Network Working Group
Request for Comments: 1352

J. Galvin
Trusted Information Systems, Inc.
K. McCloghrie
Hughes LAN Systems, Inc.
J. Davin
MIT Laboratory for Computer Science

July 1992

SNMP Security Protocols

Status of this Memo

This document specifies an IAB standards track protocol for the Internet community, and requests discussion and suggestions for improvements. Please refer to the current edition of the "IAB Official Protocol Standards" for the standardization state and status of this protocol. Distribution of this memo is unlimited.

Table of Contents

1. Abstract

The Simple Network Management Protocol (SNMP) specification [1] allows for the protection of network management operations by a variety of security protocols. The SNMP administrative model described in [2] provides a framework for securing SNMP network management. In the context of that framework, this memo defines protocols to support the following three security services:

- o data integrity,
- o data origin authentication, and
- o data confidentiality.

Please send comments to the SNMP Security Developers mailing list (snmp-sec-dev@tis.com).

2. Introduction

In the model described in [2], each SNMP party is, by definition, associated with a single authentication protocol. The authentication protocol provides a mechanism by which SNMP management communications transmitted by the party may be reliably identified as having originated from that party. The authentication protocol defined in this memo also reliably determines that the message received is the message that was sent.

Similarly, each SNMP party is, by definition, associated with a single privacy protocol. The privacy protocol provides a mechanism by which SNMP management communications transmitted to said party are protected from disclosure. The privacy protocol in this memo specifies that only authenticated messages may be protected from disclosure.

These protocols are secure alternatives to the so-called "trivial" protocol defined in [1].

USE OF THE TRIVIAL PROTOCOL ALONE DOES NOT CONSTITUTE SECURE NETWORK MANAGEMENT. THEREFORE, A NETWORK MANAGEMENT SYSTEM THAT IMPLEMENTS ONLY THE TRIVIAL PROTOCOL IS NOT CONFORMANT TO THIS SPECIFICATION.

The Digest Authentication Protocol is described in Section 4. It provides a data integrity service by transmitting a message digest -- computed by the originator and verified by the recipient -- with each SNMP message. The data origin authentication service is provided by prefixing the message with a secret value known only to the originator and recipient, prior to computing the digest. Thus, data integrity is supported explicitly while data origin authentication is supported implicitly in the verification of the digest.

The Symmetric Privacy Protocol is described in Section 5. It protects messages from disclosure by encrypting their contents according to a secret cryptographic key known only to the originator and recipient. The additional functionality afforded by this protocol is assumed to justify its additional computational cost.

The Digest Authentication Protocol depends on the existence of loosely synchronized clocks between the originator and recipient of a message. The protocol specification makes no assumptions about the strategy by which such clocks are synchronized. Section 6.3 presents one strategy that is particularly suited to the demands of SNMP network management.

Both protocols described here require the sharing of secret information between the originator of a message and its recipient. The protocol specifications assume the existence of the necessary secrets. The selection of such secrets and their secure distribution to appropriate parties may be accomplished by a variety of strategies. Section 6.4 presents one such strategy that is particularly suited to the demands of SNMP network management.

2.1 Threats

Several of the classical threats to network protocols are applicable to the network management problem and therefore would be applicable to any SNMP security protocol. Other threats are not applicable to the network management problem. This section discusses principal threats, secondary threats, and threats which are of lesser importance.

The principal threats against which any SNMP security protocol should provide protection are:

Modification of Information.

The SNMP protocol provides the means for management stations to interrogate and to manipulate the value of objects in a managed agent. The modification threat is the danger that some party may alter in-transit messages generated by an authorized party in such a way as to effect unauthorized management operations, including falsifying the value of an object.

Masquerade.

The SNMP administrative model includes an access control model. Access control necessarily depends on knowledge of the origin of a message. The masquerade threat is the danger that management operations not authorized for some party may be attempted by that party by assuming the identity of another party that has the appropriate authorizations.

Two secondary threats are also identified. The security protocols defined in this memo do provide protection against:

Message Stream Modification.

The SNMP protocol is based upon connectionless transport services. The message stream modification threat is the danger that messages may be arbitrarily re-ordered, delayed or replayed to effect unauthorized management operations. This threat may arise either by the work of a malicious attacker or by the natural operation of a subnetwork service.

Disclosure.

The disclosure threat is the danger of eavesdropping on the exchanges between managed agents and a management station. Protecting against this threat is mandatory when the SNMP is used to administer private parameters on which its security is based. Protecting against the disclosure threat may also be required as a matter of local policy.

There are at least two threats that a SNMP security protocol need not protect against. The security protocols defined in this memo do not provide protection against:

Denial of Service.

A SNMP security protocol need not attempt to address the broad range of attacks by which service to authorized parties is denied. Indeed, such denial-of-service attacks are in many cases indistinguishable from the type of network failures with which any viable network management protocol must cope as a matter of course.

Traffic Analysis.

In addition, a SNMP security protocol need not attempt to address traffic analysis attacks. Indeed, many traffic patterns are predictable -- agents 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.

2.2 Goals and Constraints

Based on the foregoing account of threats in the SNMP network management environment, the goals of a SNMP security protocol are enumerated below.

1. The protocol should provide for verification that each received SNMP message has not been modified during its transmission through the network in such a way that an unauthorized management operation might result.

2. The protocol should provide for verification of the identity of the originator of each received SNMP message.

3. The protocol should provide that the apparent time of generation for each received SNMP message is recent.

4. The protocol should provide that the apparent time of generation for each received SNMP message is subsequent to that for all previously delivered messages of similar origin.

5. The protocol should provide, when necessary, that the contents of each received SNMP message are protected from disclosure.

In addition to the principal goal of supporting secure network management, the design of any SNMP security protocol is also influenced by the following constraints:

1. When the requirements of effective management in times of network stress are inconsistent with those of security, the former are preferred.

2. Neither the security protocol nor its underlying security mechanisms should depend upon the ready availability of other network services (e.g., Network Time Protocol (NTP) or secret/key management protocols).

3. A security mechanism should entail no changes to the basic SNMP network management philosophy.

2.3 Security Services

The security services necessary to support the goals of a SNMP security protocol are as follows.

Data Integrity is the provision of the property that data and data sequences have not been altered or destroyed in an unauthorized manner.

Data Origin Authentication is the provision of the property that the claimed origin of received data is corroborated.

Data Confidentiality is the provision of the property that information is not made available or disclosed to unauthorized individuals, entities, or processes.

The protocols specified in this memo require both data integrity and data origin authentication to be used at all times. For these protocols, it is not possible to realize data integrity without data origin authentication, nor is it possible to realize data origin authentication without data integrity.

Further, there is no provision for data confidentiality without both data integrity and data origin authentication.

2.4 Mechanisms

The security protocols defined in this memo employ several types of mechanisms in order to realize the goals and security services described above:

- o In support of data integrity, a message digest algorithm is required. A digest is calculated over an appropriate portion of a SNMP message and included as part of the message sent to the recipient.

- o In support of data origin authentication and data integrity, the portion of a SNMP message that is digested is first prefixed with a secret value shared by the originator of that message and its intended recipient.

- o To protect against the threat of message reordering, a timestamp value is included in each message generated. A recipient evaluates the timestamp to determine if the message is recent and it uses the timestamp to determine if the message is ordered relative to other messages it has received. In conjunction with other readily available information (e.g., the request-id), the timestamp also indicates whether or not the message is a replay of a previous message. This protection against the threat of message reordering implies no protection against unauthorized deletion or suppression of messages.

- In support of data confidentiality, a symmetric encryption algorithm is required. An appropriate portion of the message is encrypted prior to being transmitted to its recipient.

The security protocols in this memo are defined independently of the particular choice of a message digest and encryption algorithm -- owing principally to the lack of a suitable metric by which to evaluate the security of particular algorithm choices. However, in the interests of completeness and in order to guarantee interoperability, Sections 2.4.1 and 2.4.2 specify particular choices, which are considered acceptably secure as of this writing. In the future, this memo may be updated by the publication of a memo specifying substitute or alternate choices of algorithms, i.e., a replacement for or addition to the sections below.

2.4.1 Message Digest Algorithm

In support of data integrity, the use of the MD5 [3] message digest algorithm is chosen. A 128-bit digest is calculated over the designated portion of a SNMP message and included as part of the message sent to the recipient.

An appendix of [3] contains a C Programming Language implementation of the algorithm. This code was written with portability being the principal objective. Implementors may wish to optimize the implementation with respect to the characteristics of their hardware and software platforms.

The use of this algorithm in conjunction with the Digest Authentication Protocol (see Section 4) is identified by the ASN.1 object identifier value md5AuthProtocol, defined in [4].

For any SNMP party for which the authentication protocol is md5AuthProtocol, the size of its private authentication key is 16 octets.

Within an authenticated management communication generated by such a party, the size of the authDigest component of that communication (see Section 4) is 16 octets.

2.4.2 Symmetric Encryption Algorithm

In support of data confidentiality, the use of the Data Encryption Standard (DES) in the Cipher Block Chaining mode of operation is chosen. The designated portion of a SNMP message is encrypted and included as part of the message sent to the recipient.

Two organizations have published specifications defining the DES: the National Institute of Standards and Technology (NIST) [5] and the American National Standards Institute [6]. There is a companion Modes of Operation specification for each definition (see [7] and [8], respectively).

The NIST has published three additional documents that implementors may find useful.

- o There is a document with guidelines for implementing and using the DES, including functional specifications for the DES and its modes of operation [9].

- o There is a specification of a validation test suite for the DES [10]. The suite is designed to test all aspects of the DES and is useful for pinpointing specific problems.

- o There is a specification of a maintenance test for the DES [11]. The test utilizes a minimal amount of data and processing to test all components of the DES. It provides a simple yes-or-no indication of correct operation and is useful to run as part of an initialization step, e.g., when a computer reboots.

The use of this algorithm in conjunction with the Symmetric Privacy Protocol (see Section 5) is identified by the ASN.1 object identifier value desPrivProtocol, defined in [4].

For any SNMP party for which the privacy protocol is desPrivProtocol, the size of the private privacy key is 16 octets, of which the first 8 octets are a DES key and the second 8 octets are a DES Initialization Vector. The 64-bit DES key in the first 8 octets of the private key is a 56 bit quantity used directly by the algorithm plus 8 parity bits -- arranged so that one parity bit is the least significant bit of each octet. The setting of the parity bits is ignored.

The length of the octet sequence to be encrypted by the DES must be an integral multiple of 8. When encrypting, the data should be padded at the end as necessary; the actual pad value is insignificant.

If the length of the octet sequence to be decrypted is not an integral multiple of 8 octets, the processing of the octet sequence should be halted and an appropriate exception noted. Upon decrypting, the padding should be ignored.

3. SNMP Party

Recall from [2] that a SNMP party is a conceptual, virtual execution context whose operation is restricted (for security or other purposes) to an administratively defined subset of all possible operations of a particular SNMP protocol entity. A SNMP protocol entity is an actual process which performs network management operations by generating and/or responding to SNMP protocol messages in the manner specified in [1]. Architecturally, every SNMP protocol entity maintains a local database that represents all SNMP parties known to it.

A SNMP party may be represented by an ASN.1 value with the following syntax.

```
SnmpParty ::= SEQUENCE {
  partyIdentity
     OBJECT IDENTIFIER,
  partyTDomain
     OBJECT IDENTIFIER,
  partyTAddr
     OCTET STRING,
  partyProxyFor
     OBJECT IDENTIFIER,
  partyMaxMessageSize
     INTEGER,
  partyAuthProtocol
     OBJECT IDENTIFIER,
  partyAuthClock
     INTEGER,
  partyAuthLastMsg
     INTEGER,
  partyAuthNonce
     INTEGER,
  partyAuthPrivate
     OCTET STRING,
  partyAuthPublic
     OCTET STRING,
  partyAuthLifetime
```

```
            INTEGER,
         partyPrivProtocol
            OBJECT IDENTIFIER,
         partyPrivPrivate
            OCTET STRING,
         partyPrivPublic
            OCTET STRING
       }
```

For each SnmpParty value that represents a SNMP party, the generic significance of each of its components is defined in [2]. For each SNMP party that supports the generation of messages using the Digest Authentication Protocol, additional, special significance is attributed to certain components of that party’s representation:

- o Its partyAuthProtocol component is called the authentication protocol and identifies a combination of the Digest Authentication Protocol with a particular digest algorithm (such as that defined in Section 2.4.1). This combined mechanism is used to authenticate the origin and integrity of all messages generated by the party.

- o Its partyAuthClock component is called the authentication clock and represents a notion of the current time that is specific to the party.

- o Its partyAuthLastMsg component is called the last-timestamp and represents a notion of time associated with the most recent, authentic protocol message generated by the party.

- o Its partyAuthNonce component is called the nonce and represents a monotonically increasing integer associated with the most recent, authentic protocol message generated by the party. The nonce associated with a particular message distinguishes it among all others transmitted in the same unit time interval.

- o Its partyAuthPrivate component is called the private authentication key and represents any secret value needed to support the Digest Authentication Protocol and associated digest algorithm.

- o Its partyAuthPublic component is called the public authentication key and represents any public value that may be needed to support the authentication protocol.

This component is not significant except as suggested in Section 6.4.

- o Its partyAuthLifetime component is called the lifetime and represents an administrative upper bound on acceptable delivery delay for protocol messages generated by the party.

For each SNMP party that supports the receipt of messages via the Symmetric Privacy Protocol, additional, special significance is attributed to certain components of that party's representation:

- o Its partyPrivProtocol component is called the privacy protocol and identifies a combination of the Symmetric Privacy Protocol with a particular encryption algorithm (such as that defined in Section 2.4.2). This combined mechanism is used to protect from disclosure all protocol messages received by the party.

- o Its partyPrivPrivate component is called the private privacy key and represents any secret value needed to support the Symmetric Privacy Protocol and associated encryption algorithm.

- o Its partyPrivPublic component is called the public privacy key and represents any public value that may be needed to support the privacy protocol. This component is not significant except as suggested in Section 6.4.

4. Digest Authentication Protocol

This section describes the Digest Authentication Protocol. It provides both for verifying the integrity of a received message (i.e., the message received is the message sent) and for verifying the origin of a message (i.e., the reliable identification of the originator). The integrity of the message is protected by computing a digest over an appropriate portion of a message. The digest is computed by the originator of the message, transmitted with the message, and verified by the recipient of the message.

A secret value known only to the originator and recipient of the message is prefixed to the message prior to the digest computation. Thus, the origin of the message is known implicitly with the verification of the digest.

Recall from [2] that a SNMP management communication is represented by an ASN.1 value with the following syntax.

```
SnmpMgmtCom ::= [1] IMPLICIT SEQUENCE {
  dstParty
     OBJECT IDENTIFIER,
  srcParty
     OBJECT IDENTIFIER,
  pdu   PDUs
}
```

For each SnmpMgmtCom value that represents a SNMP management communication, the following statements are true:

- o Its dstParty component is called the destination and identifies the SNMP party to which the communication is directed.

- o Its srcParty component is called the source and identifies the SNMP party from which the communication is originated.

- o Its pdu component has the form and significance attributed to it in [1].

Recall from [2] that a SNMP authenticated management communication is represented by an ASN.1 value with the following syntax.

```
SnmpAuthMsg ::= [1] IMPLICIT SEQUENCE {
  authInfo
     ANY, - defined by authentication protocol
  authData
     SnmpMgmtCom
}
```

For each SnmpAuthMsg value that represents a SNMP authenticated management communication, the following statements are true:

- o Its authInfo component is called the authentication information and represents information required in support of the authentication protocol used by the SNMP party originating the message. The detailed significance of the authentication information is specific to the authentication protocol in use; it has no effect on the application semantics of the communication other than its use by the authentication protocol in determining whether the communication is authentic or not.

- o Its authData component is called the authentication data and represents a SNMP management communication.

In support of the Digest Authentication Protocol, an authInfo component is of type AuthInformation:

```
AuthInformation ::= [1] IMPLICIT SEQUENCE {
  authTimestamp
     INTEGER (0..2147483647),
  authNonce
     INTEGER (0..2147483647),
  authDigest
     OCTET STRING
}
```

For each AuthInformation value that represents authentication information, the following statements are true:

- o Its authTimestamp component is called the authentication timestamp and represents the time of the generation of the message according to the partyAuthClock of the SNMP party that originated it. Note that the granularity of the authentication timestamp is 1 second.

- o Its authNonce component is called the authentication nonce and represents a non-negative integer value evaluated according to the authTimestamp value. In order not to limit transmission frequency of management communications to the granularity of the authentication timestamp, the authentication nonce is provided to differentiate between multiple messages sent with the same value of authTimestamp. The authentication nonce is a monotonically increasing sequence number, that is reset for each new authentication timestamp value.

- o Its authDigest component is called the authentication digest and represents the digest computed over an appropriate portion of the message, where the message is temporarily prefixed with a secret value for the purposes of computing the digest.

4.1 Generating a Message

This section describes the behavior of a SNMP protocol entity when it acts as a SNMP party for which the authentication protocol is administratively specified as the Digest Authentication Protocol. Insofar as the behavior of a SNMP protocol entity when transmitting protocol messages is defined generically in [2], only those aspects of that behavior that are specific to the Digest Authentication Protocol are described below. In particular, this section describes the encapsulation of a SNMP management communication into a SNMP authenticated management communication.

According to [2], a SnmpAuthMsg value is constructed during Step 3 of generic processing. In particular, it states the authInfo component is constructed according to the authentication protocol identified for the SNMP party originating the message. When the relevant authentication protocol is the Digest Authentication Protocol, the procedure performed by a SNMP protocol entity whenever a management communication is to be transmitted by a SNMP party is as follows.

1. The local database is consulted to determine the authentication clock, last-timestamp, nonce, and private authentication key (extracted, for example, according to the conventions defined in Section 2.4.1) of the SNMP party originating the message.

2. The authTimestamp component is set to the retrieved authentication clock value.

3. If the last-timestamp is equal to the authentication clock, the nonce is incremented. Otherwise the nonce is set to zero. The authNonce component is set to the nonce value. In the local database, the originating SNMP party’s nonce and last-timestamp are set to the nonce value and the authentication clock, respectively.

4. The authentication digest is temporarily set to the private authentication key. The SnmpAuthMsg value is serialized according to the conventions of [12] and [1]. A digest is computed over the octet sequence representing that serialized value using, for example, the algorithm specified in Section 2.4.1. The authDigest component is set to the computed digest value.

As set forth in [2], the SnmpAuthMsg value is then encapsulated according to the appropriate privacy protocol into a SnmpPrivMsg value. This latter value is then serialized and transmitted to the receiving SNMP party.

4.2 Receiving a Message

This section describes the behavior of a SNMP protocol entity upon receipt of a protocol message from a SNMP party for which the authentication protocol is administratively specified as the Digest Authentication Protocol. Insofar as the behavior of a SNMP protocol entity when receiving protocol messages is defined generically in [2], only those aspects of that behavior that are specific to the Digest Authentication Protocol are described below.

According to [2], a SnmpAuthMsg value is evaluated during Step 9 of generic processing. In particular, it states the SnmpAuthMsg value is evaluated according to the authentication protocol identified for the SNMP party that originated the message. When the relevant authentication protocol is the Digest Authentication Protocol, the procedure performed by a SNMP protocol entity whenever a management communication is received by a SNMP party is as follows.

1. If the ASN.1 type of the authInfo component is not AuthInformation, the message is evaluated as unauthentic. Otherwise, the authTimestamp, authNonce, and authDigest components are extracted from the SnmpAuthMsg value.

2. The local database is consulted to determine the authentication clock, last-timestamp, nonce, private authentication key (extracted, for example, according to the conventions defined in Section 2.4.1), and lifetime of the SNMP party that originated the message.

3. If the authTimestamp component plus the lifetime is less than the authentication clock, the message is evaluated as unauthentic.

4. If the authTimestamp component is less than the last-timestamp recorded for the originating party in the local database, the message is evaluated as unauthentic.

5. If the authTimestamp component is equal to the last-timestamp and if the authNonce component is less than or equal to the nonce, the message is evaluated as unauthentic.

6. The authDigest component is extracted and temporarily recorded.

7. A new SnmpAuthMsg value is constructed such that its authDigest component is set to the private

authentication key and its other components are set to the value of the corresponding components in the received SnmpAuthMsg value. This new SnmpAuthMsg value is serialized according to the conventions of [12] and [1]. A digest is computed over the octet sequence representing that serialized value using, for example, the algorithm specified in Section 2.4.1.

8. If the computed digest value is not equal to the previously recorded digest value, the message is evaluated as unauthentic.

9. The message is evaluated as authentic.

10. The last-timestamp and nonce values locally recorded for the originating SNMP party are set to the authTimestamp value and the authNonce value, respectively.

11. The authentication clock value locally recorded for the originating SNMP party is advanced to the authTimestamp value if this latter exceeds the recorded value.

If the SnmpAuthMsg value is evaluated as unauthentic, an authentication failure is noted and the received message is discarded without further processing. Otherwise, processing of the received message continues as specified in [2].

5. Symmetric Privacy Protocol

This section describes the Symmetric Privacy Protocol. It provides for protection from disclosure of a received message. An appropriate portion of the message is encrypted according to a secret key known only to the originator and recipient of the message.

This protocol assumes the underlying mechanism is a symmetric encryption algorithm. In addition, the message to be encrypted must be protected according to the conventions of the Digest Authentication Protocol.

Recall from [2] that a SNMP private management communication is represented by an ASN.1 value with the following syntax.

```
SnmpPrivMsg ::= [1] IMPLICIT SEQUENCE {
  privDst
     OBJECT IDENTIFIER,
  privData
     [1] IMPLICIT OCTET STRING
}
```

For each SnmpPrivMsg value that represents a SNMP private management communication, the following statements are true:

- o Its privDst component is called the privacy destination and identifies the SNMP party to which the communication is directed.

- o Its privData component is called the privacy data and represents the (possibly encrypted) serialization (according to the conventions of [12] and [1]) of a SNMP authenticated management communication.

5.1 Generating a Message

This section describes the behavior of a SNMP protocol entity when it communicates with a SNMP party for which the privacy protocol is administratively specified as the Symmetric Privacy Protocol. Insofar as the behavior of a SNMP protocol entity when transmitting a protocol message is defined generically in [2], only those aspects of that behavior that are specific to the Symmetric Privacy Protocol are described below. In particular, this section describes the encapsulation of a SNMP authenticated management communication into a SNMP private management communication.

According to [2], a SnmpPrivMsg value is constructed during Step 5 of generic processing. In particular, it states the privData component is constructed according to the privacy protocol identified for the SNMP party receiving the message. When the relevant privacy protocol is the Symmetric Privacy Protocol, the procedure performed by a SNMP protocol entity whenever a management communication is to be transmitted by a SNMP party is as follows.

1. If the SnmpAuthMsg value is not authenticated according to the conventions of the Digest Authentication Protocol, the generation of the private management communication fails according to a local procedure, without further processing.

2. The local database is consulted to determine the private privacy key of the SNMP party receiving the message

(represented, for example, according to the conventions defined in Section 2.4.2).

3. The SnmpAuthMsg value is serialized according to the conventions of [12] and [1].

4. The octet sequence representing the serialized SnmpAuthMsg value is encrypted using, for example, the algorithm specified in Section 2.4.2 and the extracted private privacy key.

5. The privData component is set to the encrypted value.

As set forth in [2], the SnmpPrivMsg value is then serialized and transmitted to the receiving SNMP party.

5.2 Receiving a Message

This section describes the behavior of a SNMP protocol entity when it acts as a SNMP party for which the privacy protocol is administratively specified as the Symmetric Privacy Protocol. Insofar as the behavior of a SNMP protocol entity when receiving a protocol message is defined generically in [2], only those aspects of that behavior that are specific to the Symmetric Privacy Protocol are described below.

According to [2], the privData component of a received SnmpPrivMsg value is evaluated during Step 4 of generic processing. In particular, it states the privData component is evaluated according to the privacy protocol identified for the SNMP party receiving the message. When the relevant privacy protocol is the Symmetric Privacy Protocol, the procedure performed by a SNMP protocol entity whenever a management communication is received by a SNMP party is as follows.

1. The local database is consulted to determine the private privacy key of the SNMP party receiving the message (represented, for example, according to the conventions defined in Section 2.4.2).

2. The contents octets of the privData component are decrypted using, for example, the algorithm specified in Section 2.4.2 and the extracted private privacy key.

Processing of the received message continues as specified in [2].

6. Clock and Secret Distribution

The protocols described in Sections 4 and 5 assume the existence of loosely synchronized clocks and shared secret values. Three requirements constrain the strategy by which clock values and secrets are distributed.

- o If the value of an authentication clock is decreased, the last-timestamp and private authentication key must be changed concurrently.

  When the value of an authentication clock is decreased, messages that have been sent with a timestamp value between the value of the authentication clock and its new value may be replayed. Changing the private authentication key obviates this threat. However, changing the authentication clock and the private authentication key is not sufficient to ensure proper operation. If the last-timestamp is not reduced similarly to the authentication clock, no message will be considered authentic until the value of the authentication clock exceeds the value of the last-timestamp.

- o The private authentication key and private privacy key must be known only to the parties requiring knowledge of them.

  Protecting the secrets from disclosure is critical to the security of the protocols. In particular, if the secrets are distributed via a network, the secrets must be protected with a protocol that supports confidentiality, e.g., the Symmetric Privacy Protocol. Further, knowledge of the secrets must be as restricted as possible within an implementation. In particular, although the secrets may be known to one or more persons during the initial configuration of a device, the secrets should be changed immediately after configuration such that their actual value is known only to the software. A management station has the additional responsibility of recovering the state of all parties whenever it boots, and it may address this responsibility by recording the secrets on a long-term storage device. Access to information on this device must be as restricted as is practically possible.

- o There must exist at least one SNMP protocol entity that assumes the role of a responsible management station.

  This management station is responsible for ensuring that all authentication clocks are synchronized and for changing the secret values when necessary. Although more than one management station may share this responsibility, their coordination is essential to the secure management of the network. The mechanism by which multiple management stations ensure that no more than one of them attempts to synchronize the clocks or update the secrets at any one time is a local implementation issue.

A responsible management station may either support clock synchronization and secret distribution as separate functions, or combine them into a single functional unit.

The first section below specifies the procedures by which a SNMP protocol entity is initially configured. The next two sections describe one strategy for distributing clock values and one for determining a synchronized clock value among SNMP parties supporting the Digest Authentication Protocol. For SNMP parties supporting the Symmetric Privacy Protocol, the next section describes a strategy for distributing secret values. The last section specifies the procedures by which a SNMP protocol entity recovers from a "crash."

6.1 Initial Configuration

This section describes the initial configuration of a SNMP protocol entity that supports the Digest Authentication Protocol or both the Digest Authentication Protocol and the Symmetric Privacy Protocol.

When a network device is first installed, its initial, secure configuration must be done manually, i.e., a person must physically visit the device and enter the initial secret values for at least its first secure SNMP party. This requirement suggests that the person will have knowledge of the initial secret values.

In general, the security of a system is enhanced as the number of entities that know a secret is reduced. Requiring a person to physically visit a device every time a SNMP party is configured not only exposes the secrets unnecessarily but is administratively prohibitive. In particular, when MD5 is used, the initial authentication secret is 128 bits long and when DES is used an additional 128 bits are needed -- 64 bits each for the key and initialization vector. Clearly, these values will need to be recorded on a medium in order to be transported between a responsible management station and a managed agent. The recommended procedure is to configure a small set of initial SNMP parties for each SNMP protocol entity, one pair of which may be used initially to configure all other SNMP parties.

In fact, there is a minimal, useful set of SNMP parties that could be configured between each responsible management station and managed agent. This minimal set includes one of each of the following for both the responsible management station and the managed agent:

- o a SNMP party for which the authentication protocol and privacy protocol are the values noAuth and noPriv, respectively,

- o a SNMP party for which the authentication protocol identifies the mechanism defined in Section 2.4.1 and its privacy protocol is the value noPriv, and

- o a SNMP party for which the authentication protocol and privacy protocol identify the mechanisms defined in Section 2.4.1 and Section 2.4.2, respectively.

The last of these SNMP parties in both the responsible management station and the managed agent could be used to configure all other SNMP parties. It is the only suitable party for this purpose because it is the only party that supports data confidentiality, which is necessary in order to protect the distributed secrets from disclosure to unauthorized entities.

Configuring one pair of SNMP parties to be used to configure all other parties has the advantage of exposing only one pair of secrets -- the secrets used to configure the minimal, useful set identified above. To limit this exposure, the responsible management station should change these values as its first operation upon completion of the initial configuration. In this way, secrets are known only to the peers requiring knowledge of them in order to communicate.

The Management Information Base (MIB) document [4] supporting these security protocols specifies 6 initial party identities and initial values, which, by convention, are assigned to the parties and their associated parameters.

All 6 parties should be configured in each new managed agent and its responsible management station. The responsible management station should be configured first, since the management station can be used to generate the initial secrets and provide them to a person, on a suitable medium, for distribution to the managed agent. The following sequence of steps describes the initial configuration of a managed agent and its responsible management station.

1. Determine the initial values for each of the attributes of the SNMP party to be configured. Some of these values may be computed by the responsible management

station, some may be specified in the MIB document, and some may be administratively determined.

2. Configure the parties in the responsible management station, according to the set of initial values. If the management station is computing some initial values to be entered into the agent, an appropriate medium must be present to record the values.

3. Configure the parties in the managed agent, according to the set of initial values.

4. The responsible management station must synchronize the authentication clock values for each party it shares with each managed agent. Section 6.3 specifies one strategy by which this could be accomplished.

5. The responsible management station should change the secret values manually configured to ensure the actual values are known only to the peers requiring knowledge of them in order to communicate. To do this, the management station generates new secrets for each party to be reconfigured and distributes those secrets with a strategy that uses a protocol that protects them from disclosure, e.g., Symmetric Privacy Protocol (see Section 6.4). Upon receiving positive acknowledgement that the new values have been distributed, the management station should update its local database with the new values.

If the managed agent does not support a protocol that protects messages from disclosure, then automatic maintenance and configuration of parties is not possible, i.e., the last step above is not possible. The secrets can only be changed by a physical visit to the device.

If there are other SNMP protocol entities requiring knowledge of the secrets, the responsible management station must distribute the information upon completion of the initial configuration. The mechanism used must protect the secrets from disclosure to unauthorized entities. The Symmetric Privacy Protocol, for example, is an acceptable mechanism.

6.2 Clock Distribution

A responsible management station must ensure that the authentication clock value for each SNMP party for which it is responsible

- o is loosely synchronized among all the local databases in which it appears,
- o is reset, as indicated below, upon reaching its maximal value, and
- o is non-decreasing, except as indicated below.

The skew among the clock values must be accounted for in the lifetime value, in addition to the expected communication delivery delay.

A skewed authentication clock may be detected by a number of strategies, including knowledge of the accuracy of the system clock, unauthenticated queries of the party database, and recognition of authentication failures originated by the party.

Whenever clock skew is detected, and whenever the SNMP entities at both the responsible management station and the relevant managed agent support an appropriate privacy protocol (e.g., the Symmetric Privacy Protocol), a straightforward strategy for the correction of clock skew is simultaneous alteration of authentication clock and private key for the relevant SNMP party. If the request to alter the key and clock for a particular party originates from that same party, then, prior to transmitting that request, the local notion of the authentication clock is artificially advanced to assure acceptance of the request as authentic.

More generally, however, since an authentication clock value need not be protected from disclosure, it is not necessary that a managed agent support a privacy protocol in order for a responsible management station to correct skewed clock values. The procedure for correcting clock skew in the general case is presented in Section 6.3.

In addition to correcting skewed notions of authentication clocks, every SNMP entity must react correctly as an authentication clock approaches its maximal value. If the authentication clock for a particular SNMP party ever reaches the maximal time value, the clock must halt at that value. (The value of interest may be the maximum less lifetime. When authenticating a message, its authentication timestamp is added to lifetime and compared to the authentication clock. A SNMP protocol entity must guarantee that the sum is never greater than the maximal time value.) In this state, the only authenticated request a management station should generate for this party is one that alters the value of at least its authentication clock and private authentication key. In order to reset these values, the responsible management station may set the authentication timestamp in the message to the maximal time value. In this case, the nonce value may be used to distinguish multiple messages.

The value of the authentication clock for a particular SNMP party must never be altered such that its new value is less than its old value, unless its last-timestamp and private authentication key are also altered at the same time.

6.3 Clock Synchronization

Unless the secrets are changed at the same time, the correct way to synchronize clocks is to advance the slower clock to be equal to the faster clock. Suppose that party agentParty is realized by the SNMP entity in a managed agent; suppose that party mgrParty is realized by the SNMP entity in the corresponding responsible management station. For any pair of parties, there are four possible conditions of the authentication clocks that could require correction:

1. The management station’s notion of the value of the authentication clock for agentParty exceeds the agent’s notion.

2. The management station’s notion of the value of the authentication clock for mgrParty exceeds the agent’s notion.

3. The agent’s notion of the value of the authentication clock for agentParty exceeds the management station’s notion.

4. The agent’s notion of the value of the authentication clock for mgrParty exceeds the management station’s notion.

The selective clock acceleration mechanism intrinsic to the protocol corrects conditions 2 and 3 as part of the normal processing of an authentic message. Therefore, the clock adjustment procedure below does not provide for any adjustments in those cases. Rather, the following sequence of steps specifies how the clocks may be synchronized when condition 1, condition 4, or both of those conditions are manifest.

1. The responsible management station saves its existing notions of the authentication clocks for the two parties agentParty and mgrParty.

2. The responsible management station retrieves the authentication clock values for both agentParty and mgrParty from the agent. This retrieval must be an

unauthenticated request, since the management station does not know if the clocks are synchronized. If the request fails, the clocks cannot be synchronized, and the clock adjustment procedure is aborted without further processing.

3. If the management station’s notion of the authentication clock for agentParty exceeds the notion just retrieved from the agent by more than the amount of the communications delay between the two protocol entities, then condition 1 is manifest. The recommended estimate of communication delay in this context is one half of the lifetime value recorded for agentParty.

4. If the notion of the authentication clock for mgrParty just retrieved from the agent exceeds the management station’s notion, then condition 4 is manifest, and the responsible management station advances its notion of the authentication clock for mgrParty to match the agent’s notion.

5. If condition 1 is manifest, then the responsible management station sends an authenticated management operation to the agent that advances the agent’s notion of the authentication clock for agentParty to be equal to the management station’s notion. If this management operation fails, then the management station restores its previously saved notions of the clock values, and the clock adjustment procedure is aborted without further processing.

6. The responsible management station retrieves the authentication clock values for both agentParty and mgrParty from the agent. This retrieval must be an authenticated request, in order that the management station may verify that the clock values are properly synchronized. If this authenticated query fails, then the management station restores its previously saved notions of the clock values, and the clock adjustment procedure is aborted without further processing. Otherwise, clock synchronization has been successfully realized.

It is important to note step 4 above must be completed before attempting step 5. Otherwise, the agent may evaluate the request in step 5 as unauthentic. Similarly, step 5 above must be completed before attempting step 6. Otherwise, the management station may evaluate the query response in step 6 as unauthentic.

Administrative advancement of a clock as described above does not introduce any new vulnerabilities, since the value of the clock is intended to increase with the passage of time. A potential operational problem is the rejection of management operations that are authenticated using a previous value of the relevant party clock. This possibility may be avoided if a management station suppresses generation of management traffic between relevant parties while this clock adjustment procedure is in progress.

6.4 Secret Distribution

This section describes one strategy by which a SNMP protocol entity that supports both the Digest Authentication Protocol and the Symmetric Privacy Protocol can change the secrets for a particular SNMP party.

The frequency with which the secrets of a SNMP party should be changed is a local administrative issue. However, the more frequently a secret is used, the more frequently it should be changed. At a minimum, the secrets must be changed whenever the associated authentication clock approaches its maximal value (see Section 7). Note that, owing to both administrative and automatic advances of the authentication clock described in this memo, the authentication clock for a SNMP party may well approach its maximal value sooner than might otherwise be expected.

The following sequence of steps specifies how a responsible management station alters a secret value (i.e., the private authentication key or the private privacy key) for a particular SNMP party.

1. The responsible management station generates a new secret value.

2. The responsible management station encapsulates a SNMP Set request in a SNMP private management communication with at least the following properties.

   o Its source supports the Digest Authentication Protocol and the Symmetric Privacy Protocol.

   o Its destination supports the Symmetric Privacy Protocol and the Digest Authentication Protocol.

3. The SNMP private management communication is transmitted to its destination.

4. Upon receiving the request, the recipient processes the

message according to [1] and [2].

5. The recipient encapsulates a SNMP Set response in a SNMP private management communication with at least the following properties.

   - Its source supports the Digest Authentication Protocol and the Symmetric Privacy Protocol.

   - Its destination supports the Symmetric Privacy Protocol and the Digest Authentication Protocol.

6. The SNMP private management communication is transmitted to its destination.

7. Upon receiving the response, the responsible management station updates its local database with the new value.

If the responsible management station does not receive a response to its request, there are two possible causes.

- The request may not have been delivered to the destination.

- The response may not have been delivered to the originator of the request.

In order to distinguish the two possible error conditions, a responsible management station could check the destination to see if the change has occurred. Unfortunately, since the secret values are unreadable, this is not directly possible.

The recommended strategy for verifying key changes is to set the public value corresponding to the secret being changed to a recognizable, novel value: that is, alter the public authentication key value for the relevant party when changing its private authentication key, or alter its public privacy key value when changing its private privacy key. In this way, the responsible management station may retrieve the public value when a response is not received, and verify whether or not the change has taken place. (This strategy is available since the public values are not used by the protocols defined in this memo. If this strategy is employed, then the public values are significant in this context. Of course, protocols using the public values may make use of this strategy directly.)

One other scenario worthy of mention is using a SNMP party to change its own secrets. In this case, the destination will change its local database prior to generating a response. Thus, the response will be constructed according to the new value. However, the responsible management station will not update its local database until after the response is received. This suggests the responsible management station may receive a response which will be evaluated as unauthentic, unless the correct secret is used. The responsible management station may either account for this scenario as a special case, or use an alteration of the relevant public values (as described above) to verify the key change.

Note, during the period of time after the request has been sent and before the response is received, the management station must keep track of both the old and new secret values. Since the delay may be the result of a network failure, the management station must be prepared to retain both values for an extended period of time, including across reboots.

6.5 Crash Recovery

This section describes the requirements for SNMP protocol entities in connection with recovery from system crashes or other service interruptions.

For each SNMP party in the local database for a particular SNMP protocol entity, its identity, authentication clock, private authentication key, and private privacy key must enjoy non-volatile, incorruptible representations. If possible, lifetime should also enjoy a non-volatile, incorruptible representation. If said protocol entity supports other security protocols or algorithms in addition to the two defined in this memo, then the authentication protocol and the privacy protocol for each party also require non-volatile, incorruptible representation.

The authentication clock of a SNMP party is a critical component of the overall security of the protocols. The inclusion of a reliable representation of a clock in a SNMP protocol entity enhances overall security. A reliable clock representation continues to increase according to the passage of time, even when the local SNMP protocol entity -- due to power loss or other system failure -- may not be operating. An example of a reliable clock representation is that provided by battery-powered clock-calendar devices incorporated into some contemporary systems. It is assumed that management stations always support reliable clock representations, where clock adjustment by a human operator during crash recovery may contribute to that reliability.

If a managed agent crashes and does not reboot in time for its responsible management station to prevent its authentication clock from reaching its maximal value, upon reboot the clock must be halted at its maximal value. The procedures specified in Section 6.3 would then apply.

If a managed network element supports a reliable clock representation, recovering from a crash requires few special actions. Upon recovery, those attributes of each SNMP party that do not enjoy non-volatile or reliable representation are initialized as follows.

- o If the private authentication key is not the OCTET STRING of zero length, the authentication protocol is set to identify use of the Digest Authentication Protocol in conjunction with the algorithm specified in Section 2.4.1.

- o The last-timestamp is initialized to the value of the authentication clock.

- o The nonce is initialized to zero.

- o If the lifetime is not retained, it should be initialized to zero.

- o If the private privacy key is not the OCTET STRING of zero length, the privacy protocol is set to identify use of the Symmetric Privacy Protocol in conjunction with the algorithm specified in Section 2.4.2.

Upon detecting that a managed agent has rebooted, a responsible management station must reset all other party attributes, including the lifetime if it was not retained. In order to reset the lifetime, the responsible management station should set the authentication timestamp in the message to the sum of the authentication clock and desired lifetime. This is an artificial advancement of the authentication timestamp in order to guarantee the message will be authentic when received by the recipient.

If, alternatively, a managed network element does not support a reliable clock representation, then those attributes of each SNMP party that do not enjoy non-volatile representation are initialized as follows.

- o If the private authentication key is not the OCTET STRING of zero length, the authentication protocol is set to identify use of the Digest Authentication Protocol in conjunction with the algorithm specified in Section 2.4.1.

- o The authentication clock is initialized to the maximal time value.

- o The last-timestamp is initialized to the maximal time value.

- o The nonce is initialized to zero.

- o If the lifetime is not retained, it should be initialized to zero.

- o If the private privacy key is not the OCTET STRING of zero length, the privacy protocol is set to identify use of the Symmetric Privacy Protocol in conjunction with the algorithm specified in Section 2.4.2.

The only authenticated request a management station should generate for a party in this initial state is one that alters the value of at least its authentication clock, private authentication key, and lifetime (if that was not retained). In order to reset these values, the responsible management station must set the authentication timestamp in the message to the maximal time value. The nonce value may be used to distinguish multiple messages.

7. Security Considerations

This section highlights security considerations relevant to the protocols and procedures defined in this memo. Practices that contribute to secure, effective operation of the mechanisms defined here are described first. Constraints on implementation behavior that are necessary to the security of the system are presented next. Finally, an informal account of the contribution of each mechanism of the protocols to the required goals is presented.

7.1 Recommended Practices

This section describes practices that contribute to the secure, effective operation of the mechanisms defined in this memo.

- o A management station should discard SNMP responses for which neither the request-id component nor the represented management information corresponds to any currently outstanding request.

  Although it would be typical for a management station to do this as a matter of course, in the context of these security protocols it is significant owing to the possibility of message duplication (malicious or otherwise).

- o A management station should not interpret an agent's lack of response to an authenticated SNMP management communication as a conclusive indication of agent or network failure.

  It is possible for authentication failure traps to be lost or suppressed as a result of authentication clock skew or inconsistent notions of shared secrets. In order either to facilitate administration of such SNMP parties or to provide for continued management in times of network stress, a management station implementation may provide for arbitrary, artificial advancement of the timestamp or selection of shared secrets on locally generated messages.

- o The lifetime value for a SNMP party should be chosen (by the local administration) to be as small as possible, given the accuracy of clock devices available, relevant round-trip communications delays, and the frequency with which a responsible management station will be able to verify all clock values.

  A large lifetime increases the vulnerability to malicious delays of SNMP messages. The implementation of a management station may, when explicitly authorized, provide for dynamic adjustment of the lifetime in order to accommodate changing network conditions.

- o When sending state altering messages to a managed agent, a management station should delay sending successive messages to the managed agent until a positive acknowledgement is received for the previous message or until the previous message expires.

  When using the noAuth protocol, no message ordering is imposed by the SNMP. Messages may be received in any order relative to their time of generation and each will be processed in the ordered received. In contrast, the security protocols guarantee that received messages are ordered insofar as each received message must have been sent subsequent to the sending of a previously received message.

  When an authenticated message is sent to a managed agent, it will be valid for a period of time that does not exceed lifetime under normal circumstances. During the period of time this message is valid, if the management station sends another authenticated message to the managed agent that is received and processed prior to the first message, the first message will be considered unauthentic when it is received by the managed agent.

Indeed, a management station must cope with the loss and re-ordering of messages resulting from anomalies in the network as a matter of course. A management station implementation may choose to prevent the loss of messages resulting from re-ordering when using the security protocols defined in this memo by delaying sending successive messages.

o The frequency with which the secrets of a SNMP party should be changed is indirectly related to the frequency of their use.

  Protecting the secrets from disclosure is critical to the overall security of the protocols. Frequent use of a secret provides a continued source of data that may be useful to a cryptanalyst in exploiting known or perceived weaknesses in an algorithm. Frequent changes to the secret avoid this vulnerability.

  Changing a secret after each use is is generally regarded as the most secure practice, but a significant amount of overhead may be associated with that approach.

  Note, too, in a local environment the threat of disclosure may be insignificant, and as such the changing of secrets may be less frequent. However, when public data networks are the communication paths, more caution is prudent.

o In order to foster the greatest degree of security, a management station implementation must support constrained, pairwise sharing of secrets among SNMP entities as its default mode of operation.

  Owing to the use of symmetric cryptography in the protocols defined here, the secrets associated with a particular SNMP party must be known to all other SNMP parties with which that party may wish to communicate. As the number of locations at which secrets are known and used increases, the likelihood of their disclosure also increases, as does the potential impact of that disclosure. Moreover, if the set of SNMP protocol entities with knowledge of a particular secret numbers more than two, data origin cannot be reliably authenticated because it is impossible to determine with any assurance which entity of that set may be the originator of a particular SNMP message. Thus, the greatest degree of security is afforded by configurations in which the secrets for each SNMP party are known to at most two protocol entities.

7.2 Conformance

A SNMP protocol entity implementation that claims conformance to this memo must satisfy the following requirements:

1. It must implement the noAuth and noPriv protocols whose object identifiers are defined in [4].

   noAuth This protocol signifies that messages generated by a party using it are not protected as to origin or integrity. It is required to ensure that a party's authentication clock is always accessible.

   noPriv This protocol signifies that messages received by a party using it are not protected from disclosure. It is required to ensure that a party's authentication clock is always accessible.

2. It must implement the Digest Authentication Protocol in conjunction with the algorithm defined in Section 2.4.1.

3. It must include in its local database at least one SNMP party with the following parameters set as follows:

   o partyAuthProtocol is set to noAuth and
   o partyPrivProtocol is set to noPriv.

   This party must have a MIB view [2] specified that includes at least the authentication clock of all other parties. Alternatively, the authentication clocks of the other parties may be partitioned among several similarly configured parties according to a local implementation convention.

4. For each SNMP party about which it maintains information in a local database, an implementation must satisfy the following requirements:

   (a) It must not allow a party's parameters to be set to a value inconsistent with its expected syntax. In particular, Section 2.4 specifies constraints for the chosen mechanisms.

(b) It must, to the maximal extent possible, prohibit read-access to the private authentication key and private encryption key under all circumstances except as required to generate and/or validate SNMP messages with respect to that party. This prohibition includes prevention of read-access by the entity's human operators.
(c) It must allow the party's authentication clock to be publicly accessible. The correct operation of the Digest Authentication Protocol requires that it be possible to determine this value at all times in order to guarantee that skewed authentication clocks can be resynchronized.
(d) It must prohibit alterations to its record of the authentication clock for that party independently of alterations to its record of the private authentication key (unless the clock alteration is an advancement).
(e) It must never allow its record of the authentication clock for that party to be incremented beyond the maximal time value and so "roll-over" to zero.
(f) It must never increase its record of the lifetime for that party except as may be explicitly authorized (via imperative command or securely represented configuration information) by the responsible network administrator.
(g) In the event that the non-volatile, incorruptible representations of a party's parameters (in particular, either the private authentication key or private encryption key) are lost or destroyed, it must alter its record of these quantities to random values so subsequent interaction with that party requires manual redistribution of new secrets and other parameters.

5. If it selects new value(s) for a party's secret(s), it must avoid bad or obvious choices for said secret(s). Choices to be avoided are boundary values (such as all-zeros) and predictable values (such as the same value as previously or selecting from a predetermined set).

7.3 Protocol Correctness

The correctness of these SNMP security protocols with respect to the stated goals depends on the following assumptions:

1. The chosen message digest algorithm satisfies its design criteria. In particular, it must be computationally infeasible to discover two messages that share the same digest value.

2. It is computationally infeasible to determine the secret used in calculating a digest on the concatenation of the secret and a message when both the digest and the message are known.

3. The chosen symmetric encryption algorithm satisfies its design criteria. In particular, it must be computationally infeasible to determine the cleartext message from the ciphertext message without knowledge of the key used in the transformation.

4. Local notions of a party's authentication clock while it is associated with a specific private key value are monotonically non-decreasing (i.e., they never run backwards) in the absence of administrative manipulations.

5. The secrets for a particular SNMP party are known only to authorized SNMP protocol entities.

6. Local notions of the authentication clock for a particular SNMP party are never altered such that the authentication clock's new value is less than the current value without also altering the private authentication key.

For each mechanism of the protocol, an informal account of its contribution to the required goals is presented below. Pseudocode fragments are provided where appropriate to exemplify possible implementations; they are intended to be self-explanatory.

7.3.1 Clock Monotonicity Mechanism

By pairing each sequence of a clock's values with a unique key, the protocols partially realize goals 3 and 4, and the conjunction of this property with assumption 6 above is sufficient for the claim that, with respect to a specific private key value, all local notions of a party's authentication clock are, in general, non-decreasing with time.

7.3.2 Data Integrity Mechanism

The protocols require computation of a message digest computed over the SNMP message prepended by the secret for the relevant party. By virtue of this mechanism and assumptions 1 and 2, the protocols realize goal 1.

Normally, the inclusion of the message digest value with the digested message would not be sufficient to guarantee data integrity, since the digest value can be modified in addition to the message while it is enroute. However, since not all of the digested message is included in the transmission to the destination, it is not possible to substitute both a message and a digest value while enroute to a destination.

Strictly speaking, the specified strategy for data integrity does not detect a SNMP message modification which appends extraneous material to the end of such messages. However, owing to the representation of SNMP messages as ASN.1 values, such modifications cannot -- consistent with goal 1 -- result in unauthorized management operations.

The data integrity mechanism specified in this memo protects only against unauthorized modification of individual SNMP messages. A more general data integrity service that affords protection against the threat of message stream modification is not realized by this mechanism, although limited protection against reordering, delay, and duplication of messages within a message stream are provided by other mechanisms of the protocol.

7.3.3 Data Origin Authentication Mechanism

The data integrity mechanism requires the use of a secret value known only to communicating parties. By virtue of this mechanism and assumptions 1 and 2, the protocols explicitly prevent unauthorized modification of messages. Data origin authentication is implicit if the message digest value can be verified. That is, the protocols realize goal 2.

7.3.4 Restricted Administration Mechanism

This memo requires that implementations preclude administrative alterations of the authentication clock for a particular party independently from its private authentication key (unless that clock alteration is an advancement). An example of an efficient implementation of this restriction is provided in a pseudocode fragment below. This pseudocode fragment meets the requirements of assumption 6.

Pseudocode Fragment. Observe that the requirement is not for simultaneous alteration but to preclude independent alteration. This latter requirement is fairly easily realized in a way that is consistent with the defined semantics of the SNMP Set operation.

```
Void partySetKey (party, newKeyValue)
{
    if (party->clockAltered) {
       party->clockAltered = FALSE;
       party->keyAltered = FALSE;
       party->keyInUse = newKeyValue;
       party->clockInUse = party->clockCache;
    }
    else {
       party->keyAltered = TRUE;
       party->keyCache = newKeyValue;
    }
}

Void partySetClock (party, newClockValue)
{
    if (party->keyAltered) {
       party->keyAltered = FALSE;
       party->clockAltered = FALSE;
       party->clockInUse = newClockValue;
       party->keyInUse = party->keyCache;
    }
    else {
       party->clockAltered = TRUE;
       party->clockCache = newClockValue;
    }
}
```

7.3.5 Ordered Delivery Mechanism

The definition of the Digest Authentication Protocol requires that, if the timestamp value on a received message does not exceed the timestamp of the most recent validated message locally delivered from the originating party, then that message is not delivered. Otherwise, the record of the timestamp for the most recent locally delivered validated message is updated.

```
if (msgIsValidated) {
    if (timestampOfReceivedMsg >
       party->timestampOfLastDeliveredMsg) {
```

```
         party->timestampOfLastDeliveredMsg =
            timestampOfReceivedMsg;
      }
      else {
         msgIsValidated = FALSE;
      }
   }
```

Although not explicitly represented in the pseudocode above, in the Digest Authentication Protocol, the ordered delivery mechanism must ensure that, when the authentication timestamp of the received message is equal to the last-timestamp, received messages continue to be delivered as long as their nonce values are monotonically increasing. By virtue of this mechanism, the protocols realize goal 4.

7.3.6 Message Timeliness Mechanism

The definition of the SNMP security protocols requires that, if the authentication timestamp value on a received message -- augmented by an administratively chosen lifetime value -- is less than the local notion of the clock for the originating SNMP party, the message is not delivered.

```
   if (timestampOfReceivedMsg +
          party->administrativeLifetime <=
          party->localNotionOfClock) {
          msgIsValidated = FALSE;
   }
```

By virtue of this mechanism, the protocols realize goal 3. In cases in which the local notions of a particular SNMP party clock are moderately well-synchronized, the timeliness mechanism effectively limits the age of validly delivered messages. Thus, if an attacker diverts all validated messages for replay much later, the delay introduced by this attack is limited to a period that is proportional to the skew among local notions of the party clock.

7.3.7 Selective Clock Acceleration Mechanism

The definition of the SNMP security protocols requires that, if the timestamp value on a received, validated message exceeds the local notion of the clock for the originating party, then that notion is adjusted forward to correspond to said timestamp value. This mechanism is neither strictly necessary nor sufficient to the security of the protocol; rather, it fosters the clock synchronization on which valid message delivery depends -- thereby enhancing the effectiveness of the protocol in a management context.

```
if (msgIsValidated) {
       if (timestampOfReceivedMsg >
             party->localNotionOfClock) {
             party->localNotionOfClock =
                   timestampOfReceivedMsg;
       }
}
```

The effect of this mechanism is to synchronize local notions of the party clock more closely in the case where a sender's notion is more advanced than a receiver's. In the opposite case, this mechanism has no effect on local notions of the party clock and either the received message is validly delivered or not according to other mechanisms of the protocol.

Operation of this mechanism does not, in general, improve the probability of validated delivery for messages generated by party participants whose local notion of the party clock is relatively less advanced. In this case, queries from a management station may not be validly delivered and the management station needs to react appropriately (e.g., by administratively resynchronizing local notions of the clock in conjunction with a key change). In contrast, the delivery of SNMP trap messages generated by an agent that suffers from a less advanced notion of a party clock is more problematic, for an agent may lack the capacity to recognize and react to security failures that prevent delivery of its messages. Thus, the inherently unreliable character of trap messages is likely to be compounded by attempts to provide for their validated delivery.

7.3.8 Confidentiality Mechanism

The protocols require the use of a symmetric encryption algorithm when the data confidentiality service is required. By virtue of this mechanism and assumption 3, the protocols realize goal 5.

8. Acknowledgements

The authors would like to thank the members of the SNMP Security Working Group of the IETF for their patience and comments. Special thanks go to Jeff Case who provided the first implementation of the protocols. Dave Balenson, John Linn, Dan Nessett, and all the members of the Privacy and Security Research Group provided many valuable and

detailed comments.

9. References

[1] Case, J., M. Fedor, M. Schoffstall, and J. Davin, The Simple Network Management Protocol", RFC 1157, University of Tennessee at Knoxville, Performance Systems International, Performance Systems International, and the MIT Laboratory for Computer Science, May 1990. (Obsoletes RFC 1098.)

[2] Davin, J., Galvin, J., and K. McCloghrie, "SNMP Administrative Model", RFC 1351, MIT Laboratory for Computer Science, Trusted Information Systems, Inc., Hughes LAN Systems, Inc., July 1992.

[3] Rivest, R., "The MD5 Message-Digest Algorithm", RFC 1321, MIT Laboratory for Computer Science, April 1992.

[4] McCloghrie, K., Davin, J., and J. Galvin, "Definitions of Managed Objects for Administration of SNMP Parties", RFC 1353, Hughes LAN Systems, Inc., MIT Laboratory for Computer Science, Trusted Information Systems, Inc., July 1992.

[5] FIPS Publication 46-1, "Data Encryption Standard", National Institute of Standards and Technology, Federal Information Processing Standard (FIPS); Supersedes FIPS Publication 46, January 15, 1977; Reaffirmed January 22, 1988.

[6] ANSI X3.92-1981, "Data Encryption Algorithm", American National Standards Institute, December 30, 1980.

[7] FIPS Publication 81, "DES Modes of Operation", National Institute of Standards and Technology, December 2, 1980, Federal Information Processing Standard (FIPS).

[8] ANSI X3.106-1983, "Data Encryption Algorithm - Modes of Operation", American National Standards Institute, May 16, 1983.

[9] FIPS Publication 74, "Guidelines for Implementing and Using the NBS Data Encryption Standard", National Institute of Standards and Technology, April 1, 1981. Federal Information Processing Standard (FIPS).

[10] Special Publication 500-20, "Validating the Correctness of Hardware Implementations of the NBS Data Encryption Standard", National Institute of Standards and Technology.

[11] Special Publication 500-61, "Maintenance Testing for the Data Encryption Standard", National Institute of Standards and

Technology, August 1980.

[12] Information Processing -- Open Systems Interconnection -- Specification of Basic Encoding Rules for Abstract Syntax Notation One (ASN.1), International Organization for Standardization/International Electrotechnical Institute, 1987, International Standard 8825.

10. Authors' Addresses

James M. Galvin
Trusted Information Systems, Inc.
3060 Washington Road, Route 97
Glenwood, MD 21738

Phone: (301) 854-6889
EMail: galvin@tis.com

Keith McCloghrie
Hughes LAN Systems, Inc.
1225 Charleston Road
Mountain View, CA 94043

Phone: (415) 966-7934
EMail: kzm@hls.com

James R. Davin
MIT Laboratory for Computer Science
545 Technology Square
Cambridge, MA 02139

Phone: (617) 253-6020
EMail: jrd@ptt.lcs.mit.edu