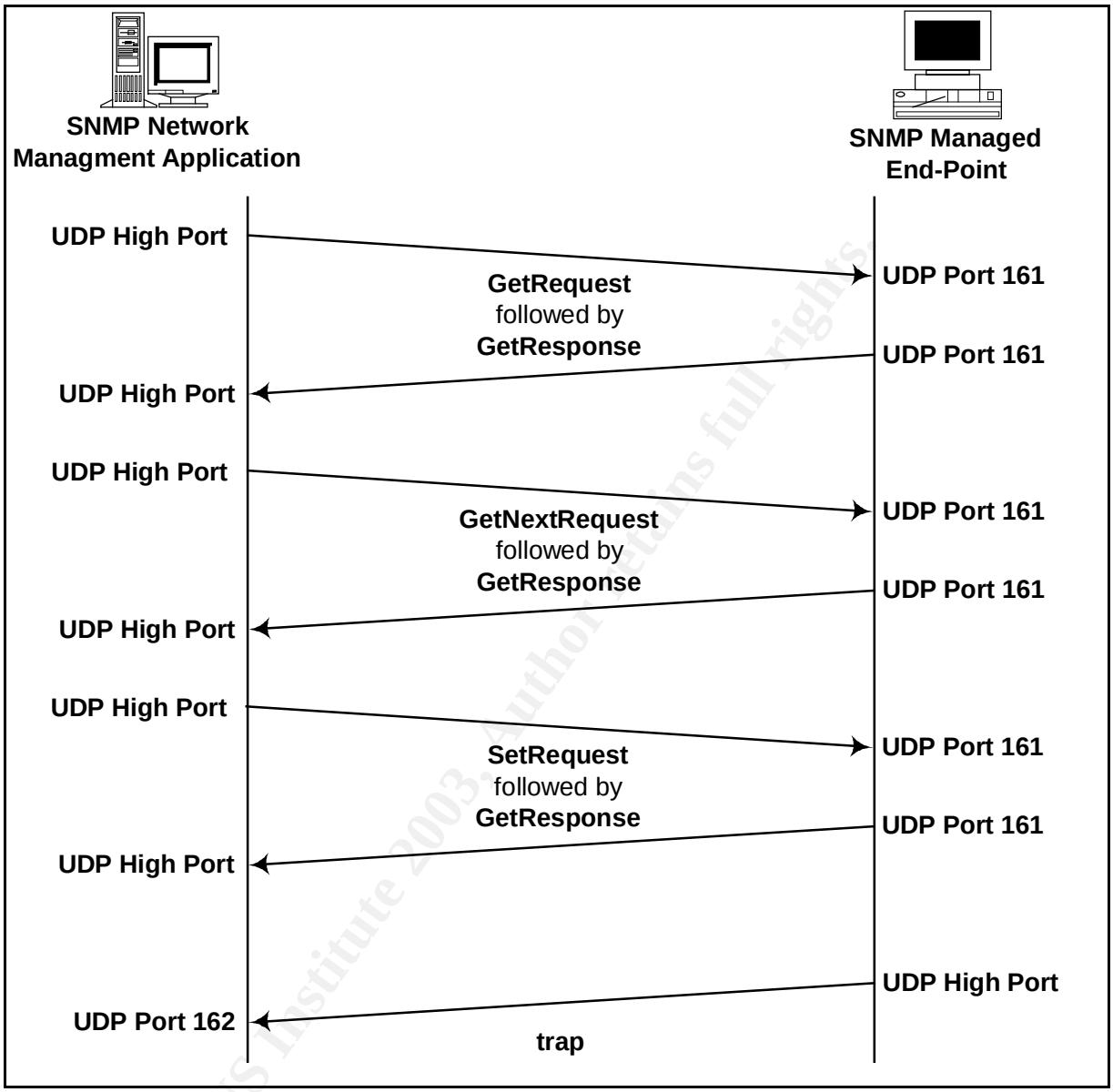
- GetResponse

The SNMP agent initiates the last SNMPv1 operation type and there is no response:

- Trap

[26] "Webster's New Collegiate Dictionary": by G. & C. Merriam Co.

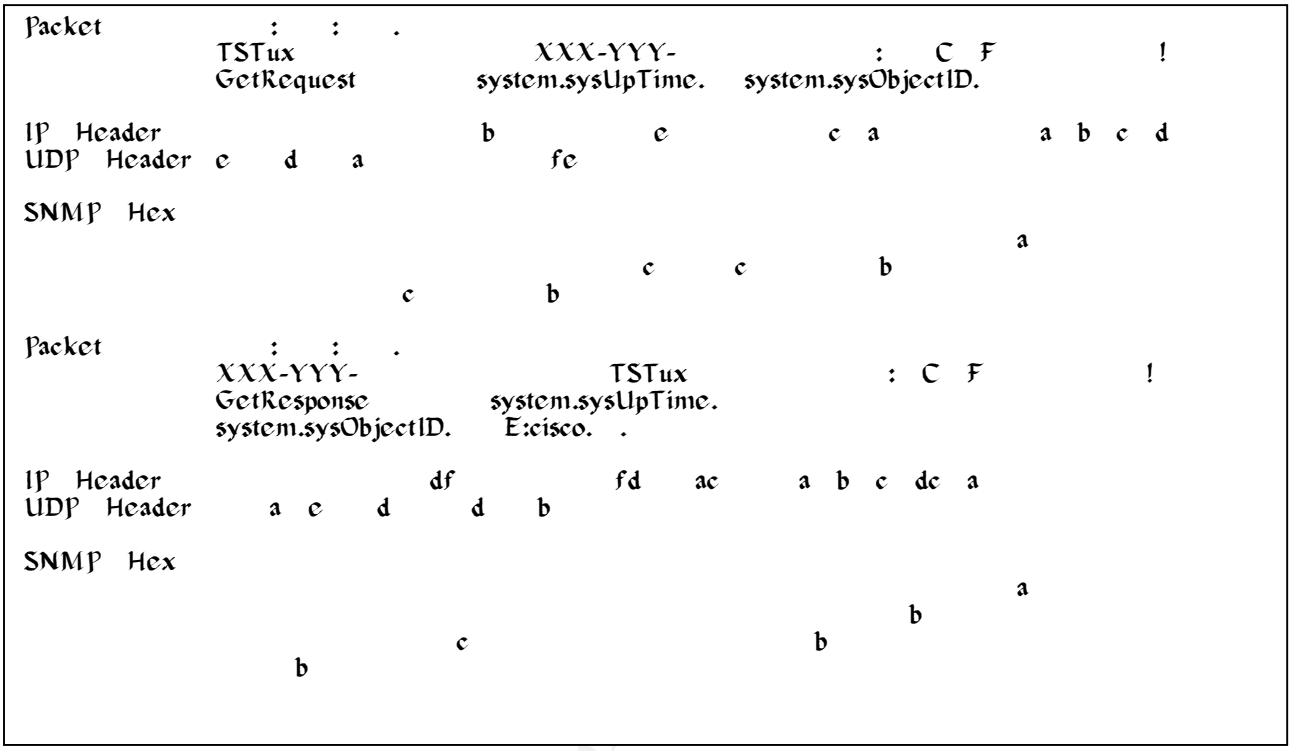
In diagram form:



• Figure 5: SNMP Communications Flows

A "tcpdump -l -s -x host XXX-YYY- " command run on a CiscoWorks™ system captured the following two SNMP operations. A perl script reformatted the captured output to produce the following report.

[27] "tcpdump" command run on an IBM AIX™ version 4.3 server



• Figure 6: SNMP Packet Dumps

The first packet is an SNMP packet travelling from server TSTux01 on port 59789 to device XXX-YYY-1 on port 161. The SNMP operation is a "GetRequest" operation requesting two variables "system.sysUpTime" and "system.sysObjectID". The return packet, from XXX-YYY-1 port 161 to TSTux01 port 59789, is a "GetResponse" operation and contains the values of the requested variable.

The hexadecimal output of the SNMP datagram is a collection of information triplets. These nested triplets have the following form and make up the SNMP datagram:

SNMP Tag	Length	Value

Nesting occurs when the "Value" portion of a parent triplet contains more "Tag-Length-Value" triplets. The following SNMP data types tags identify the various SNMP data types for versions 1 and 2[28]:

SNMP Tag Value	SNMP Data Type (version 1/version 2)
0x02	INTEGER/Integer32
0x04	OCTET STRING
0x05	NULL
0x06	OBJECT IDENTIFIER
0x30	SEQUENCE (of more triplets)
0x40	IpAddress
0x41	Counter/Counter32

[28] "Understanding SNMP MIBs" David Perkins Evan McGinnis – Prentice Hall 1997 ISBN 0-13-437708-7

0x42	Gauge/Gauge32
0x43	TimeTicks
0x44	Opaque
0x46	na / Counter64
0xA0	GetRequest
0xA1	GetNextRequest
0xA2	GetResponse
0xA3	SetRequest
0xA4	Trap

• Table 3: SNMP Tag Values

The following table decodes the first captured packet:

Line#	SNMP Data Type	Field Length	Value & Length	SNMP Datagram Hexadecimal Data
1	Sequence	1		30
2	Length	1	0x36 (54) Bytes	36
3	Integer	1		Θ;;;;;;;;;;;;;;;202
4	Length	1	1 Byte	: 01
5	Value	1	Version (SNMPv1)	: 00
6	Octet String	1		: 04
7	Length	1	8 Bytes	: 08
8	Value	8	F123456!	: 4631323334353621
9	Get-request	1	A0	: A0
10	Length	1	0x27 (39) Bytes	: 27
11	Integer	1		: Θ;;;;;;;;;;;;;;;202
12	Length	1	1 Byte	: : 01
13	Value	1	Message#	: : 49
14	Integer	1		0 : 02
15	Length	1	1 Byte	x : 01
16	Value	1	Error-Status	3 : 00
17	Integer	1		6 : 02
18	Length	1	1 Byte	: 01
19	Value	1	Error-Index	B : 00
20	Sequence	1		y 0 30
21	Length	1	0x1c (28) Bytes	t x 1c
22	Sequence	1		e 2 Θ;;;;;;;;;;;;;230
23	Length	1	0x0c (12) Bytes	s 7 : 0c
24	OID	1		: : Θ;;;;;;;;;;;206
25	Length	1	8 Bytes	: B 0 : 08
26	Value	8	1.3.6.1.2.1.1.3.0	: y x 0x0c Bytes 2B06010201010300
27	Null	1		: t 1 : 05
28	Length	1	0 Bytes	: e c A;;;;;;;;;;;;;;;00!
29	Sequence	1		: s 30
30	Length	1	0x0c (12) Bytes	: : B 0c
31	OID	1		: : y Θ;;;;;;;;;;;206
32	Length	1	8 Bytes	: : t : 08
33	Value	8	1.3.6.1.2.1.1.2.0	: : e 0x0c Bytes 2B06010201010200
34	Null	1		: : s : 05
35	Length	1	0 Bytes	A;;;;4;;;;4;;;;4;;;;;;;;;;;;;;;00!

• Table 4: SNMP "GetRequest" Packet

The SNMP tag in line 24 is 0x06. It identifies an OID triplet. Line 25 specifies an 8-byte length for the OID. The value of this OID (2B06010201010300) is located in

lines 26. The first 2 levels of all OIDs can be only a few possible values. The following formula[29] translates these values into a hexadecimal value:

$$.i.j \Rightarrow 40*i + j$$

Thus the left most ".1.3" of an OID becomes (40*1 + 3) or 43 (or 0x2B).

The OID specified in line 26 (2B06010201010300) decodes to "1.3.6.1.2.1.1.3.0" in numbers or ".iso.org.dod.internet.mgmt.mib-2.system.sysUpTime.0" in words. The trailing ".0" is a demarcation specifying the end of the OID and that a single specific data value should be returned.

The OID specified in line 33 (2B06010201010200) becomes "1.3.6.1.2.1.1.2.0" in numbers or ".iso.org.dod.internet.mgmt.mib-2.system.sysObjectID.0" in words.

The entire SNMP datagram decodes into the following triplets:

SNMP Datagram Triplet Information	Nesting Level	Tag/Type	Length	Value or Location
First SNMP Datagram Sequence Triplet	0	30/Sequence	0x36	Lines 3 to 35
SNMP version	1	02/Integer	0x01	0x00 (SNMPv1)
Read community string	1	04/Octets	0x08	F123456!
GetRequest	1	A0/GetRequest	0x27	Lines 11 to 35
Message number	2	02/Integer	0x01	0x49
Error status	2	02/Integer	0x01	No Error
Error index	2	02/Integer	0x01	No Error
List of requested variables	2	30/Sequence	0x1C	Lines 22 to 35
First sequence of requests	3	30/Sequence	0x0C	Lines 24 to 28
OID (system.sysUpTime.0)	4	06/OID	0x08	1.3.6.1.2.1.1.3.0
Null padding	4	05/Null	0x00	
Second sequence of requests	3	30/Sequence	0x0C	Lines 31 to 35
OID (system.sysObjectId.0)	4	06/OID	0x08	1.3.6.1.2.1.1.2.0
Null padding	4	05/Null	0x00	

• Table 5: SNMP GetRequest Triplet Information

The first SNMP triplet is a sequence (0x30) of nested triplets. Within this triplet are the SNMP version, the read community string, and the GetRequest triplets. Nested within the GetRequest triplet are the message number, error status, error index, and list of requested variables triplets.

Within the list of requested variables triplet are the first sequence of requests and the second sequence of requests. Within both of these are the requested OIDs padded with a null triplet with 0 length.

[29] "Object Identifier Encoding" Appendix A page 394 of Perkins' and McGinnis': Understanding SNMP MIBS; Prentice Hall, 1997

The following table decodes the second captured packet:

Line#	SNMP Data Type	Field Length	Value & Length	SNMP Datagram Hexadecimal Data
1	Sequence	1		30
2	Length	1	0x43 (67) Bytes	43
3	Integer	1		Θ;;;;;;;;;;;;;;;202
4	Length	1	1 Byte	: 01
5	Value	1	Version (SNMPv1)	: 00
6	Octet String	1		: 04
7	Length	1	8 Bytes	: 08
8	Value	8	F123456!	: 4631323334353621
9	Get-response	1		: A2
10	Length	1	0x34 (52) Bytes	: 34
11	Integer	1		: Θ;;;;;;;;;;;;;;;202
12	Length	1	1 Byte	: : 01
13	Value	1	Message#	: : 49
14	Integer	1		**0** : 02
15	Length	1	1 Byte	**x** : 01
16	Value	1	Error-Status	**4** : 00
17	Integer	1		**3** : 02
18	Length	1	1 Byte	: 01
19	Value	1	Error-Index	**B** : 00
20	Sequence	1		**y** **0** 30
21	Length	1	0x29 (41) Bytes	**t** **x** 29
22	Sequence	1		**e** **3** Θ;;;;;;;;;;;;;230
23	Length	1	0x10 (16) Bytes	**s** **4** : 10
24	OID	1		: : Θ;;;;;;;;;;;;206
25	Length	1	8 Bytes	: **B** : : 08
26	Value	8	1.3.6.1.2.1.1.3.0	: **y** **0** **0x10 Bytes** 2B06010201010300
27	TimeTicks	1		: **t** **x** : 43
28	Length	1	4 Bytes	: **e** **2** : 04
29	Value	4	942,432,309	: **s** **9** A;;;;;;;;;;;;;;;;382C6035!
30	Sequence	1		: : 30
31	Length	1	0x15 (21) Bytes	: : **B** 15
32	OID	1		: : **y** Θ;;;;;;;;;;;;06
33	Length	1	8 Bytes	: : **t** : 08
34	Value	8	1.3.6.1.2.1.1.2.0	: : **e** **0x15 Bytes** 2B06010201010200
35	OID	1		: : **s** : 06
36	Length	1	9 Bytes	: : : : 09
37	Value	9	1.3.6.1.4.1.9.1.217	A;;;;4;;;;4;;;;4;;;;;;;;;;;;;;;;2B0601040109018159!

• Table 6: SNMP GetRespose Packet

The return packet in the GetResponse operation contains the OIDs from the preceding GetRequest followed by its value.

Lines 26 and 34 are the OIDs specified in the preceding GetRequest operation. The values returned with these OIDs are “0x38c26035” (line 29) and 2B0601040109018159 (line 37). As specified by line 27 the “0x38c26035” is a TimeTicks value and the value “2B0601040109018159” denotes an OID as specified in line 35. This OID requires more information to translate.

Using the formula above (40*i + j), the first part “2B060104010901” translates into “1.3.6.1.4.1.9.1”. The trailing “0x8159” requires more explanation.

The ASN.1 standard uses only the seven least significant bits for data. The eighth most significant bit is a continuation identifier. When this bit is set, the least significant seven bits of the next byte are included as part of the data.

Hence:

```
From "0x8159":
81 => 10000001(binary) has most significant bit turned on
indicating a continuation
59 => 01011001(binary) is the expected continuation


The 7 least significant (those representing data in ASN.1) bits
of each byte are concatenated into a single binary string and
converted to hexadecimal and then decimal


"0000001" concatenated to "1011001" => 11011001 => 0xD9 => 217
```

• Figure 7: Decoding OID Triplets with Continuations

Thus the OID "2B0601040109018159" becomes "1.3.6.1.4.1.9.1.217" in numbers or "iso.org.dod.internet.private.enterprise.cisco.products.217" in words, where "217" relates to a specific Cisco device type.

Protocols Used

SNMP may use some of the following protocols as specified in the associated RFCs:

- RFC 1089 SNMP directly over Ethernet
- RFC 1461 SNMP directly over X.25
- RFC1420 SNMP over IPX (sockets 36879 and 36880)
- RFC 1419 SNMP over Appletalk/DDP
- RFC 1592 SNMP over TCP for IBM MIB extensions

In some of these protocols, the SNMP data representation is changed. For example, SNMP tag type "0x40" represents an IP address. This makes no sense when running SNMP directly over Ethernet, which has no IP address.

Generally, SNMP runs over UDP/IP. Where User Datagram Protocol (UDP) is a connectionless protocol at the transport layer and Internet Protocol (IP) is the protocol at the network layer. Neither protocol provides delivery reliability so there is no network mechanism that will verify that datagrams reached their target.

The UDP header contains only 4 pieces of information:

- Source port
- Destination port

- Datagram length
- Datagram checksum

The UDP header has no fields for preventing spoofing or forging datagrams. Unlike TCP, which has sequence numbers, acknowledgement numbers, and a three-way handshake mechanism that make spoofing difficult, spoofing a UDP datagram is essentially trivial.

SNMP durability is the vendor's responsibility for completeness and reliability. SNMP over UDP/IP is essentially a "fire and forget" communication stream.

The following table shows a representation of the layers in the TCP/IP protocol suite generally associated with SNMP.

Layer	Protocol	Comments
Application Layer	SNMP	Vendor implemented capabilities
Transport Layer	UDP	Connectionless and unreliable
Network Layer	IP	Routing packets only
Link Layer	Ethernet	MAC

Vulnerabilities of the SNMP

The vulnerabilities of SNMP fall into two categories:

- inherent vulnerabilities
- vendor's implementation vulnerabilities

Type 1: Vulnerabilities Inherent to SNMP Versions 1 and 2

Vendors provide legitimate management functions using SNMP and many vendors provide SNMP turned on by default with well-known community names. The security problems inherent to versions 1 and 2 of SNMP provide the perfect basis for exposing these functions to unauthorized disclosure or use.

SNMP versions 1 and 2 have the following four security issues:

- no end-point authentication or validation
- no protection against the modification of SNMP datagrams
- no protection of the flow of datagrams sent between end-points
- no protection against viewing the SNMP data directly from datagrams

In SNMP, versions 1 and 2, datagrams transit the network in clear text. A sniffer watching traffic going between an SNMP management station and a managed end-point can retrieve all SNMP data as it passes. This disclosure includes the read (and write) community words contained within the datagrams.

A properly crafted packet, spoofed from an authorized SNMP management station, containing an appropriately defined "setRequest" command can be used to cause managed end-points to execute unauthorized tasks.

The following table entries identify a small set of variables that identify legitimate management functions on various vendor platforms. In the examples, "X" translates into "iso.org.dod.internet":

Vendor	Cisco
Source MIB	CISCO-C6200-MIB-V1SMI.my[30]
Function	Reboot Cisco 6200 device
OID by Number	1.3.6.1.4.1.9.10.26.1.4
OID by Name	X.private.enterprises.cisco.ciscoExperiment.cisco6200MibObjects.c62System.systemReset

Vendor	Cisco
Source MIB	BGP4-MIB-V1SMI.my
Function	Stop a BGP peer connection
OID by Number	1.3.6.1.2.1.15.3.1.3
OID by Name	X.mgmt.mib-2.bgp.bgpPeerTable.bgpPeerEntry.bgpPeerAdminStatus

Vendor	Microsoft (LANMAN)
Source MIB	LNMAN2.MIB[31]
Function	Set idle session timeout value
OID by Number	1.3.6.1.4.1.77.1.2.22
OID by Name	X.private.enterprises.lanmanager.manmgr2.server.svDisConTime

Vendor	Microsoft
Source MIB	WINS.MIB
Function	Delete all data records pertaining to a particular WINS IP entry.
OID by Number	1.3.6.1.4.1.311.1.2.5.9
OID by Name	X.private.enterprises.Microsoft.software.wins.cmd.cmdDeleteDbRcds

Interpreting these functions:

- the 1st forces a network device to reboot
- the 2nd interferes with a network routing protocol
- the 3rd can cause sessions to terminate after 0 seconds of idle time
- the 4th modifies the Microsoft® WINS database

The above examples describe some of the functionality that vendors provide via SNMP. Since SNMP (versions 1 and 2) is inherently insecure then these functions

[30] "Cisco MIBs Download Site" July 3 2003 URL: ftp.cisco.com\pub\mibs\v1
[31] "Microsoft MIBs Download" July 10 2003 URL: http://www.wtcs.org/snmp4tpc/snmp4tpc.htm

are not secure. Recall that there are over 8000 modifiable variables in Cisco's MIB space alone.

Another area of vulnerability with SNMP versions 1 and 2 is the ability to intercept and alter SNMP datagrams travelling between management station and end-point (or vice versa). Due to SNMP's inability to verify data authenticity, neither end would be aware of the altered datagram.

Also, it is possible to delay, restrict, re-order or prevent the transmission of datagrams and due to SNMP's inability to protect against communications flow interference neither the management station nor managed end-point would be aware of the change.

Type 2: Vulnerable Vendor Implementations

As with any software, vendors' implementations are prone to the standard set of coding vulnerabilities. The "PROTOS" project[32] (detailed below) determined the robustness of several vendor implementations of SNMPv1. The results of this targeted testing can be found at:

http://www.kb.cert.org/vuls/id/854306

A tabulated summary of the results follows:

Status	Vendor Count	Percent of Total Tested
Not Vulnerable	43	43/286 = 15%
Vulnerable	93	93/286 = 33%
Unknown	150	150/286 = 52%

• Table 7: SNMPv1 Robustness Summary

As this table highlights, only 15% of tested vendors passed the PROTOS tests. Thirty three percent had specific identifiable failures ranging from SNMP agent failures, to uncontrolled consumption of resources (predominantly memory leaks), to buffer over-flows. The 52% categorized as unknowns are a little misleading. Within these tests, a problem occurred with the vendor's SNMP implementation but failure details were not identifiable.

CVEs and Candidates for CVEs

The Mitre Corporation[33] manages and publishes a database of vulnerabilities for computing equipment and software. Two forms of vulnerabilities exist in the Mitre database:

- Common Vulnerabilities and Exposures (CVE®)
- Candidate Vulnerabilities (CAN)

The following tables are a brief survey of the current list of CVEs and CANs that exist and involve SNMP.

[32] "PROTOS – Security Testing of Protocol Implementations" July 10 2004 URL: http://www.ee.oulu.fi/research/ouspg/protos/
[33] "MITRE" July 20 2003 URL: www.mitre.org

The following CVEs relating to SNMP are from http://cve.mitre.org/cve/downloads[34]:

CVE Number	Summary Description
CVE-1999-0294	delete all WINS database entries
CVE-1999-0472	default community word “public” cannot be removed
CVE-1999-0815	memory leak in Windows NT 4.0
CVE-1999-0888	escalating privileged in Oracle
CVE-1999-1335	sensitive information disclosure in Red Hat Linux 4.0
CVE-2000-0221	DOS on Nautica Marlin bridges
CVE-2000-0379	cannot disable SNMP-write capabilities on Netopia router
CVE-2000-0515	escalated privilege in HP-UX
CVE-2000-1058	DOS or privilege escalation in HP OpenView
CVE-2001-0236	execute arbitrary code in Solaris snmpXdmid
CVE-2001-0487	DOS on AIX snmp agent
CVE-2001-0514	information disclosure, DOS, gain access on Atme1 802.11b WAP
CVE-2001-0564	DOS on American Power Corporation (APC) management card
CVE-2001-0888	DOS on Atme1 firmware version 1.3 Wireless Access Point
CVE-2001-0998	DOS and SNMP agent failure on IBM HACMP
CVE-2002-0017	arbitrary code execution on SGI IRIX
CVE-2002-0069	memory leak on Squid 2.4
CVE-2002-0302	loss of information on Symantec Enterprise Firewall

• Table 8: CVE Summary

From http://cve.mitre.org/cve/candidates/downloads-can.html[35] the following CANs

Candidate #	Description
CAN-2002-0012	describes PROTOS results for SNMPTRAP
CAN-2002-0013	describes PROTOS results for SNMPGET
CAN-2002-0053	buffer overflows in Windows Systems
CAN-2002-0109	information disclosure on Linksys Etherfast routers
CAN-2002-0305	inability to disable SNMP on ZOT printers
CAN-2002-0478	arbitrary SNMP data modification on Foundry Networks EdgeIron 4802F
CAN-2002-0540	disclosure and modification of sensitive data on Nortel CVX 1800 allows
CAN-2002-0796	escalated privilege on Solaris 5.6 through 8
CAN-2002-0797	escalated privilege on Solaris 5.6 through 8
CAN-2002-0812	sensitive information disclosure on Compaq wireless access point
CAN-2002-0858	escalated privilege in Oracle 8i an 9i

[34] Downloaded and summarized on July 10, 2003.
[35] Down loaded and summarized on July 10, 2003.

CAN-2002-1048	sensitive information disclosure on HP JetDirect printers
CAN-2002-1170	DOS on Net-SNMP package
CAN-2002-1408	sensitive information disclosure on HP OpenView
CAN-2002-1426	DOS on HP ProCurve switch 4000M C.07.03
CAN-2002-1448	privilege escalation in Avaya Cajun products
CAN-2002-1555	sensitive information disclosure on Cisco ONS15454 and ONS15327
CAN-2003-0137	sensitive information disclosure on DX200 based networks for Nokia

- Table 9: Vulnerability Candidate Description

The summaries above refer to many vendors and many products. Vendors have responded to most of these vulnerabilities and candidates by creating patches or mitigation strategies.

Part 2 – Specific Exploit

Exploit Details

The following vulnerability description is at: http://www.securityfocus.com/bid/6030/info/. Vulnerable systems include all versions of Windows 2000 Server, Advanced Server, Datacenter Server, Terminal Services, and Professional running at base install, service pack 1 or service pack 2. There are some other requirements for the exploit to be effective described below.

There is no CVE defined for this problem.

Any SNMP management tool (or software) can exploit this vulnerability. The impact of this exploit is a denial of service against the victim system. The results of this impact depend entirely on the roll the victim machine plays in an environment.

Microsoft was notified in November of 2001 and addressed the problem is service pack 3 as described at: http://support.microsoft.com/default.aspx?scid=kb;en-us;Q296815.

Description of Variants

Using standard SNMP operations, in an appropriate manner, perform this exploit. Any SNMP tool or utility can cause this problem and each would have its own command syntax. The following products and commands are examples:

Vendor	Tool	Command
NGS Software	snmplib	snmplib get(next) 161 <community> 1.3.6.1.4.1.77.1.2.28.0
IBM	NetView	snmpget –c <community> hostname|IP_address 1.3.6.1.4.1.77.1.2.28.0
Microsoft	SNMPUtil	snmputil get(next) hostname|IP_address <community> 1.3.6.1.4.1.77.1.2.28.0
Perl	SNMP::Util	snmpget hostname|IP_address <community> 1.3.6.1.4.1.77.1.2.28.0 snmpwalk hostname|IP_address <community> 1.3.6.1.4.1.77.1.2.28.0

• Table 10: SNMP Management Products

Protocol Description

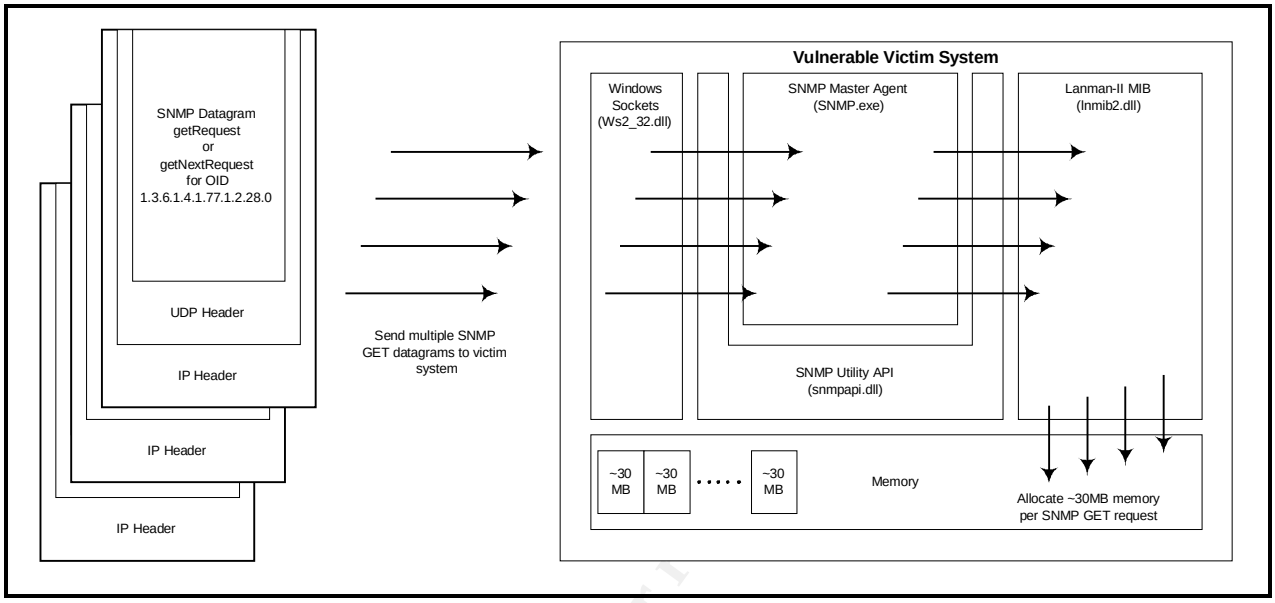
A description of the SNMP is located in part 1 of this report.

How the Exploit Works

The vulnerability is only exploitable on systems running the SNMP service and not running the print spooler service. When an SNMP management product makes a request for print queue information the lnmib-II.dll program allocates memory (as much as 30 MB) and does not release it. If requested repeatedly the problematic dll on the victim system will consume all available memory and the system will halt. Rebooting the system will free the memory resources.

Diagram

The following diagram shows how to exploit this particular SNMP vulnerability.



• Figure 8: Exploiting the SNMP Vulnerability[36]

1) Send multiple SNMP datagrams requesting object identifier 1.3.6.1.4.1.77.1.2.28.0 to a vulnerable host (target port UDP 161). Since SNMP works over UDP/IP there are no expected responses or session establishment packets (as you would expect with TCP/IP) and therefore a spoofed source can be used.
2) The victim (target) system receives the packets and the Windows Sockets program passes the SNMP datagram (via the Windows SNMP API Utility used between all SNMP agents on Windows 2000) to the SNMP master agent (SNMP.exe).
3) The SNMP master agent parses the request (recognizing the subagent that handles requests for 1.3.6.1.4.1.77.1.2.28.0) and passes it through to the Lanman-II MIB program (lnman2.dll) (also via the SNMP API utility).
4) The SNMP subagent (lnman2.dll) allocates approximately 30 megabytes of memory for each SNMP GET request.
5) A large number of GET requests will consume all available memory on the victim machine and the machine will experience a performance degradation due to lack of available memory.

[36] "Microsoft Windows 2000 SNMP Documentation" August 5th, 2003 URL: http://www.microsoft.com/technet/treeview/default.asp?url=/technet/prodtechnol/windows2000serv/reskit/tcpip/part3/tcpch10.asp

How to Use the Exploit

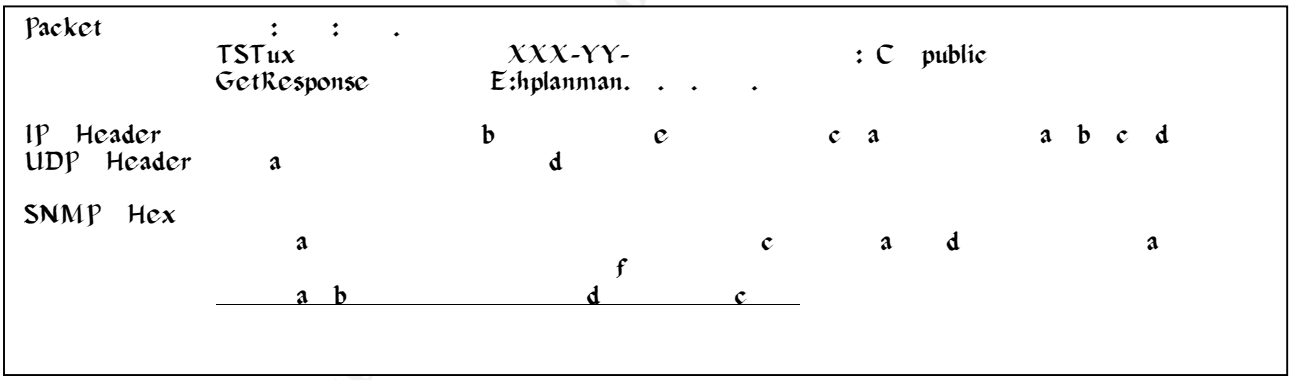
Using any SNMP management software or utility, a brief list is tabled above, target many SNMP "getRequest" or "getNextRequest" at a vulnerable host requesting the print queue information in the LANMAN-II OID 1.3.6.1.4.1.77.1.2.28.0 (in numbers) or iso.org.dod.internet.private.enterprises.lanManager.lanmgr_2.server.svPrintQNumber.0 (in words).

A second mechanism for exploiting the vulnerability is to create a properly crafted SNMP datagram and repeatedly direct this at a vulnerable host using UDP port 161. The program netcat[37] can send the datagrams to the victim.

Signature of the Attack

Network Signature

A network based identification method is to watch for and create an alert on a SNMP datagram containing the OID "1.3.6.1.4.1.77.1.2.28.0". A "getResponse" packets with these characteristics is shown below (packet collected with "tcpdump" and reformatted with a "perl" program):



• Figure 9: Captured SNMP Datagram with Specified OID

Within the SNMP_Hex portion of the captured SNMP packet shown above is the SNMP triplet "060a2b060104014d01021c00". This triplet translates as:

- Type "06" or Object Identifier
- Length 0a or 10 bytes
- Value "2b060104014d01021c00" or "1.3.6.1.4.1.77.1.2.28.0"

This is the "getResponse" packet returning from a "getRequest" or "getNextRequest" datagram.

[37] "netcat" can be found at: http://www.atstake.com/research/tools/network_utilities/

Target Host Signature

It is possible to observe the effects of the attack with the Windows 2000 task manager. By using the "View", then "Select Columns", and then menu options, the "VM Size" column can be added to the display.

This column will show an increase in memory consumption for the Windows master SNMP agent process (SNMP.EXE) each time the SNMP action occurs. Each SNMP "get" attack consumes approximately 30 Megabytes of memory.

As the attack continues, the victim system's performance will degrade. A shortage of available memory will prevent new processes from starting. Detailed information about this vulnerability and its effects are at:

http://support.microsoft.com/default.aspx?scid=kb;en-us;Q296815

Protection from the Exploit

General Strategies for Mitigating SNMP Vulnerabilities

Turn it off – if SNMP is not required disable it. By default, many vendors provide SNMP turned on using well-known community words. A scan of 8 Class "C" networks (managed by the author) resulted in approximately 100 devices running SNMPv1, using "public" and "private" community words. Foundstone®[38] provides a scanner called SNSCAN[39] which can be used to quickly determine the number of systems running SNMP.

Upgrade it – where possible upgrade to SNMP version 3. SNMPv3 eliminates the inherent vulnerabilities found in versions 1 and 2.

Isolate it – if SNMP is required for management then restrict it to a separate "out of band" management network. Often, this option is prohibitive but provides an excellent method for putting SNMP into a more restricted network.

Block it – if SNMP traffic should not cross a network control point then prevent it. SNMP should not transit a network perimeter.

Alter it – never rely on the vendor's default implementation of blank or universally known community words.

Control it – most SNMP managed devices provide the ability to specify access control lists that limit SNMP access to authorized devices. These access controls use IP addresses and can be of limited value since the SNMP/UDP/IP datagram is so easily spoofed.

[38] www.foundstone.com (July 29 2003)
[39] "Scanning Tools" July 29 2003 URL: www.foundstone.com/index.htl?subnav=resources/navigation.htm&subcontent=/resources/proddesc/snscan.htm

Specific SNMP Problem Mitigation Strategies for Windows 2000

Microsoft Windows 2000 systems are vulnerable to this exploit. Service Pack 3 fixes this vulnerability so the most effective mechanism for protecting a system from this specific exploit is to patch the system.

If a system is not patchable, then disabling SNMP on the machine is the next most effective mitigation strategy. A web search on "Windows 2000 removing SNMP" returns many documentation references that describe the process. The basic process is to use the "Add/Remove" programs option in the "Control Panel" followed by the "Add/Remove Windows Components". Choose "Management and Monitoring Software", click details, deselect SNMP, and click next. The following gives a much more detailed and graphic description of the process (the site also has a decent Windows 2000 security checklist): http://www.lokbox.net/SecureWin2k/snmp.asp.

If SNMP is required in your un-patchable Windows 2000 environment then altering the default installation and controlling access to it can have some mitigation effect. A web search for "Windows 2000 configure SNMP" produces many good document references. This is of limited value and may even result in a false sense of security. Recall that the SNMP (versions 1 and 2) protocols are inherently insecure. Sniffing network traffic will produce both non-standard community words and the IP address of "appropriately defined" SNMP managers. Spoofing an SNMP request is a trivial task.

Source Code/ Pseudo Code

Standard SNMP management tools can exploit the vulnerability using standard and appropriate SNMP requests. Source code or pseudo code are from those tools. A description of various vendors and products are in the "Variants" section above.

Additional Information

Impact of the Exploit

This exploit essentially performs a denial of service attack on the function provided by the victim. As the attack continues, the victim machine will essentially halt as available memory is exhausted. Specific impact from an end-user's perspective is obviously dependent on the function of the victim machine.

Alternate Sources of Information

The following sites all describe the vulnerability and most suggest the exploit mechanism via any standard SNMP management tool:

- http://www.securiteam.com/windowsntfocus/6S00M0K5SM.html
- http://xforce.iss.net/xforce/xfdb/10431
- http://www.kb.cert.org/vuls/id/887393
- http://www.nextgenss.com/advisories/snmp_dos.txt

The PROTOS Project

Though not specifically related to the Microsoft Windows 2000 vulnerability the PROTOS project identified dozens of vendors with vulnerable SNMP implementations on hundreds of product. The information contained below is the reason for defining the general SNMP vulnerability mitigation strategies.

The PROTOS project[40] ran at the Department of Electrical and Information Engineering at the University of Oulu in Finland. This project tested various vendor implementations of protocols via a test suite/tool developed for the project. The SNMPv1 implementations of many vendors were verified with a specific set of test data[41] and the results were tabulated and presented.

The SNMP testing project was broken into 2 main groups:

- GetRequest, GetNextRequest, and SetRequest tests
- Trap tests

The first group tests SNMP agents the second group test SNMP management software. Within each of these 2 main groups 2 test streams were defined:

- Application tests
- Basic Encoding Rules (ASN.1/BER) tests

The first test stream directly tested the vendor implementations by coding the following into the SNMP datagrams:

- Bit pattern errors
- Format string errors
- Integer value errors
- Missing symbol errors
- Overflow errors

The second stream tested the BER encoder/decoder that "front-ends" each SNMP implementation by deliberately using SNMP datagrams with BER errors.

Over 18,000 test PDUs were used against SNMP agents and more than 15,000 test PDUs were used against SNMP management software implementations.

The test results for each particular implementation were defined as follows:

- **Failed**: a vendors' SNMP implementation failed and a particular datagram can be identified as the cause
- **Failed Unconditionally**: a vendors' SNMP implementation failed and a particular datagram can be identified as the cause and the failure is NOT dependent on community string

[40] "PROTOS – Security Testing of Protocol Implementations" : July 10 2004 URL: http://www.ee.oulu.fi/research/ouspg/protos/
[41] "c06-snmpv1": July 10 2003 URL: http://www.ee.oulu.fi/research/ouspg/protos/testing/c06/snmpv1/index.html

- **Inconclusive**: a vendors' SNMP implementation failed and a particular datagram cannot be identified as the cause
- **Unknown**: a target machine became so corrupt that test results could not be collected
- **Passed**: none of the above

The above definitions give the vendor the benefit of the doubt. Nothing bad happens in only in the "Passed" category. All other categories were used to designate a failure of some sort. The test team notified the vendors and published the results:

- http://icat.nist.gov/icat.cfm?cvename=can-2002-0013
- http://icat.nist.gov/icat.cfm?cvename=can-2002-0012
- http://www.cert.org/advisories/CA-2002-03.html

ISS identified the PROTOS test suite as an "attack tool" and created the following advisory for it:

- http://xforce.iss.net/xforce/alerts/id/advise110

The list of vendors and the results of the testing can be found at:

- http://www.kb.cert.org/vuls/id/854306

As shown in the above web site the vendor's implementations of SNMP are not stellar:

Status	Vendor Count
Not Vulnerable	43
Vulnerable	93
Unknown	150

Most vendors have responded and remedied the flaws found by PROTOS but this is clearly not the entire story. PROTOS pounded the SNMP agents of these vendors with a few thousand different SNMPv1 datagrams and sent the results to the vendors.

This may sound like a lot of combinations but it is actually quite small. If someone were to take the PROTOS suite, which is actually just a directory full of SNMPv1 datagrams, create a few million datagram permutations, and run these against vendor SNMP implementations then other flaws would undoubtedly be found.

Reference List

General SNMP References

www.simpleweb.org

www.snmplink.org

http://silver.he.net/~rrg/snmpworld.htm

www.faqs.org/faqs/snmp-faq

Cited References

http://www.sans.org/top20/

http://www.sans.org/top20/#U4

http://www.iana.org/assignments/port-numbers

http://www.cis.ohio-state.edu/cgi-bin/rfc/rfc1700.html

http://isc.incidents.org/trends.html

TCP/IP Illustrated, Volume1 The Protocols: W. Richard Stevens

http://www.garlic.com/~lynn/rfcterms.htm

http://www.isi.edu/in-notes/rfc1157.txt

http://www.ietf.org/iesg/1rfc_index.txt

http://www.itu.int/ITU-T/studygroups/com17/languages/X.690-0207.pdf

http://www.itu.int/ITU-T/studygroups/com17/languages/X.683-0207.pdf

http://www.isi.edu/in-notes/rfc3416.txt

http://www.isi.edu/in-notes/rfc3411.txt

http://www.isi.edu/in-notes/rfc3414.txt

http://community.roxen.com/developers/idocs/rfc/rfc2403.html

http://www.faqs.org/rfcs/rfc3174.html

http://www.zvon.org/tmRFC/RFC3414/Output/chapter8.html

http://www.isi.edu/in-notes/rfc2578.txt

ftp.cisco.com/pub/mibs/v1

ftp.cisco.com/pub/mibs/v2

http://www.mg-soft.si/RFC1213-MIB.html

http://www.iana.org/assignments/enterprise-numbers

"Webster's New Collegiate Dictionary": by G. & C. Merriam Co.

"Understanding SNMP MIBs" David Perkins Evan McGinnis – Prentice Hall 1997 ISBN 0-13-437708-7

http://www.wtcs.org/snmp4tpc/snmp4tpc.htm

http://www.kb.cert.org/vuls/id/854306

http://www.securiteam.com/windowsntfocus/6S00M0K5SM.html

http://xforce.iss.net/xforce/xfdb/10431

http://www.kb.cert.org/vuls/id/887393

http://www.nextgenss.com/advisories/snmp_dos.txt

http://www.ee.oulu.fi/research/ouspg/protos/

http://cve.mitre.org/cve/downloads

http://cve.mitre.org/cve/candidates/downloads-can.html

http://www.securityfocus.com/bid/6030/info/

http://support.microsoft.com/default.aspx?scid=kb;en-us;Q296815

http://www.microsoft.com/technet/treeview/default.asp?url=/technet/prodtechnol/windows2000serv/reskit/tcpip/part3/tcpch10.asp

http://www.atstake.com/research/tools/network_utilities/

www.foundstone.com

http://www.lokbox.net/SecureWin2k/snmp.asp.

http://icat.nist.gov/icat.cfm?cvename=can-2002-0013

http://icat.nist.gov/icat.cfm?cvename=can-2002-0012

http://www.cert.org/advisories/CA-2002-03.html

http://xforce.iss.net/xforce/alerts/id/advise110

http://www.kb.cert.org/vuls/id/854306