

# Security Issues in SNMP

Andrew R. Calder

University of Abertay

## Abstract

The SNMP protocol has been around for many years now and during that time there have been multiple versions. This paper aims to investigate how security varies between versions and how vulnerable the most commonly used version is, as well as identifying the reason those vulnerabilities exist.

---

# TABLE OF CONTENTS

<b>1. INTRODUCTION</b>	<b>4</b>
1.1 WHAT IS SNMP?	4
1.1.1 MANAGED DEVICE	4
1.1.2 SNMP AGENT	4
1.1.3 MANAGEMENT INFORMATION BASE & OBJECT IDENTIFIERS	5
1.1.4 NETWORK MANAGEMENT SYSTEM	6
1.1.5 A PROVISIONAL SOLUTION	7
1.2 SNMPv1	8
1.2.1 SECURITY IN SNMPv1	8
1.2.2 INSECURITY OF SNMPv1	10
1.2.2.1 TRANSPORT FLAWS	10
1.2.2.2 AUTHENTICATION FLAWS	10
1.2.2.3 DISCLOSURE OF INFORMATION	11
1.2.2.4 IMPLICATIONS	11
1.3 SNMPv2	11
1.3.1 SECURITY IN SNMPv2	11
1.3.2 INSECURITY OF SNMPv2	12
1.3.3 FAILURE TO LAUNCH	12
1.5 SNMPv3	13
1.5.1 SECURITY OF SNMPv3	15
1.5.2 INSECURITY OF SNMPv3	16
1.5.2.1 AUTHENTICATION FLAWS	16
1.5.2.2 OTHER SECURITY FLAWS	17
<b>2. OBJECTIVES</b>	<b>18</b>
<b>3. PROCEDURES AND RESULTS</b>	<b>19</b>
3.1 SETTING UP THE VIRTUAL NETWORK	19
3.1.1 SETTING UP WINDOWS SERVER 2008	21
3.1.2 SETTING UP WINDOWS 7	22
3.1.3 SETTING UP KALI LINUX	23
3.2 CONFIGURING SNMP	25
3.3 ATTACKING SNMP	28
3.3.1 INFORMATION DISCLOSURE/MAN IN THE MIDDLE	28
3.1.2 CRACKING THE COMMUNITY STRING	29
<b>4. DISCUSSION</b>	<b>31</b>
4.1 VENDOR IMPLEMENTATION	31
4.2 MITIGATIONS	32
<b>5. CONCLUSION</b>	<b>33</b>
<b>6. TABLE OF TERMS</b>	<b>34</b>
<b>8. APPENDIX X – SNMPv3 SPECIFICATIONS (IEFT.ORG)</b>	<b>36</b>
<b>9. REFERENCES</b>	<b>37</b>

# TABLE OF FIGURES

<b>FIGURE</b>	<b>PAGE</b>
Figure 1.1.3a - "MIB-2"	4
Figure 1.1.3b - "MIB-2 subtree"	5
Figure 1.1.4a - "A simplified model of SNMP"	6
Figure 1.2.1a - "Trap Types"	8
Figure 1.2.1b - "SNMPv1 Simplified"	8
Figure 1.5a - "SNMPv2* vs SNMPv3"	12
Figure 1.5.1a - "SNMPv3 Packet Format"	15
Figure 3.1a - "GNS3 Network Configuration"	19
Figure 3.1.1a- Server 2008 Static IP	20
Figure 3.1.2 - Windows 7 Static IP	21
Figure 3.1.3a - Kali Static IP	22
Figure 3.1.3b - IP Addresses Set (Confirmation)	23
Figure 3.2a - Enabling SNMP in services	24
Figure 3.2b - Default SNMP Configuration	25
Figure 3.2c - Query Uptime	26
Figure 3.3.1a - Configure Wireshark for GNS3	27
Figure 3.3.1b - Man in The Middle (MITM) Attack	28
Figure 3.3.2a - 'Securer' SNMP Configuration	28
Figure 3.2.2b - Hydra Attack	29
Figure 3.2.2c - Hydra Guesses Password Correctly	29
Figure 3.2.2d - Hydra Triggers Trap Security Alerts	29

## 1. INTRODUCTION

For the past two decades, network administrators have relied on the Simple Network Management Protocol (SNMP), to manage and monitor networked devices. With little competition, it has become the de facto standard when it comes to network management. However, a report from United States Computer Emergency Readiness Team (US-CERT) suggests that many implementations contain vulnerabilities that have left several hundred products by many different vendors open to exploitation. An attacker able to successfully exploit these vulnerabilities would be able to perform undesirable actions; "These vulnerabilities may cause denial-of-service conditions, service interruptions, and in some cases may allow an attacker to gain access to the affected device. Specific impacts will vary from product to product." (*us-cert.gov*).

SNMP lowers costs significantly by eliminating the need for local administrators at multiple locations, instead allowing for remote administration, it coherently presents data and makes network management easier and more convenient. However, is the convenience worth the risk?

### 1.1 WHAT IS SNMP?

The SNMP is a popular network management protocol implemented on the application layer of the networking stack that eases the exchange between managed devices and network management systems. It primarily operates over User Datagram Protocol (UDP) -using port 161- but may also use OSI Connectionless Network Services (CLNS), AppleTalk Datagram Delivery Protocol (DDP) or Novell Internet Packet Exchange (IPX) (*Zobel, D, 2010*). It allows the monitoring of services such as Dynamic Host Configuration Protocol (DHCP) and for configuration and collection of information.

At the highest-level SNMP consists of three main components:

- Managed Devices
- Agents
- Network Management Systems (NMS)

#### 1.1.1 MANAGED DEVICE

A *Managed Device* is quite simply a device residing on a managed network, it could be, but is not limited to, printers, routers, alarms, switches, servers and clients – just about anything on the network (*Technet, 2003*). Although SNMP can be used with a diverse range of devices, the way in which information is accessed is standardized.

#### 1.1.2 SNMP AGENT

Any SNMP Managed Device will have an SNMP *Agent*. An agent is program or service that listens for SNMP requests and handles responses to/from the manager. Agents do the majority of the work; they effectively translate device information into a SNMP-usable format, in order to present the device information to a Network

Management System. The information gathered by agents is stored in a database called the "*Management Information Base*" or MIB.

### 1.1.3 MANAGEMENT INFORMATION BASE & OBJECT IDENTIFIERS

The Management Information Base (MIB) is a collection of specifications that define the particular properties of a managed device. The MIB is particularly important in SNMP because if an object of a managed device isn't described in the MIB then as far as the NMS and agents know, it doesn't exist. The MIB effectively acts as a translator; it helps the NMS understand SNMP responses obtained from managed devices. RFC 1213 describes "MIB-2" which is a standardised MIB that is supposed to be included in all SNMP devices (*ietf.org, rfc1213*) it allows the NMS to request various information, as detailed in the figure 1.1.3a below.

Subtree Name	OID	Description
<i>system</i>	<i>1.3.6.1.2.1.1</i>	Defines a list of objects that pertain to system operation, such as the system uptime, system contact, and system name.
<i>interfaces</i>	<i>1.3.6.1.2.1.2</i>	Keeps track of the status of each interface on a managed entity. The <i>interfaces</i> group monitors which interfaces are up or down and tracks such things as octets sent and received, errors and discards, etc.
<i>at</i>	<i>1.3.6.1.2.1.3</i>	The address translation ( <i>at</i> ) group is deprecated and is provided only for backward compatibility. It will probably be dropped from MIB-III.
<i>ip</i>	<i>1.3.6.1.2.1.4</i>	Keeps track of many aspects of IP, including IP routing.
<i>icmp</i>	<i>1.3.6.1.2.1.5</i>	Tracks things such as ICMP errors, discards, etc.
<i>tcp</i>	<i>1.3.6.1.2.1.6</i>	Tracks, among other things, the state of the TCP connection (e.g., <i>closed</i> , <i>listen</i> , <i>synSent</i> , etc.).
<i>udp</i>	<i>1.3.6.1.2.1.7</i>	Tracks UDP statistics, datagrams in and out, etc.
<i>egp</i>	<i>1.3.6.1.2.1.8</i>	Tracks various statistics about EGP and keeps an EGP neighbor table.
<i>transmission</i>	<i>1.3.6.1.2.1.10</i>	There are currently no objects defined for this group, but other media-specific MIBs are defined using this subtree.
<i>snmp</i>	<i>1.3.6.1.2.1.11</i>	Measures the performance of the underlying SNMP implementation on the managed entity and tracks things such as the number of SNMP packets sent and received.

Figure 1.1.3a - "MIB-2" (Mauro and Schmidt, 2005)

As touched on above, the MIB files use an SNMP-specific format; managed devices may have multiple unique *Object Identifiers* (OIDs), which are organized in a hierarchical manner. For example, typical objects that could be monitored on a switch may include the volume of incoming and outgoing traffic, or the rate of packet loss. MIBs are flexible and there may be vendor-specific additions, but generally most well-defined areas are globally standardised.

OIDs also follow a hierarchical structure, it can be represented as a multi-level tree much like a directory on a computer. However, unlike a directory, OIDs typically use a numerical format as opposed to the clearer textual representation.

The structure of a *Management Information Base* can be thought of as a top-down hierarchical tree. At each branch there is a unique identifying string and number; the strings and numbers may be used interchangeably (Ellingwood, J, *Digitalocean.com*, Aug 18, 2014).

In order to refer to a specific object in the tree, a path must be traced from the 'root' of the tree to the object in question. Each node in the path is concatenated to produce an address, each node is separated by a colon. The entire address is the Object Identifier. For example, the object identifier for MIB-2's UDP statistics can be represented as "1.3.6.1.2.1.7" or as "iso.org.dod.internet.mgmt.mib-2.UDP", the structure itself is represented in Figure 1.1.3b below.

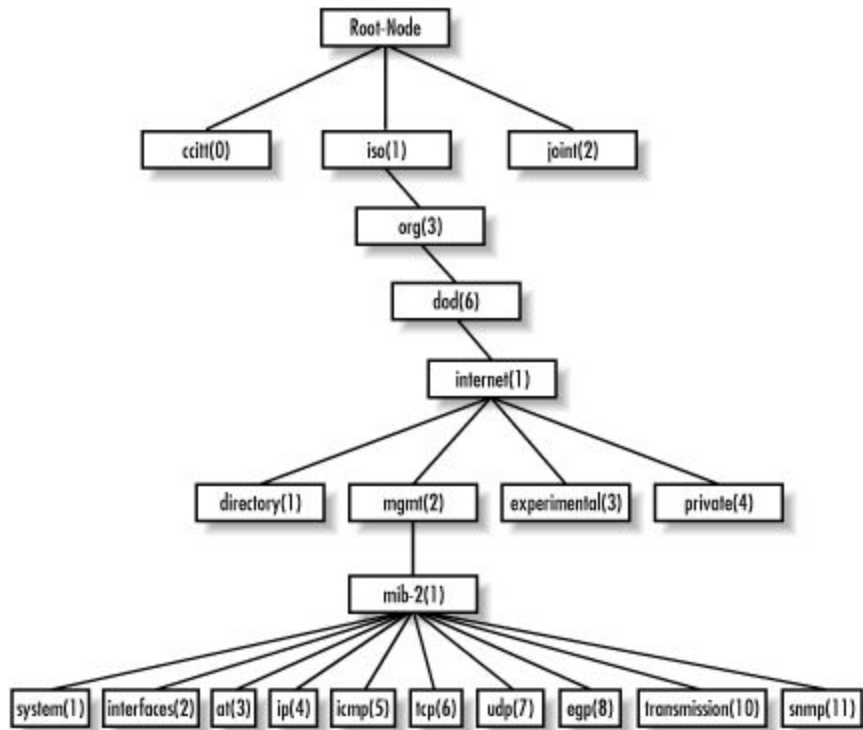


Figure 1.1.3b - "MIB-2 subtree" (Mauro and Schmidt, 2005)

### 1.1.4 NETWORK MANAGEMENT SYSTEM

The *Network Management System* is a program that queries SNMP agents and handles responses to/from them. It provides network administrators the ability to manage and monitor Managed Devices. The status information of any device can be requested by a NMS; it may contain, among others, versions of installed packages, IP addresses, available disk space, session tables and ARP tables. A manager can proactively request information using messages such as "*SetRequest*" or "*GetRequest*", or it may wait for the agent to send information at a preset interval (digitalocean.com). In addition, a manager should also respond to "*Response*" and "*Trap*" messages. Each message has a corresponding *Protocol Data Unit* which is a numerical value. For example, *GetRequest* has a PDU of 0.

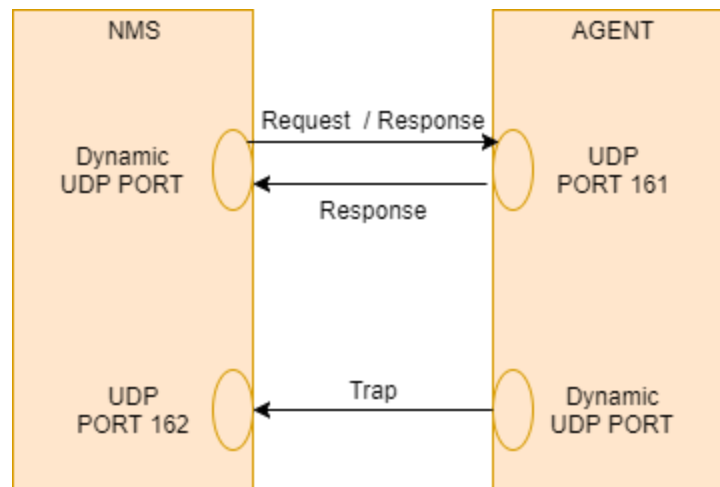


Figure 1.1.4a - "A simplified model of SNMP"

There are three main releases of the Simple Network Management Protocol, and most networked devices will offer some form of SNMP support. The most commonly used version is SNMPv2 which came out just over 20 years ago and well known for its (in-)security. Its popularity seems to be orientated around the ease of implementation; windows server doesn't natively support any higher than SNMPv2 -if at all (*Technet, 2012*).

### 1.1.5 A PROVISIONAL SOLUTION

SNMP was initially thought of as a provisional solution by its creators, as such they made it modular to ease transition to an official standard. International Organisation for Standardization (ISO) did not develop an official standard however, and with no competition SNMP became widely used (*Etingof, 2017*).

SNMP milestones:

- Research Project - a successor of Simple Gateway Monitoring Protocol (SGMP)
- SNMPv1- 1988: first appearance
- SNMPv2c - 1993: Additions to data types, counter size and new protocol operations
- SNMPv3 - 2003: Complete redesign, recognized as Internet Standard (*STD0062*) by IETF

Since 2004, the IETF has recognized SNMPv3 as the current standard. SNMPv3 has been designated an Internet Standard which is the highest level of maturity an RFC can reach (*Hoffman, P, IEF.T.org*). Despite being the *newest*, most secure version, SNMPv2c is still used far more often as it has most of the features and unlike SNMPv3 it is easy to implement.



## 1.2 SNMPv1

The management of large networks can be complex. Ensuring that all devices, be it printers, routers, switches, servers or clients, perform as expected may offer a significant challenge. With the introduction of the Simple Network Management Protocol (SNMP), a successor to the Simple Gateway Monitoring Protocol (SGMP), operators were given an alternative to pinging problem-devices or relying on on-site workers. This tool could consistently gather the information required.

SNMPv1 was defined in RFC 1212, RFC 1155 and RFC 1157 as "A Simple Network Management Protocol (SNMP)", "a standard that defines how communication occurs between SNMP-capable devices and defines SNMP message types." (*ietf.org, 1988-1991*). It uses MIB which was specified in RFC 1155 and RFC 1158 to define access to aspects of a managed device.

The protocol itself was written in Abstract Syntax Notation One (ASN.1) which is an Interface Definition Language, in that it allows for communication between software components that cannot directly interact, for example, two different operating systems. These days this isn't so much of an issue due to standardization but when SNMPv1 was developed it provided a bridge between completely different systems. The use of ASN.1 in the development of SNMP provides some context to the unusual naming conventions; ASN.1's Basic Encoding Rules (BER) describe a string of characters as an 'octet string'. In RFC 2578 the community string is defined as an "octet string", and the two terms are used interchangeably throughout the documentation.

### 1.2.1 SECURITY IN SNMPv1

SNMPv1 introduced the '*Community string*', also known as the '*octet string*'. The community string is effectively a password shared between agent(s) and manager. The general idea is that in a network you would have certain *communities* of devices that you may wish to access at one time; this 'community string' would allow access to said devices when required whilst providing basic security. An SNMP agent may belong to more than one SNMP community and it will not respond to NMS that are not part of one of its communities.

The default communities are:

- private – READ/WRITE ACCESS
- public – READ ONLY

If the community string is invalid when attempting to access a managed device the device may send out a Trap which is effectively an alert. This can be used to notify the NMS of unauthorized access attempts among other things.

Trap Type	Name	Description
0	Cold start	Agent is booting
1	Warm start	Agent is rebooting
2	Link down	An interface has gone down
3	Link up	An interface has come up
4	Authentication Failure	An invalid community was received in a message
5	EGP neighbor loss	An EGP peer has gone down
6	<vendor specific>	<vendor specific>

Figure 1.2.1a - Trap Types - Based on comptechdoc.org's "Net-SNMP Trap Types" (Comptechdoc.org, n.d.)

There are two other security parameters in SNMPv1, these are:

- Accepted Community Names -Only allows requests from NMS in list of communities
- Accept SNMP Packets From:
  - ...any host – This is the default configuration, very insecure.
  - ...these hosts – Only allows requests from hosts on a list of IP addresses.

In SNMPv1 there were only five simple messages, three were manager-only and the remaining two were agent-only; these are detailed in the table below. Each message has a corresponding Protocol Data Unit or PDU, which is the way the protocol identifies the request type. A breakdown of the PDUs available in SNMPv1 can be found in Appendix A – SNMPv1 PDUs. The messages provided basic functionality such as checking volume of traffic and giving managed devices a way to alert NMS of any issues (Trap). A simplified representation of SNMPv1 can be seen in figure 1.2.1b below.

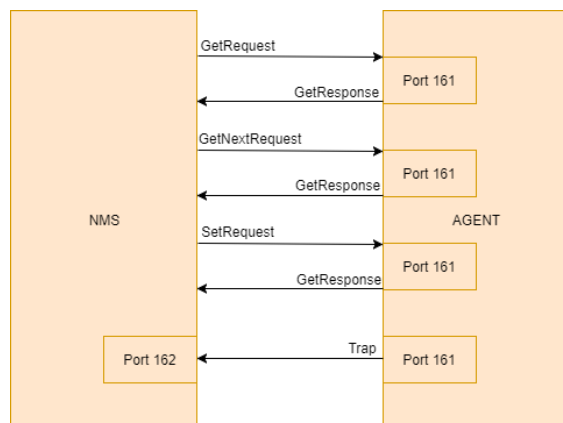


Figure 1.2.1b - "SNMPv1 Simplified" - N.B. Interestingly, a response to a 'SetRequest' is still a 'GetResponse'

## 1.2.2 INSECURITY OF SNMPv1

### 1.2.2.1 TRANSPORT FLAWS

As UDP is used to transport the PDUs, packets may or may not be received, coupled with the fact that Traps do not require a response; it is conceivable that a situation may arise where something unintended is happening that the NMS is not aware of.

In comparison to TCP, UDP is trivial to spoof; there is no 'handshake' with the device that would serve as a basic confirmation of identity. With UDP the source can be forged which negates the 'accepted hosts' security feature – if an attacker is able to discover or guess the IP of a NMS.

In RFC 1812, a solution to UDP source spoofing was discussed but it just was not viable with current networks – *"We considered suggesting routers also validate the source IP address of the sender as suggested in 8, but that methodology will not operate well in the real networks out there today. The method suggested is to look up source addresses to see that the return path to that address would flow out the same interface as the packet arrived upon. With the number of asymmetric routes in the Internet, this would clearly be problematic."* (Baker. F, *ieft.org*, 1995).

To this day it still problematic to try and implement any sort of checking of the source IP, if source filtering is enabled then fast switching will be disabled, increasing traffic latency from a few milliseconds to a few seconds. It is simply not practical.

### 1.2.2.2 AUTHENTICATION FLAWS

As Community strings are sent in cleartext, there is no attempt to hide them and thus anyone sniffing traffic between a NMS and an agent will effectively be handed the password.

Like passwords, community strings are often very easy to predict. Typically people will leave the default communities or pick an unimaginative community string like '*switch*' or '*router*'.

Nowhere in the specification does it state that there should be a limit on the number of community string attempts, as such an attacker may attempt to brute-force the community string without worry of being locked out. In theory each invalid attempt of the community string in a brute-force attack should cause the targeted agent to send out a trap detailing such an event has occurred but as it uses UDP, and no response is required the agent wouldn't know if the NMS received its trap or not.

Another thing that isn't monitored in this version of SNMP is the message issue number or associated time stamp; there is no consequence for a PDU arriving that is not the next expected one e.g. 556 after 555, nor is there consequence for there being a time skew between the sent time and received time.

### 1.2.2.3 DISCLOSURE OF INFORMATION

Assuming the attacker could gain access to the targeted agent what sort of information could they gain? Depending on the type of device targeted the attacker may have access to the following:

- Network Topology
- Routing Tables
- Network Traffic Statistics
- Filter Rules
- Any proprietary vendor information

### 1.2.2.4 IMPLICATIONS

Since there was no controls in place to handle clock skew between PDUs or mismatching message numbers it is possible for replay attacks to be performed. A replay attack is when a valid PDU is fraudulently delayed or repeated (*microsoft.com*, 2017).

An unauthorized individual may be able to alter or manipulate the network; modification of access control lists would allow access to areas that might otherwise have been untouchable, Denial of services attacks can be conducted internally, a clear network topology of the network would reduce volume of enumeration an attacker would have to perform, and finally, it makes more sophisticated attacks easier.

## 1.3 SNMPv2

SNMPv2 (Technically SNMPv2\*) was defined in RFC 1441 and RFC 1446 as "...version 2 of the Internet-standard Network Management Framework.", and "Security Protocols for version 2 of the Simple Network Management Protocol". It was written to address the feature and security deficiencies of SNMPv1.

In terms of design, SNMPv2 was just an extension of SNMPv1; providing some highly sought features and an attempt to improve upon the weak security found in its predecessor.

It added two new PDU types; '*GetBulkRequest*' – allows for the retrieval of multiple objects from a managed device, and '*InformRequest*' – a way for NMS to acknowledge the receipt of traps.

### 1.3.1 SECURITY IN SNMPv2

New, complex security services were defined which would mitigate many the threats faced in the previous version, were as follows:

- Data Integrity
  - Ensures that the content of the message has not been altered or destroyed by an unauthorized party, nor have the contents been altered to a greater extent than would be expected to occur non-maliciously.
- Data Origin Authentication
  - Ensures that the claimed origin can be corroborated.
- Data confidentiality

- The contents of the message should not be disclosed to unauthorized parties.

The concepts behind the offered security services were solid, they are very similar to what is offered in the more successful SNMPv3. Unfortunately however, SNMPv2's specification of the services was ill-defined and subsequently had several failings.

### 1.3.2 INSECURITY OF SNMPv2

An attempt was made to reduce the window in which replay attacks could occur. By recording the time, the PDU was sent it could be compared to the clock of the receiving party (agent or NMS) +/- a number of seconds to allow for transportation time. For this to work properly the clocks of the devices would have to be synced, while this might seem obvious it was not mentioned in the specification. If a developer was to follow the specification exactly the feature would not work and thus in some implementations this security feature was not included at all.

The chosen implementation of the specified encryption – DES – had known issues. It requires an Initialization Vector – which is effectively a seed in modern terms. Requiring an initialization vector isn't inherently bad, however, not specifying how to generate it is. If the initialization vector isn't randomized, it becomes significantly easier to brute-force and in some cases even guess the content of messages.

Assuming the initialization vector was generated properly, DES is still an issue. A DES key is only 56-bit, this is too short. In a 1996 paper a group of well known cryptographers looked at key lengths. They suggested a minimum of 90 bits would be required to consider a cipher secure (*Blaze. M, Diffie. W, Rivest. B, Schneier. B, Shimomura. T, Thompson. E, Weiner. M, 1996*). As proof of its weakness, in 1998 the *Electronic Frontier Foundation* developed a DES-cracker that could find a DES key in less than days' worth of searching and in 2004 the updated version was able to compute every possible DES key in 22 hours (*EFF.org, 1999*). Using modern tools such as '*John the Ripper*' or '*Hashcat*' it is possible to do this in minutes.

This version actually included MD5 for authentication however it was not enforced to allow for backwards compatibility with devices which may not support the version. By providing backward compatibility certain '*mitigated*' vulnerabilities from SNMPv1 may still affect this version and as it still uses UDP, all the transportation flaws are still valid.

### 1.3.3 FAILURE TO LAUNCH

SNMPv2 provided only marginal security improvements at best and a lack of definition in the specification meant that implementation of the protocol was overly complicated. This did not provide developers and network administrators much motivation to upgrade from SNMPv1. The complexity was also prevalent in the protocol itself; devices were a whole lot slower when using SNMPv2 as its technical complexity meant that additional resources were needed; many devices did not have the ram or computational power to adequately function.

## 1.4 SNMPv2C

When vendors mention support of SNMPv2, they almost always turn out to be using SNMPv2C. In RFC 1901 this is noted - "The administrative framework for SNMPv2 identified in this document is the same framework as was defined for SNMPv1. Use of this administrative framework with SNMP Version 2 is commonly known as "Community-based SNMPv2 (SNMPv2C)."(*ietf.org*).

SNMPv2C is a 'dirty solution' to the problems presented by SNMPv2 (hereby referred to as SNMPv2\*), it provides the additional PDU types introduced in SNMPv2\* but still only uses the basic community string-based authentication found in SNMPv1. In short, it is all the features of SNMPv2\* with the security of SNMPv1; if anything, v2C is a downgrade regarding security as it provides more features to exploit and no new protections.

Despite being arguably the weakest version of SNMP, SNMPv2c is still the most commonly used. This is largely due to support issues; Windows Server 2008 supports SNMPv1 and SNMPv2C, Windows Server 2012 does not seem to support SNMP "SNMP is deprecated." – but SNMPv2C can still be installed, and Windows Server 2016 does not support SNMP at all.

## 1.5 SNMPv3

To address the failings of the original SNMPv2 (SNMPv2\*), SNMPv3 was developed. Having learned what happens when something is ill-defined, authors ensured SNMPv3 was very well defined – the specifications for SNMPv3 spans 13 documents. A summary of these documents can be seen in Appendix B – SNMPv3 Specifications.

By design it is easily implementable and far more secure than SNMPv2C; it is what SNMPv2\* should have been. In fact, if the two are compared, it is immediately noticeable how similar they are.

	<i>SNMPv2*</i>	<i>SNMPv3</i>
<i>Transport Protocol</i>	UDP	UDP
<i>PDU Types</i>	SNMPv2 Types	SNMPv2 Types
<i>PDU Format</i>	SNMPv2 Format	SNMPv2 Format
<i>Security Model</i>	User-based	User-based
<i>Authentication</i>	MD5/None	User-defined/None

*Figure 1.5a – SNMPv2\* vs SNMPv3*

SNMPv3 still uses the same PDU format and types as SNMPv2 – no new PDU types were part of the specification. It also still uses UDP, which has been consistently used throughout every revision of SNMP so no surprises here. The changes to SNMPv3 are primarily related to security; it has a *new* user-based security model and offers much stronger encryption and HMAC based authentication, as well as a new view based access control model which is effectively an enhancement to MIB. Unfortunately, SNMPv3 shares the resource-hungriness that plagued SNMPv2\*; fortunately, it has been around a decade and most networks can now handle the additional load.

### 1.5.1 SECURITY OF SNMPv3

In comparison to the lack-luster security offered by previous versions of SNMP, SNMPv3 offered a sizable improvement – even going as far as removing backwards compatibility. In RFC 3414 (*ietf.org*), Authors *Blumenthal & Wijnen* outline the goals for the new security module as follows.

1. *“Provide for verification that each received SNMP message has not been modified during its transmission through the network.”*
2. *“Provide for verification of the identity of the user on whose behalf a received SNMP message claims to have been generated.”*
3. *“Provide for detection of received SNMP messages, which request or contain management information, whose time of generation was not recent.”*
4. *“Provide, when necessary, that the contents of each received SNMP message are protected from disclosure.”*

The goals of the new security model are clearly a direct response to the failings of previous versions. Without goal 3, for example, replay attacks would still be a big issue and without goal 4 it would still be possible for people to obtain the community strings simply by viewing SNMP packets.

The biggest security-related changes to SNMP in this version are the addition of good (at the time) encryption and authentication. The PDU is encrypted with DES in Cipher Block Chaining (CBC) mode, this adheres to goal 4 as it ensures the contents of the PDU are not disclosed and thus neither are the community strings. Also introduced was a password to key mechanism which maps a password to an MD5 or SHA-1 private key, in an attempt to slow down brute-force attacks as per the *A.2. Password to Key Algorithm* specification in RFC 3414 (*ietf.org*). Hash-based Message Authentication Code (HMAC) is to be used but authentication still mostly dependent on implementation - although SHA1 and MD5 are suggested meaning there is little excuse for not including any at all.

The way the new authentication and encryption affects SNMP packets can be seen below in Figure 1.5.1a -" SNMPv3 Packet Format".



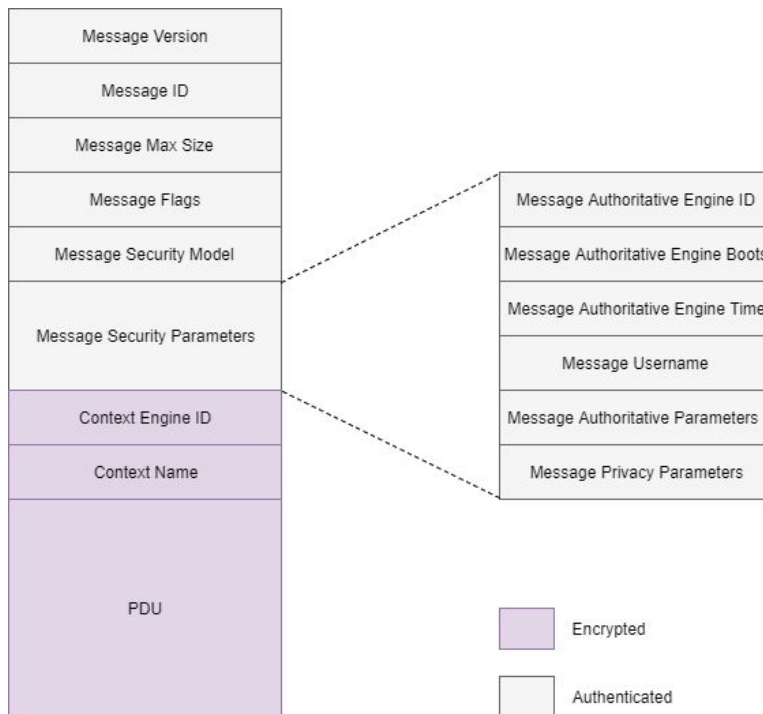


Figure 1.5.1a - "SNMPv3 Packet Format" from "SNMP, SNMPv2, SNMPv3 and RMON 1 and 2, 3rd Edition", William Stallings, 1998

## 1.5.2 INSECURITY OF SNMPv3

Although SNMPv3 is a massive improvement over its predecessors, it isn't perfect. Many of the issues that were prevalent in previous versions haven't been mitigated as such, but rather become more difficult to abuse. Exploitation is still possible, but it will take longer.

DES in SNMPv3 is still DES and thus is prone to the same vulnerabilities touched on in SNMPv2\* - it can be cracked in a short length of time. CBC mode for DES depends on the Initialization vector being 64-bit. The initialization vector is created by taking the last 8 octets of a private key and XORing it with an 8 octet salt value. This salt value is transmitted in "Message Privacy Parameters". The issue with this lies in the generation of the salt value - once again it is left up to the implementer. The salt value should be randomized but by not specifying how to do this, the likelihood of poor implementation increases.

While it is still easy to brute-force or dictionary attack bad passwords, longer and more complex passwords are now harder to obtain than before as the addition of the *Password to Key Algorithm* slows the process. It's not impossible to break more complex passwords, but it would take the attacker significantly longer to do so.

### 1.5.2.1 AUTHENTICATION FLAWS

As mentioned earlier, authentication is handled via HMAC. There isn't necessarily anything wrong with HMAC itself but rather in SNMP's implementation of it. MD5 produces a 16 octet

authentication key and SHA-1 produces a 20 octet authentication key. SNMPv3 HMAC truncates output to 16 octets. Effectively, regardless of the size of the outputted authentication key, it will be truncated to 16 octets, rendering the additional security that could otherwise have been provided by more advanced algorithms mostly useless. The generated authentication key is stored in *Message Authorization Parameters*.

*Message Authorization Parameters* is victim to the same issues as *Message Privacy Parameters*, but to a greater degree. Since the hashes are truncated to 16-octets even more hash collisions will be possible and thus a situation may arise where an attacker has a working password without having the correct password.

### 1.5.2.2 OTHER SECURITY FLAWS

In an effort to reduce replay attacks SNMPv3 has implemented a mechanism which relies on the number of times the device has booted and the time since last device reboot. This is stored in the SNMP packet as *Message Authoritative Engine Boots*, *Message Authoritative Engine Time* and a generated *Message Authoritative Engine ID*. If any of the values do not match what is expected then the packet is dropped and a trap of type 4 (Authentication Failure) is sent to the NMS. However, the variance on the expected value is not part of the specification and thus once again, it is up to implementers.

As with all other versions of SNMP, SNMPv3 uses UDP and thus it is still vulnerable to denial of service attacks – but so are a lot of other services that use UDP. Forgery of SNMPv3 packets is technically possible, but absurdly difficult. Brute forcing is possible but far slower than it used to be, and while rather unlikely, there is now the possibility of unexpected hash collisions.

SNMPv3 certainly isn't a golden child – it does not fix everything, but it is a step in the right direction.

## 2. OBJECTIVES

While SNMPv3 is the most secure version of SNMP, many vendors still use SNMPv2C. Version 2C is easier to implement, has backwards compatibility and has most of the non-security features that SNMPv3 has. Then there is the support issue, as mentioned earlier SNMPv2c is also the best supported version of SNMP, windows server does not natively support version 3 at all. This is by no fault of the protocol itself, but rather a lack of awareness amongst people using it. Network management systems are probably most to blame for the current situation as they provide no indication that using SNMPv2C is considered insecure.

Since SNMPv2 is the most commonly used, I will be on it. In my practical I aim to do the following.

1. show just how quickly unauthorized access can be achieved
2. show how easily community strings may be stolen in transit
3. demonstrate the threat of default /improper configuration

To do this I will create a very small simulation network consisting of an attacker, a Network Management System and a Managed Device (Agent). By testing multiple configurations of SNMPv2C I will prove the lack of security and aim to demonstrate the consequences of it.

### 3. PROCEDURES AND RESULTS

In order to evaluate the security of SNMPv2c a suitable environment is needed. In this investigation the testing environment consists of three computers – a NMS, an Agent, and an Attacker- as well as basic infrastructure to allow them to communicate. In a more realistic example there would likely be more network devices and a far more complicated infrastructure setup. However, including more devices does not add to the experiment as it needlessly adds additional steps that lead to the same end result.

This section will be broken up into three parts, starting with the setup of the virtual network, following on with the configuration of SNMP and finishing up with the exploitation of SNMP.

#### 3.1 SETTING UP THE VIRTUAL NETWORK

To setup a suitable network to conduct the test virtualization was required. The cost of physically acquiring the same hardware was far too high, both in terms of time and money. In order to perform this experiment the following devices were required.

1. Windows 2008 Server
2. Windows 7 PC
3. Kali Linux PC
4. Generic Ethernet Switch

Technically, the server and computers could be virtualized and communicate directly, however that would make some of the attacks impossible to conduct; a man in the middle relies directly on capturing traffic between two points. If the devices were directly connected reconfiguration would be required between regular operation and this attack which is completely unrealistic.

In order to virtualize the network GNS3 was used. GNS3 is an emulator for networks – *“GNS3 allows you to visualize, plan, test and troubleshoot network environments across any vendor platform at scale - without the need to directly interact with the network hardware.”* (gns3.com). GNS3 does support windows but on their website they recommend using the Linux version which provides an increase in performance and stability over its Windows counterpart. By word of mouth the author was informed that on Windows GNS3 installs some programs that cannot be easily uninstalled. Between the better performance and stability, and the unsatisfactory Windows experience it was decided that the GNS3 environment itself would exist on an Ubuntu virtual machine. This provides several benefits aside from those already covered. Firstly, virtual machines support snapshots which allows for saving the current state – this is useful as if anything goes wrong starting over completely is not required. Secondly, It makes the environment portable which is useful for debugging

performance issues.

Whilst virtualization solves the availability of devices issue, it also introduces limitations on the environment in the form of hardware limitations. In order to ensure adequate performance on the networked devices their corresponding virtual machines were required to have at least 2GB of memory. Since GNS3 was being run within a virtual machine, the virtualized networked devices had to be run in the virtual machine. This created a lot of overhead and between the virtualization of the GNS3 host and the devices virtualized within the GNS3 network. The entire setup required 70GB of storage and over 10GB of RAM was being used when active. It might have been possible to reduce the memory usage by launching the network devices in headless mode where possible as virtualbox separates video memory usage from system memory usage.

The GNS3 host (Ubuntu) was running in VMware with 10GBs of memory and a 70GB HDD; the 2008 Server required 25GB HDD and 2GB RAM, the Windows 7 network manager also required 2GB of RAM but only a 20GB HDD, the Kali based attacker only required 15GB HDD and 2GB RAM. The switch did not require such values to be manually defined. The virtual network representation as per GNS3 can be seen below in 3.1a.

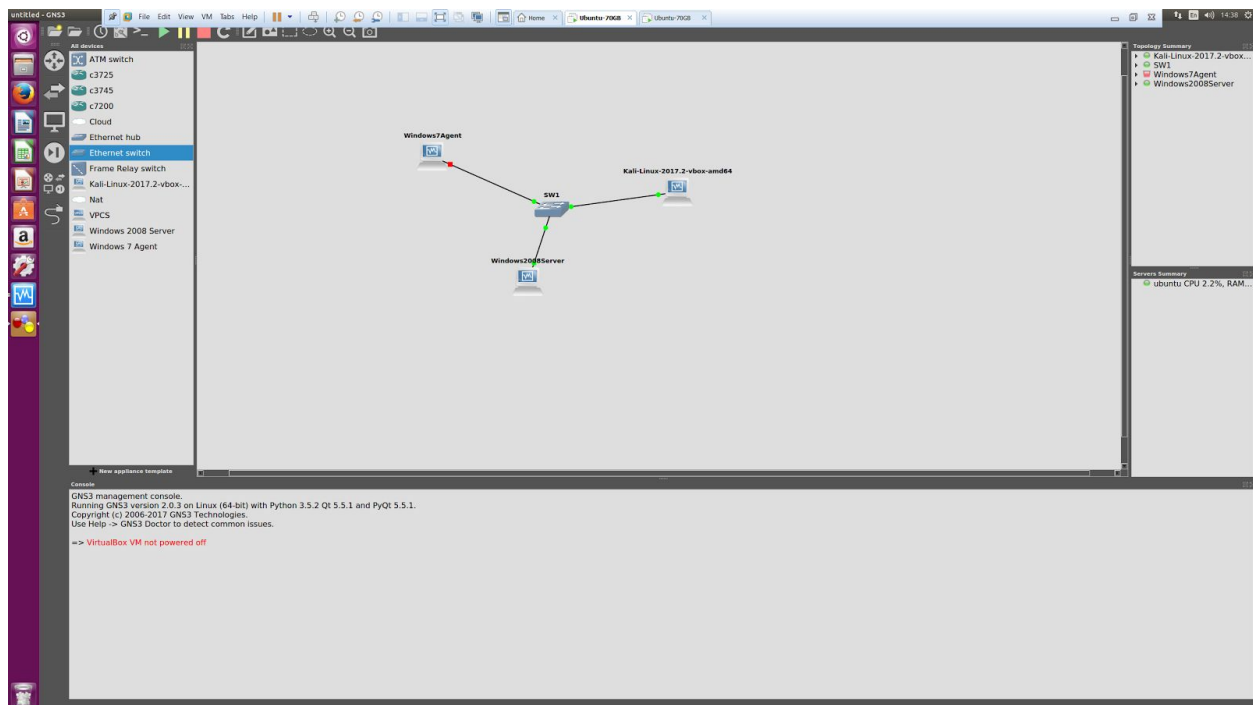


Figure 3.1a - GNS3 Network Configuration

Since this is a simple network, there is no DHCP server and due to encountered issues there is only a switch, not a router. Thus, IP addresses must be static and assigned manually.

### **3.1.1 SETTING UP WINDOWS SERVER 2008**

After installing Windows Server 2008 as a virtualbox based VM it needed to be configured to work with GNS3. GNS3 requires any included virtual machines to be using the Generic adapter so that they may communicate with other devices on the network. There isn't more to it than that; that is just how it works. Having selected the Intel MT1000 generic adapter and importing the virtual machine it can now be connected to the network. It is possible to tell if the device has connected properly or not depending on the network adapter status. If it says "no connections" something has gone wrong, if it says "unidentified network" then it is likely working properly. Unidentified network doesn't seem like it would be the working version but since there is no router the device is unable to identify any networks.

Having connected the device to the network the next thing to do is set the static IP address. On Windows Server 2008 this option is under "Internet Protocol Version 4 (TCP/Ipv4) Properties". For ease of identification the IP address is set as "192.168.0.8", this can be seen in figure 3.1.1a below.

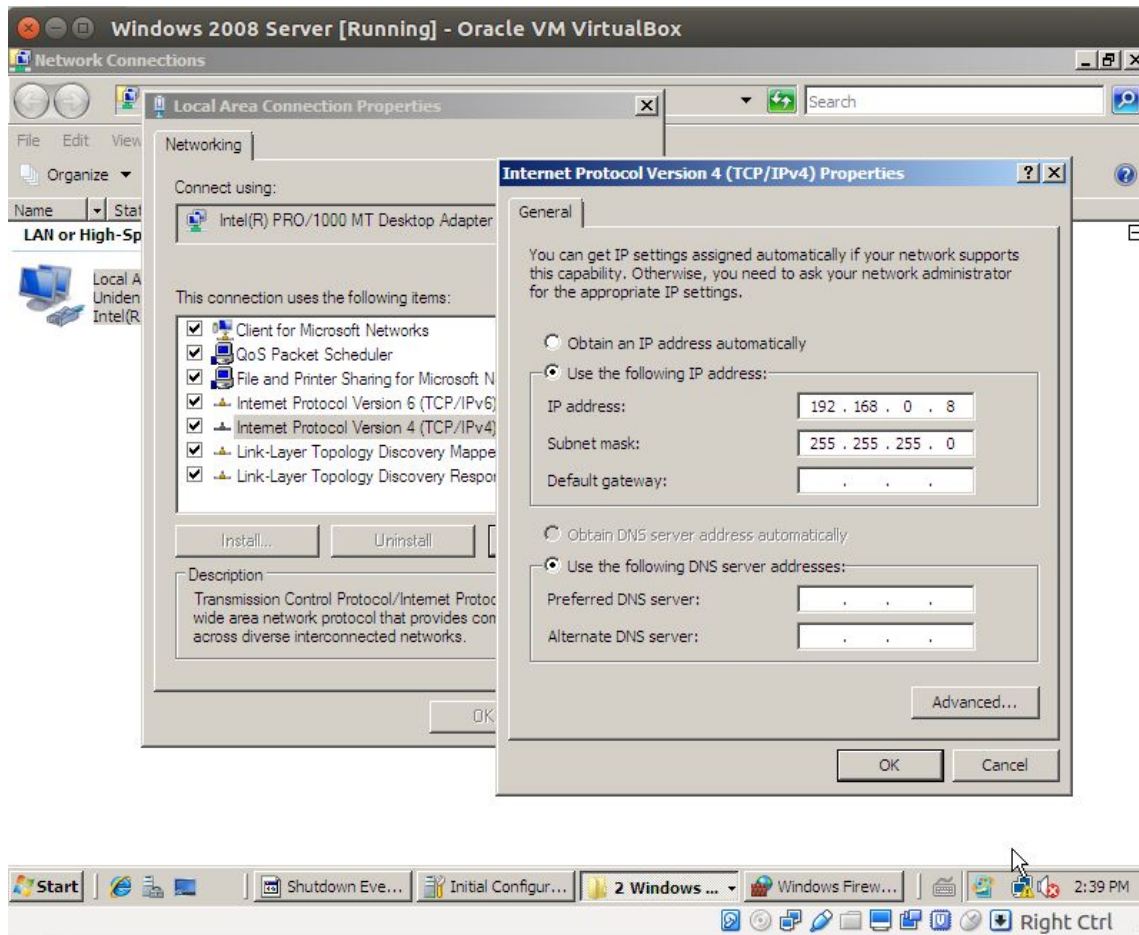


Figure 3.1.1a- Server 2008 Static IP

### 3.1.2 SETTING UP WINDOWS 7

The setup of the windows 7 machine was pretty much identical to that of the Windows 2008 Server. The slight exception to this was in regard to the generic adapter. For some reason Windows 7 would not recognise the default adapter (intel MT1000). In order to get windows 7 working the generic server adapter had to be selected – which was really counterintuitive.

Once again, a static IP address was set. This time “192.168.0.7” was chosen as can be seen in figure 3.1.2a below.

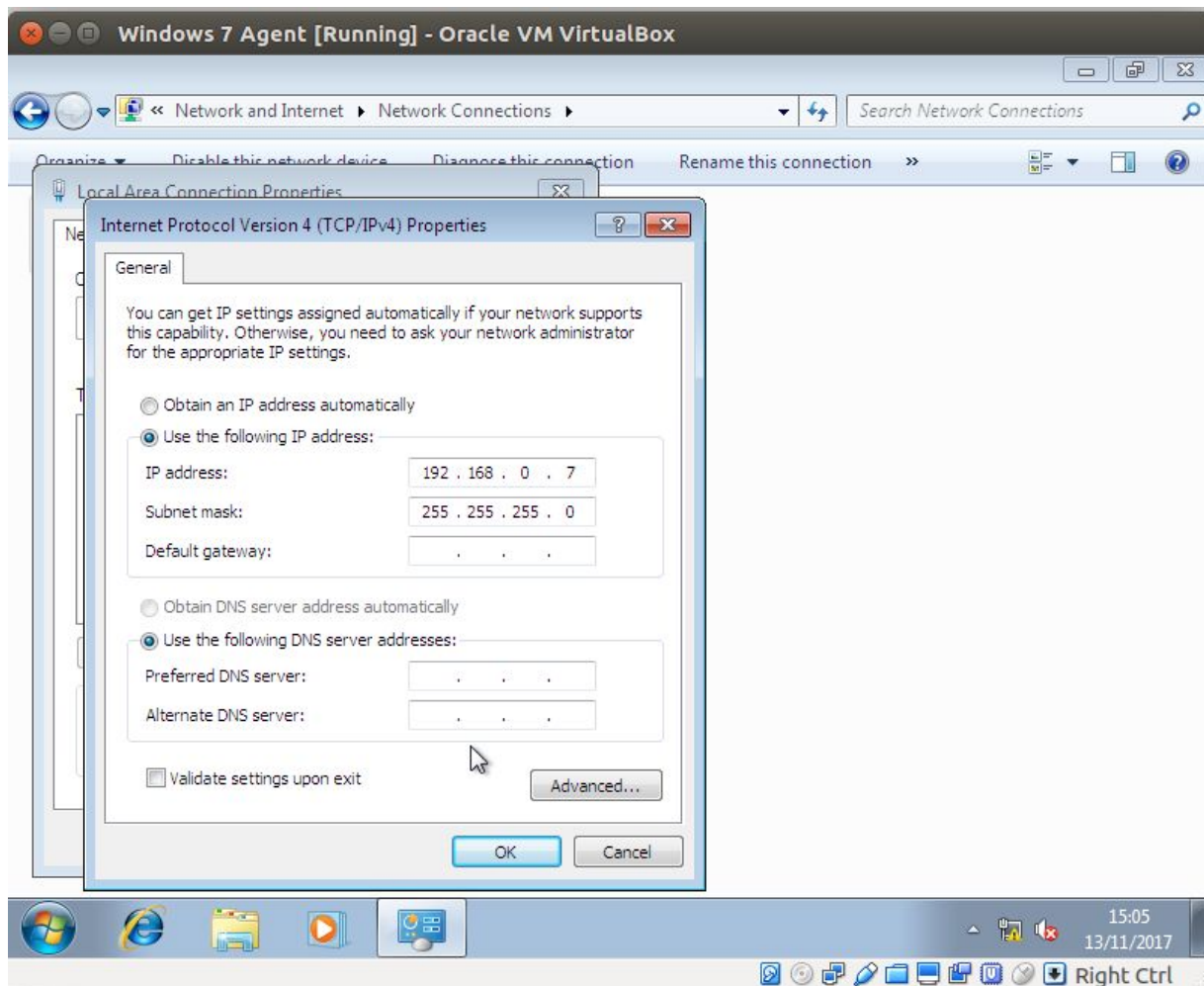


Figure 3.1.2 - Windows 7 Static IP



### 3.1.3 SETTING UP KALI LINUX

Once again, setting up Kali was rather similar to setting up the other devices. The generic adapters worked with no additional configuration and overall the Kali system was the easiest to setup. Under IPV4 settings it was possible to set a static IP of "192.168.0.123" as can be seen in figure 3.1.3a below.

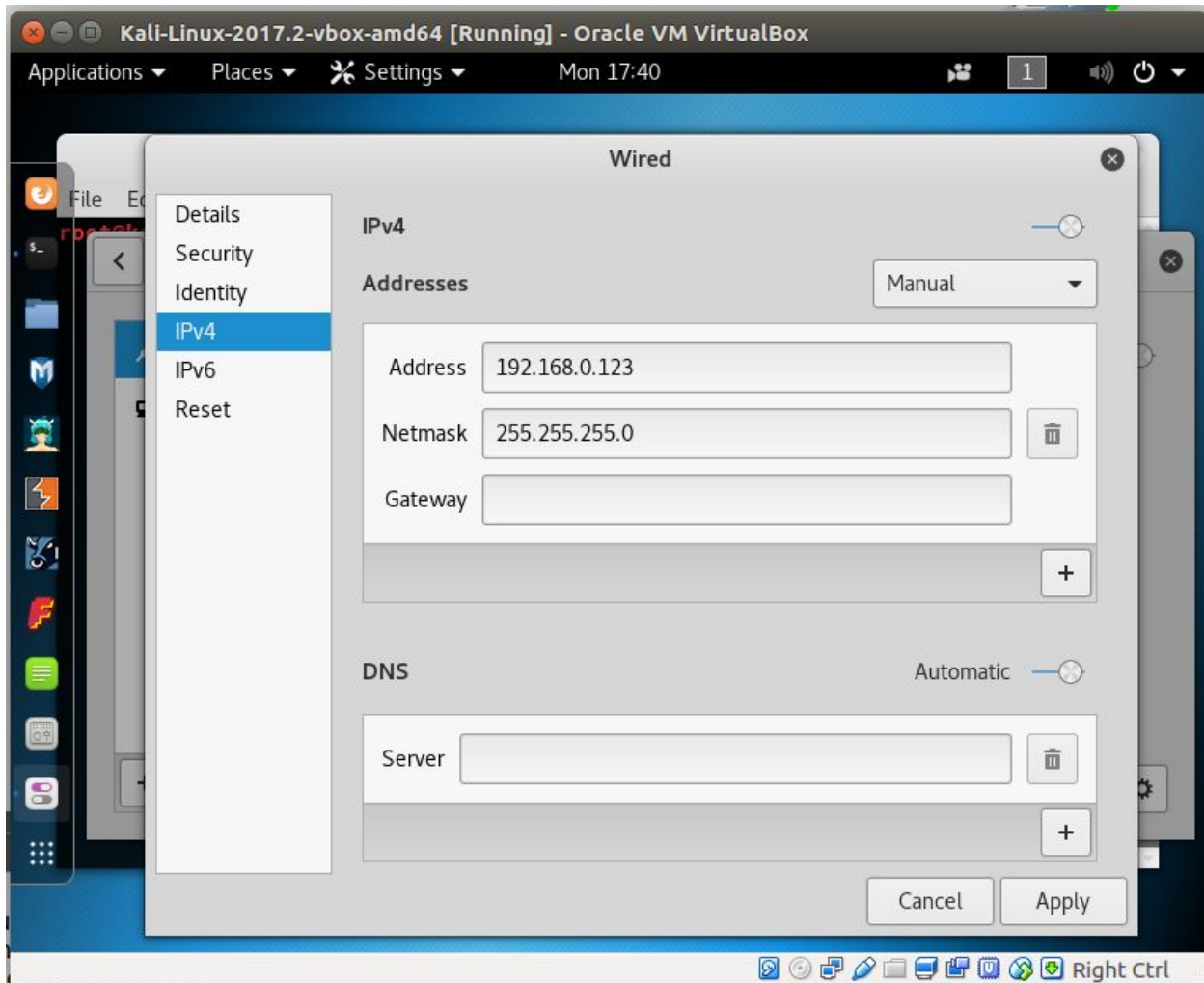
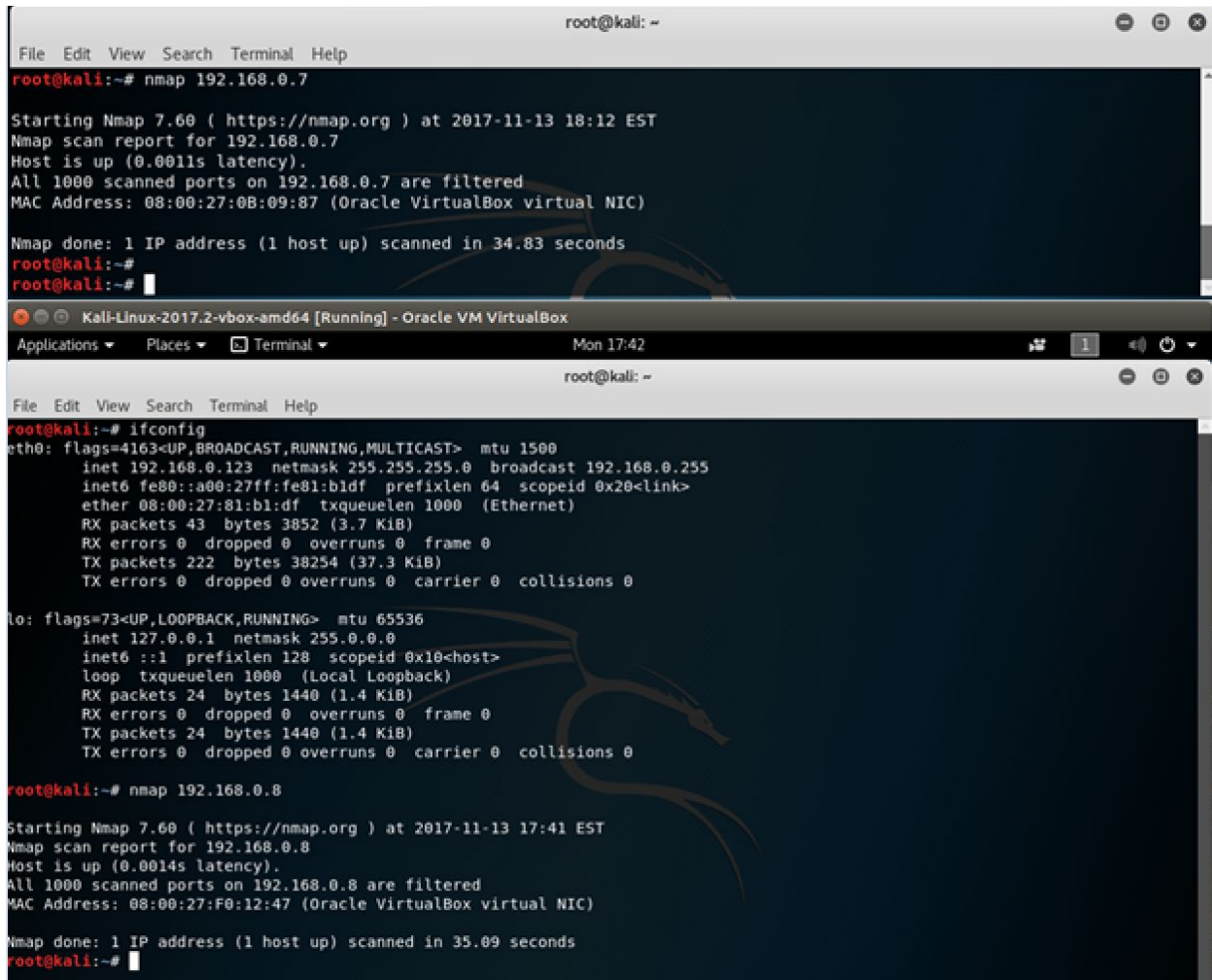


Figure 3.1.3a - Kali Static IP

In order to confirm that the IP addresses have been set properly, an NMAP scan was performed from Kali. The devices should appear as 'up' if they work with the corresponding IP address, this is slightly different from kali which would not be able to scan the network at all if its IP address was not set. As can be seen in figure 3.1.3b below, all IP addresses were as expected.



```
root@kali: ~  
File Edit View Search Terminal Help  
root@kali:~# nmap 192.168.0.7  
Starting Nmap 7.60 ( https://nmap.org ) at 2017-11-13 18:12 EST  
Nmap scan report for 192.168.0.7  
Host is up (0.0011s latency).  
All 1000 scanned ports on 192.168.0.7 are filtered  
MAC Address: 08:00:27:0B:09:87 (Oracle VirtualBox virtual NIC)  
Nmap done: 1 IP address (1 host up) scanned in 34.83 seconds  
root@kali:~#  
root@kali:~#  
Kali-Linux-2017.2-vbox-amd64 [Running] - Oracle VM VirtualBox  
Applications Places Terminal Mon 17:42  
root@kali: ~  
File Edit View Search Terminal Help  
root@kali:~# ifconfig  
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500  
    inet 192.168.0.123 netmask 255.255.255.0 broadcast 192.168.0.255  
    inet6 fe80::a00:27ff:fe81:b1df prefixlen 64 scopeid 0x20<link>  
    ether 08:00:27:81:b1:df txqueuelen 1000 (Ethernet)  
    RX packets 43 bytes 3852 (3.7 KiB)  
    RX errors 0 dropped 0 overruns 0 frame 0  
    TX packets 222 bytes 38254 (37.3 KiB)  
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0  
lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536  
    inet 127.0.0.1 netmask 255.0.0.0  
    inet6 ::1 prefixlen 128 scopeid 0x10<host>  
    loop txqueuelen 1000 (Local Loopback)  
    RX packets 24 bytes 1440 (1.4 KiB)  
    RX errors 0 dropped 0 overruns 0 frame 0  
    TX packets 24 bytes 1440 (1.4 KiB)  
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0  
root@kali:~# nmap 192.168.0.8  
Starting Nmap 7.60 ( https://nmap.org ) at 2017-11-13 17:41 EST  
Nmap scan report for 192.168.0.8  
Host is up (0.0014s latency).  
All 1000 scanned ports on 192.168.0.8 are filtered  
MAC Address: 08:00:27:F0:12:47 (Oracle VirtualBox virtual NIC)  
Nmap done: 1 IP address (1 host up) scanned in 35.09 seconds  
root@kali:~#
```

Figure 3.1.3b - IP Addresses Set (Confirmation)

### 3.2 CONFIGURING SNMP

On Windows, configuring SNMP is rather easy – especially on the agent side. To setup an agent the SNMP and SNMP Trap services must be enabled within services which can be found under computer management. Shown below in figure 3.2a are the enabled services on Windows 7; setup on Windows Server 2008 is very similar and thus will not be shown.

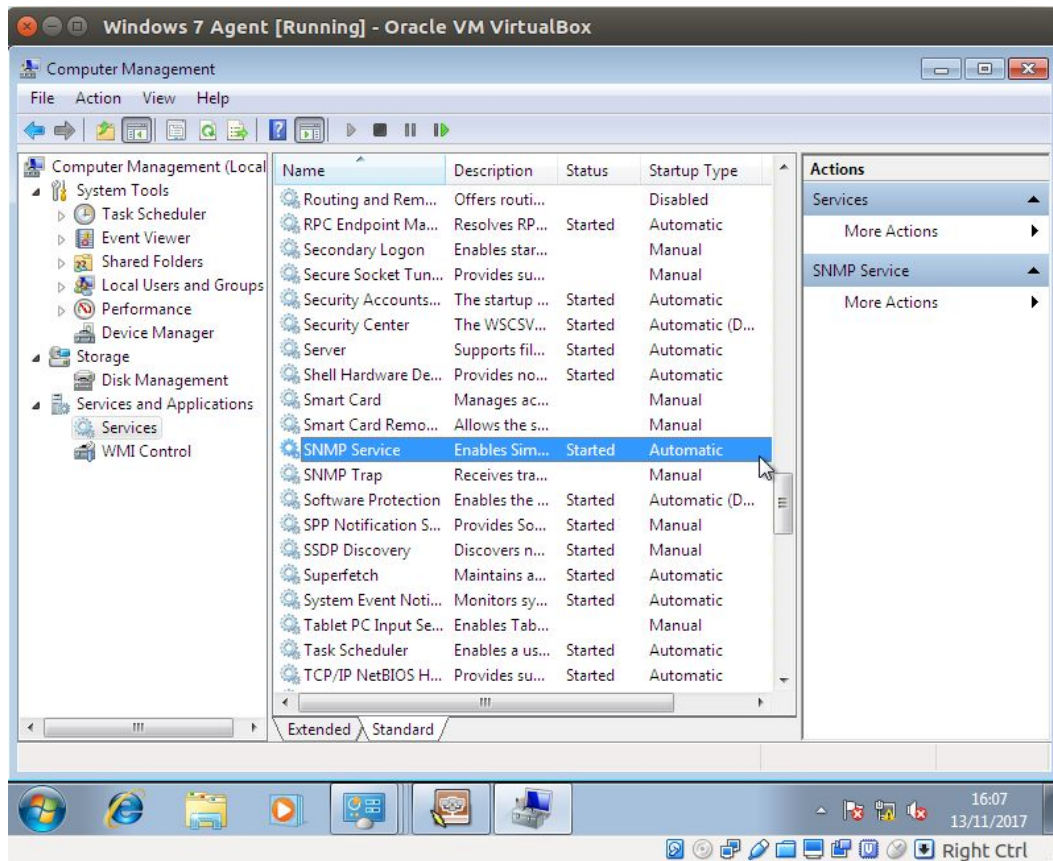


Figure 3.2a - Enabling SNMP in services

Next on the agent, the communities and IP addresses to accept SNMP traffic from must be configured. The default configuration for this is “public” for the read only community, and “private” for the read/write community. The default configuration is incredible insecure but as discussed, some users just want something that works and will not update it. The default configuration can be seen in figure 3.2b below.

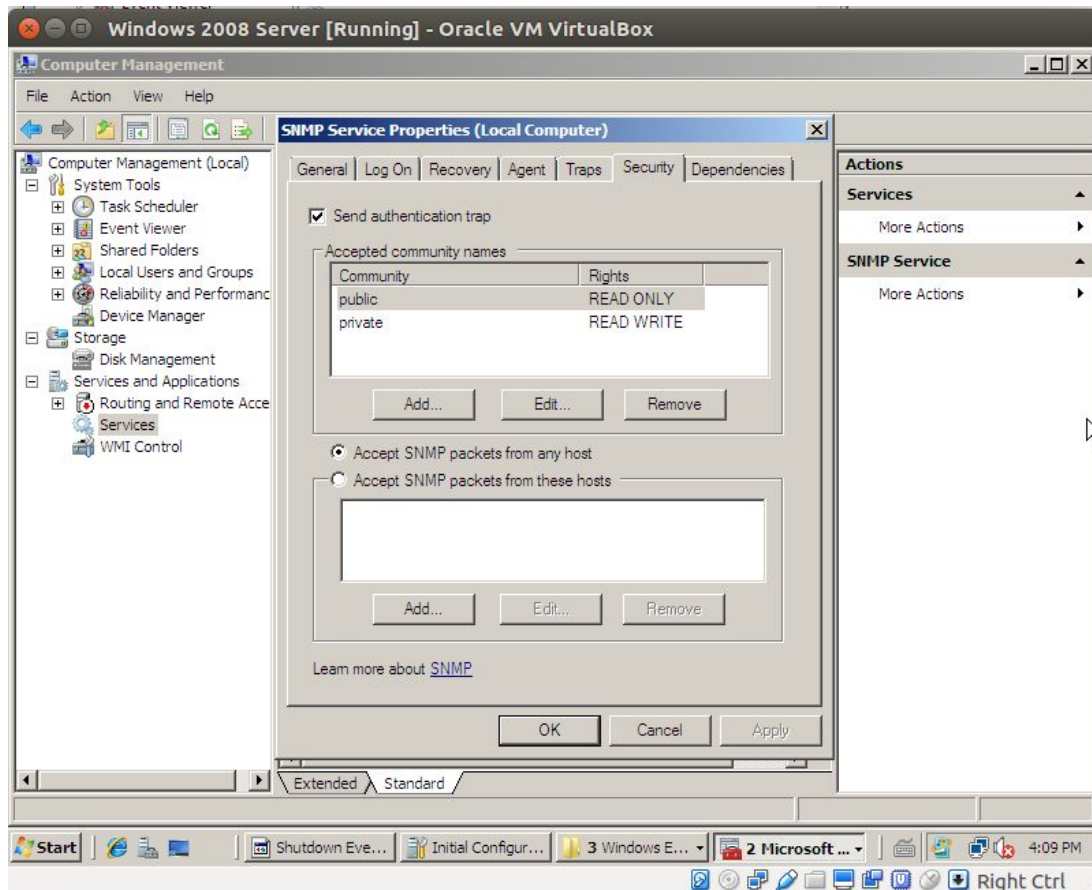


Figure 3.2b - Default SNMP Configuration

Then on the acting NMS, the agent (192.168.0.8 in this case) has to be added to the Network Managers (PowerSNMP) list of known agents by inputting the agent's IP and its community octet strings. A query of uptime to prove it is working as expected can be seen in figure 3.2c below.

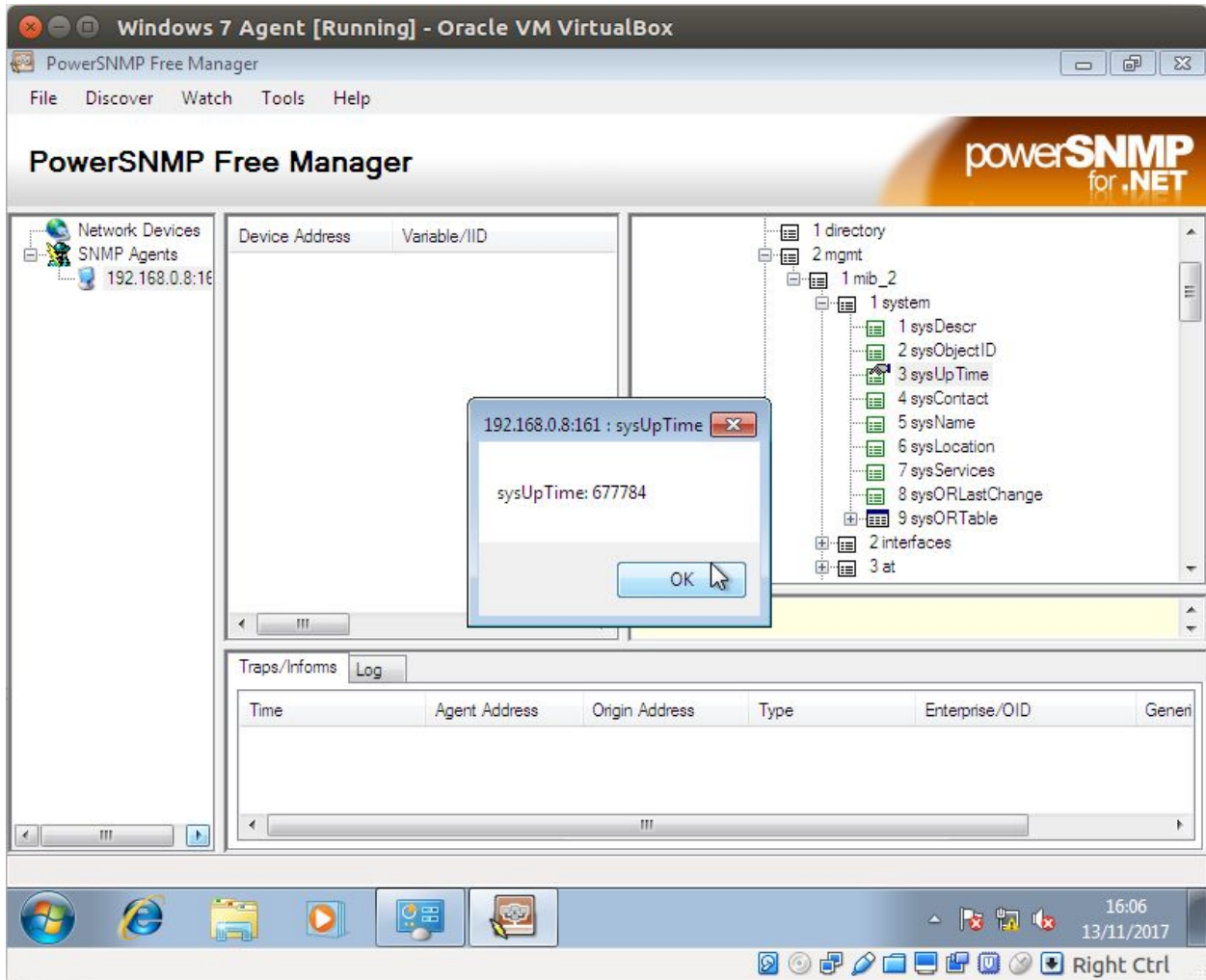


Figure 3.2c - Query Uptime

### 3.3 ATTACKING SNMP

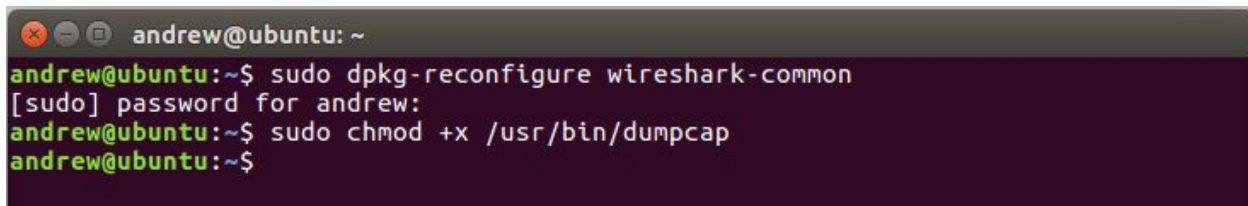
As touched on earlier there are several ways to exploit SNMP, this section will demonstrate how an attacker could exploit the way SNMPv2c works to gain access to its information. There are other ways to conduct these attacks and in a non-simulated network there would likely be extra steps however, those extra steps would not rely on the weakness of SNMP, but rather flaws with other systems and protocols. As this is not the goal of this investigation these will only be commented on and absolutely will not be tested.

Since SNMPv2x has been setup and tests have shown it is working as expected the exploitation of its vulnerabilities may begin.

#### 3.3.1 INFORMATION DISCLOSURE/MAN IN THE MIDDLE

If this was a real network and not a virtualized network, when performing a man in the middle attack, the attacker would likely be positioned between the devices they were observing traffic from. However, since this network is virtualized some problems are presented; it is not possible to directly intercept the traffic. If the virtualized environment included a router it may be possible to conduct DNS poisoning/spoofing to trick the router into sending its traffic via the attacker. However, since this investigation focuses on SNMP vulnerabilities and not general networking vulnerabilities this step will be skipped.

In order to demonstrate what an attacker could see if set up as a man in the middle I had the GNS3 host run Wireshark on the connection between the switch and the Windows 2008 Server. By default Wireshark cannot run within other programs, to resolve this Wireshark must be allowed capture packets on non-sudo accounts. The commands required to make this change can be seen in figure 3.3.1a below.

A terminal window with a dark background and light text. The window title is 'andrew@ubuntu: ~'. The terminal shows the following commands and output:

```
andrew@ubuntu:~$ sudo dpkg-reconfigure wireshark-common
[sudo] password for andrew:
andrew@ubuntu:~$ sudo chmod +x /usr/bin/dumpcap
andrew@ubuntu:~$
```

Figure 3.3.1a - Configure Wireshark for GNS3

In the early example where system uptime was being collected, our attacker was actually monitoring the traffic. Within the packet capture -which can be seen below in FIGURE X, the SNMP *get-request* as well as the *get-response* containing the read/write community string and the system uptime can be seen. As previously discussed SNMPv2c transmits the community string in plain text and thus the hacker has been able to obtain it by performing this man in the middle attack.

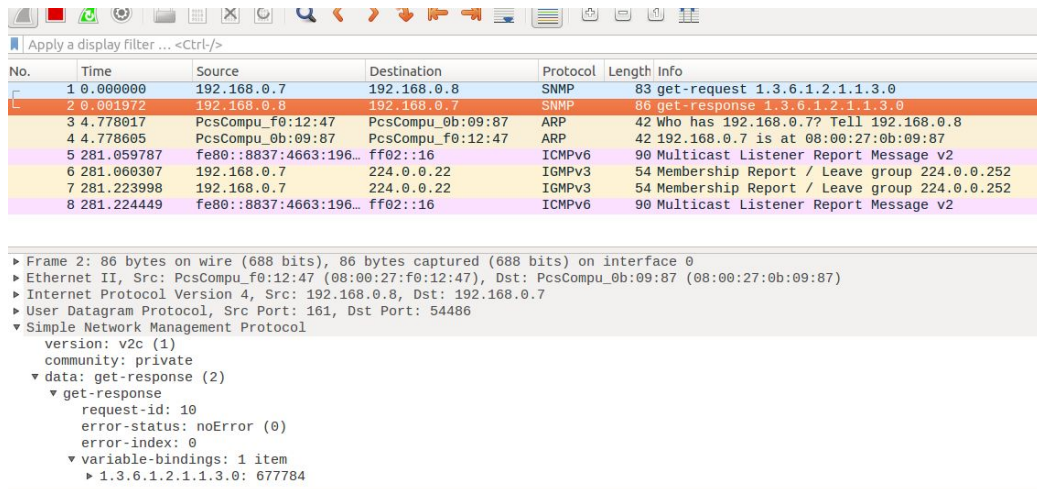


Figure 3.3.1b - Man in The Middle (MITM) Attack

### 3.3.2 CRACKING THE COMMUNITY STRING

The man in the middle attack relied on there being communication between the manager and the agent. If there was no communication between the NMS and the agent then it would not be possible for the MITM attack to obtain the information it did. Luckily for the attacker there is another way to get the community string – cracking it!

For this attack we will assume the community strings are slightly more secure, and that the hosts to accept SNMP packets from has been set specifically to the NMS (192.168.0.7). It would otherwise be even more trivial to conduct the attack. In figure 3.3.2a below the *securer* SNMP configuration can be seen.

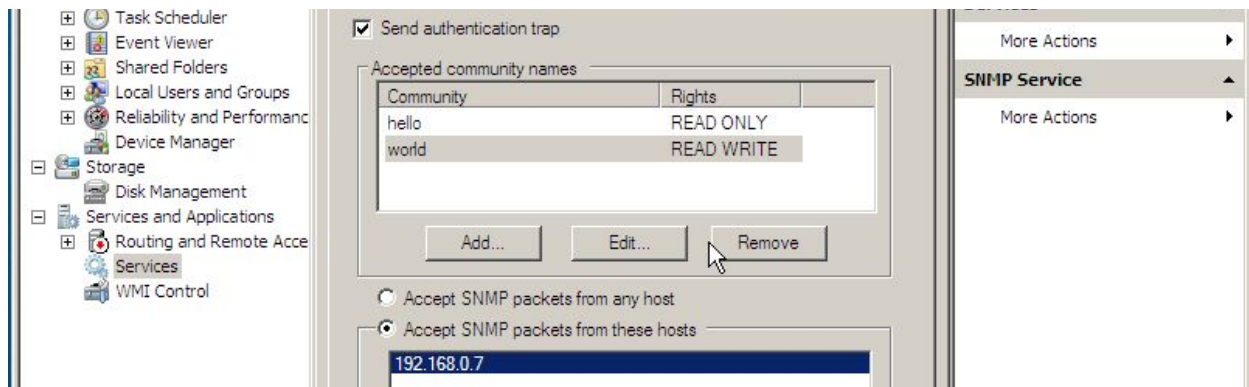


Figure 3.3.2a - 'Securer' SNMP Configuration

Assuming our attacker can conduct DNS poisoning/spoofing as mentioned earlier, the Agent will not realize that the attacker is any different from the NMS. In this particular example the attackers IP has been set to mimic the NMS' IP (192.168.0.7) by configuring the IP on the 'Eth0' interface. Using Hydra (*Kali.org*) – a parallelized login cracker- it is still trivial to attack SNMPv2c. By supplying hydra with a dictionary file (a list of common passwords) made specifically for SNMP (*github.com/danielmiessler*) it was possible to break this particular

configuration of SNMP in a minute. Figure 3.2.2b below shows the output of the Hydra attack.

```

root@kali: ~
File Edit View Search Terminal Help
root@kali:~# hydra -P ~/Desktop/common-snmp-community-strings.txt 192.168.0.8 snmp
Hydra v8.6 (c) 2017 by van Hauser/THC - Please do not use in military or secret service organizations, or for illegal purposes.

Hydra (http://www.thc.org/thc-hydra) starting at 2017-11-13 19:26:51
[DATA] max 16 tasks per 1 server, overall 16 tasks, 118 login tries (l:1/p:118), ~8 tries per task
[DATA] attacking snmp://192.168.0.8:161/
[161][snmp] host: 192.168.0.8 password: hello
[161][snmp] host: 192.168.0.8 password: world
[STATUS] 118.00 tries/min, 118 tries in 00:01h, 1 to do in 00:01h, 4 active
1 of 1 target successfully completed, 2 valid passwords found
Hydra (http://www.thc.org/thc-hydra) finished at 2017-11-13 19:27:55
root@kali:~#

```

Figure 3.2.2b - Hydra Attack

The moment when Hydra successfully guessed the read/write community string is shown in figure 3.2.2c.

748	56.104981	192.168.0.8	192.168.0.7	SNMP	83	get-response	1.3.6.1.2.1.1.1
749	276.520803	fe80::8837:4663:196...	ff02::1:2	DHCPv6	150	Solicit	XID: 0x96ae0b CID: 00010001219063800800270b0987
750	277.517188	fe80::8837:4663:196...	ff02::1:2	DHCPv6	150	Solicit	XID: 0x96ae0b CID: 00010001219063800800270b0987
751	279.520679	fe80::8837:4663:196...	ff02::1:2	DHCPv6	150	Solicit	XID: 0x96ae0b CID: 00010001219063800800270b0987
752	283.525576	fe80::8837:4663:196...	ff02::1:2	DHCPv6	150	Solicit	XID: 0x96ae0b CID: 00010001219063800800270b0987
753	291.526974	fe80::8837:4663:196...	ff02::1:2	DHCPv6	150	Solicit	XID: 0x96ae0b CID: 00010001219063800800270b0987

```

▶ Frame 748: 83 bytes on wire (664 bits), 83 bytes captured (664 bits) on interface 0
▶ Ethernet II, Src: PcsCompu_f0:12:47 (08:00:27:f0:12:47), Dst: PcsCompu_81:b1:df (08:00:27:81:b1:df)
▶ Internet Protocol Version 4, Src: 192.168.0.8, Dst: 192.168.0.7
▶ User Datagram Protocol, Src Port: 161, Dst Port: 34822
▼ Simple Network Management Protocol
  version: version-1 (0)
  community: world
  ▼ data: get-response (2)
    ▼ get-response
      request-id: 442406656
      error-status: noSuchName (2)
      error-index: 1
      ▶ variable-bindings: 1 item

```

Figure 3.2.2c - Hydra Guesses Password Correctly

It is worth noting that whilst the Hydra attack did obtain the password after a minute, the Agent was sending trap after trap – attempting to notify the NMS that someone had attempted to connect with an invalid community string as can be seen in 3.2.2d. This likely would not go unnoticed.

715	50.087411	192.168.0.7	192.168.0.8	SNMP	85	get-request	1.3.6.1.2.1.1.1
716	50.087522	192.168.0.7	192.168.0.8	SNMP	81	get-request	1.3.6.1.2.1.1.1
717	50.087621	192.168.0.7	192.168.0.8	SNMP	87	get-request	1.3.6.1.2.1.1.1
718	50.087716	192.168.0.7	192.168.0.8	SNMP	84	get-request	1.3.6.1.2.1.1.1
719	50.087811	192.168.0.7	192.168.0.8	SNMP	84	get-request	1.3.6.1.2.1.1.1
720	50.088348	192.168.0.8	192.168.0.7	SNMP	90	trap	iso.3.6.1.4.1.311.1.1.3.1.2
721	50.089214	192.168.0.8	192.168.0.7	SNMP	90	trap	iso.3.6.1.4.1.311.1.1.3.1.2
722	50.090112	192.168.0.8	192.168.0.7	SNMP	90	trap	iso.3.6.1.4.1.311.1.1.3.1.2
723	50.090985	192.168.0.8	192.168.0.7	SNMP	90	trap	iso.3.6.1.4.1.311.1.1.3.1.2
724	50.092126	192.168.0.8	192.168.0.7	SNMP	90	trap	iso.3.6.1.4.1.311.1.1.3.1.2

Figure 3.2.2d - Hydra Triggers Trap Security Alerts



## 4. DISCUSSION

Is it really surprising that SNMPv2C is so weak, after all it is nearly 15 years old and it relies on a security model that is approaching 3 decades old. This investigation has proved that SNMPv2C should not be used unless absolutely necessary. The researchers who first implemented SNMP can likely be forgiven for their negligence, SNMPv2C on the other hand should never have existed.

While some liberties were taken in regard to the hacker's access to the virtual network, the final results are similar to what could occur in a real network. Beyond ensuring that the hacker is not able to breach the internal network in the first place, there are not many positive mitigations to the issues SNMPv2C presents.

Despite a clear lack of security, SNMPv2C isn't the main weak point in SNMP security. That title goes to vendor implementations of SNMP.

### 4.1 VENDOR IMPLEMENTATION

Vendor implementation is a threat to all versions of SNMP. While vendor implementations were particularly bad in SNMPv2C and below, SNMPv3 is still affected. There are many, many examples of poor implementations causing significant issues.

Default communities are the worst of all vendor introduced vulnerabilities. A default community becomes a vulnerability when a vendor doesn't properly secure their devices by either not changing SNMP communities from the SNMP defaults or using their own community that is hard-coded (cannot be changed) on the device. Hard coded community strings are a very big issue on IOT devices; manufacturers may be using SNMP to gather statistics from their devices as opposed to manual collection as reported by (Rowe, K, 2014).

Another common issue introduced by vendors is excessive information disclosure through poor MIB design. In January 2002 a *seclists* user reported that their D-Link router would report the router password if queried (*seclists.org*). As the router also had a default community issue, an attacker could potentially gain access to the internal network if they could externally request the password from the router. The attack would look something along the lines of this:

```
Attacker-PC# snmpwalk 192.168.0.10 public enterprises.937.2.1.2.2
enterprises.937.2.1.2.2.0 = "mypw"
```

If instead a set was used instead of a get (*snmpwalk* uses get by default) it is possible that the attacker may have been able to change the password to that user's own network.

By looking through an MIB depot for specific terms other potentially vulnerable designs may be identified. An attacker could limit their search to a particular vendor or products and likely find something exploitable quite easily.

An area implementers seem to be particularly bad with is error handling, a quick search for "SNMP buffer overflow" will reveal hundreds of CVEs. A particularly notable one is the Solaris snmpXdmid buffer overflow (exploit-db.com). The underlying issue is that many SNMP

implementations rely on the groundwork of others. NET-SNMP is one of the most common implementations and is absolutely riddled with vulnerabilities (*cvedetails.com*).

## 4.2 MITIGATIONS

The easiest way to mitigate the threats presented by SNMP is to minimize usage; unless absolutely critical SNMP should not be used and if it has to be used then SNMPv3 is the least terrible option. By disabling set-type requests it is possible to prevent anyone from using SNMP itself to tamper with a network.

When setting up SNMP the community strings should never be left as default, as proven in the D-Link example a default community string can make a bad vulnerability even worse. Eliminating default strings is also one of the easiest mitigations to implement.

If ingress and egress filtering are not already set up then they should be. Ingress filtering allows a firewall (or any device that supports it) to check that incoming packets are actually from the source claimed. Egress filtering is restricting the types of traffic that can come from one network to another. Ingress filtering is definitely less situational than egress which would rely on there being no SNMP traffic from (for example) external sources (*experts-exchange.com*). In that example it would not be possible for a remote admin to query SNMP devices within that network but it would also not be possible for an attacker to attempt the same.

If using SNMPv3 then it is much the same, except to make use of the *new* security features; there is an option for encryption and authentication, there is no reason not to use it – so use it.

## 5. CONCLUSION

In conclusion, despite the many promised features of SNMP a lot is left to be desired. None of the current releases are perfect and all the releases suffer from damaging oversights. While things have improved with SNMPv3 it still has some major drawbacks, although it is the most secure version of SNMP it should not be considered secure. The protocol itself is only part of the issue though. Implementers and vendors are equally to blame. Whether it be default communities or just downright poor implementations, they are the ones who can make a bad vulnerability even worse.

Another part of the problem is that it is so convenient for larger organizations to remain on their current versions. Upgrades cause downtime and are hard to justify beyond "But security!". There likely will not be major pushes for upgrades until some catastrophic risk has the market demanding change.

All in all, things are improving between versions. Hopefully by SNMPv4 things have improved, but until then the market is unlikely to budge.

## 6. TABLE OF TERMS

TERM	MEANING
AGENT	software or service that responds to requests from, and sends traps to a manager
MANAGER /NMS	Software/service that gets and sets information on agents
PDU	Protocol Data Unit - used for communication between managers and agents
OID	Object Identifier
MIB	Management Information Base
VM	Virtual Machine

**7. APPENDIX A - SNMPv1 PDUs**

<i>Manager Messages</i>		<i>Agent Messages</i>	
<b>Message</b>	<b>PDU</b>	<b>Message</b>	<b>PDU</b>
<i>GetRequest</i>	0	<i>GetResponse</i>	2
<i>GetNextRequest</i>	1	<i>Trap</i>	4
<i>SetRequest</i>	3		

## 8. APPENDIX B – SNMPv3 SPECIFICATIONS (*IEFT.ORG/RFC*)

RFC	YEAR	SUMMARY
<b>2576</b>	2000	Coexistence between Version 1, Version 2, and Version 3 of the Internet-standard Network Management Framework
<b>2578</b>	2000	Structure of Management Information Version 2 (SMIV2)
<b>2579</b>	2000	Textual Conventions for SMIV2
<b>2580</b>	2000	Conformance Statements for SMIV2
<b>3410</b>	2002	Introduction and Applicability Statements for Internet Standard Management Framework
<b>3411</b>	2002	An Architecture for Describing Simple Network Management Protocol (SNMP) Management Frameworks
<b>3412</b>	2002	Message Processing and Dispatching for the Simple Network Management Protocol (SNMP)
<b>3413</b>	2002	Simple Network Management Protocol (SNMP) Applications
<b>3414</b>	2002	User-based Security Model (USM) for version 3 of the Simple Network Management Protocol (SNMPv3)
<b>3415</b>	2002	View-based Access Control Model (VACM) for the Simple Network Management Protocol (SNMP)
<b>3416</b>	2002	Version 2 of the Protocol Operations for the Simple Network Management Protocol (SNMP)
<b>3417</b>	2002	Transport Mappings for the Simple Network Management Protocol (SNMP)
<b>3418</b>	2002	Management Information Base (MIB) for the Simple Network Management Protocol (SNMP)

## 9. References

Microsoft TechNet (2003) What Is SNMP?. [ONLINE] Available at:

<https://technet.microsoft.com/en-us/library/cc776379>. [Accessed 18 October 2017]

Etingof (2017). SNMP history — SNMP library for Python 4.4 documentation. [online] Snmplabs.com. Available at:

<http://snmplabs.com/pysnmp/docs/snmp-history.html> [Accessed 19 Oct. 2017].

US-CERT. (2017). CA-2003-06. [online] Available at: <https://www.cert.org/historical/advisories/CA-2003-06.cfm?>

[Accessed 5 Nov. 2017].

Zobel, D. (2010). How do SNMP, MIBs and OIDs work? | Paessler Knowledge Base. [online] Kb.paessler.com.

Available at: <https://kb.paessler.com/en/topic/653-how-do-snmp-mibs-and-oids-work> [Accessed 14 Nov. 2017].

K. McCloghrie (1986). RFC 1213 - Management Information Base for Network Management of TCP/IP-based

internets: MIB-II. [online] Tools.ietf.org. Available at: <https://tools.ietf.org/html/rfc1213> [Accessed 14 Nov. 2017].

Ellingwood, J. (2017). An Introduction to SNMP (Simple Network Management Protocol) | DigitalOcean. [online]

Digitalocean.com. Available at:

<https://www.digitalocean.com/community/tutorials/an-introduction-to-snmp-simple-network-management-protoco>

l [Accessed 14 Nov. 2017].

Mauro, D. and Schmidt, K. (2005). Essential SNMP. 2nd ed. O'REILLY, p.25, p26.

Technet.microsoft.com. (2012). Features Removed or Deprecated in Windows Server 2012. [online] Available at:

<https://technet.microsoft.com/en-us/library/hh831568.aspx> [Accessed 9 Oct. 2017].

Hoffman, P. (2012). The Tao of IETF: A Novice's Guide to the Internet Engineering Task Force. [online] Ietf.org.

Available at: <https://www.ietf.org/tao.html#anchor45> [Accessed 14 Nov. 2017].

Rose, M. (1991). RFC1212. [online] Ietf.org. Available at: <https://www.ietf.org/rfc/rfc1212> [Accessed 14 Nov. 2017].

Rose, M. (1991). RFC1155. [online] Ietf.org. Available at: <https://www.ietf.org/rfc/rfc1155> [Accessed 14 Nov. 2017].

Case, J. (1990). RFC1157. [online] Ietf.org. Available at: <https://www.ietf.org/rfc/rfc1157> [Accessed 14 Nov. 2017].

Rose, M. (1991). RFC1158. [online] Ietf.org. Available at: <https://www.ietf.org/rfc/rfc1158> [Accessed 14 Nov. 2017].

McCloghrie, K. (1991). RFC2578. [online] Ietf.org. Available at: <https://www.ietf.org/rfc/RFC2578> [Accessed 14 Nov. 2017].

Comptechdoc.org. (n.d.). Simple Network Management Protocol. [online] Available at: <http://www.comptechdoc.org/independent/networking/guide/netsnmp.html> [Accessed 14 Nov. 2017].

Baker, F. (1995). RFC 1812. [online] Ietf.org. Available at: <https://www.ietf.org/rfc/rfc1812> [Accessed 14 Nov. 2017].

EFF.org (1999) EFF DES Cracker Project [online] Available at: <https://web.archive.org/web/19990506023227/https://www.eff.org/descracker.html> [Accessed 14 Nov. 2017]

Blaze, M, Diffie, W, Rivest, B, Schneier, B, Shimomura, T, Thompson, E, Weiner, M (1996) Minimal Key Lengths for Symmetric Ciphers to Provide Adequate Commercial Security [online] Available at: <https://web.archive.org/web/20011005180016/http://www.counterpane.com/keylength.html>

Case, J. (1996). RFC 1901. [online] Ietf.org. Available at: <https://www.ietf.org/rfc/rfc1901> [Accessed 14 Nov. 2017].

Blumenthal, U & Wijnen, B (2002). RFC 3414 . [online] Ietf.org. Available at: <https://www.ietf.org/rfc/rfc3414> [Accessed 14 Nov. 2017].

GNS3.com (2017) GNS3 [online] Available at: <https://www.gns3.com> [Accessed 14 Nov. 2017]

Kali.org (2014) THC-Hydra [online] Available at: <https://tools.kali.org/password-attacks/hydra> [Accessed 14 Nov. 2017]

Rowe, K (2014) Internet of Things requirements and protocols [online] Available at: <http://www.embedded-computing.com/embedded-computing-design/internet-of-things-requirements-and-protocols> [Accessed 14 Nov. 2017]



experts-exchange.com (2009) Understanding Ingress and Egress Traffic [online] Available at:

<https://www.experts-exchange.com/questions/24875642/Understanding-Ingress-and-Egress-Traffic.html> [Accessed 14 Nov. 2017]

OTHER RFC PAGE [online] RFC # Available at: <https://www.ietf.org/rfc/rfc#>