Enhancing Syslog and SNMP Traps/Notifications Security

Faisal M. Sibai , Linh Khanh Nguyen, Phirun C Thach, Yan Wu
George Mason University
fsibai@gmu.edu, lnguyeno@gmu.edu , pthach@gmu.edu, yanwu3@gmail.com

Abstract

The current Simple Network Management Protocol version 3 (SNMPv3) mentioned in RFC (3414)[1] and (3411)[2] implements some security measures opposed to the previous versions, yet the security measures which have been implemented are still weak and insufficient in the face of current threats from hackers and attackers in the world, on the other hand Syslog mentioned in RFC (3164)[3] or (3195)[4] does not incorporate any security features or mechanisms in the protocol design; information is sent in clear text which can be captured internally or externally in any given environment. The proposed solution modifies the current protocol built for both SNMPv3 and Syslog to incorporate higher security measures which impose confidentiality, integrity and authenticity into their architecture.

1. Introduction

Security Operation Centers (SOCs) and Network Operation Centers (NOCs) nowadays are deployed in different scale organizations which can range from small to large. These centers basically process incoming messages from different systems and network components scattered all around the network. These messages contain valuable network, system and configuration information which are part of the network. The information received by the devices is analyzed and decisions are made based on the analysis whereby these decisions are usually based on security concerns inside or outside the organization or they can be based on simple network management related issues. The messages are sent either through SNMP traps/notifications from SNMP agent enabled devices or Syslog enabled devices; SNMP traps in version 1 & 2 are sent in clear text while SNMP traps in version 3 (RFC 3414)[1] provides some message privacy using Code Block Chaining-Data Encryption Security (CBC-DES) and some authentication mechanisms using HMAC-MD5-96 or HMAC-SHA1-96 [1]. Syslog (RFC 3164 [3] & 3195[4]), on the other hand, provides no privacy or authentication, and therefore it is also sent in clear text. The messages which are sent in clear text cause a very large concern since they disclose private information on the internal or external network which could be sniffed or snooped and could be used for other hacking attempts later on. Clear text can also enable replay attacks which happens when a third party is sitting in the middle of the communication and tries to forge and send edited information to the final destination. Authentication of the origin is also not available in this model which could cause repudiation to a sent message

2. Traps and Syslog Notification Messages

SNMP Traps are asynchronous notifications sent from SNMP enabled agents to what are called SNMP managers or collection stations. SNMP was originally defined in RFC 1157[5]. SNMP Traps are sent using UDP port 162 which makes these messages unreliable as both sender and receiver cannot guarantee the delivery of the messages sent. Traps are a bundle of data that are defined by a MIB (Management Information Base); seven generic trap numbers (0-6) are defined. They range from (coldStart) to enterpriseSpecific); enterprise-specific traps are what make the trap mechanism so powerful as it gives companies the opportunity to define enterprise-specific traps to whatever conditions they consider worth monitoring. The trap is divided into enterprise ID (the organization ID) and a specific number assigned by the organization. The notion for the trap is extremely flexible since organization can subdivide their enterprise as much as they like, therefore, defining customized traps carrying specific information like IP addresses, security threats, port numbers is a very common thing and is widely used. Currently there are three different versions for SNMP in use which are v1,v2c and v3. v1 is still heavily used whereas v2c is not frequently used due to its different variations and protocol complexity and lastly v3 which has not yet been adopted by most companies thus has not been widely used to date [6].

Syslog protocol was originally defined in RFC 3164. This protocol provides a transport to allow a device to send event notification messages across IP networks to event message collectors known as syslog servers, or managers or correlation services and no acknowledgments are sent back for the receipt of the messages. This service is usually under a Unix service called “syslogd” which means syslog demon. Syslog uses UDP port 514 for its communication which basically does not guarantee the delivery of the message or the receipt of the receiver. The Syslog packet is limited to 1024 bytes and carries the following information in its packets: facility, severity, hostname, timestamp and message. Facility is a categorization of source generated from which could be operating system, process or an application. Severity is the severity of the message which is generated from the specific source that can range from 0 (Emergency) to 7 (Debugging). The Hostname is the hostname or IP address which the message was transmitted from. Timestamp is the local time when the message was generated. Message is simply a message which gives more information about the notification. Syslog has been implemented using both UDP and TCP. TCP was introduced to provide reliability to notifications but at the same time adds overhead on devices. Although reliability was introduced for notifications delivery assurance, companies mostly use the UDP based syslog[7].

2. SNMP v 1 Trap security characteristics

SNMPv1 traps are sent in clear text on the network, therefore the source device IP address, severity, enterprise-id information, community name are all sent in clear text and can be sniffed off the network and reused at a later stage. v1 does not provide any encryption, integrity or non-repudiation mechanisms for the messages sent. Messages can be intercepted and viewed or even changed using a man-in-the-middle attack or messages could even be replayed at a later stage. The below diagram (figure1) shows the packet layout for a SNMPv1 packet:

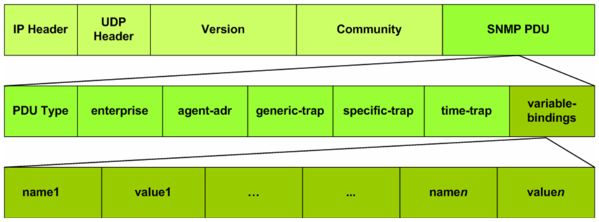

Figure (1)

3. SNMP v 2/2c Trap security characteristics

SNMPv2 and its variations (2c, 2u, 2) were introduced to solve some problems in v1 most of which were internal protocol problems that, to a large extent, don’t affect the user or traps. Community strings and information was still being sent in clear text so the main differences were the addition of what is called GETBULK PDU to the protocol which enables the requester to request a large number of GET or GETNEXT in one packet. SMI in v1 was upgraded to SMIv2 with more data types like 64-bit counters. The above changes did not reflect any changes on the trap security delivery mechanisms. Some of the changes which did affect traps in v2 however is the change of TRAP-TYPE in MIB v1 to NOTIFICATION-TYPE in v2, hence, no big difference between the old and the new. An INFORM type of traps was also added which is basically an acknowledged type of trap instead of the old unacknowledged trap. Furthermore, in v2 the generic-traps were replaced with many specific traps or better saying notifications. So as we can see from the above. no changes on the security mechanisms were implemented in v1[6].

4. SNMP v 3 Trap security characteristics

SNMPv3 incorporated major security enhancements which included authentication and encryption of the messages   in both TRAPS and INFORMS. Authentication of the message is done by the use of MD5 or SHA and encryption of the message was originally done by the use of DES. Recently, AES has been introduced to some v3 developments [8].

The following diagram (figure 2) shows a typical v3 message format:

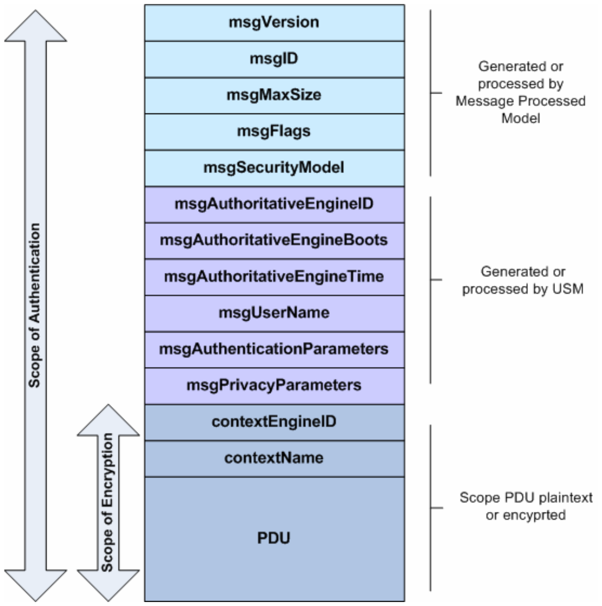
Scope of Authentication
Scope of Encryption
msgVersion
msgID
msgMaxSize
msgFlags
msgSecurityModel
msgAuthoritativeEngineID
msgAuthoritativeEngineBoots
msgAuthoritativeEngineTime
msgUserName
msgAuthenticationParameters
msgPrivacyParameters
contextEngineID
contextName
PDU
Generated or processed by Message Processed Model
Generated or processed by USM
Scope PDU plaintext or encyprted

Figure (2)

The msgVersion is set to 3 which is the SNMP version. The msgID is used between the manager and the agent for response coordination. The msgMaxSize is the maximum message size supported and the msgFlags is an 8 bit value important flag which shows whether a report PDU is to be generated, or whether privacy or authentication is used. The msgSecurityModel specifies which security model was used (could be 1, 2 or 3) and msgSecurityParameters is also important here since it contains security specific information, the contectEngineID identifies an SNMP entity (discussed later), conextName identifies a particular context within an SNMP engine, scopePDUa block of data made up of contextEngineID, contextName, and SNMP PDU. The following are the important added security features in SNMP: msgAuthoritativeEngineID,
msgAuthoritativeEngineBoots,and
msgAuthoritativeEngineTime
which are respectively the snmpEngine ID , Boots and Time for the authoritative engine . msgUserName is the user who will be authenticating and encrypting the message, msgAuthenticationParameters is what contains the HMAC message digest MD5 and SHA that are currently being used, and msgPrivacyParameters which could contain the encryption mechanism used like DES or AES depending on the SNMP implementation[6].

In order for all the above to work, we need to understand the composition of SNMPv3 entity .The SNMPv3 entity is composed of an engine and applications; the engine is divided into four pieces: the Message processing subsystem, the Dispatcher, the security subsystem, and the access Control Subsystem. The applications are divided into a number of applications like the command generator, command responder, notification originator, etc. The following diagram (figure 3) shows exactly what the entity contains:

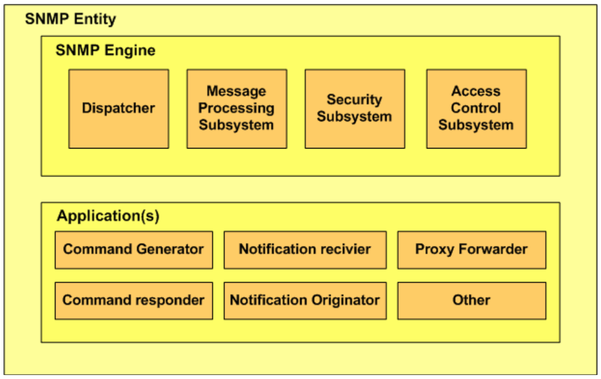

Figure (3)

In the Engine part we find that the dispatcher's job is to send and receive messages from different SNMP versions. The message processing subsystem prepares a message to be sent and extracts data from received messages. The security subsystem then provides authentication and encryption services for the message and the access control subsystem is responsible for controlling MIB objects. With regards to the applications, we can find that most of the applications existed in SNMPv1 and v2 .

In reality, when traps are configured, the following parameters will be configured: Username, Security level which could be noAuthNoPriv which has no authentication and no privacy, it could be authNoPriv which has authentication but no privacy, or it could be authPriv which is authentication and privacy; authentication protocol, authentication passphrase, privacy protocol and lastly privacy passphrase [6].

From the information which has been collected, it is quite evident that SNMPv3 added message authentication and encryption to its message layout, and they can also be enabled and disabled based on user needs.

5. Syslog security characteristics

Syslog in its basic implementation does not provide confidentiality, integrity or availability. Some syslog reliability mechanisms were proposed in RFC 3195 which propose a reliable syslog architecture but do not add anything on security [7]. Some other initiatives tried to add to syslog to make it more secure. Some of these were adding TLS on top of syslog , or using SSH based clients to add authenticity and privacy to the syslog packets. Third party software is available on the net and provides this functionality. Other initiatives also took place recently with the end of 2006 which proposes a new syslog-sec protocol and another one which involves syslog signing but none have been approved yet and no feedback has been given.

6. Research Problem

Syslog and SNMP Traps are highly used in most organizations and environments in either SOCs or NOCs or for normal testing purposes. Most environments, to this day, tend to send all their information with no security on top of it. The reason behind this is due to the lack of security in the first place in the protocol (as in SNMPv1 and SNMPv2) and people are use to it since its wide spread among technologies and its ease of implementation. Another reason is that this technology can support a number of security mechanisms like privacy and integrity (SNMPv3) but thus far, many technology producers have not yet adopted the new protocol as well as the difficulties while implementing the security features in the protocol. In addition, there are different security levels the protocol can be configured to (e.g. noAuthNoPriv, authNoPriv, or authPriv) which gives the user the option to not use any security, which people tend to prefer, due to the easiness of its configuration. Another reason is that some organizations, such as the government and banking sectors, rely on high speed based messages which contain text messages and severity details for correlation requirements (Syslog messages). These messages contain no security mechanisms on top of them. Some workarounds like using third party TLS (SSL) and SSH utilities try to solve the problem but not fully since they are still relying on a third party software and that software does not cover all communications (which means that some traffic can slip without encryption). Recently, a drafted document by the IETF proposed syslog using TLS which sounds good , but the problem with that solution is that it works in a reliable environment using TCP while most users tend to use UDP due to its considerably higher speed and because there is no need for a reliable transport layer. The following diagram (figure 4) sums up the above discussion:

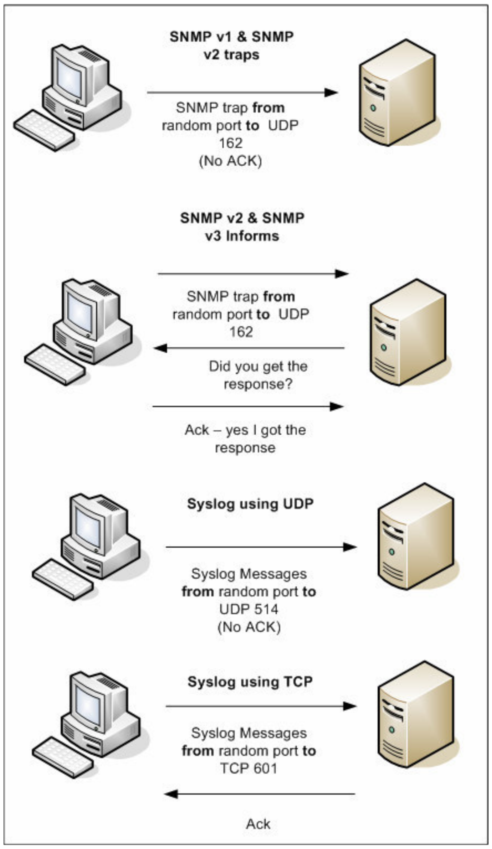
SNMP v1 & SNMP v2 traps
SNMP trap from random port to UDP 162
(No ACK)
SNMP v2 & SNMP v3 Informs
SNMP trap from random port to UDP 162
Did you get the response?
Ack – yes I got the response
Syslog using UDP
Syslog Messages from random port to UDP 514
(No ACK)
Syslog using TCP
Syslog Messages from random port to TCP 601
Ack

Figure (4)

7. Related Research

While researching the different security mechanisms to secure SNMP Traps/Notifications and Syslog, we came across some mechanisms that were used by some organizations. This section describes the mechanisms which are currently being used:

7.1 IPsec

IPsec (IP security) is a suite of protocols for securing Internet Protocol (IP) communications by authenticating and/or encrypting each IP packet in a data stream. IPsec also includes protocols for cryptographic key establishment. IPSec operates on the network layer. This makes IPsec more flexible, as it can be used for protecting both TCP and UDP based protocols, but it increases its complexity and processing overhead, as it cannot rely on TCP (OSI layer 4) to manage reliability and fragmentation [9]. While IPsec provides encryption for flowing SNMP trap/notifications and Syslog traffic, IPsec is not usually deployed in LAN networks; it is usually deployed in virtual private networks and WAN connections. The robust key distribution and key exchange, processing requirements needed, and the complexity and overhead make IPSec difficult to deploy on large scale networks in the LAN segments. On the other hand, IPSec could still be deployed on a more controlled small-medium network.

7.2 Stunnel (Secure Tunnel)

Stunnel is a form of SSL wrapper which can be used to provide secure encrypted connections for clients or servers that do not speak TLS or SSL natively. It runs on a variety of operating systems, including most Unix-like operating systems and Windows. It relies on a separate library such as OpenSSL or SSLeay to implement the underlying TLS or SSL protocol [10]. Syslog messages can be tunneled using Stunnel which uses the syslog-ng server and client mechanisms. Although Stunnel is good for some popular operating systems, it lacks support on other proprietary appliances from different vendors which produce different security based appliances (i.e., firewalls, IDS, IPS, Anti-spam, etc.). And by using the Stunnel architecture, the traffic will only be secure partially, and not fully, as in the diagram below:

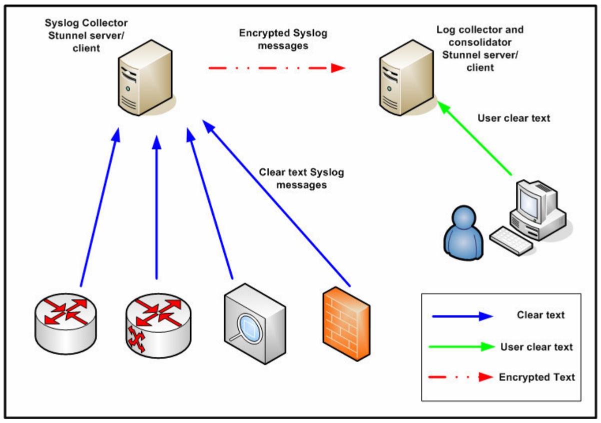
Syslog Collector Stunnel server/ client
Encrypted Syslog messages
Log collector and consolidator Stunnel server/ client
User clear text
Clear text Syslog messages
Clear text
User clear text
Encrypted Text

Figure (5)

7.3 SSH (Secure Shell)

Secure Shell or SSH is a set of standards and an associated network protocol that allows establishing a secure channel between a local and a remote computer. It uses public-key cryptography to authenticate the remote computer and (optionally) to allow the remote computer to authenticate the user. SSH provides confidentiality and integrity of data exchanged between the two computers using encryption and message authentication codes (MACs).[11] While SSH is typically used to log into a remote machine and execute commands, it also supports tunneling for other protocols like Syslog messages by establishing a tunnel between the server and the client. No matter how trustworthy this type of implementation seems, in reality, it still suffers from the previous problems in Stunnel, adding to it the problem of maintaining a session; if a session is lost, messages will also be lost, too.

7.4 Nsyslogd, Secure Remote Streaming (SRS) and other variants

Secure Remote Streaming[12], Nsyslogd[13] and other variants suffer from similar inherited problems of Stunnel and SSH; the un-scalable support for different operating systems limits the ability for these mechanisms to work on more proprietary OSs which are deployed on a wide variety of network and security based appliances.

7.5 Signed Syslog

A draft was submitted in 2007 to the IETF on a solution for the following syslog security problems:

- ❖ Origin authentication
- ❖ Message integrity
- ❖ Replay resistance
- ❖ Message sequencing
- ❖ Detection of missing messages [14].

Although this draft seems to solve a lot of issues related to Syslog security, its main concentration nevertheless is on signing the message and does not give encryption of the payload high importance. Therefore, this draft has certain benefits but lacks some other important concerns.

7.6 Syslog over TLS

Another draft was also submitted in 2007 in regards to syslog security using TLS as it can provide the following:

- Confidentiality
- Authenticity
- Integrity
- Reliable message flow due to the TCP connection [15].

While this looks like an excellent solution, it causes tremendous overhead on the network in addition to constant session maintenance and slow speed of message sending.

As observed from all the different implementations discussed earlier, it is clear that these protocols depend on another layer to secure the traffic sent or received. Our goal in the next part of this paper is to concentrate on the enhancement of the security prospective of the protocol instead of depending on third party layers or software.

8. Solutions/Analysis

The goal of our solution is to enhance the current protocol supporting structure to integrate security as a core requirement in the protocol. An overview of the enhancement for each protocol will be detailed in this section:

8.1 SNMPv1 and SNMPv2

SNMPv1 and SNMPv2 as mentioned earlier in this document use community names (strings) to send traps/notifications. These community names (strings) are sent in clear text. SNMPv1 and SNMPv2 are considered a weak mechanism to support any security enhancements due to their limited architecture which necessitates the introduction of a newer version for this protocol which is SNMPv3. Therefore, further use of SNMPv1 and SNMPv2 protocols should be stopped and both should be completely retired.

8.2 SNMPv3

SNMPv3, as mentioned earlier in this document, was launched with a great deal of added security features. It mainly was developed to address the main security issues which are: protection against modification of information, protection against masquerade, protection from message disclosure, and finally protection against message replays and stream modifications. The protocol based on RFC[3414] using the USM or User based model addressed these components, but unfortunately did not take into consideration two main issues which will be discussed below with their solutions:

8.2.1 Security levels

SNMPv3 implements three different types of security levels in the User-Based model:

-without authentication and without privacy (noAuthNoPriv)
- with authentication but without privacy (authNoPriv)
- with authentication and with privacy (authPriv)

By default the SNMPv3 protocol defaults to noAuthNoPriv which basically gives the same results as SNMPv1 or SNMPv2. Most users tend to always use the easier and simpler implementations therefore they are inclined to use the default and actually stick to it with no changes. The following information is usually entered by the user while configuring an SNMP entity:

- User name
- Security Level
- Authentication algorithm used
- Authentication passphrase
- Privacy algorithm used
- Privacy passphrase

We can also see from the Management Information Base (MIB) below that the three security levels are actually enabled based on the integer number order they have :

```
INTEGER { noAuthNoPriv(1),
          authNoPriv(2),
          authPriv(3) }
```

The above parameters are configured based on the inputs which are placed by the user while configuring the SNMP entity on both the SNMP agent and the SNMP manager. Through our solution, we simply believe that noAuthNoPriv and AuthNoPrivacy should be completely eliminated as choices in the MIB, and the user interface should force the user to enter the parameters required for both authentication and privacy and eliminate the security level choice as following:

- User name
- Authentication algorithm used
- Authentication passphrase
- Privacy algorithm used
- Privacy passphrase

The MIB should also only contain the following :

INTEGER {authPriv (1)}

The only situation where noAuthNoPriv is required is in SNMP Discovery where the non-authorative SNMP entity (manager) queries the authorative (Agent) for its snmpEngineID. Since no authentication and no privacy is used at that stage, an exceptional case can be solely configured for SNMP Discovery but it should not affect SNMP traps or the notification mechanism. This exception could be implemented by adding a different separate module that handles the SNMP Discovery or by enabling the option of noAuthNoPriv to the SNMP Discovery module only.

8.2.2 Symmetric encryption with Automated key exchange method

Currently, based on RFC [3411] as well as [3414], encryption and authentication is implemented as following:

- Static or even dynamically generated passphrases are stored within a privacy key and authentication key parameters are inside of the SNMP engine.
- These keys are manually entered on different hosts so symmetric encryption can take place between the two SNMP entities at any time.
- These keys are changed based on manual user intervention.
- No expiration or revocation is done without user intervention.

The above solution suffers from main security threats like:

- Password/passphrase guessing
- Password/passphrase weakness
- Password/passphrase aging

The reason behind the above implementation is the outcome of using UDP on its own by means of the SNMP protocol for messages.

UDP is mainly used in SNMP traps because of its speed, low impact and overhead on the network which justifies its use.

Our solution solves this problem by using a Hybrid of TCP for key exchange based on the Diffie-hellman algorithm which are negotiated on specific time intervals with the use of UDP as the main subsystem for trap/notification sending.

This solution basically includes the following core items as the foundation for the new SNMPv3-Secure Exchange created by our solution:

1. The SNMPv3 entities. (both Agent and Manager)
2. Randomly generated large prime number P and a primitive base G mod P. where both should be stored in a new MIB variable, for example: snmpPrimeNumber and snmpPrimitiveBase
3. The UDP connectionless protocol which sends and receives the traps/notifications
4. The TCP connection-oriented protocol to exchange the large prime numbers for the use of the Diffie-Hellman algorithm at each SNMP entity
6. A Message buffer
5. The Diffie-Hellman algorithm in the SNMP entity
7. Random large generated integers

The below diagram summarizes the main components of the SNMPv3-SecureExchange method and further details will be given following the diagram:

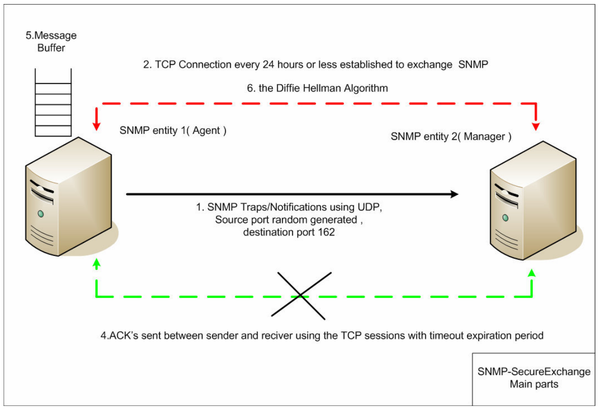
5.Message Buffer
2. TCP Connection every 24 hours or less established to exchange SNMP
6. the Diffie Hellman Algorithm
SNMP entity 1(Agent)
SNMP entity 2(Manager)
1. SNMP Traps/Notifications using UDP, Source port random generated , destination port 162
4.ACK's sent between sender and reciver using the TCP sessions with timeout expiration period
SNMP-SecureExchange Main parts

Figure (6)

The following is the sequence of operations:

1. The SNMP entity in the Agent (authorative) generates a large prime number P and a primitive base of Mod P which is called G. A secret is also generated and shared between the SNMP entity which gets appended to the messages for Message integrity and

authentication of origin. A message digest is created for all three values and appended to the message which will be sent (HMAC-MD5-96 or HMAC-SHA1-96 could be used here)

2. After the generation of the numbers, a TCP session is established by the entity (Agent) to the other entity (the Manager) and timers for ACK start at the same time.

3. The entity authenticates the user which established the connection.

4. If the connection is not authenticated, the connection is dropped.

5. An ACK is sent back to the Agent for both Authentication and connection establishment. If it is not sent, it will be repeated for 3 times and if the connection fails, a trap is generated.

6. If the connection is authenticated P, G ,secret and the message digest are transported across the session in clear text from the agent to the manager.

7. The Agent will have a timer start from the moment it transfers the payload with all the contents to the Manager. If an ACK is received before the expiration of the timer, the operation will be completed and another timer starts for a second ACK.

8. If the timer expires before the ACK is received, a TCP session is started again and the whole process is repeated (the process is repeated 3 times. If the 3rd time fails, a trap is generated to the Manager).

9. If the first ACK is received by the Agent successfully, that means that the payload is received by the Manager. The Manager then verifies that the message digest is correct and the contents are not modified. However, if the content is modified, a second ACK will not be sent back to the Agent and the timer expires on the Agent's side. On the other hand, if the content is received correctly, a second ACK is sent back to the Agent which is the flag to start the negotiation of computed values.

10. The Agent generates a large random integer (a) which is not shared, (G^a mod P) is computed and then a message digest is computed and attached to the payload and sent to the manager and a timer starts for an ACK.

11. If it is received successfully, an ACK is returned if not the same process of prior ACK is repeated here
(see steps 8 & 9). The Manager makes sure that the payload is not modified on the way, but if it is modified, a second ACK is not sent back so the timer of the Agent will realize that it has expired and resends the message. If it is received correctly, a second ACK is sent back.

12. After the manager sends out the ACK, the manager generates a random large integer (b) which is not shared. (G^b mod P) is computed and then a message digest is computed and attached to the payload and sent to the Agent.

13. If it is received successfully and ACK is returned if not the same process of prior ACK is repeated here
(see steps 8 & 9).

14. Both the agent and manager now compute the secret key as following:

- ❖ Agent: (G^b mod P)^a mod P
- ❖ Manager: (G^a mod P)^b mod P.

Both will generate the same key which can now be used in the SNMP privacy key attribute in the MIB.

15. Now the message can be sent using UDP by encrypting the traffic using the new generated shared key, and can also be decrypted on the other side using the same shared key.

16. While the process of key exchange is occurring, the messages are buffered at the Agent's side and will not be sent until the process of key exchange has been either completed or not (which will result in using the older key in case of a failure).

The following diagram shows the Protocol interaction which happens between each SNMP Entity:

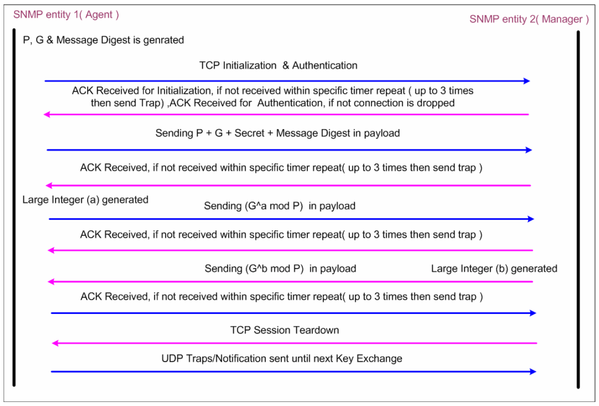
SNMP entity 1(Agent)
SNMP entity 2(Manager)
P, G & Message Digest is genrated
TCP Initialization & Authentication
ACK Received for Initialization, if not received within specific timer repeat (up to 3 times then send Trap) ,ACK Received for Authentication, if not connection is dropped
Sending P + G + Secret + Message Digest in payload
ACK Received, if not received within specific timer repeat(up to 3 times then send trap)
Large Integer (a) generated
Sending (G^a mod P) in payload
ACK Received, if not received within specific timer repeat(up to 3 times then send trap)
Sending (G^b mod P) in payload
Large Integer (b) generated
ACK Received, if not received within specific timer repeat(up to 3 times then send trap)
TCP Session Teardown
UDP Traps/Notification sent until next Key Exchange

Figure (7)

As we can see from the above the mechanisms, it is easy and straightforward using the Diffie-Hellman algorithm and at the same time, the keys are exchanged in a very secure manner without much complication.

The same method used for key encryption exchange can also be used for authentication key exchange by using a different TCP session .

8.2.3 Other mechanisms which can be used in the same manner

Other mechanisms can also be used in the same manner such as:

- ❖ The integration and support of a PKI infrastructure by having Asymmetric cryptographic key (public and private keys)
- ❖ A Password-authenticated key agreement algorithm (such as, EKE, PAK or PPK)

The reason why we personally don’t recommend these is due to their complicated implementation; SNMP is at the end of the day called Simple Network Management Protocol which means that it should, at least, be simple to use.

8.3 Syslog

Syslog, as mentioned earlier in this document, was introduced when security wasn’t a very big concern, hence, this protocol was developed with a lot of major security loopholes like:

- ❖ Confidentiality
- ❖ Integrity
- ❖ Authenticity
- ❖ Message forgery
- ❖ Message Replay

Our solution concentrates on solving some main problems like:

- ❖ Confidentiality
- ❖ Integrity
- ❖ Authenticity

The following two solutions will be proposed by our group.

8.3.1 TLS Syslog (from draft RFC draft-ietf-syslog-transport-tls-07)

This solution was proposed in 2007 by two men working in Huawei in Beijing. Not much detailed information has been provided by the group on the actual implementation of this mechanism, but in simple terms, this solution uses TLS (Transport Layer Protocol) as the main transport layer for the messages. By implementing TLS, the following would be achieved:

- ❖ Confidentiality
- ❖ Authenticity
- ❖ Integrity
- ❖ Reliable message flow due to the TCP connection.

A sender and relay is always considered a client and a receiver is always considered a server.

The following shows the details on how the exchange of messages works:

1. Client sends a ClientHello message specifying the list of cipher suites, compression methods and the highest protocol version it supports.

2. The client receives a ServerHello where the server chooses the connection parameters from the choice offered by the client.

3. When the connection parameters are known, the client and server exchange certificates which are based on the X.509 standard.

4. The Server requests the certificate from the client so that the connection can be mutually authenticated.

5. The Client and server negotiate a common secret called the “master secret”, possibly using the result of DH (Diffie-Hellman Exchange) or simply encrypting a secret with a public key that is decrypted with the peer’s private key.

6. After that, the encrypted path will exist and Syslog messages can be sent through the encrypted path.

7. If for any reason the encrypted path goes down and needs to be re-established, Syslog traffic can be buffered for that period and continues to be sent after the connection is resumed.

8. ACKs can also be implemented using this mechanism to acknowledge the received messages.

The following is a diagram which shows how the communication is done in real time:

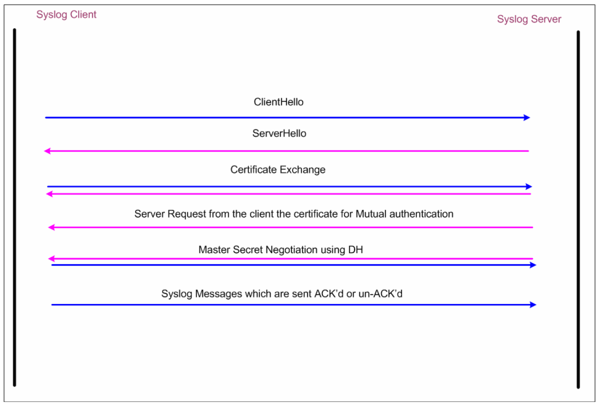
Syslog Client
Syslog Server
ClientHello
ServerHello
Certificate Exchange
Server Request from the client the certificate for Mutual authentication
Master Secret Negotiation using DH
Syslog Messages which are sent ACK'd or un-ACK'd

Figure (8)

Indeed, the TLS mechanism seems secure and provides reliability, nonetheless, TLS mechanism has the following drawbacks with Syslog:

- High overhead
- High impact on network
- Slower due to the permanent connection established with all the encryption and decryption taking place.
- Messages are sent at a slower pace.

Therefore, this solution would be ideal for a small environment that does not have high traffic and has high requirements. A better solution will be proposed in the following section.

8.3.2 Syslog-SecureCIA

This solution of Syslog-SecureCIA provides the following security mechanisms:

- Integrity
- Confidentiality
- Authenticity

This will be achieved by implementing a hybrid mechanism that uses TCP for key exchange for short periods of time and UDP for sending out Syslog messages.

The following are the main additions to the new protocol:

1. TCP Session establishment
2. User IDs with passwords on each Syslog host (client, relay or server)
3. Use of a Message Authentication protocol (like HMAC-MD5 or HMAC-SHA1) Note: the algorithm used can be changed in the future
4. Key generation algorithm
5. Key Exchange Algorithm
6. Encryption algorithm (such as 3DES & AES) Note: the algorithm used can be changed in the future
7. In the Syslog client/relay/server fields such as the following should be added:
   a. SyslogUser (this is the user)
   b. SyslogPassphrase (Hashed)
   c. SyslogAuthKey (used for Integrity and Authenticity)
   d. SyslogAuthProtocol
   e. SyslogPrivKey (used for Payload Encryption)
   f. SyslogPrivProtocol
8. Each Client/Relay should have one entity which contains the above fields
9. Each Server should be able to create many entities which contain the above fields. The number of entities should be limited by the amount of traffic sent to each server. Theoretically speaking, a maximum of 1000 entities should be created

The following diagram shows the relationship between the entities and client/relay model:

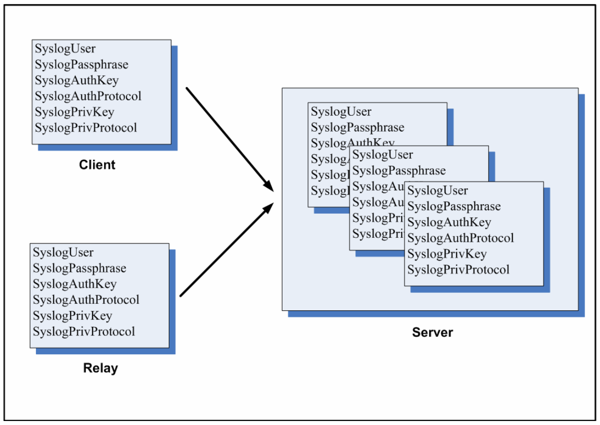

Figure (9)

The following shows the steps of how this new protocol should operate:

1. The Syslog Client/Relay generates a large prime number P and a primitive base of Mod P which is called G. A secret is also generated and shared between the Syslog entity which gets appended to the messages for message integrity and Authentication of origin. A message digest is created for all three values and appended to the message which will be sent (HMAC-MD5-96 or HMAC-SHA1-96 could be used here).

2. After the generation of the numbers, a TCP session is established by the entity (Client/Relay) to the other entity (the server), and timers for ACK start at the same time.

3. The entity authenticates the user which established the connection.

4. If the connection is not authenticated, the connection is dropped.

5. An ACK is sent back to the Client/Relay for both Authentication and connection establishment. If not sent, it will be repeated for 3 times, times and if the connection fails, a trap is generated.

6. If the connection is authenticated P, G ,secret and the message digest are transported across the session in clear text from the Client/Relay to the Server.

7. The Client/Relay will have a timer start from the moment it transfers the payload with all the contents to the Server. If an ACK is received before the expiration of the timer, the operation will be completed and another timer starts for a second ACK.

8. If the timer expires before the ACK is received, a TCP session is started again and the whole process is repeated (the process is repeated 3 times, however, if the 3rd time fails, a trap is generated to the Server).

9. If the first ACK is received by the Client/Relay successfully, that means that the payload is received by the Server. The Server then verifies that the message digest is correct and the contents are modified. If the content is modified, a second ACK will not be sent back to the Client/Relay and the timer expires on the Client/Relay side. On the other hand, if the content is received correctly, a second ACK is sent back to the Client/Relay which is the flag to start the negotiation of computed values.

10. The Client/Relay generates a large random integer (a) which is not shared. (G^a mod P) is computed and then a message digest is computed and attached to the payload and sent to the Server and a timer starts for an ACK.

11. If it is received successfully, an ACK is returned if not the same process of prior ACK is repeated here
(see steps 8 & 9). The Server makes sure that the payload is not modified on the way. If it is modified, a second ACK is not sent back so the timer of the Client/Relay will realize that it has expired and resends the message. If it is received correctly, a second ACK is sent back.

12. After the Server sends out the ACK, the Server generates a random large integer (b) which is not shared. (G^b mod P) is computed and then a message digest is computed and attached to the payload and sent to the Client/Relay.

13. If it is received successfully and ACK is returned if not the same process of prior ACK is repeated here
(see steps 8 & 9).

14. Both the Client/Relay and Server now compute the secret key as following:

- Client/Relay: (G^b mod P)^a mod P
- Server: (G^a mod P)^b mod P.

Both will generate the same key which can now be used in the SNMP privacy key attribute in the MIB.

15. Now the message can be sent using the UDP by encrypting the traffic using the new generated shared key, and can also be decrypted on the other side using the same shared key. (for encryption and authentication, the same mechanisms of SNMP are used in this context).

16. While the process of key exchange is occurring, the messages are buffered at the Client/Relay side and not sent until the process of key exchange has been either completed or not (which will result in using the older key).

The Following diagram shows the interaction of the Syslog Entities:

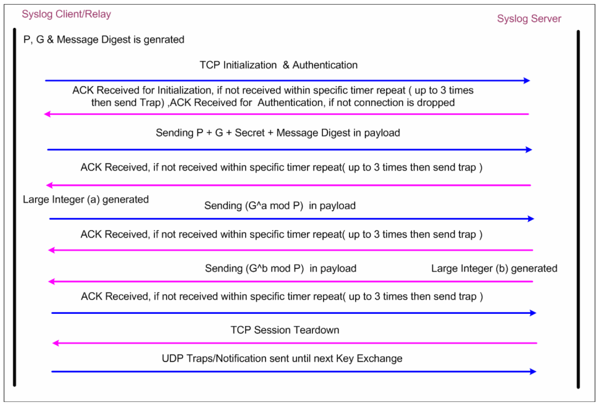
Syslog Client/Relay
Syslog Server
P, G & Message Digest is genrated
TCP Initialization & Authentication
ACK Received for Initialization, if not received within specific timer repeat (up to 3 times then send Trap) ,ACK Received for Authentication, if not connection is dropped
Sending P + G + Secret + Message Digest in payload
ACK Received, if not received within specific timer repeat(up to 3 times then send trap)
Large Integer (a) generated
Sending (G^a mod P) in payload
ACK Received, if not received within specific timer repeat(up to 3 times then send trap)
Sending (G^b mod P) in payload
Large Integer (b) generated
ACK Received, if not received within specific timer repeat(up to 3 times then send trap)
TCP Session Teardown
UDP Traps/Notification sent until next Key Exchange

Figure (10)

9. Summary

The additions over both SNMP and Syslog are considered very small in protocol and programming terms, yet they add extra security on top of both protocols which enables users to truly rely on both for current and future notifications which could be integrated into more complicated and different applications.

Since both are very similar in usage and architecture, the future would look better if both were unified and one secure reliable mechanism was introduced. As for now, since a lot of companies are already using both, it would be beneficial at least to truly secure both protocols by considering what we have presented in our research to produce a safer, better secured SNMP and Syslog.

10. References

[1] U. Blumenthal, B. Wijnen "RFC 3414", Standard, Internet Society, Dec. 2002, pp. 1-87.

[2] D. Harrington,R. Presuhn, B. Wijnen "RFC 3411", Standard, Internet Society, Dec. 2002, pp. 1-64.

[3] C. Lonvick "RFC 3164", Standard, Internet Society, Aug. 2001, pp. 1-29.

[4] D. New , M. Rose "RFC 3195", Standard, Internet Society, Nov. 2001, pp. 1-36.

[5] J. Case, M. Fedor , M. Schoffstall , J. Davin "RFC 1157", Standard, Internet Society, May. 1990, pp. 1-35.

[6] Douglas Mauro, Kevin Schmidt, *Essential SNMP Second Edition*, O'Reilly, California, September 2005.Chapter10

[7] Anand Deveriya, *An Overview of Syslog Protocol*, Cisco Press, Dec. 2005.

[8] Auto Secure Manager ,Manual, Enterasys Network, <http://www.enterasys.com/support/manuals/AutoSecMgr_2.1-web/asmhelp/docs/asm_cf_traps_informs.html>

[9] IPSec ,Article, Wikipedia, <http://en.wikipedia.org/wiki/IPsec>, Apr., 2007.

[10] Stunnel ,Article, Wikipedia, <http://en.wikipedia.org/wiki/Stunnel>, Feb., 2007.

[11] SSH ,Article, Wikipedia, <http://en.wikipedia.org/wiki/Ssh>, Apr., 2007.

[12] Secure Remote Streaming ,Article, <http://www.w00w00.org/files/sectools/SRS/>,

[13] Nsyslogd ,Article, ANU <http://coombs.anu.edu.au/~avalon/nsyslog.html>

[14] J. Kelsey, J. Callas , A. Clemm "draft-ietf-syslog-sign-21.txt", Draft, Internet Society, March. 2007, pp. 1-36.

[15] F. Miao , M. Yuzhi "draft-ietf-syslog-transport-tls-07.txt ", Draft, Internet Society, March. 2007, pp. 1-11.