Interested in learning more about security?

SANS Institute InfoSec Reading Room

SNMP and Potential ASN.1 Vulnerabilities

Earlier this year a number of vulnerabilities in the Simple Network Management Protocol (SNMP) were publicized by the University of Oulu Secure Programming Group. This paper briefly describes the SNMP protocol, with emphasis on the underlying ASN.1 notation, discusses the vulnerabilities identified by Oulu and demonstrates the Oulu Protos SNMP testing tool. A number of protocols critical to the secure use of the Internet, such as SSL/TLS, S/MIME, Kerberos, LDAP and H.323 also rely on ASN.1 and the potential for further...

SNMP and Potential ASN.1 Vulnerabilities

Dr. Edmund Whelan, CISSP

December 2002

Abstract

Earlier this year a number of vulnerabilities in the Simple Network Management Protocol (SNMP) were publicized by the University of Oulu Secure Programming Group. This paper briefly describes the SNMP protocol, with emphasis on the underlying ASN.1 notation, discusses the vulnerabilities identified by Oulu and demonstrates the Oulu Protos SNMP testing tool. A number of protocols critical to the secure use of the Internet, such as SSL/TLS, S/MIME, Kerberos, LDAP and H.323 also rely on ASN.1 and the potential for further, more serious and less easily addressed vulnerabilities within such protocols is also discussed. These protocols are considered to be potentially at risk and it is noted that a large scale, successful attack on a protocol such as SSL/TLS would damage the credibility of the Internet as a secure place to do business and would discourage a large number of corporations who currently see the Internet as a core part of their business strategy. On the evidence currently available it seems that the underlying ASN.1 standard itself is not primarily at fault. Rather the ASN.1 encoders and decoders do not seem to handle malformed encodings robustly. It would be expected that this may allow such vulnerabilities to be successfully addressed, hopefully before large scale attacks can be launched. The complexity of ASN.1 may hinder this as could the possibility of a single attack vector, such as a worm, being deployed to target similar underlying ASN.1 vulnerabilities in a number of protocols simultaneously.

This paper was written in part fulfilment of the SANS GSEC requirements.

Table of Contents

1. Introduction

Earlier this year a number of issues with the Simple Network Management Protocol (SNMP) [RFC1157] were highlighted by the University of Oulu Secure Programming Group [OSPG]. This led to the release of a CERT vulnerability alert [CA0203] and a flurry of activity by vendors to release patches to address the issues highlighted. Following this initial activity there has been, despite little press attention, a sustained rumble within the IT security industry with concerns being voiced that the issues raised by Oulu are not solely related to SNMP. As ASN.1 is a fundamental part of a number of widely used protocols there is concern that these too may be susceptible to the same kind of issues and, whereas SNMP could be filtered at an organization's firewall, many other potentially vulnerable protocols would be much harder to protect and have a much more detrimental effect on the Internet as a whole were a successful attack to take place.

2. Simple Network Management Protocol (SNMP)

2.1. History of SNMP

In 1988, the Internet Architecture Board [IAB] recommended in RFC1052 [RFC1052] that all TCP/IP implementations be network manageable and determined a strategy of using the already defined SNMP [SNMP88] in the short term and moving to the OSI network management framework, which was not yet a full standard, in the longer term. Several standards were prepared by the Management Information Base (MIB) working group of the Internet Engineering Task Force [IETF] to define the management information, including:

- RFC1065 which defined the Structure of Management Information (SMI)
- RFC1066 which defined the Management Information Base (MIB)

These two standards were designed to be compatible with both the SNMP and OSI management frameworks. Of particular interest for this current paper was the decision taken at the time to base the management information protocols on a subset of ASN.1 as this was the favoured and successful notation used within a number of OSI standards.

However, it was found that designing a management structure compatible with both frameworks was more difficult than expected and the requirement for compatibility with OSI was dropped [RFC1109]. The SMI and MIB documents were altered and the recommended management framework became based around the following:

- RFC1155 Structure and Identification of Management Information for TCP/IP
- RFC1156 Management Information Base for Network Management of TCP/IP
- RFC1157 Simple Network Management Protocol (SNMP)

RFC1157 was the output of the SNMP Extensions working group and changed the original SNMP definition to keep in step with the changes in the MIB. SNMP was based upon an earlier protocol called the "Simple Gateway Monitoring Protocol (SGMP)" [RFC1028]. However, since its inception in RFC 1067 in August 1988, SNMP was not backwardly compatible with SGMP and so new UDP ports had been assigned to avoid confusion.

In addition, RFC1215 [RFC1215] defines the use of Traps within SNMP. Since the original RFCs, SNMP has undergone a number of revisions, the latest version being SNMPv3 with earlier versions being SNMPv1 and SNMPv2. The changes have, in general, been minor and the underlying framework has remained the same. One significant difference is that version 3 includes more sophisticated mechanisms for authentication etc. However, SNMPv1 is still widely used within the Internet community.

2.2. Overview of SNMP

SNMP is an application level management protocol. This has the advantage that it can be used without regard to the underlying network hardware and the subsequent use of one set of protocols is desirable from a management perspective as all devices will respond to the same set of commands. This is in contrast to earlier link-level management protocols which differed depending on the device. There are disadvantages in that, unless the operating system, IP software and transport protocol are performing correctly it may not be possible to contact a router in order to manage it. Nevertheless, SNMP has worked very well in practice.

The architecture of SNMP is simple, with a network being seen as consisting of:

- Network Management Stations (NMS)
  - These execute management applications to monitor and control network elements
- Network Elements
  - These are devices such as hosts, gateways, switches etc and have management agents responsible for performing the management functions requested by the NMS.

SNMP is used for the communication between the NMS and the agents. The agents only alter or inspect variables and, therefore, the NMS uses "set" and "get" calls only. The network is polled on a regular basis and a number of unsolicited messages from the agents are allowed, known as "traps", which guide the timing and focus of the polling. Traps can be set up so they are sent depending on the condition of an interface for example. Although seemingly inflexible, a number of more complex commands can be executed on network devices by the use of simple "set" commands. An example of this would be rebooting a device by changing the "number of seconds to reboot" variable on a device. Moreover, UDP [RFC768] is generally used to minimize complexity which has the effect that each message must be represented by a single transport datagram. Other protocols could also be used but UDP is by far the most common.

SNMP divides the management protocol into two distinct parts and specifies separate standards for each. These parts are concerned with:

1) The data being managed
   - The standard defines the data which a managed device must maintain as well as how the data is identified.
2) Communication of information
   - The protocol defines the communication between the software running on the management station and the agent, including the specification of messages as well as the format of the names and addresses.

2.3. Management Information Base (MIB)

SNMP does not specify exactly what information can be accessed on which devices. Rather, the Management Information Base (MIB) standard defines which data items a device must keep, together with the operations allowed on each. Even if a device does not have information about MIB items defined in newer MIB versions, the fact that all devices communicate using the same protocol means that all devices can parse a query for that item and either provide the information or send an error message explaining that they do not have the item.

SNMPv1 and SNMPv2 collated variables into a single large MIB documented in a single standard. However, once the second version was produced (MIB-II) the IETF allowed the publication of many individual MIB documents each specifying the MIB for a specific type of device. Thus there are now RFCs which specify MIB variables for devices such as routers, switches and modems, as well as vendor-specific MIBs.

Each MIB variable is held within a category, such as system, interfaces, ip and tcp all of which contain related items, with each category being identified by a specific identifier. Examples of such objects are:

- system
  - sysUpTime – the time since last reboot
- interfaces
  - ifNumber – the number of interfaces
  - ifMtu – the MTU of a particular interface
- ip
  - ipDefaultTTL – the value used by IP in the time-to-live field
  - ipInReceives – the number of datagrams received
  - ipFragOKs – the number of datagrams fragmented
  - ipRoutingTable – the IP Routing Table

2.4. Structure of Management Information (SMI)

As well as the definitions of management information contained in the MIB, the SMI standard defines the rules used to define and identify these variables. This restricts the type of variables allowed in the MIB, specifies the rules for naming these variables and creates rules for defining the variable types. For example, *ipAddress* is defined as a 4-octet string, *counter* is defined as an integer between 0 and 2^{32}-1 and *ipRoutingTable* is defined as a table.

The SMI standard specifies that all MIB variables must be defined using ISO's Abstract Syntax Notation 1 [ASN.1] which is a formal language allowing a human readable form as well as a compact encoded representation which can be used in communication protocols and prevents any ambiguity in the form or content of any variable. The use of ASN.1 was in part due to earlier success with the use of ASN.1 in SGMP, the predecessor to SNMP, and partly because there was originally a requirement to ease eventual transition to OSI based network management protocols. A slightly more complex subset of ASN.1 than that of SGMP is used in SNMP to define the managed objects and the protocol data units (PDUs) used for managing the objects, but there are a number of restrictions. Notably, SNMP uses only a subset of the ASN.1 Basic Encoding Rules [BER]. Notably: the definite length form is used; non-constructor encodings rather than constructor encodings are used whenever possible; and the restrictions are placed on all aspects of the protocol.

ASN.1 is an ITU standard and can be found on their website [ITU]. The ASN.1 standards used in SNMP are the X.208 (ASN.1) and X.209 (BER) ITU-T standards. There are newer versions (X680-X693) which were approved in August 2002 and have been pre-published. However, current protocols generally use the older standards. The ITU do not make their standards freely available but, rather, charge for them. At the time of writing, it was possible to download up to three ITU standards free of charge and, notably, the latest ASN.1 standards could be downloaded as a single item containing all the related standards (X.680-X.693). These latter standards are currently also available free from [ITUSG].

Names used for MIB variables are taken from the *object identifier (OID)* namespace administered by the ISO/ITU and which is used to unambiguously identify items with globally unique identifiers. In order to be globally unique, each identifier is structured, with authority for various branches being delegated to different organisations in much the same way as the Internet namespace. Each branch is assigned both a number, for encoded representation of the names, and a short text string which is used for human understanding. The IAB obtained use of a sub-branch of the US Department of Defence namespace as indicated in Figure 1. It should be noted that a number of vendors, including Cisco, Cabletron and IBM, use differing namespaces such as that below private (1.3.6.1.4).

An example is the *ipAddrTable* variable under the *ip* subtree (4). This has the prefix *"iso.org.dod.internet.mgmt.mib.ip.ipAddrtable"* or, equivalently, *"1.3.6.1.2.1.4.20"* as the ipAddrTable variable has been assigned the identifier 20. This variable is actually defined as an array, with each element being a structure containing five items. This can be written, with text following double hyphens being comments, as:

```
ipAddrTable ::= SEQUENCE OF IpAddrEntry

ipAddrEntry ::= SEQUENCE {
        ipAdEntAddr             ipAddress,          -- IP address
        ipAdEntIfIndex          INTEGER,            -- index of the interface
        ipAdEntNetMask          ipAddress,          -- IP subnet mask
        ipAdEntBcastAddr        ipAddress,          -- IP broadcast address
        ipAdEntReasmMaxSize     INTEGER (0..65535)  – Max datagram size
}
```

With this notation, each element is identified by a descriptive name followed by a declaration of its type, such as an Integer.

When using ASN.1, members of an array are identified by adding a suffix to the end of the ASN.1 identifier. For simple numeric variables the suffix 0 is used but in the case of the *ipAddrTable* the standard defines the suffix used to be an IP address. Hence, to specify the net mask in the address table corresponding to IP address "10.2.1.3" one would write either of the following:

iso.org.dod.internet.mgmt.mib.ip.ipAddrtable.ipAddrEntry.ipAdEntNetMask.10.2.1.3

or

1.3.6.1.2.1.4.20.1.3.10.2.1.3

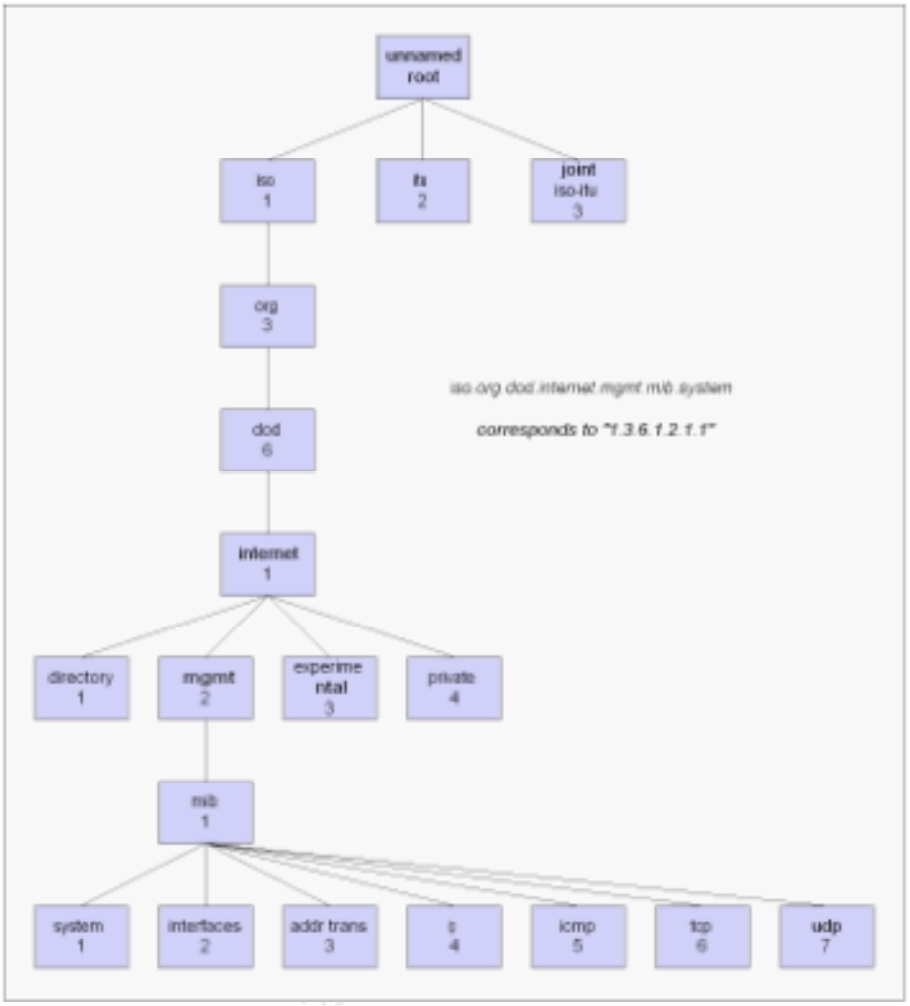

Figure 1: Part of the *mib* variable namespace, under "1.3.6.1.2.1" [1]

2.5. SNMP Message Format

SNMP messages vary depending on the protocol version (1, 2 or 3). Restricting ourselves to the case of SNMPv1, the message format is of the form:

```
SNMPv1Message ::=
        SEQUENCE{
                version         INTEGER (0..2147483647),        -- 1 for SNMPv1
                community       OCTET STRING,                   -- community string
                data            ANY                     -- e.g. PDUs if trivial
                                                        -- authentication is used
        }
```

The PDU data can be, for example, a "get -request", "set-request" or one of a few other related messages. Typical examples can be seen in the Sniffer network captures later in this document.

The ASN.1 Basic Encoding Rules (BER) are mandated but will not be dealt with fully here. There are many full descriptions of ASN.1 apart from the standards already mentioned. Particularly useful resources are [LG], [JL], [HS] and [OSS]. BER is not

[1] Based on a representation by Comer [COMER]

the only method of encoding ASN.1, others include Distinguished Encoding Rules (DER), Packet Encoding Rules (PER) and Lightweight Encoding Rules (LWER). However, SNMP uses a subset of BER for simplicity. BER adopts a "Type, Length, Value" (TLV) notation with each element of the encoding carrying information on the type of the following field, its length and then the actual element value itself. Where the element is itself structured, the value part of the element is itself a series of embedded TLV components and this can be continued until no more elements are structured.

2.6. SNMP Architecture

As mentioned in the Oulu paper [OUSNMP], SNMP is a widely accepted protocol and, even if not used for monitoring and management, is often present in network devices. It has been around for a considerable amount of time and so would be expected to have reached a satisfactory level of robustness. Moreover, successful Denial of Service (DoS) attacks on critical network devices may cause serious issues and problems decoding exceptional BER encodings may cause issues before the completion of the authentication process using community strings. There are some SNMP test suites available such as [IWL,SMPL] but none really tests for robustness.

The SNMP architecture considers a network to consist of a number of Network Management Stations (NMS) and SNMP agents. The NMS communicates with the agents, generally using UDP on port 161 although SNMP can be used over other protocols, for example see [RFC1089, RFC1161 and RFC1298]. The NMS can send "get" commands to retrieve data from the agent or "set" commands to write information to the device's configuration. A minimal level of security is attained by the use of "Community Strings", sometimes known as "Community Names" which are used to provide trivial access control to the agents. There are two types, namely "public" and "private", allowing READ-ONLY access and READ-WRITE access respectively. However, as these are often left at their default settings of "public" and "private" and are, in addition, susceptible to sniffing as they are sent in clear text these offer little real security. SNMPv2 and SNMPv3 do include enhanced security features.

In addition to the NMS issuing "get/set" commands, the agents can send "traps" to the NMS in order to inform it of their current status and/or any changes. These "traps" are sent to UDP port 162 on the NMS. A simplified SNMP architecture is shown in Figure 2.

There are basically three types of agents:

- Normal Agent
  - These accept requests from the NMS and send responses/traps to it
- Master Agent
  - Also known as Extensible Agent and usually transparent to the NMS, these use subagent protocols, such as AgentX, DMI, SMUX and Emanate to talk directly to subagents. For example see [RFC1227, RFC2741]
- Proxy Agents
  - These act as gateways to either bypass application level firewalls or map between SNMP and other management protocols or between SNMP versions. In general, these are visible to the NMS as it must select special parameters to address the final target agent.

The use of ASN.1 Basic Encoding Rules (BER) is mandated by the SNMP RFC 1157. An agent will receive a data object from the BER decoder and reply to the manager through the BER encoder.

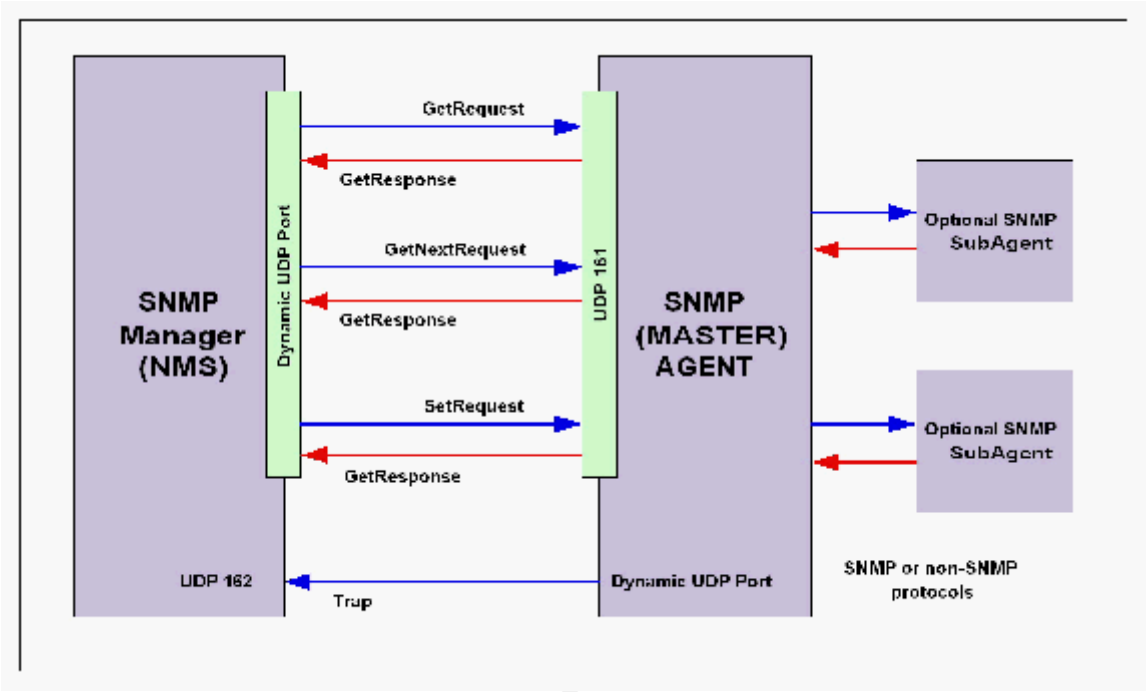

Figure 2: Simplified architecture of the Simple Network Management Protocol [2]

In the case of the Protos test suite, once a test packet has been sent, a valid Protocol Data Unit (PDU) of zero size (*zerocase*) is sent in order to determine if the SNMP service is still running. If a response is not received to this valid PDU it is assumed that the previous test packet had caused a problem in the SNMP service preventing it from responding. Clearly, this approach is not possible when testing "trap" messages as there should never be a reply to a "trap" message.

3. Vulnerabilities

3.1. University of Oulu Secure Programming Group

The University of Oulu Secure Programming Group [OSPG] initially looked in to Lightweight Directory Access Protocol (LDAP) vulnerabilities [OULDAP, CA0218] and, following this, looked into any potential SNMP issues. They specifically tested SNMPv1 but mentioned that the issues found would also be expected to affect implementations of later versions of the SNMP standard. The test suite used was "Protos" [OUSNMP] and was based around their LDAP test suite. Protos works by creating test SNMP packets containing overlong or malformed Object Identifiers (OIDs) and other exceptional data in various fields of the SNMP datagrams and includes over 50,000 test cases. This suite allowed Oulu to find vulnerabilities in the way Network Management Stations decode and process "trap" messages, as well as in the way that SNMP agents decode and process request messages such as "get" and "set". These were largely due to insufficient checking of the messages as they

[2] Based on a diagram in the Oulu SNMPv1 paper [OUSNMP]

were received and caused issues including Denial of Service, format string vulnerabilities and buffer overflows.

Some exploits do not require the use of the correct community string as the issue arose before the community string checking was applied. Due to the lack of even the simple community string authentication within SNMP such exploits are trivial. Moreover, UDP source addresses provide little protection as they can be spoofed, one could easily spoof the address of an authorized NMS for example, and some agents by default accept SNMP packets sent to the network broadcast address. Consequently, a number of exploits identified by Oulu using this suite are possible even when one knows neither the community string nor the device address.

A number of steps can be taken to provide some level of protection. However, their effectiveness is variable. Some of the steps which have been highlighted, and which are mentioned here for completeness, include:

- Using one of the free SNMP scanning tools such as SNMPing from SANS [SNMPing] or SNScan from Foundstone [FSTN] to identify SNMP devices.
- Most vendors of critical equipment have released patches against the identified vulnerabilities and these can be installed.
- SNMP can be disabled, although some systems were found by Oulu to be vulnerable to certain attacks even when this was done.
- Filter inbound traffic using a firewall to block UDP ports 161 and 162. It should be borne in mind however that there are a number of SNMP related services which may be vulnerable to attack and which do not use these ports.
- Filter outbound traffic in the same way in order to avoid being used as a launch pad.
- Change the default community strings, although this does not offer full protection.
- Use an IDS with up-to-date signatures to identify possible attacks and take preventive action.
- Use a separate Management network or VLAN for SNMP traffic

3.2. The Protos Test Suite

The protos testing suite developed by Oulu consists of the following test cases:

Group	Protocol Data Units (PDUs)	Test Cases
Req-App	GetRequest, GetNextRequest and SetRequest PDUs with application exceptions	10,601
Req-Enc	GetRequest, GetNextRequest and SetRequest PDUs with encoding exceptions	18.915
Trap-App	Trap PDUs with application exceptions	15,323
Trap-Enc	Trap PDUs with encoding exceptions	8,777

The packages can be downloaded from [OUSNMP] as jar files of which there are four, each one corresponding to a row in the above table. Once the jar files are downloaded they can be unzipped, if desired, allowing access to the source code which is used to inject the various test cases which are held in a separate directory. The suite is released under the GNU General Public License (GPL) version 2. Once downloaded, one can use individual test cases but the simplest method of testing an SNMPv1 implementation is by making use of the bundled test cases through use of

the java utility. One needs the Java Runtime environment [JSE] but, once this is installed, it is trivial to launch the test suite.

3.2.1. Simple Examples Using the Protos Test Suite

As examples to include here, the first two parts of the protos test suite, namely the Req-App and Req-Enc test cases, were tested against a Netgear ME102 Wireless Access Point (IP 192.168.1.122) on a simple network. The ME102 has an SNMPv1 management utility using community strings. The system used to launch the attacks was a Windows XP Professional laptop, equipped with an Enterasys 802.11b PCMCIA wireless card with which it connects to the Access Point in infrastructure mode. This machine was 192.168.1.5 and is alternatively known as DELLC610 in some of the following.

3.2.1.1. Protos Req-App Test Cases

The software was downloaded from Oulu and the following command used:

C:\> java -jar c06-snmpv1-req-app-r1.jar -host 192.168.1.122 –zerocase

This sends the test cases to port 161 of the SNMP agent, in this case the Access Point (AP), with the *zerocase* option sending a valid, zero length PDU and awaiting a reply in between each of the test cases. This provides a means of testing whether the SNMP agent is still functioning before sending any other test cases and allows one to determine which of the tests caused a problem. However, in order for the *zerocase* to work, the access point had to be configured with the READ-ONLY community string of "public" otherwise a reply would not be received. The tests would still work without the *zerocase* option being used. The *zerocase* does not work with Trap PDUs as the trap handling in SNMP does not involve responses. The READ-WRITE community string was set to a non-standard value and was not known or used by the protos test suite. Figure 3 shows the command line output.

```
test-case #2340: injecting valid case...
waiting 100 ms for reply...40 bytes received
test-case #2341: injecting meta-data 0 bytes, data 60025 bytes
waiting 100 ms for reply...0 bytes received
test-case #2341: injecting valid case...
waiting 100 ms for reply...40 bytes received
test-case #2342: injecting meta-data 0 bytes, data 42 bytes
waiting 100 ms for reply...42 bytes received
test-case #2342: injecting valid case...
waiting 100 ms for reply...40 bytes received
test-case #2343: injecting meta-data 0 bytes, data 100 bytes
waiting 100 ms for reply...0 bytes received
test-case #2343: injecting valid case...
waiting 100 ms for reply...0 bytes received
test-case #2343: No reply to valid case within 100 ms. Retrying...
waiting 200 ms for reply...0 bytes received
test-case #2343: No reply to valid case within 200 ms. Retrying...
waiting 400 ms for reply...0 bytes received
test-case #2343: No reply to valid case within 400 ms. Retrying...
waiting 800 ms for reply...0 bytes received
test-case #2343: No reply to valid case within 800 ms. Retrying...
ERROR: No route to host: Datagram send failed
```

Figure 3: Output from Protos "Req-App" test cases

It can be seen in Figure 3 that the AP failed to respond to the valid *zerocase* PDU after test case 2343. In addition, the wireless network failed. In this case, the ME102 did recover after several seconds but a stream of such packets would be expected to cause a sustained DoS. Figure 4 shows the captured network traffic corresponding to the failure shown in Figure 3 (test packet 2343). Note the object requested. Lines

194/195 correspond to the zerocase, 196/197 to testca se 2342, and 198/199 again correspond to the valid *zerocase.* Line 200 is test case 2343.

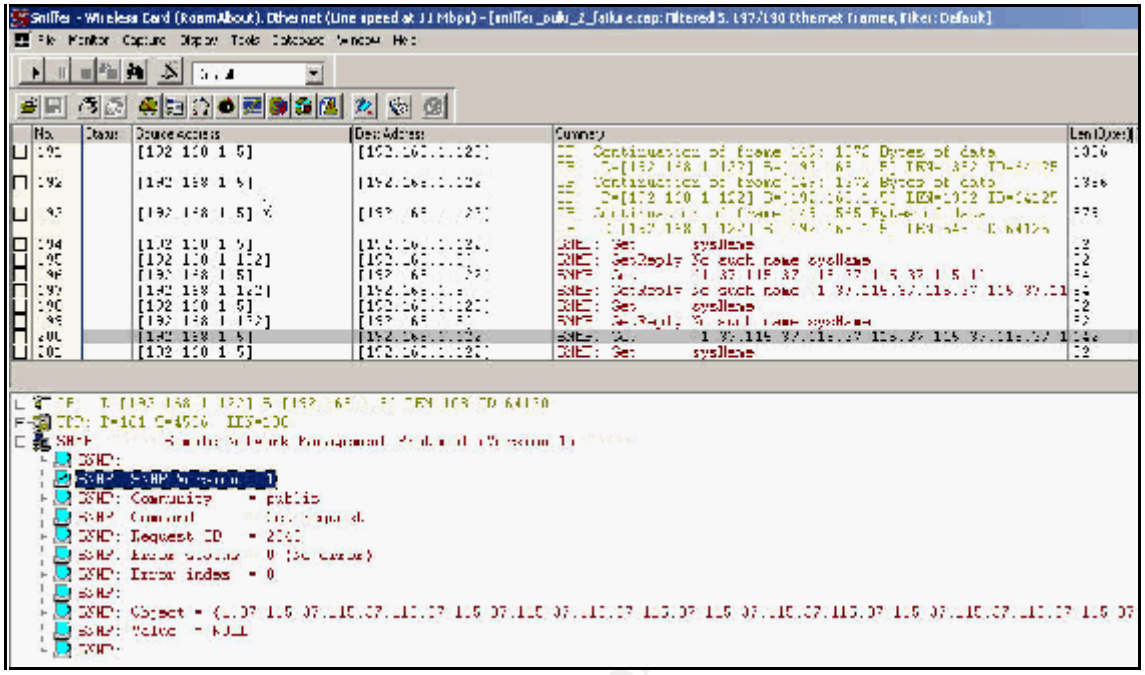

Figure 4: Sniffer output showing details of the Protos (Req -App) traffic causing AP failure

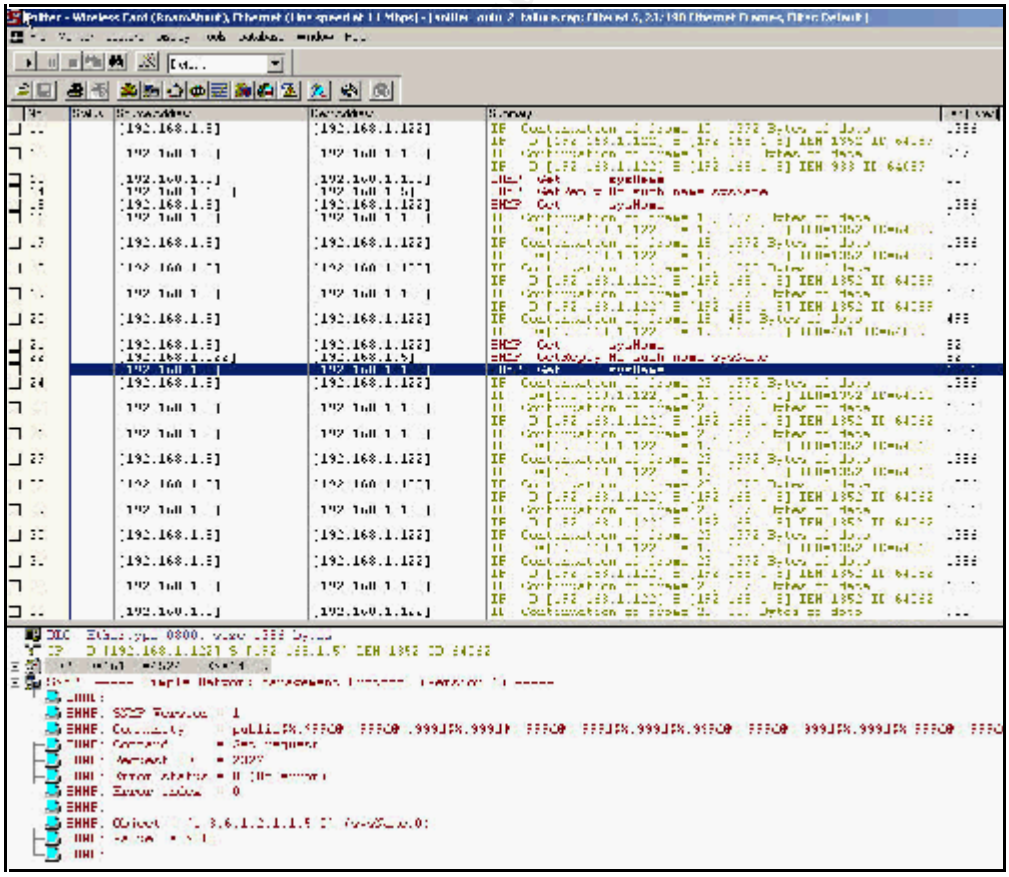

Figure 5: Sniffer output showing SNMP traffic generated by the Protos (Req-App) test cases. Note the Community String. The test cases shown did not cause the Access Point to fail.

The network traffic generated by the test suite is shown in the Sniffer Pro [SNIF] capture in Figure 5. These test cases with malformed data being sent to the Access Point, interleaved with *zerocases*, did not cause it to fail. It should be noted that there are command line options allowing one just to send a single test case, a specified range or indeed some of your own. Indeed this was used to verify that test case 2343 caused a problem even when the READ-ONLY community string was not "public". This is a powerful tool which could be dangerous in the wrong hands.

3.2.1.2. Protos Req-Enc Test Cases

A similar process was performed for the Req-Enc test cases using the command:

C:\> java -jar c06-snmpv1-req-enc-r1.jar -host 192.168.1.122 –zerocase –showreply

The command is similar to the Req-App case, but the option "-showreply" was used whereas we chose to omit this in the previous example. The command line output is shown in Figure 6 with Figure 7 showing the corresponding network traffic captured by Sniffer Pro. The packet shown in Figure 7 (row 1475, test case 1911) shows that a malformed ASN.1 packet was sent to the AP which then stopped responding, leading to a failure of the wireless network. As before, the AP recovered after a few seconds but a stream of such packets could cause a sustained DoS.

```
test-case #1909: injecting meta-data 0 bytes, data 41 bytes
waiting 100 ms for reply...0 bytes received
test-case #1909: injecting valid case...
waiting 100 ms for reply...40 bytes received
00000000  30 26 02 01 00 04 06 70 75 62 6c 69 63 a2 19 02  0&.....public...
00000016  01 00 02 01 02 02 01 01 30 0e 30 0c 06 08 2b 06  ........0.0...+.
00000032  01 02 01 01 05 00 05 00                          ........
test-case #1910: injecting meta-data 0 bytes, data 94 bytes
waiting 100 ms for reply...0 bytes received
test-case #1910: injecting valid case...
waiting 100 ms for reply...40 bytes received
00000000  30 26 02 01 00 04 06 70 75 62 6c 69 63 a2 19 02  0&.....public...
00000016  01 00 02 01 02 02 01 01 30 0e 30 0c 06 08 2b 06  ........0.0...+.
00000032  01 02 01 01 05 00 05 00                          ........
test-case #1911: injecting meta-data 0 bytes, data 163 bytes
waiting 100 ms for reply...0 bytes received
test-case #1911: injecting valid case...
waiting 100 ms for reply...0 bytes received
test-case #1911: No reply to valid case within 100 ms. Retrying...
waiting 200 ms for reply...0 bytes received
test-case #1911: No reply to valid case within 200 ms. Retrying...
waiting 400 ms for reply...0 bytes received
test-case #1911: No reply to valid case within 400 ms. Retrying...
waiting 800 ms for reply...0 bytes received
test-case #1911: No reply to valid case within 800 ms. Retrying...
ERROR: No route to host: Datagram send failed
```

Figure 6: Output from the Req-Enc test cases. The AP failing to respond after testcase 1911

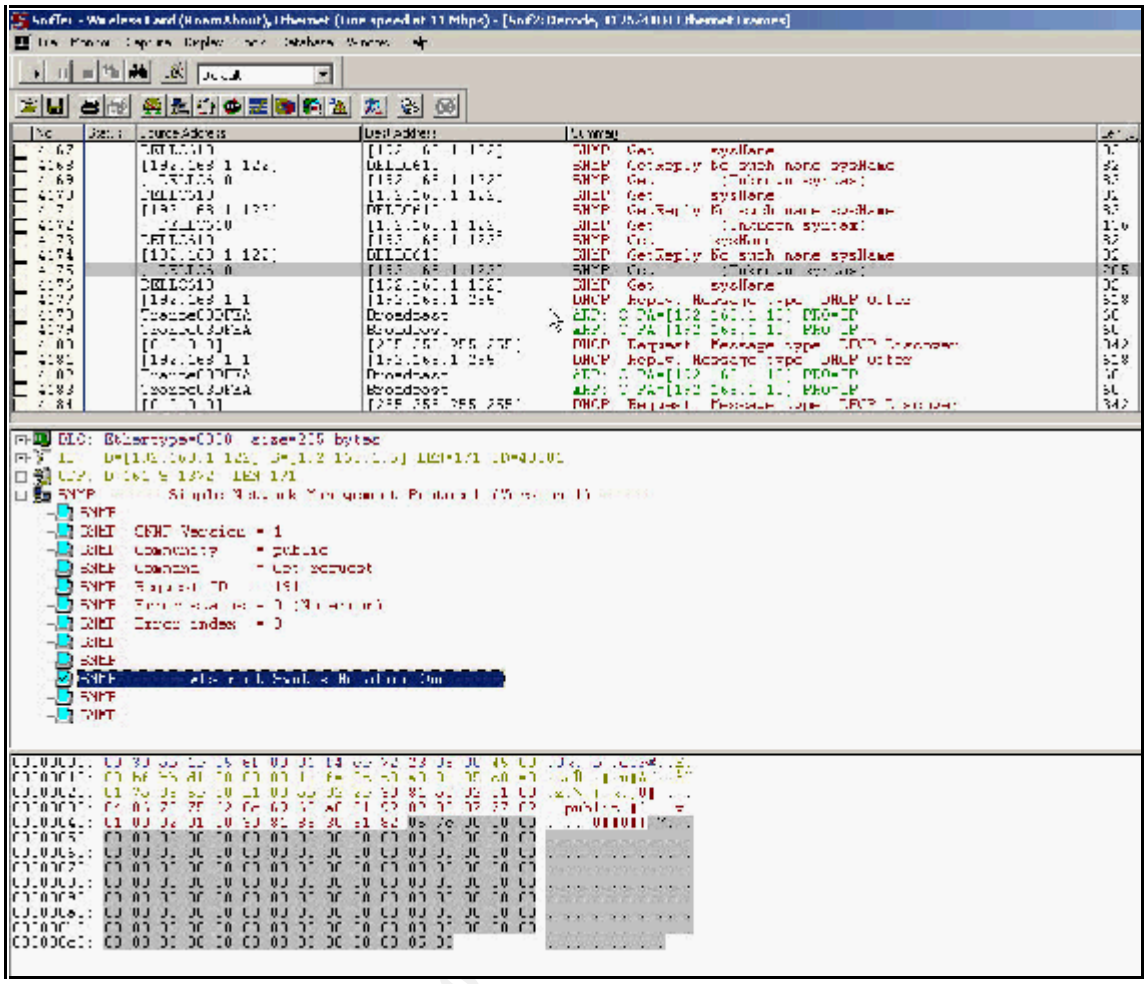

Figure 7: Sniffer output showing protos (Req -Enc) traffic which affected the AP

3.2.2. SNMP Scanners

In the examples above, the IP address of the AP was known. However, there are a number of ways to find unknown devices with SNMP listening, or the IP address of known devices, within a network. A number of devices will respond to SNMP requests sent to the network address but there are also numerous scanners available. Scanners such as Nessus, ISS, Cybercop and nmap could be used but for this simple protocol there are much faster, specialized scanners available. One such scanner is SNScan which is one of the freeware Foundstone tools [FSTN]. Another similar tool is SNMPing which was developed by SANS and is available from [SNMPing]. Using SNScan on the simple local network in use, with both the READ-ONLY and READ-WRITE community strings of the Access Point set to non-trivial values unknown to SNScan, the whole network was scanned in less than 10 seconds and the Wireless Access Point (IP 192.168.1.122) found as can be seen in Figure 8. As this utility scans quickly it would be trivial to find a large number of potentially vulnerable SNMP listening devices in a short time. Indeed, tests have confirmed this to be the case with public class C networks (up to 254 hosts) being scanned in little longer than the local network.

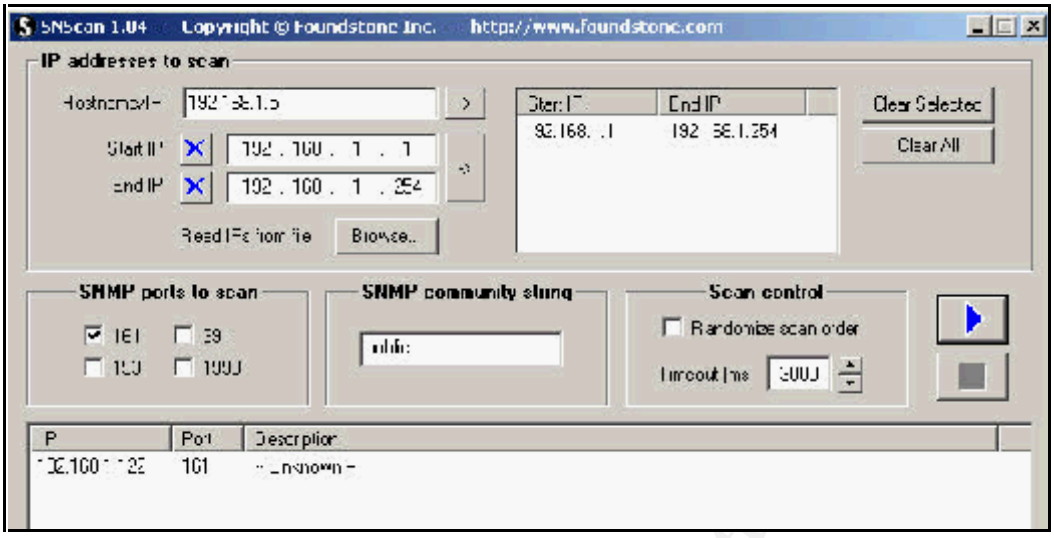

Figure 8: SNScan finds 192.168.1.122 (AP). The community strings were non -trivial & unknown

3.3. Implications

The Oulu Secure Programming Group performed a wide range of tests on numerous well known SNMP products and found a number of them to be vulnerable to the attacks included in the test suite. This was alarming as SNMPv1 is used for a number of critical elements in the Internet as well as within many corporate intranets. Device based access controls do little to help as UDP addresses are not difficult to spoof and most implementations accept SNMP packets sent to the network broadcast address by default. As mentioned earlier, this work resulted in a CERT vulnerability being issued and much work by hardware and software vendors in order to patch their products. At the time of writing it is believed that most critical products have patches available or other mechanisms have been put in place as recommended by Oulu and CERT in order to protect the devices from malicious SNMP packets. It should be noted that SNMP is also used as the basis for a number of other less obviously vulnerable services.

4. Potential Future Vulnerabilities

It has been seen that SNMP is vulnerable to a number of attacks which send malformed ASN.1 encoded data. SNMP is a widespread protocol used to manage a wide variety of devices which are often critical to a network and so there was a considerable amount of worry when the vulnerabilities were publicized by Oulu and CERT. However, a significant proportion of the critical infrastructure was already patched against the vulnerabilities as CERT released information to a number of organizations before releasing the information publicly. However, there was still the potential for a serious infrastructure issue but, for a number of not very well understood reasons, the threat failed to materialize to the extent anticipated. There is anecdotal evidence that this vulnerability was a little too complex to be exploited by "script kiddies" and more elite hackers were reluctant to spend the time and energy understanding ASN.1, which has been a little discussed protocol until recently, in order to develop an automated attack tool. Coupled with this is the fact that there would seem to be little to gain from this exploit which could not be gained via other simpler, well-understood exploits.

However, the SNMP vulnerabilities have raised awareness of ASN.1 and it would be a secluded hacker who had not heard that ASN.1 underpins a number of core Internet protocols. It would be remiss of the security community to sit back waiting for an exploit and there is in fact active research being undertaken in a race to identify the critical vulnerabilities in the core protocols in order to address them before they are exploited by hackers. SNMP had the great advantage that it was relatively easy to protect as it could be separated off a local network or filtered at the firewall with little real impact on functionality. In most cases there was little need to have SNMP access through corporate firewalls and the very fact that the normal security of SNMPv1 in particular was so lacking meant that a number of protective measures had already been taken by conscientious security managers well before this vulnerability was identified.

Unfortunately, there are a number of key protocols which rely on ASN.1 and are used precisely to provide security which cannot be filtered at a firewall without destroying their functionality. The remaining part of this paper briefly discusses a few of these.

4.1. Secure Sockets Layer (SSL) / Transport Layer Security (TLS)

4.1.1. Overview of SSL/TLS

Secure Sockets Layer (SSL) was originally developed by Netscape [SSL]. The Transport Layer Security 1.0 [RFC2246] standard was based on SSLv3 and was written to standardize the popular and widely used SSL protocol within the IETF, mandating the use of freely available algorithms. There are few differences between TLS 1.0 and SSLv3 but the two are not generally interoperable. However, TLS 1.0 can be "backed down" to be interoperable with SSLv3 implementations if required. In this discussion TLS is taken to mean both SSL and TLS unless specifically stated otherwise. In addition, SSL/TLS is a generic underlying protocol and is assumed here to be used to protect HTTP traffic. Similar comments will apply for other traffic protected by TLS.

TLS comprises the TLS Record Protocol and the TLS Handshake Protocol. The Record protocol is used to provide connection security and integrity and is used to encapsulate the higher level protocols such as the handshake protocol, the alert protocol, the change cipher spec protocol, and the application data protocol. The handshake protocol, shown in Figure 9, allows the server and, optionally, the client to be authenticated and set up the necessary encryption mechanisms before the application data is transmitted. This handshake protocol is felt to be of most relevance for this discussion.

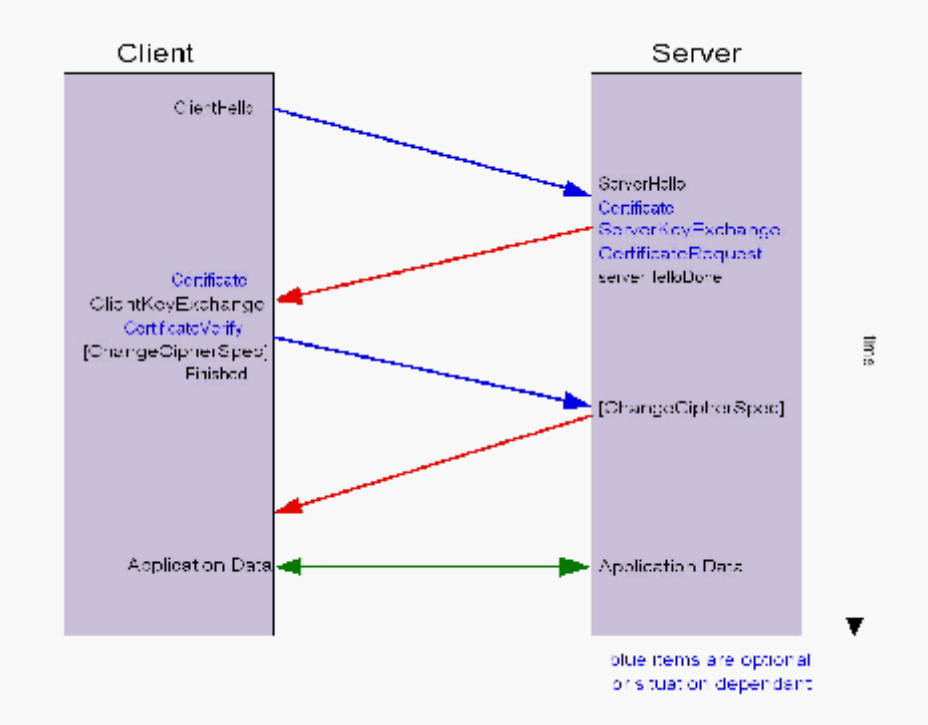

Figure 9: The SSL/TLS Handshake Protocol

The full details are in [RFC2246] but simply the handshake process is as follows:

1) Client sends "Client Hello" message to server and includes
   - a version number identifying the TLS version in use on the client
   - a random structure including the time and a random number
   - a session ID which can be empty or relate to an earlier session
   - a Cipher Suite list of supported algorithms with the favourite first
   - a list of supported compression algorithms, again with the favourite first
2) Server sends "Server Hello" message including
   - the highest TLS version supported by both server and client
   - an independent random structure
   - a session ID
   - a single cipher suite selected from the list in the Client Hello
   - a compression method selected from the list in the Client Hello
3) Server sends "Server Certificate" message including
   - Generally an X.509v3 certificate is included
   - A certificate chain can be included
4) Server sends "Key Exchange" message.
   - This is only sent if the Server Certificate message does not contain enough information for the client to exchange a premaster secret. This is unusual for the algorithms currently in common usage. Generally, a public key is included in the Server Certificate which can be used by the client to either encrypt the premaster secret in the case of RSA, or complete a key exchange in the case of Diffie-Hellman.
5) Server sends "Certificate Request" message if client authentication is required
   - list of certificate types requested in order of preference

- list of distinguished names of acceptable Certificate Authorities, derived from X.509

6) Server sends "Server Hello Done" message.
7) Client sends "Client Certificate" message if it was requested by the server
   - As in the Server Certificate message an X.509v3 certificate is usual
8) Client sends "Client Key Exchange" message
   - A random premaster secret is securely generated and sent to the server encrypted with the server's RSA public key
   - Alternatively, the client's Diffie-Hellman parameters are sent to the server if not already included in the client certificate. This allows both client and server to generate a shared premaster secret.
9) Client sends "Certificate Verify" message
   - If a client has a signing certificate it signs a concatenation of all the messages it has sent or received since the Client Hello and signs the hash using MD5 or SHA-1
10) Client sends "Change Cipher Spec" message
   - This simply signals that all future messages will be protected under the newly negotiated keys and algorithms.
11) Client sends a "Finished" message
   - This is the first message sent using the negotiated keys, and algorithms
12) Server sends "Change Cipher Spec" message

The master secret is generated from the premaster secret by using a pseudo-random function and the two random numbers generated by the client and server and sent in steps 1 and 2 of the handshake. Following this handshake process the data is protected by the TLS record protocol.

4.1.2. Potential SSL/TLS Issues

Although there is only passing mention of ASN.1 in the TLS RFC, the standard is actually fundamentally reliant on it. ASN.1 is used to encode X.509v3 certificates which include the Server and Client certificates transmitted in steps 3 and 7 of the handshake protocol. Malformed ASN.1 encodings of server certificates sent to browsers could cause vulnerabilities on a user's machine ranging from DoS to buffer overflows and the execution of arbitrary code. Moreover, if a web server is configured to accept client certificates, then a malformed client certificate could have the same effect on the server with much more serious consequences. In particular, as it tends to be the more critical web servers which are configured to accept or require client authentication through client certificates, it is ironic that these may be the most at risk. In addition, although these are the most obvious possible avenues of attack there are a number of more subtle areas which may be open to exploitation such as the signature encodings used in a number of the steps.

Furthermore, if there are issues it may be difficult to protect the servers unless a patch is available as the servers have to be accessible from the Internet. There may be some protection afforded by the fact that TLS is more complex than a stateless UDP based protocol like SNMP, which may limit the use of worms or other such attack vectors, but this has not yet been fully examined. Notably, there has already

been a CERT vulnerability [CA0 223] released which mentions specific ASN.1 encoding and decoding errors in OpenSSL. Moreover, the web server market is dominated by Apache, Microsoft and iPlanet but the market share of each is a hotly debated topic [SHAR]. It seems clear that if ASN.1 en coding and decoding issues are present in one or more of these products there would be a significant impact on the Internet and users' trust in online security. In addition, if a number of corporations were to suffer serious loss of revenue or reputation b ecause of SSL vulnerabilities this would probably make them and most of their competitors rethink their online strategy.

4.2. Other Protocols

A number of other protocols developed to provide security make use of ASN.1 encoding. In particular a number are based on RSA encryption standards PKCS#1 [RFC2313] and PKCS#7 [RFC2315].

4.2.1. S/MIME

Secure/Multipurpose Internet Mail Extensions (S/MIME) [RFC2630] specifies a way of sending and receiving secure email. It provides authentication, message integrity, non-repudiation and confidentiality by the use of digital signatures and encryption using Public Key Cryptography. It is fundamentally based on the Cryptographic Message Syntax (CMS) specification [RFC2630] which specifies encapsulation syntax for such cryptographically e nhanced data. CMS is derived from the RSA labs PKCS#7 [RFC2315] and is based heavily on ASN.1 data structures. Consequently, any protocol making use of CMS would be expected to be at considerable risk if issues were found with ASN.1.

With regards to S/MIME it is difficult to see how a critical infrastructure attack would occur as the mail transfer agents would be expected to generally transfer messages without handling any of the ASN.1 encoding. The ASN.1 would only be decoded once an email reached its end point and was decrypted and/or had its signature checked. It may also prove difficult to filter ASN.1 traffic before it reached the endpoint as it may be encrypted within the S/MIME message. In addition, there may be mechanisms to lever ASN.1 vulnerabiliti es to bypass the security features of S/MIME but the relevant standards have not yet been studied in sufficient depth to ascertain if this is the case.

4.2.2. Internet Key Exchange (IKE)

Any protocol making use of RSA encryption [RFC2313] will generally be making use of ASN.1 for handling the RSA keys, certificates and signatures and this includes the already mentioned SSL/TLS and S/MIME. In addition, protocols making use of other forms of Public Key Cryptography, such as Diffie -Hellman and DSS, may be vulnerable as they will generally be using encoding based on PKCS#1. Indeed, this is the case for the Internet Key Exchange (IKE) [RFC2409], which is used with IPsec Virtual Private Networks (VPNs) in cases where certificates are required because pre-shared secrets are either not considered secure enough or cannot scale sufficiently.

4.2.3. Others

In principle any protocol transferring information related to Public Key Cryptography may be vulnerable as ASN.1 underpins a large proportion of such protocols. In particular the X.509v3 certificate and X.509v2 Certificate Revocation List (CRL) specification [RFC2459] is based on ASN.1 and is generally used by any protocol using certificates and CRLs. Affected protocols may include such things as the PKCS #10 Certification Request format [RFC2314] which is used to request certificates from a Certificate Authority (CA).

There are other protocols directly based on ASN.1 which may be affected by such vulnerabilities including Kerberos [RFC1510], LDAP and the ITU H.323 Teleconferencing protocol. Moreover, LDAP has already, as mentioned earlier, been a subject of a CERT advisory [CA0118] based around such issues. It is expected that a number of other protocols which use ASN.1 but are not detailed here may also be vulnerable to similar issues. A list of some other protocols using ASN.1 can be found at [OSS1]. However, the protocols mentioned in this paper give an idea of the kind of issues which may exist and the kinds of protocols which may be affected.

5. Conclusion

A number of SNMP vulnerabilities were identified by the University of Oulu earlier this year which were due to the underlying ASN.1, although no large scale exploit was seen. ASN.1 is prevalent in a number of widely used protocols and, in particular, those protocols developed to secure the use of the Internet such as SSL/TLS, IKE and S/MIME as well as others such as LDAP, Kerberos and H.323. If ASN.1 vulnerabilities were found to be present in protocols such as these there could be a more serious threat than that posed by SNMP. SSL/TLS and S/MIME issues have been seen which are due to ASN.1 and it is believed that a large amount of work is being done to identify and address any serious issues in critical protocols.

However, it is not clear that the ASN.1 vulnerabilities are conceptually different to any other vulnerability. The current evidence indicates that the issues are caused by the ASN.1 encoders and decoders failing to handle malformed data in a robust way rather than being an underlying fault in the ASN.1 protocol itself, although ASN.1 is a complex protocol. If these encoders, decoders and tools can be modified to handle exceptional data more robustly, the issues raised should be resolved in the same way as for any other identified vulnerability. There is the problem that ASN.1 underlies a wide range of protocols and it would be hoped that any issues be resolved before a large scale attack is launched. If a serious SSL/TLS attack, for instance, were to be witnessed it would damage the credibility of the Internet as a secure place to do business and would discourage a large number of corporations who currently see the Internet as a core part of their business strategy. The complexity of ASN.1 may hinder a solution as could the possibility of a single attack vector, such as a worm, being deployed to target similar underlying ASN.1 vulnerabilities in a number of protocols simultaneously.

Appendix A: References

[ASN1] "Specification of Abstract Syntax Notation One (ASN.1)" ITU-T Recommendation X.208 (November 1988) http://www.itu.int/rec/recommendation.asp?type=items&lang=e&parent=T-REC-X.208-198811-W

[BER] "Specification of Basic Encoding Rules for Abstract Syntax Notation One (ASN.1)" ITU-T Recommendation X.209 (November 1988) http://www.itu.int/rec/recommendation.asp?type=items&lang=e&parent=T-REC-X.209-198811-W

[CA0118] CERT Advisory CA-2001-18 "Multiple Vulnerabilities in Several Implementations of the Lightweight Directory Access Protocol" http://www.cert.org/advisories/CA-2001-18.html (16 July 2001)

[CA0203] CERT Advisory CA-2002-03 http://www.cert.org/advisories/CA-2002-03.html "Multiple Vulnerabilities in Many Implementations of the Simple Network Management Protocol (SNMP)" (12 Feb 2002)

[CA0223] CERT Advisory CA-2002-23 http://www.cert.org/advisories/CA-2002-23.html "Multiple Vulnerabilities in OpenSSL" (30 July 2002)

[CERT] The Computer Emergency Response Team Coordination Centre http://www.cert.org (CERT/CC)

[COMER] "Internetworking with TCP/IP, Volume 1" pp 553-574 Comer, Douglas. 4th Edition (2000, Prentice Hall) ISBN: 0-13-018380-6

[FSTN] Free Foundstone Tools http://www.foundstone.com/knowledge/free_tools.html

[HS] Henning Schulzrinne's ASN.1 Resource Page http://www.cs.columbia.edu/~hgs/internet/asn.1.html

[IAB] Internet Architecture Board http://www.iab.org/

[IETF] Internet Engineering Task Force http://www.ietf.org

[ITU] International Telecommunication Union (ITU) Formerly "International Telephone and Telegraph Consultative Committee (CCITT)" http://www.itu.int/

[ITUSG] ITU-T "Study Group 17: Languages for Telecommunication Systems" http://www.itu.int/ITU-T/studygroups/com17/languages/index.html

[IWL] http://www.iwl.com
SilverCreek SNMP Testing Utility

[JL] "ASN.1 Complete" (1999). Larmouth, John. (1999, Morgan Kaufmann)
ISBN: 0-12233-435-3
Available online at http://www.oss.com/asn1/larmouth.html

[JSE] The Java 2 Platform, Second Edition is available from
http://java.sun.com/j2se (the latest version is 1.4.1)

[LG] "A Layman's Guide to a Subset of ASN.1, BER and DER"
Kaliski, B. (November 1993, RSA Labs Technical Note)
http://security.polito.it/asn1/layman.pdf (or numerous other sites)

[OSPG] University of Oulu Secure Programming Group (OUSPG)
http://www.ee.oulu.fi/research/ouspg/

[OSS] OSS Nokalva ASN.1 Resources
http://www.oss.com/asn1/

[OSS1] "Standard Using ASN.1" (OSS Nokalva Web Site)
http://www.oss.com/asn1/usage.html

[OULDAP] Protos Test Suite: c06-ldapv3
http://www.ee.oulu.fi/research/ouspg/protos/testing/c06/ldapv3/

[OUSNMP] Protos Test Suite: c06-snmpv1
http://www.ee.oulu.fi/research/ouspg/protos/testing/c06/snmpv1/

[RFC768] RFC 768, "User Datagram Protocol"
Postel, J (August 1980) http://www.ietf.org/rfc/rfc768.txt

[RFC1028] RFC 1028, "A Simple Gateway Monitoring Protocol (SGMP)"
Davin, J. et al. (November 1987) http://www.ietf.org/rfc/rfc1028.txt

[RFC1052] RFC 1052, "IAB Recommendations for the Development of
Internet Network Management Standards"
Cerf, V. (April 1988) http://www.ietf.org/rfc/rfc1052.txt

[RFC1065] RFC 1065, "Structure and Identification of Management Information for
TCP/IP-based internets" (Obsoleted by RFC 1155)
Rose, M and McCloghrie, K. (August 1988)
http://www.ietf.org/rfc/rfc1065.txt

[RFC1066] RFC 1066, "Management Information Base for Network Management
of TCP/IP-based internets" (Obsoleted by RFC 1156)
McCloghrie, K. and Rose, M (August 1988)
http://www.ietf.org/rfc/rfc1066.txt

[RFC1089] RFC 1089, "SNMP Over Ethernet"
Schoffstall, M et al. (February 1989)
http://www.ietf.org/rfc/rfc1089.txt

[RFC1109] RFC 1109, "Report of the Second Ad Hoc Network Management Review Group", Cerf, V (August 1989)
http://www.ietf.org/rfc/rfc1109.txt

[RFC1155] RFC 1155, "Structure and Identification of Management Information for TCP/IP-based Internets (SMI)" (Obsoletes RFC 1065)
Rose, M and McCloghrie, K. (May 1990)
http://www.ietf.org/rfc/rfc1155.txt

[RFC1156] RFC 1156, "Management Information Base for Network Management of TCP/IP-based internets (MIB)" (Obsoletes RFC 1066)
McCloghrie, K. and Rose, M (May 1990)
http://www.ietf.org/rfc/rfc1156.txt

[RFC1157] RFC 1157, "A Simple Network Management Protocol (SNMP)"
Case, J. et al. (May 1990) (Obsoletes RFC 1098)
http://www.ietf.org/rfc/rfc1157.txt

[RFC1161] RFC 1161, "SNMP over OSI"
Rose, M (June 1990), http://www.ietf.org/rfc/rfc1161.txt

[RFC1215] RFC 1215, "A Convention for Defining Traps for use with the SNMP"
Rose, M. (March 1991), http://www.ietf.org/rfc/rfc1215.txt

[RFC1227] RFC 1227, "SNMP MUX Protocol and MIB"
Rose, M (May 1991), http://www.ietf.org/rfc/rfc1227.txt

[RFC1298] RFC 1298, "SNMP over IPX"
Wormley, R. and Bostock, S. (February 1992)
http://www.ietf.org/rfc/rfc1298.txt

[RFC1510] RFC 1510, "The Kerberos Network Authentication Service (V5)"
Kohl, J. and Neuman, C. (September 1993)
http://www.ietf.org/rfc/rfc1510.txt

[RFC2246] RFC 2246, "The TLS Protocol Version 1.0"
Dierks, T and Allen, C. (January 1999)
http://www.ietf.org/rfc/rfc2246.txt

[RFC2313] RFC 2313, "PKCS #1: RSA Encryption Version 1.5"
Kaliski, B (March 1998) http://www.ietf.org/rfc/rfc2313.txt

[RFC2314] RFC 2314 "KCS '10: Certification Request Syntax Version 1.5"
Kaliski, B (March 1998) http://www.ietf.org/rfc/rfc2314.txt

[RFC2315] RFC 2315, "PKCS #7: Cryptographic Message Syntax Version 1.5" Kaliski, B (March 1998) http://www.ietf.org/rfc/rfc2315.txt

[RFC2409] RFC 2409, "The Internet Key Exchange" Harkins, D and Carrel, D (November 1998) http://www.ietf.org/rfc/rfc2409.txt

[RFC2459] RFC 2459, "Internet X.509 Public Key Infrastructure Certificate and CRL Profile" Housley, R. et al (January 1999) http://www.ietf.org/rfc/rfc2459.txt

[RFC2630] RFC 2630, "Cryptographic Message Syntax" Housley, R (June 1999) http://www.ietf.org/rfc/rfc2630.txt

[RFC2633] RFC 2633, "S/MIME Version 3 Message Specification" Ramsdell, B. (June 1999) http://www.ietf.org/rfc/rfc2633.txt

[RFC2741] RFC 2741, "Agent Extensibility (AgentX) Protocol Version 1" Daniele, M. et al (January 2000), http://www.ietf.org/rfc/rfc2741.txt

[SHAR] http://www.netcraft.com/survey/
http://apachetoday.com/news_story.php3?ltsn=2000-07-27-001-01-NW-LF-MR
http://www.win2000mag.net/Articles/Index.cfm?ArticleID=15713

[SMPL] SimpleTester SNMP Testing Utility
http://www.smplsft.com/stest.html

[SNIF] Sniffer Technologies, a division of Network Associates
http://www.sniffer.com/

[SNMP88] RFC 1067, "A Simple Network Management Protocol (SNMP)" Case, J. et al. http://www.ietf.org/rfc/rfc1067.txt (August 1988) Obsoleted by RFC 1098 (89) which was obsoleted by RFC 1157 (90)

[SNMPing] SNMPing is available from http://www.sans.org/snmp

[SSL] "The SSL Protocol Version 3.0", Freier, A et al (November 1996)
http://wp.netscape.com/eng/ssl3/draft302.txt
Alternatively, see http://wp.netscape.com/eng/ssl3/

General SNMP References

SNMP FAQ (2 parts)	http://www.faqs.org/faqs/snmp-faq/
"SNMPLink.org"	http://www.snmplink.org/
"SNMPWorld"	http://silver.he.net/~rrg/snmpworld.htm
Securing SNMP (Solaris)	http://ist.uwaterloo.ca/security/howto/2000-10-04/
Securing SNMP (MS)	http://www.sans.org/infosecFAQ/incident/SNMP.htm
Securing Cisco Routers	http://www.sans.org/infosecFAQ/netdevices/router.htm
SNMP	http://www.nwfusion.com/newsletters/sec/1004sec1.html
SNMP Security (MS)	http://www.securityfocus.com/focus/microsoft/2k/snmp.html

Last Updated: June 18th, 2018

Upcoming SANS Training

Click Here for a full list of all Upcoming SANS Events by Location

SANS Paris June 2018	Paris, FR	Jun 25, 2018 - Jun 30, 2018	Live Event
SANS Minneapolis 2018	Minneapolis, MNUS	Jun 25, 2018 - Jun 30, 2018	Live Event
SANS Vancouver 2018	Vancouver, BCCA	Jun 25, 2018 - Jun 30, 2018	Live Event
SANS London July 2018	London, GB	Jul 02, 2018 - Jul 07, 2018	Live Event
SANS Cyber Defence Singapore 2018	Singapore, SG	Jul 09, 2018 - Jul 14, 2018	Live Event
SANS Charlotte 2018	Charlotte, NCUS	Jul 09, 2018 - Jul 14, 2018	Live Event
SANSFIRE 2018	Washington, DCUS	Jul 14, 2018 - Jul 21, 2018	Live Event
SANS Cyber Defence Bangalore 2018	Bangalore, IN	Jul 16, 2018 - Jul 28, 2018	Live Event
SANS Pen Test Berlin 2018	Berlin, DE	Jul 23, 2018 - Jul 28, 2018	Live Event
SANS Riyadh July 2018	Riyadh, SA	Jul 28, 2018 - Aug 02, 2018	Live Event
Security Operations Summit & Training 2018	New Orleans, LAUS	Jul 30, 2018 - Aug 06, 2018	Live Event
SANS Pittsburgh 2018	Pittsburgh, PAUS	Jul 30, 2018 - Aug 04, 2018	Live Event
SANS San Antonio 2018	San Antonio, TXUS	Aug 06, 2018 - Aug 11, 2018	Live Event
SANS August Sydney 2018	Sydney, AU	Aug 06, 2018 - Aug 25, 2018	Live Event
SANS Boston Summer 2018	Boston, MAUS	Aug 06, 2018 - Aug 11, 2018	Live Event
Security Awareness Summit & Training 2018	Charleston, SCUS	Aug 06, 2018 - Aug 15, 2018	Live Event
SANS Hyderabad 2018	Hyderabad, IN	Aug 06, 2018 - Aug 11, 2018	Live Event
SANS New York City Summer 2018	New York City, NYUS	Aug 13, 2018 - Aug 18, 2018	Live Event
SANS Northern Virginia- Alexandria 2018	Alexandria, VAUS	Aug 13, 2018 - Aug 18, 2018	Live Event
SANS Krakow 2018	Krakow, PL	Aug 20, 2018 - Aug 25, 2018	Live Event
SANS Chicago 2018	Chicago, ILUS	Aug 20, 2018 - Aug 25, 2018	Live Event
Data Breach Summit & Training 2018	New York City, NYUS	Aug 20, 2018 - Aug 27, 2018	Live Event
SANS Prague 2018	Prague, CZ	Aug 20, 2018 - Aug 25, 2018	Live Event
SANS Virginia Beach 2018	Virginia Beach, VAUS	Aug 20, 2018 - Aug 31, 2018	Live Event
SANS San Francisco Summer 2018	San Francisco, CAUS	Aug 26, 2018 - Aug 31, 2018	Live Event
SANS Copenhagen August 2018	Copenhagen, DK	Aug 27, 2018 - Sep 01, 2018	Live Event
SANS SEC504 @ Bangalore 2018	Bangalore, IN	Aug 27, 2018 - Sep 01, 2018	Live Event
SANS Wellington 2018	Wellington, NZ	Sep 03, 2018 - Sep 08, 2018	Live Event
SANS Amsterdam September 2018	Amsterdam, NL	Sep 03, 2018 - Sep 08, 2018	Live Event
SANS Tokyo Autumn 2018	Tokyo, JP	Sep 03, 2018 - Sep 15, 2018	Live Event
SANS Tampa-Clearwater 2018	Tampa, FLUS	Sep 04, 2018 - Sep 09, 2018	Live Event
SANS MGT516 Beta One 2018	Arlington, VAUS	Sep 04, 2018 - Sep 08, 2018	Live Event
SANS Cyber Defence Canberra 2018	OnlineAU	Jun 25, 2018 - Jul 07, 2018	Live Event
SANS OnDemand	Books & MP3s OnlyUS	Anytime	Self Paced