Zeitschrift: Comtec : Informations- und Telekommunikationstechnologie = information and telecommunication technology

Herausgeber: Swisscom

Band: 77 (1999)

Heft: 7-8

Artikel: Security within SNMP version 3

Autor: Sellin, Rüdiger

DOI: https://doi.org/10.5169/seals-877032

Security Within SNMP Version 3

The new version 3 of SNMP, the Simple Network Management Protocol from the Internet Engineering Task Force (IETF), comes with a new architecture including a number of long expected security functions. The former SNMP version 1 has no means to guarantee neither a secure transmission of management commands nor a secure implementation of management applications without potential threats. Nonetheless, SNMP version 1 is still in widespread use and still popular because of its simplicity and robustness. It is expected that the new SNMP version 3 will supersede version 1 soon because it will meet the requirements of the growing SNMP users community much better than the previous two versions. This article describes the security parts of the new SNMP architecture.

RÜDIGER SELLIN, BERN

The SNMP History

SNMP version 1 (SNMPv1) has been very successful over the past decade. After the standard was launched in May 1990, SNMP gained more and more success especially on the LAN market (Local Area Network). Only after a few years more than 50 LAN equipment suppliers supported SNMP by putting SNMP agents on their routers, bridges, servers and hosts. It was the first time where systems management with one single management protocol became possible. Coupling this major advantage with SNMP's simplicity there were almost no doubts that SNMP is the industry standard management protocol for almost every multivendor LAN environment. Being that successful in the LAN market segment, SNMP increasingly got a foot into the WAN markets door (Wide Area Network). Many major data communications suppliers (like Cisco or Ascend) who offer, or computer manufacturers (like IBM) who use data communications equipment, deliver global solutions for broadband communications e.g. for ATM networks in both segments, LANs and WANs. Therefore it was only a question of time as to when SNMP would become a simple and easy-to-implement option for telecommunication networks as well. With the development of SNMP's version 2 (SNMPv2), the IETF tried to extend the capabilities of SNMPv1 by adding new PDUs (Protocol Data Units) for the transfer of bulk data (GET-BULK PDU) and for the manager-to-manager communication (INFORM PDU), and a new security concept.

Especially the new security concept which was developed between 1991 and 1992 [RFC 1351 to RFC 1353] drew the attention of the SNMP users because SNMPv1's recognised lack of security. But SNMPv2's misfortune was that the U.S. DoD (Department of Defence) which still has its hands on the Internet did not agree to publish the security part of SNMPv2 due to export rules within the USA. After a longer period of debating the ongoing negotiations between the participating parties did not lead to an acceptable compromise, so SNMPv2 was published without the security part. Therefore, the needs of the growing SNMP user's community still were not met. In addition, many dialects of SN-

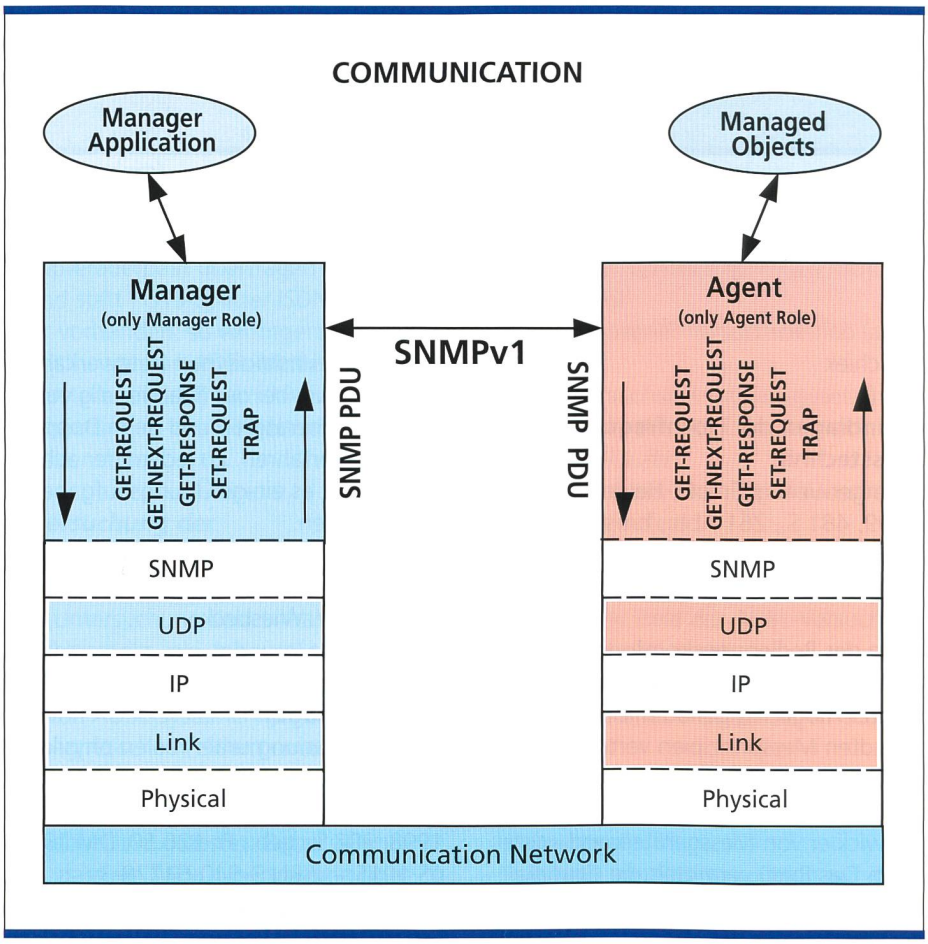

Fig. 1. Communication between manager and agent via SNPMv1. Abbreviations: SNMP: Simple Network Management Protocol; UDP: User Datagram Protocol; IP: Internet Protocol; PDU: Protocol Data Unit.

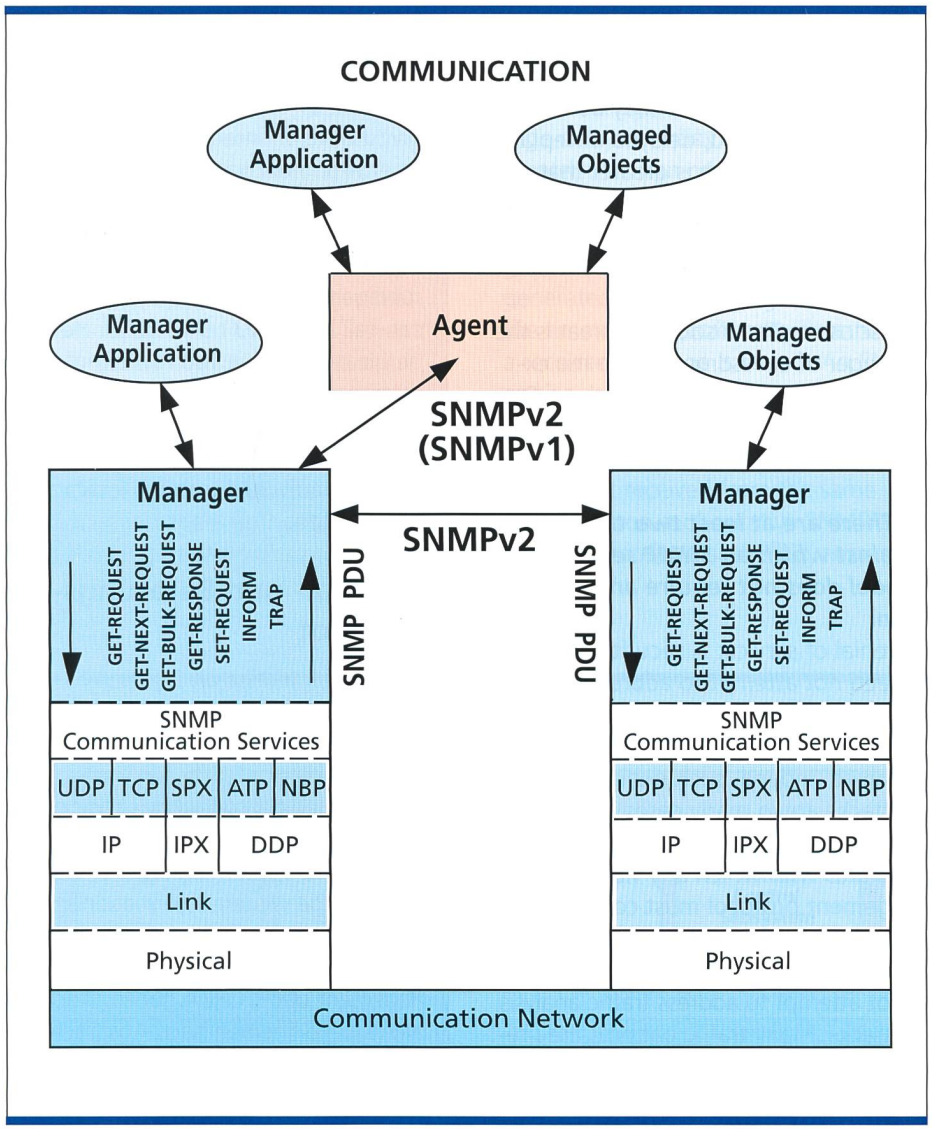

Fig. 2. Communication between Manager and agent via SNMPv2. Abbreviations: UDP: User Datagram Protocol; TCP: Transport Control Protocol; IP: Internet Protocol; PDU: Protocol Data Unit; SPX: Internet Packet Exchange; NBP: Name Binding Protocol; DDP: Datagram Delivery Protocol (SDX and IPX are Novell, NBP, ATP and DDP are Apple).

MPv2 appeared on the market which more or less led to an incompatibility. Considering the above mentioned background it is obvious that the specifications for SNMP version 3 (SNMPv3) were developed under high expectations. Therefore, the architecture for SNMPv3 [1] has to meet all the requirements which were already stated for SNMPv2, and it has to be in a way flexible to be backwards compatible at least with SNMPv1 and the "official" IETF specification of SNMPv2. SNMPv3 is an extensible SNMP framework which supplements the SNMPv2 framework, by supporting the following:

– a new SNMP message format,
– Security for messages,
– Access control, and
– Remote configuration of SNMP parameters.

Other SNMP frameworks, i.e., other configurations of implemented subsystems, are expected to be consistent with this architecture, too.

The SNMP Engine

An SNMP engine (in SNMPv1 called a protocol entity) as one part of the SNMP entity provides services for sending and receiving messages, authenticating and encrypting messages, and controlling access to managed objects. There is a one-to-one association between an SNMP engine and the SNMP entity which contains it (fig. 1).

The engine contains:

– a dispatcher,
– a message processing subsystem,
– a security subsystem, and
– an access control subsystem.

Within an administrative domain, an snmpEngineID is the unique and unambiguous identifier of an SNMP engine. Since there is a one-to-one association between SNMP engines and SNMP entities, it also uniquely and unambiguously identifies the SNMP entity within that administrative domain. Note that it is possible for SNMP entities in different administrative domains to have the same value for a snmpEngineID. In case that administrative domains are merged, it may be necessary to assign new values.

Dispatcher

There is only one dispatcher in an SNMP engine. It allows for concurrent support of multiple versions of SNMP messages in the SNMP engine. It does so by:

– sending and receiving SNMP messages to/from the network,
– determining the version of an SNMP message and interacting with the corresponding message processing model,
– providing an abstract interface to SNMP applications for delivery of a PDU to an application,
– providing an abstract interface for SNMP applications that allows them to send a PDU to a remote SNMP entity.

Message Processing Subsystem

The Message Processing Subsystem is responsible for preparing messages for sending, and extracting data from received messages. It potentially contains multiple message processing models as shown in figure 2.

Each message processing model defines the format of a particular version of an SNMP message and co-ordinates the preparation and extraction of each such version-specific message format.

Security Subsystem

The security subsystem provides security services such as the authentication and privacy of messages and potentially contains multiple security models as shown in the figure 3. One or more security models may be present.

A security model specifies the following:

– the threats against which it protects,
– the goals of its services,
– the security protocols used to provide security services such as

– authentication and privacy, and
– the mechanisms, procedures, and MIB objects used to provide a security service such as authentication or privacy.

Access Control Subsystem
The access control subsystem provides authorisation services by means of one or more access control models. An access control model defines a particular access decision function in order to support decisions regarding access rights.

SNMP Security Model
From the SNMP user's perspective, the extension of the former SNMP framework by adding useful security mechanisms is the major new feature of SNMPv3. The access control subsystem and the security subsystem guarantee this step forward in the evolution of SNMP. For this purpose, a security model was developed for the architecture of SNMPv3, protecting the network management application of a number of classical threats which apply to any network protocols.
Within the SNMP management framework, principal threats, secondary threats, and less important threats are considered [3]:

1. The principal threats against which any security model should provide protection are:
– Modification of information: The modification threat is the danger that some unauthorised entity may alter in-transit SNMP messages generated on behalf of an authorised principal in such a way as to effect unauthorised management operations, including falsifying the value of an object.
– Masquerade: The masquerade threat is the danger that management operations not authorised for some principal may be attempted by assuming the identity of another principal that has the appropriate authorisations.

2. Secondary threats against which any security model used within the SNMPv3 architecture should provide protection are:
– Message stream modification: The SNMP protocol is typically based upon a connectionless transport service which may operate over any subnetwork service. The re-ordering, delay or replay of messages can and does occur through the natural operation of many such subnetwork services. The message stream modification threat is the danger that messages may be maliciously re-ordered, delayed or replayed to an extent which is greater than can occur through the natural operation of a subnetwork service, in order to effect unauthorised management operations.
– Disclosure: The disclosure threat is the danger of eavesdropping on the exchanges between SNMP engines. Protecting against this threat may be required as a matter of local policy.

3. There are at least two threats against which an SNMP security model does not require any protection:
– Denial of service: A security model need not attempt to address the broad range of attacks by which service on behalf of authorised users is denied. Indeed, such denial-of-service attacks are in many cases indistinguishable from the type of network failures with which any viable management protocol must cope as a matter of course.
– Traffic analysis: A security model need not attempt to address traffic analysis attacks. Many traffic patterns are predictable – entities may be managed on a regular basis by a relatively small number of management stations – and therefore there is no significant advantage afforded by protecting against traffic analysis.

Security Services and Design Goals
Based on the above listed threats in the SNMP network management environment, the goals of this SNMP Security Model are as follows [3]:
– Provide for verification that each received SNMP message has not been modified during its transmission through the network.
– Provide for verification of the identity of the user on whose behalf a received SNMP message claims to have been generated.
– Provide for detection of received SNMP messages, which request or contain management information, whose time of generation was not recent.
– Provide, when necessary, that the contents of each received SNMP message are protected from disclosure.

The security services necessary to support these goals are as follows [3]:
– Data integrity is the provision of the property that data has not been altered or destroyed in an unauthorised manner, nor have data sequences been altered to an extent greater than can occur non-maliciously.
– Data origin authentication is the provision of the property that the claimed identity of the user, on whose behalf received data was originated, is corroborated.
– Data confidentiality is the provision of the property that information is not made available or disclosed to unauthorised individuals, entities, or processes.

Message timeliness and Limited replay protection is the provision of the property that a message whose generation time is outside of a specified time window is not accepted. Note that message reordering is not dealt with and can occur in normal conditions too.

SNMP Security Functions

Protection against Message Replay, Delay and Redirection [4]

In order to protect against message replay, delay and redirection, one of the SNMP engines involved in each communication is designated to be the authoritative SNMP engine. When an SNMP message contains a payload which expects a response (those messages that contain a Confirmed Class PDU), then the receiver of such messages is authoritative. When an SNMP message contains a payload which does not expect a response (those messages that contain an Unconfirmed Class PDU), then the sender of such a message is authoritative.

The following mechanisms provide for the detection of authenticated messages whose time of generation was not recent:

– To protect against the threat of message delay or replay (to an extent greater than can occur through normal operation), a set of timeliness indicators (for the authoritative SNMP engine) are included in each message generated. An SNMP engine evaluates the timeliness indicators to determine if a received message is recent. An SNMP engine may evaluate the timeliness indicators to ensure that a received message is at least as recent as the last message it received from the same source. A non-authoritative SNMP engine uses received authentic messages to advance its notion of the timeliness indicators at the remote authoritative source.

An SNMP engine must also use a mechanism to match incoming Responses to outstanding Requests and it MUST drop any Responses that do not match an outstanding request. For example, a msgID can be inserted in every message to cater for this functionality.

This protection against the threat of message delay or replay does neither imply nor provide any protection against unauthorised deletion or suppression of messages. Also, an SNMP engine may not be able to detect message reordering if all the messages involved are sent within the Time Window[1] interval. Other mechanisms defined independently of the security protocol can also be used to detect the re-ordering replay, deletion, or suppression of messages containing Set operations (e.g., the MIB variable snmpSetSerialNo).

Message Verification [4]

To verify that a message sent to/from one authoritative SNMP engine cannot be replayed to/as-if-from another authoritative SNMP engine, each message includes an identifier unique to the authoritative SNMP engine associated with the sender or intended recipient of the message. A message containing an Unconfirmed Class PDU sent by an authoritative SNMP engine to one non-authoritative SNMP engine can potentially be replayed to another non-authoritative SNMP engine. The latter non-authoritative SNMP engine might (if it knows about the same userName with the same secrets at the authoritative SNMP engine) as a result update its notion of timeliness indicators of the authoritative SNMP engine, but that is not considered a threat. In this case, A Report or Response message will be discarded by the Message Processing Model, because there should not be an outstanding Request message. A Trap will possibly be accepted. Again, that is not considered a threat, because the communication was authenticated and timely. It is as if the authoritative SNMP engine was configured to start sending Traps to the second SNMP engine, which theoretically can happen without the knowledge of the second

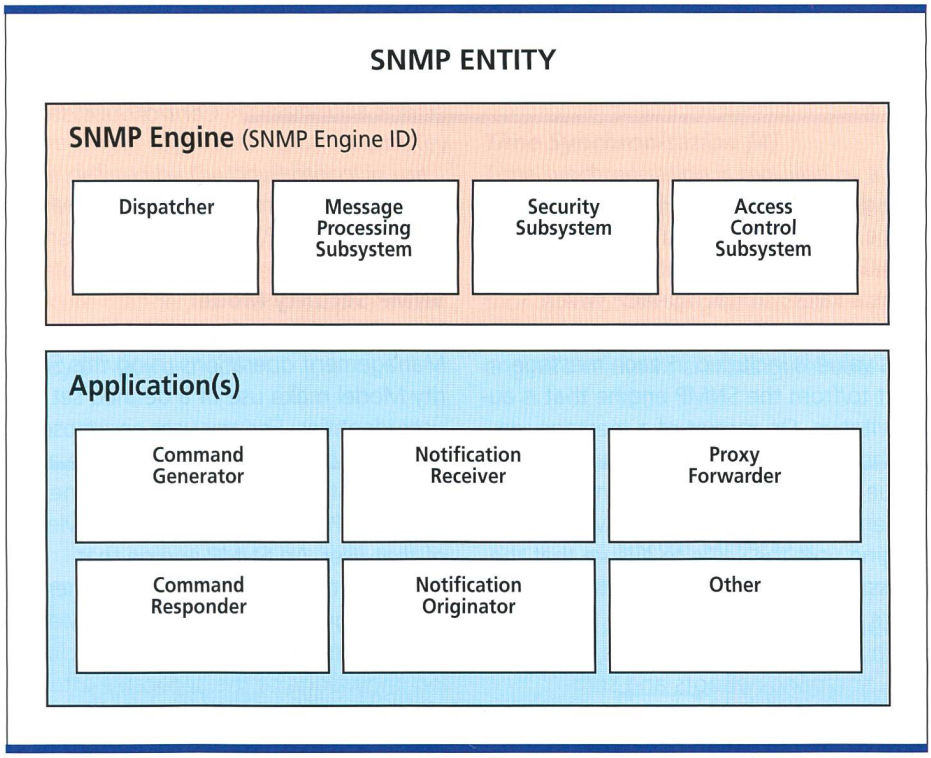

Fig. 3. An SNMP Entity and ist compenents.

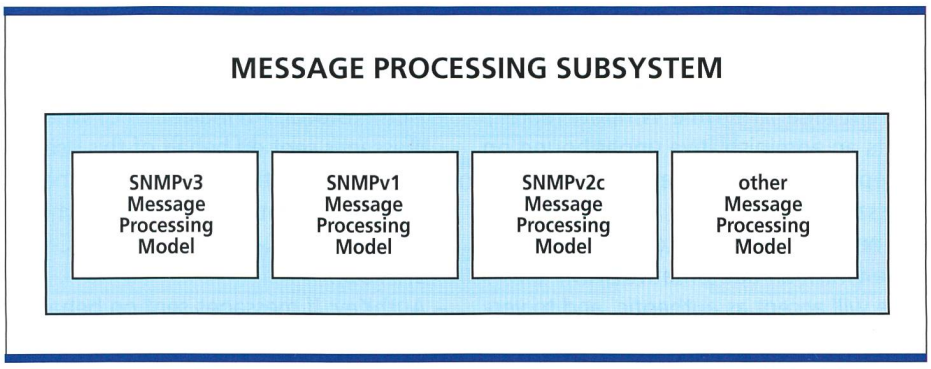

Fig. 4. Message Processing Subsystem.

[1] The Time Window is a value that specifies the window of time in which a message generated on behalf of any user is valid. The same value of the Time Window (150 seconds) is used for all users.

References: related IETF Requests for Comments (RFC) about SNMP Version 3

(can be downloaded from www.ietf.org free of charge)

[1] An Architecture for Describing SNMP Management frameworks, RFC 2271, Internet Engineering Task Force, Network Working Group, January 1998

[2] Message Processing and Dispatching for the Simple Network Management Protocol (SNMP), RFC 2272, Internet Engineering Task Force, Network Working Group, January 1998

[3] User-based Security Model (USM) for version 3 of the Simple Network Management Protocol (SNMPv3), RFC 2274, Internet Engineering Task Force, Network Working Group, January 1998

[4] User-based Security Model (USM) for version 3 of the Simple Network Management Protocol (SNMPv3), Internet-Draft <draft-ietf-snmpv3-usm-v2-04.txt> (will obsolete RFC2274), T. J. Watson/U. Blumenthal/B. Wijnen, IBM T. J. Watson Research, 20 January 1999

[5] View-based Access Control Model (VACM) for the Simple Network Management Protocol (SNMP), RFC 2275, Internet Engineering Task Force, Network Working Group, January 1998

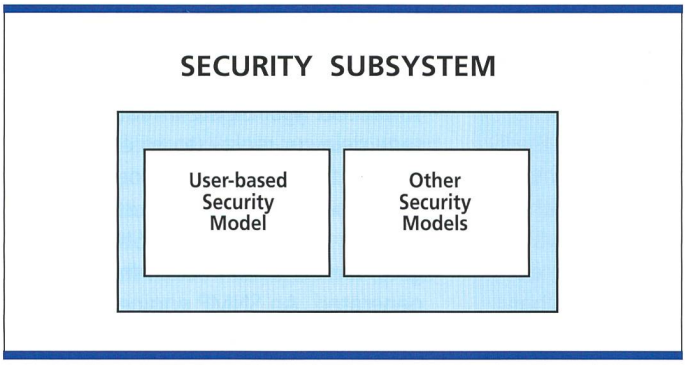

Fig. 5. Security Subsystem.

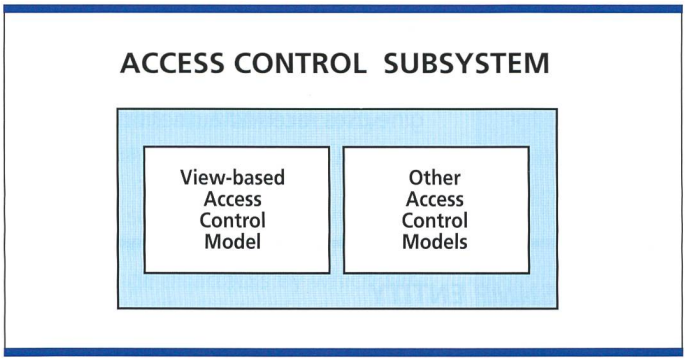

Fig. 6. Acess Control Subsystem.

SNMP engine anyway. In either cases, the second SNMP engine may not expect to receive this Trap, but is allowed to see the management information contained in it.

Detection of Messages which were not Recently Generated [4]

A set of time indicators are included in the message, indicating the time of generation. Messages without recent time indicators are not considered authentic. In addition, an SNMP engine MUST drop any responses that do not match an outstanding request. This however is the responsibility of the Message Processing Model.

The definitions given in [4] allow the same user to be defined on multiple SNMP engines. Each SNMP engine maintains a value, snmpEngineID, which uniquely identifies the SNMP engine. This value is included in each message sent to/from the SNMP engine that is authoritative. On receipt of a message, an authoritative SNMP engine checks the value to ensure that it is the intended recipient, and a non-authoritative SNMP engine uses the value to ensure that the message is processed using the correct state information.

Each SNMP engine maintains two values, snmpEngineBoots and snmpEngineTime, which taken together provide an indication of time at that SNMP engine. Both of these values are included in an authenticated message sent to or received from that SNMP engine. On receipt, the values are checked to ensure that the indicated timeliness value is within a Time Window of the current time. The Time Window represents an administrative upper bound on acceptable delivery delay for protocol messages.

For an SNMP engine to generate a message which an authoritative SNMP engine will accept as authentic, and to verify that a message received from that authoritative SNMP engine is authentic, such an SNMP engine must first achieve timeliness synchronisation with the authoritative SNMP engine.

SNMP Security Model

User-based Security Model Users [4]

Management operations using this Security Model make use of a defined set of user identities. For any user on whose behalf management operations are authorized at a particular SNMP engine, that SNMP engine must have knowledge of that user. An SNMP engine that wishes to communicate with another SNMP engine must also have knowledge of a user known to that engine, including knowledge of the applicable attributes of that user.

A user and its attributes are defined as follows:

- userName: A string representing the name of the user.
- SecurityName: A human-readable string representing the user in a format that is Security Model independent.
- AuthProtocol: An indication of whether messages sent on behalf of this user can be authenticated, and if so, the type of authentication protocol which is used (HMAC-MD5-96 or HMAC-SHA-96).
- AuthKey: If messages sent on behalf of this user can be authenticated, the (private) authentication key for use with the authentication protocol. Note that

a user's authentication key will normally be different at different authoritative SNMP engines. The authKey is not accessible via SNMP. The length requirements of the authKey are defined by the authProtocol in use.
- authKeyChange and authOwnKeyChange: The only way to remotely update the authentication key. Does that in a secure manner, so that the update can be completed without the need to employ privacy protection.
- PrivProtocol: An indication of whether messages sent on behalf of this user can be protected from disclosure, and if so, the type of privacy protocol which is used ([4] defines the CBC-DES Symmetric Encryption Protocol).
- PrivKey: If messages sent on behalf of this user can be en/decrypted, the (private) privacy key for use with the privacy protocol. Note that a user's privacy key will normally be different at different authoritative SNMP engines. The privKey is not accessible via SNMP. The length requirements of the privKey are defined by the privProtocol in use.
- privKeyChange and privOwnKeyChange: The only way to remotely update the encryption key. Does that in a secure manner, so that the update can be completed without the need to employ privacy protection.

Replay Protection [4]

Each SNMP engine maintains three objects:
- snmpEngineID, which (at least within an administrative domain) uniquely and unambiguously identifies an SNMP engine.
- snmpEngineBoots, which is a count of the number of times the SNMP engine has re-booted/re-initialized since snmpEngineID was last configured; and,
- snmpEngineTime, which is the number of seconds since the snmpEngineBoots counter was last incremented.

Each SNMP engine is always authoritative with respect to these objects in its own SNMP entity. It is the responsibility of a non-authoritative SNMP engine to synchronise with the authoritative SNMP engine, as appropriate. An authoritative SNMP engine is required to maintain the values of its snmpEngineID and snmpEngineBoots in non-volatile storage.

Time Synchronisation [4]

Time synchronisation is required by a non-authoritative SNMP engine in order to proceed with authentic communications. It has occurred when the non-authoritative SNMP engine has obtained a local notion of the authoritative SNMP engine's values of snmpEngineBoots and snmpEngineTime from the authoritative SNMP engine. These values must be (and remain) within the authoritative SNMP engine's Time Window. Therefore, the local notion of the authoritative SNMP engine's values must be kept loosely synchronised with the values stored at the authoritative SNMP engine. To keep a local copy of snmpEngineBoots and snmpEngineTime from the authoritative SNMP engine, a non-authoritative SNMP engine must also keep one local variable, latestReceivedEngineTime. This value records the highest value of snmpEngineTime that was received by the non-authoritative SNMP engine from the authoritative SNMP engine and is used to eliminate the possibility of replaying messages that would prevent the non-authoritative SNMP engine's notion of the snmpEngineTime from advancing. A non-authoritative SNMP engine must keep local notions of these values (snmpEngineBoots, snmpEngineTime and latestReceivedEngineTime) for each authoritative SNMP engine with which it wishes to communicate. Since each authoritative SNMP engine is uniquely and unambiguously identified by its value of snmpEngineID, the non-authoriative SNMP engine may use this value as a key in order to cache its local notions of these values.

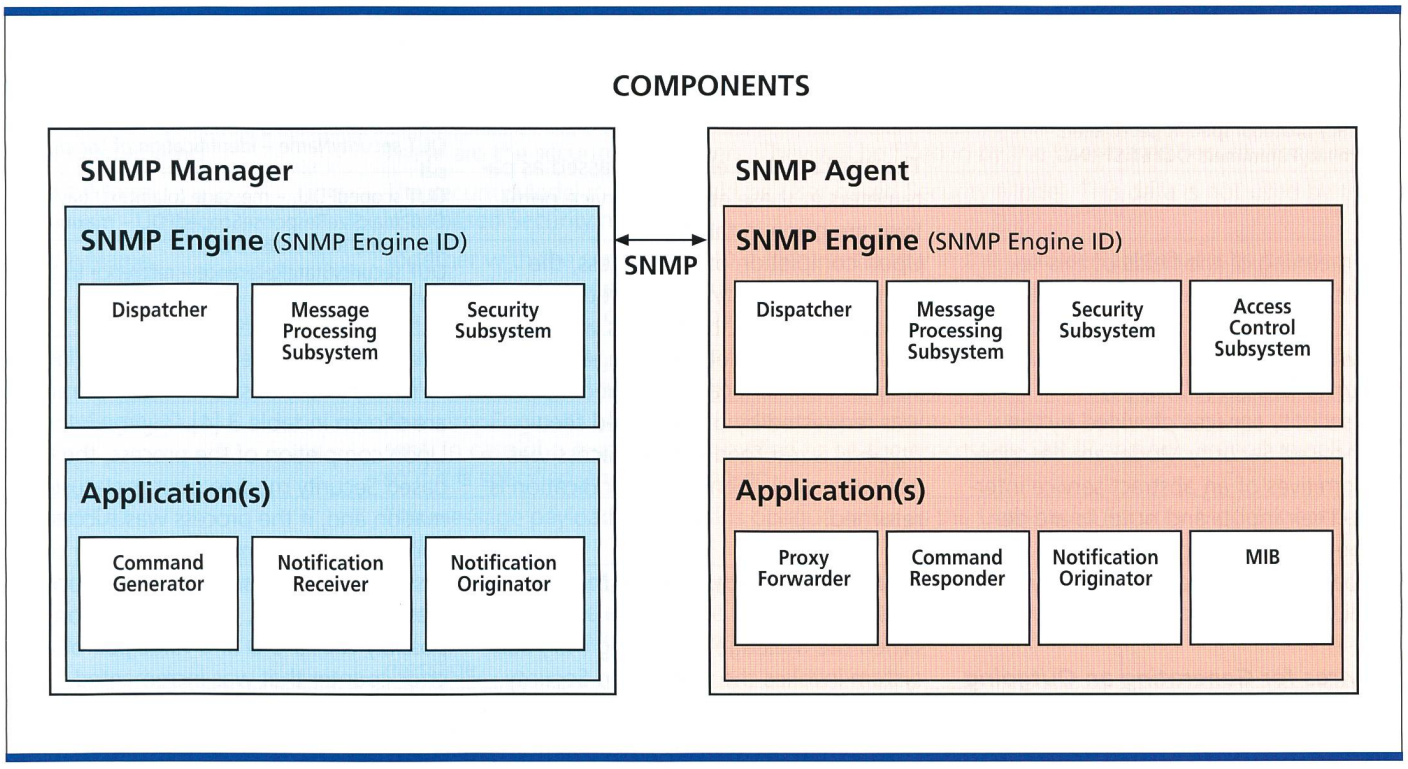

Fig. 7. Components of an SNMP manager and an SNMP agent.

Time synchronisation occurs as part of the procedures of receiving an SNMP message. As such, no explicit time synchronisation procedure is required by a non-authoritative SNMP engine. Note, that whenever the local value of snmpEngineID is changed (e.g., through discovery) or when secure communications are first established with an authoritative SNMP engine, the local values of snmpEngineBoots and latestReceivedEngineTime should be set to zero. This will cause the time synchronisation to occur when the next authentic message is received.

SNMP Messages Using SNMP Security Model [4]

The syntax of an SNMP message using this Security Model adheres to the common SNMP message format which is based on the SNMP Message Processing Model document. The field msgSecurityParameters in SNMPv3 messages has a data type of OCTET STRING. Its value is the BER (Basic Encoding Rules) serialisation of the following ASN.1 sequence:

```
USMSecurityParametersSyntax DEFINITIONS IMPLICIT TAGS ::= BEGIN
UsmSecurityParameters ::=
SEQUENCE {
– global User-based security parameters
msgAuthoritativeEngineID OCTET STRING,
msgAuthoritativeEngineBoots INTEGER
(0..2147483647),
msgAuthoritativeEngineTime INTEGER
(0..2147483647),
msgUserName OCTET STRING (SIZE(0..32),
– authentication protocol specific parameters
msgAuthenticationParameters OCTET STRING,
– privacy protocol specific parameters
msgPrivacyParameters OCTET STRING
}
END
```

The meaning of the fields of this sequence can be obtained from table 1 [4].

Services provided by the User-based Security Model (USM) [4]

The security services provided by the User-based Security Model are described as primitives of an abstract service interface. Their inputs and outputs are described as abstract data elements as they are passed in these abstract service primitives.

Services for Generating an Outgoing SNMP Message

When the Message Processing (MP) Subsystem invokes the User-based Security module to secure an outgoing SNMP message, it must use the appropriate service as provided by the Security module. The following two services are provided: A service to generate a Request message. The abstract service primitive is:

```
statusInformation = – success or errorIndication
generateRequestMsg(
INmessageProcessingModel – typically, SNMP version
INglobalData – message header, admin data
INmaxMessageSize – of the sending SNMP entity
INsecurityModel – for the outgoing message
INsecurityEngineID – authoritative SNMP entity
INsecurityName – on behalf of this principal
INsecurityLevel – Level of Security requested
INscopedPDU – message (plaintext) payload
OUT securityParameters – filled in by Security Module
OUT wholeMsg – complete generated message
OUT wholeMsgLength – length of generated message
)
```

A service to generate a Response message. The abstract service primitive is:

```
statusInformation = – success or errorIndication
generateResponseMsg(
INmessageProcessingModel – typically, SNMP version
INglobalData – message header, admin data
INmaxMessageSize – of the sending SNMP entity
INsecurityModel – for the outgoing message
INsecurityEngineID – authoritative SNMP entity
INsecurityName – on behalf of this principal
INsecurityLevel – Level of Security requested
INscopedPDU – message (plaintext) payload
INsecurityStateReference – reference to security state
– information from original
– request
OUT securityParameters – filled in by Security Module
OUT wholeMsg – complete generated message
OUT wholeMsgLength – length of generated message
)
```

The abstract data elements passed as parameters in these abstract service primitives are given in table 2 [4].
Upon completion of the process, the User-based Security module returns statusInformation. If the process was successful, the completed message with privacy and authentication applied if such was requested by the specified securityLevel is returned. If the process was not successful, then an errorIndication is returned.

Services for Processing an Incoming SNMP Message

When the Message Processing (MP) Subsystem invokes the User-based Security module to verify proper security of an incoming message, it must use the service provided for an incoming message. The abstract service primitive is:

SNMP-related articles and books from the author in German

TMN – die Basis für das Telekom-Management der Zukunft, R. Sellin, dpunkt-Verlag Heidelberg, 1995, ISBN 3-7685-4294-7

CMIP (Common Management Information Protocol) – das OSI Network Management Protokoll, R. Sellin, Technische Mitteilungen Telecom PTT, Juli 1992, Hallwag Verlags AG Bern

SNMP (Simple Network Management Protocol) – das Internet Network Management Protokoll, R. Sellin, Technische Mitteilungen Telecom PTT, Januar 1994, Hallwag Verlags AG Bern

CORBA – die Lösung für das Netzmanagement? R. Sellin, ComTec, November 1998, Hallwag Verlags AG Bern

ATM und ATM-Management – die Basis für das B-ISDN der Zukunft, R. Sellin, VDE-Verlag Offenbach/Berlin, 1997, ISBN

```
statusInformation = – errorIndication or success
– error counter OID/value if error
processIncomingMsg(
INmessageProcessingModel – typically, SNMP version
INmaxMessageSize – of the sending SNMP entity
INsecurityParameters – for the received message
INsecurityModel – for the received message
INsecurityLevel – Level of Security
INwholeMsg – as received on the wire
INwholeMsgLength – length as received on the wire
OUT securityEngineID – authoritative SNMP entity
OUT securityName – identification of the principal
OUT scopedPDU, – message (plaintext) payload
OUT maxSizeResponseScopedPDU – maximum size of the Response PDU
OUT securityStateReference – reference to security state
) – information, needed for response
```

The abstract data elements passed as parameters in the abstract service primitives are shown in table 3 [4].
Upon completion of the process, the User-based Security module returns statusInformation and, if the process was successful, additional data elements for further processing of the message or, If the process was not successful, an errorIndication, possibly with a OID and value pair of an error counter that was incremented.

Key Localisation Algorithm [4]

A localised key is a secret key shared between a user U and one authoritative

SNMP engine E. Even though a user may have only one password and therefore one key for the whole network, the actual secrets shared between the user and each authoritative SNMP engine will be different. This is achieved by key localisation (Localised-key). First, if a user uses a password, then the user's password is converted into a key Ku using one of the two algorithms (as described in Appendixes A.2.1 and A.2.2 of [4]).
To convert key Ku into a localised key Kul of user U at the authoritative SNMP engine E, one appends the snmpEngineID of the authoritative SNMP engine to the key Ku and then appends the key Ku to the result, thus enveloping the snmpEngineID within the two copies of user's key Ku. Then one runs a secure hash function (which one depends on the authentication protocol defined for this user U at authoritative SNMP engine E; this document defines two authentication protocols with their associated algorithms based on MD5 and SHA). The output of the hash-function is the localised key Kul for user U at the authoritative SNMP engine E.

Sequence fields	Meaning
msgAuthoritativeEngineID	specifies the snmpEngineID of the authoritative SNMP engine involved in the exchange of the message
msgAuthoritativeEngineBoots	specifies the snmpEngineBoots value at the authoritative SNMP engine involved in the exchange of the message
msgAuthoritativeEngineTime	specifies the snmpEngineTime value at the authoritative SNMP engine involved in the exchange of the message
msgUserName	specifies the user (principal) on whose behalf the message is being exchanged. Note that a zero-length userName will not match any user, but it can be used for snmpEngineID discovery
msgAuthenticationParameters	defined by the authentication protocol in use for the message, as defined by the usmUserAuthProtocol column in the user's entry in the usmUserTable
msgPrivacyParameters	defined by the privacy protocol in use for the message, as defined by the usmUserPrivProtocol column in the user's entry in the usmUserTable)

Table 1. ASN.1 sequence of the field msgSecurityParameters in SNMPv3 messages.

Data Elements / Parameters	Meaning
statusInformation	An indication of whether the encoding and securing of the message was successful. If not it is an indication of the problem.
messageProcessingModel	The SNMP version number for the message to be generated. This data is not used by the User-based Security module.
globalData	The message header (i.e., its administrative information). This data is not used by the User-based Security module.
maxMessageSize	The maximum message size as included in the message. This data is not used by the User-based Security module.
securityParameters	These are the security parameters. They will be filled in by the User-based Security module.
securityModel	The securityModel in use. Should be User-based Security Model. This data is not used by the User-based Security module.
securityName	Together with the snmpEngineID it identifies a row in the usmUserTable that is to be used for securing the message. The securityName has a format that is independent of the Security Model. In case of a response this parameter is ignored and the value from the cache is used.
securityLevel	The Level of Security from which the User-based Security module determines if the message needs to be protected from disclosure and if the message needs to be authenticated.
securityEngineID	The snmpEngineID of the authoritative SNMP engine to which a Request message is to be sent. In case of a response it is implied to be the processing SNMP engine's snmpEngineID and so if it is specified, then it is ignored.
scopedPDU	The message payload. The data is opaque as far as the User-based Security Model is concerned.
securityStateReference	A handle/reference to cachedSecurityData to be used when securing an outgoing Response message. This is the exact same handle/reference as it was generated by the User-based Security module when processing the incoming Request message to which this is the Response message.
wholeMsg	The fully encoded and secured message ready for sending on the wire.
wholeMsgLength	The length of the encoded and secured message (wholeMsg).

Table 2. Parameters of abstract service primitives of an outgoing SNMP message.

Data Elements / Parameters	Meaning
statusInformation	An indication of whether the process was successful or not. If not, then the statusInformation includes the OID and the value of the error counter that was incremented.
messageProcessingModel	The SNMP version number as received in the message. This data is not used by the User-based Security module.
MaxMessageSize	The maximum message size as included in the message. The User-based Security module uses this value to calculate the maxSizeResponseScopedPDU.
securityParameters	These are the security parameters as received in the message.
securityModel	The securityModel in use. Should be the User-based Security Model. This data is not used by the User-based Security module.
securityLevel	The Level of Security from which the User-based Security module determines if the message needs to be protected from disclosure and if the message needs to be authenticated.
wholeMsg	The whole message as it was received.
wholeMsgLength	The length of the message as it was received (wholeMsg).
securityEngineID	The snmpEngineID that was extracted from the field msgAuthoritativeEngineID and that was used to lookup the secrets in the usmUserTable.
securityName	The security name representing the user on whose behalf the message was received. The securityName has a format that is independent of the Security Model.
scopedPDU	The message payload. The data is opaque as far as the User-based Security Model is concerned.
maxSizeResponseScopedPDU	The maximum size of a scopedPDU to be included in a possible Response message. The User-based Security module calculates this size based on the msgMaxSize (as received in the message) and the space required for the message header (including the securityParameters) for such a Response message.
securityStateReference	A handle/reference to cachedSecurityData to be used when securing an outgoing Response message. When the Message Processing Subsystem calls the User-based Security module to generate a response to this incoming message it must pass this handle/reference.

Table 3. Parameters of abstract service primitives of an incoming SNMP message.

Zusammenfassung

Sicherheit bei der SNMP-Version 3

Die neue Version 3 von SNMP, dem Simple Network Management Protokoll der Internet Engineering Task Force (IETF), weist eine neue Architektur mit einigen seit langem erwarteten Sicherheitsfunktionen auf. Die frühere SNMP-Version 1 kann weder die sichere Übertragung von Management-Befehlen noch die sichere Implementierung von Management-Applikationen ohne potentielle Risiken garantieren. Trotzdem ist die SNMP-Version 1 aber noch immer weit verbreitet und beliebt, denn sie ist einfach und robust. Sie soll jedoch schon bald durch die neue SNMP-Version 3 ersetzt werden, da diese den Bedürfnissen des ständig wachsenden SNMP-Benutzerkreises weit besser entspricht als ihre beiden Vorgänger. Der folgende Artikel beschreibt die Sicherheitsaspekte der neuen SNMP-Architektur.

Outlook

Within SNMP version 3 (SNMPv3) a number of former or completely new defined security functions have been introduced. With the implementation of these functions, the lack of security which the former versions suffered from will disappear. Although former version 1 still dominates the market, it can be foreseen that the new SNMPv3 will overcome the older protocol versions because it offers features which have been expected by the SNMP user's community since years. Major vendors have already announced their will to offer SNMPv3-based network management products during this year. But the most interesting question from the security perspective is: Will SNMPv3 be able to cover the extended security needs of the growing SNMP user's community? We will see soon. 9.4

Rüdiger Sellin, *dipl. Ing., schloss das Studium der Nachrichtentechnik 1986 erfolgreich ab und ist seitdem in den Branchen Telekommunikation und angewandte Informatik tätig. Er bekleidete verschiedene Positionen bei Netzbetreibern und Systemhäusern in Deutschland und in der Schweiz, unter anderem als Systems Engineer in der OSI-Entwicklung und als Product Manager im Marketing von Network Support Systems. Rüdiger Sellin ist seit 1992 bei Swisscom AG beschäftigt und hier seit 1. Juli 1999 als Consultant Innovations bei Marketing & Sales, Major Accounts, Consulting & Design für das Aufspüren und die Nutzung neuester Trends und Techniken zum Vorteil der grössten Geschäftskunden von Swisscom mitverantwortlich. Er ist zudem Autor von zwei Fachbüchern zu den Themen ATM und TMN sowie Verfasser von zahlreichen Fachbeiträgen für Kommunikationsmagazine im In- und Ausland. Er leitet darüber hinaus in Westeuropa Fachseminare im Gebiet der Telekommunikation und tritt gelegentlich als Referent an internationalen Kongressen auf. Rüdiger Sellin ist unter der E-Mail-Adresse ruediger.sellin@swisscom.com erreichbar.*