[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.

References

[1] Jeffrey D. Case, Mark S. Fedor, Martin L. Schoffstall, and James R. Davin. A Simple Network Management Protocol (SNMP). RFC 1157, DDN Network Information Center, SRI International, May 1990. Obsoletes RFC1098.

[2] James R. Davin, James M. Galvin, and Keith McCloghrie. SNMP Administrative Model. Internet draft, Internet Engineering Task Force, December 1991. draft-ietf-snmpsec-admin-02.txt.

[3] Ronald L. Rivest and Steve Dusse. The MD5 Message-Digest Algorithm. Internet draft, Internet Engineering Task Force, July 1991. draft-rsadsi-rivest-md5-01.txt.

[4] Keith McCloghrie, James R. Davin, and James M. Galvin. Definitions of Managed Objects for Administration of SNMP Parties. Internet draft, Internet Engineering Task Force, December 1991. draft-ietf-snmpsec-mib-02.txt.

[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.

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.

9.2.8 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 2, the protocols realize goal 5.

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.

9.2.7 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

```
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.

9.2.6 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

```
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;
    }
}
```

9.2.5 Ordered Delivery Mechanism

The definition of the SNMP security 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.

way that is consistent with the defined semantics of the SNMP Set operation.

9.2.2 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 4, 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.

9.2.3 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 4, 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.

9.2.4 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 5.

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

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.

9.2.1 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 1 above is sufficient for the claim that all local notions of a party's authentication clock are, in general, non-decreasing with time.

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).

9.2 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 concatentation of the

2. It must implement the Digest Authentication Protocol in conjunction with the mechanism defined in Section 4.4.1.

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

   - **partyAuthProtocol** is set to **noAuth** and
   - **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:

   (a) It must not allow a party's parameters to be set to a value inconsistent with its expected syntax. In particular, Section 4.4 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

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 equivalent to a one-time pad algorithm, which is generally regarded as the most secure algorithm. However, there is potentially a significant amount of overhead associated with that approach.

Note, too, in a local environment the threat of disclosure may be insignificant, and as such the frequency of secret changes may be small. However, when public data networks are the communication paths more caution is prudent.

9.1 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 as to disclosure. It is required to ensure that a party's authentication clock is always accessible.

The intent of the data integrity service is that authentic messages be ordered. Of course, a message that may be a replay of a previous message must not be considered authentic. In the absence of secure, connection-oriented transport services, it is not possible to distinguish messages that were subject to ordinary network delays from messages that are being replayed.

- 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 previously received message was sent.

  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.

- The frequency with which the secrets of a SNMP party should be changed is indirectly related to the frequency

security protocols it is significant owing to the possibility of message duplication (malicious or otherwise).

- 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.

- 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 replay attacks. The implementation of a management station may, when explicitly authorized, provide for dynamic adjustment of the lifetime in order to accommodate changing network conditions.

- The data integrity service, which is defined to ensure that data and data sequences have not been altered or destroyed, is consistent with Goal 4, which states that the apparent time of generation for each received SNMP message is subsequent to previously received messages.

  The intent of Goal 4 is that messages received out of order be discarded as unauthentic. This is necessary to protect against the replay of messages in the absence of secure, connection-oriented transport services.

- The nonce is initialized to zero.
- If the lifetime is not retained, it should be initialized to zero.
- If the private privacy key is the **OCTET STRING** of zero length, the privacy protocol is set to the Symmetric Privacy Protocol according to the mechanism defined in Section 4.4.2.

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, private authentication key, and lifetime if it 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 is used to distinguish multiple messages.

9 Security Considerations

Although the protocols defined in this memo explicitly address security considerations, there are a number of considerations upon which their security depends. In the next section the critical issues for implementors to be aware of are listed for ease of reference. In the last section an informal account of the contribution of each mechanism of the protocols to the required goals is presented. Additional considerations are listed below.

- 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

- If the lifetime is not retained, it should be initialized to zero.
- If the private privacy key is not the **OCTET STRING** of zero length, the privacy protocol is set to the Symmetric Privacy Protocol according to the mechanism defined in Section 4.4.2.

Upon detecting that a managed agent has rebooted, a responsible management station must reset all other values, 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, however, a managed network element does not support a reliable clock representation, the following set of procedures apply to all SNMP protocol entities residing at that element. For each SNMP party in the local database for such a SNMP protocol entity, its identity, private authentication key, and private privacy key must enjoy non-volatile, incorruptible representations. If possible, lifetime should also enjoy a non-volatile, incorruptible representation. Upon reboot, the remaining elements of each SNMP party are set as follows. (As indicated above, it is assumed the only security protocols are the two specified in this memo.)

- If the private authentication key is the **OCTET STRING** of zero length, the authentication protocol is set to the Digest Authentication Protocol according to the mechanism defined in Section 4.4.1.
- The authentication clock is initialized to the maximal time value.
- The last-timestamp is initialized to the maximal time value.

protocol entity enhances the 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.

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 8.3 would then apply.

If a managed network element supports a reliable clock representation, recovering from a crash requires no special actions. (It is assumed that management stations will always support reliable clock representations, where a person present during the recovery setting the clock is considered a reliable representation.) For each SNMP party in the local database for such a 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. Upon reboot, the remaining elements of each SNMP party are set as follows. (It is assumed the only security protocols are the two specified in this memo. Should there exist other security protocols, the authentication protocol and the privacy protocol would also require non-volatile, incorruptible representations.)

- If the private authentication key is not the **OCTET STRING** of zero length, the authentication protocol is set to the Digest Authentication Protocol according to the mechanism defined in Section 4.4.1.
- The last-timestamp is initialized to the value of the authentication clock.
- The nonce is initialized to zero.

One possible strategy is to set the public value corresponding to the secret being changed to a recognizable, novel value, i.e., set **partyAuthPublic** when changing **partyAuthPrivate**, or set **partyPrivPublic** when changing **partyPrivPrivate**. In this way, the responsible management station may retrieve the public value when a response is not received, and verify if 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 the mechanism described above to verify if the change has taken place, i.e., setting the public value to a recognizable, novel value when changing a secret value.

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.

8.5 Crash Recovery

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

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

It is important to note that the authenticated command to advance the agent's notion of the value of P1, to fix case 1, must be done before the authenticated retrieval of the values from the agent. Otherwise, the management station will evaluate the response as unauthentic. Administratively advancing a clock does not introduce any new vulnerabilities, since the value of the clock is intended to increase with the passage of time. Operationally, the only potential issue is messages sent assuming knowledge of the previous value of the clock may be evaluated as unauthentic. A management station can easily account for this scenario by not sending any additional message to an agent during the period of time it is synchronizing the clocks.

8.4 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 at least every maximal time value (see Section 9).

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

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.

2. The responsible management station should retrieve the authentication clock values for both P1 and P2 from the agent. This 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 remaining steps in this sequence are skipped.

3. The retrieved values must be checked to determine which of case 1 or case 4 is true. Note that the test for case 1 needs to account for the communications delay in retrieving the P1 value; that is, the actions below for case 1 should only be invoked if the management station's notion of the value of P1's authentication clock exceeds the agent's notion by the amount of communications delay. The recommended value for use as the communications delay in this calculation is one-half the lifetime.

   Case 4: The responsible management station should advance its notion of the value of P2's authentication clock to be equal to the agent's notion.

   Case 1: The responsible management station should send an authenticated command to the agent that advances the agent's notion of the value of P1's authentication clock to be equal to the management station's notion. If the authenticated command fails the management station should restore the existing values and exit this sequence of steps.

4. The responsible management station should retrieve the authentication clock values for both P1 and P2 from the agent. This must be an authenticated request, since the management station must verify that the clock values are properly synchronized. If the authenticated command fails the management station should restore the existing values and the remaining steps in this sequence are skipped. Otherwise, clock synchronization has been achieved.

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.

8.3 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. For any pair of parties, there are four possible scenarios for the state of the authentication clocks.

Let party P1 execute in the agent and party P2 execute in the management station. The four scenarios for the state of the authentication clocks are:

1. The management station's notion of the value of P1's authentication clock exceeds the agent's notion.
2. The management station's notion of the value of P2's authentication clock exceeds the agent's notion.
3. The agent's notion of the value of P1's authentication clock exceeds the management station's notion.
4. The agent's notion of the value of P2's authentication clock exceeds the management station's notion.

Note the selective clock acceleration mechanism intrinsic to the protocol corrects cases 2 and 3 as part of the normal processing of an authentic message. Therefore, no special instructions are included below to correct those cases.

The following sequence of events specifies how the clocks should be synchronized when either case 1 or case 4 represent the status of the authentication clocks.

1. The responsible management station should save its existing values of P1's and P2's authentication clocks.

- is reset, as indicated below, upon reaching its maximal value, and
- 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.

The detection of a skewed authentication clock may be identified by a number of strategies, including knowledge of the accuracy of the system clock, unauthenticated queries of the local database, and recognition of authentication failures originated by the party. The sequence of steps by which an appropriate replacement value for a skewed value may be determined can be found in Section 8.3.

The sequence of steps by which a responsible management station changes the authentication clock for a particular SNMP party is similar to that used to change a secret value (see Section 8.4). However, since the clock value need not be protected from disclosure, it is not necessary for the destination to support a privacy protocol to distribute clock values.

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, private authentication key, last-timestamp, and nonce. 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 is used to distinguish multiple messages.

with each managed agent. Section 8.3 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 8.4). 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, e.g., the Symmetric Privacy Protocol is an acceptable mechanism.

8.2 Clock Distribution

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

- is loosely synchronized among all the local databases in which it appears,

Configuring one pair of SNMP parties to be used to configure all other parties has the advantage of only exposing 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 supporting these security protocols specifies 6 initial party identities and initial values, which, by convention, are assigned to the parties and their associated parameters [4].

All 6 parties should be configured in each new managed agent and its responsible management station. The responsible management station should be configured before the managed agent, 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 specifies how to initially configure a managed agent and its responsible management station upon its installation.

1. Determine the initial values for each of the components 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

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 initially configure a small set of SNMP parties for each SNMP protocol entity, one pair of which may be used to initially 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.

- a SNMP party identity with its **partyAuthProtocol** and **partyPrivProtocol** set to **noAuth** and **noPriv**, respectively

- a SNMP party identity with its **partyAuthProtocol** set according to the mechanism defined in Section 4.4.1 and its **partyPrivProtocol** set to **noPriv**

- a SNMP party identity with its **partyAuthProtocol** and **partyPrivProtocol** set according to the mechanisms defined in Section 4.4.1 and Section 4.4.2, 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 identity for this purpose because it is the only identity that supports data confidentiality, which is necessary in order to protect the distributed secrets from disclosure to unauthorized entities.

changing the secret values when necessary. Although more than one management station may participate in the 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".

8.1 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 suggests 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

- 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.

- 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 immediately changed 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. This may require recording the secrets on a long-term storage device. Access to information on this device must be as restricted as is practically possible.

- 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

7.2 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], a **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 **partyPrivProtocol** value of the SNMP party receiving the message. When that value is the Symmetric Privacy Protocol, the procedures executed by a SNMP protocol entity whenever a management communication is received by a SNMP party are as follows.

1. The local database is consulted to determine the private privacy key of the SNMP party receiving the message, according to the conventions of the mechanism defined in Section 4.4.2.

2. The contents octets of the **privData** component are decrypted according to the conventions of the mechanism defined in Section 4.4.2, using the extracted private privacy key.

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

8 Clock and Secret Distribution

The protocols described in Sections 6 and 7 assume the existence of loosely synchronized clocks and secret values. Although a variety of strategies may be employed, there are three requirements that apply to any strategy employed.

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 **partyPrivProtocol** value of the SNMP party receiving the message. When that value is the Symmetric Privacy Protocol, the procedures executed by a SNMP protocol entity whenever a management communication is to be transmitted by a SNMP party are as follows.

1. If the **SnmpAuthMsg** is not authenticated according to the conventions of the Digest Authentication Protocol, the generation of the **SnmpPrivMsg** 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, according to the conventions of the mechanism defined in Section 4.4.2.

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

4. The octet sequence representing the serialized value of **SnmpAuthMsg** is encrypted according to the conventions of the mechanism defined in Section 4.4.2, using 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.

be encrypted must be protected according to the conventions of the Digest Authentication Protocol.

Recall from [2], 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:

- Its **privDst** component is called the *private destination* and identifies the SNMP party to which the communication is directed.
- Its **privData** component is called the *private data* and represents the (possibly encrypted) serialization (according to the conventions of [12] and [1]) of a SNMP authenticated management communication (according to the conventions of the Digest Authentication Protocol).

The Symmetric Privacy Protocol is identified by the ASN.1 object identifier listed in Section 4.4.2.

7.1 Generating 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

authentication key and its other components are set to the value of the corresponding components in the received **SnmpAuthMsg**. This new **SnmpAuthMsg** value is serialized according to the conventions of [12] and [1]. A digest is computed over the octet sequence representing the serialized value of the new **SnmpAuthMsg** according to the conventions of the mechanism defined in Section 4.4.1.

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. In the local database the originating SNMP party's:

   - last-timestamp and nonce values are set to the **authTimestamp** value and the **authNonce** value, respectively, and
   - authentication clock is set to the **authTimestamp** value if its value is less than the **authTimestamp** 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].

7 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

According to [2], a **SnmpAuthMsg** value is evaluated during Step 9 of generic processing. In particular, it states the **SnmpAuthMsg** is evaluated according to the **partyAuthProtocol** value of the SNMP party that originated the message. When that value is the Digest Authentication Protocol, the procedures executed by a SNMP protocol entity whenever a management communication is received by a SNMP party are as follows.

1. If the contents octets of the **authInfo** component value is not an **AuthInformation** value, 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 according to the conventions of the mechanism defined in Section 4.4.1) 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** is constructed such that its **authDigest** component is set to the private

authentication key (extracted according to the conventions of the mechanism defined in Section 4.4.1) 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 **partyAuthNonce** and **partyAuthLastMsg** 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** is serialized according to the conventions of [12] and [1]. A digest is computed over the octet sequence representing the serialized value of **SnmpAuthMsg** according to the conventions of the mechanism defined in Section 4.4.1. 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.

6.2 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.

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.

- 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.

In this context, the Digest Authentication Protocol is identified by the ASN.1 object identifier listed in Section 4.4.1.

6.1 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 **partyAuthProtocol** value of the SNMP party originating the message. When that value is the Digest Authentication Protocol, the procedures executed by a SNMP protocol entity whenever a management communication is to be transmitted by a SNMP party are as follows.

1. The local database is consulted to determine the authentication clock, last-timestamp, nonce, and private

determining whether the communication is authentic or not.

- 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 data type **AuthInformation**. It is represented by an ASN.1 value with the following syntax.

```
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:

- 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.

- Its **authNonce** component is called the *authentication nonce* and represents a non-negative integer value evaluated according to the **authTimestamp** value. Note that the granularity of the authentication timestamp is 1 tick per second. 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

true:

- Its **dstParty** component is called the *destination* and identifies the SNMP party to which the communication is directed.
- Its **srcParty** component is called the *source* and identifies the SNMP party from which the communication is originated.
- Its **pdu** component has the form and significance attributed to it in [1].

Recall from [2], 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:

- 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

- Its **partyPrivPublic** component is called the *public privacy key* and represents any public value that may be needed to support the privacy protocol. In this context, this component is not significant (see Section 8.4).

6 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], 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

- 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.

- 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.

- Its **partyAuthPrivate** component is called the *private authentication key* and represents any secret value needed to support the authentication protocol. Valid values in this context are defined according to the value of **partyAuthProtocol**.

- Its **partyAuthPublic** component is called the *public authentication key* and represents any public value that may be needed to support the authentication protocol. In this context, this component is not significant (see Section 8.4).

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

- Its **partyPrivProtocol** component is called the *privacy protocol* and identifies a protocol and a mechanism by which all protocol messages received by the party are protected from disclosure. Valid values in this context are listed in Section 9.1.

- Its **partyPrivPrivate** component is called the *private privacy key* and represents any secret value needed to support the privacy protocol. Valid values in this context are defined according to the value of **partyPrivProtocol**.

For each **SnmpParty** value that supports the generation of messages using the Digest Authentication Protocol or the receipt of messages via the Symmetric Privacy Protocol, the significance of each of its components is as follows.

- Its **partyIdentity** component is called the *identity* of the party and corresponds to the party identity.
- Its **partyTDomain** component is called the *transport domain* and indicates the kind of transport service by which the party receives network management traffic. An example of a transport domain is **rfcXXXXDomain** (SNMP over UDP).
- Its **partyTAddr** component is called the *transport addressing information* and represents a transport service address by which the party receives network management traffic.
- Its **partyProxyFor** component is called the *proxied party* and represents the identity of a second SNMP party or other management entity with which interaction may be necessary to satisfy received management requests. In this context, the value **noProxy** signifies that the party responds to received management requests by entirely local mechanisms.
- Its **partyMaxMessageSize** component is called the *maximum message size* and identifies the maximum number of octets the party is prepared to accept in a SNMP message.
- Its **partyAuthProtocol** component is called the *authentication protocol* and identifies a protocol and a mechanism by which all messages generated by the party are authenticated as to integrity and origin. Valid values in this context are listed in Section 9.1.
- Its **partyAuthClock** component is called the *authentication clock* and represents a notion of the current time that is specific to the party.

```
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
}
```

- the size of the **partyPrivPrivate** component 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 **partyPrivPrivate** is a 56 bit quantity used directly by the algorithm plus 8 parity bits, with one parity bit in 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 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.

5 SNMP Party

Recall from [2], 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 can be represented by an ASN.1 value with the following syntax.

4.4.2 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: [7] and [8], respectively.

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

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

- 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.

- 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.

When used in conjunction with the Symmetric Privacy Protocol (see Section 7):

- this mechanism is identified by the ASN.1 object identifier value **desPrivProtocol**, defined in [4].

algorithm. The principal reason for this is to acknowledge the lack of a suitable metric with which to measure the security of a particular choice of an algorithm. However, in the interests of completeness and in order to guarantee interoperability, Sections 4.4.1 and 4.4.2 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.

4.4.1 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 version of source code that implements the algorithm. This code was written with portability being the principal objective. Implementors may wish to consider optimizing the implementation with respect to the characteristics of their hardware and software platforms.

When used in conjunction with the Digest Authentication Protocol (see Section 6):

- this mechanism is identified by the ASN.1 object identifier value **md5AuthProtocol**, defined in [4].

- the size of the **partyAuthPrivate** component of each **SnmpParty** (see Section 5) that specifies the use of **md5AuthProtocol** is 16 octets.

- the size of the **authDigest** component of an **AuthInformation** value (see Section 6) is 16 octets.

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.

4.4 Mechanisms

The types of mechanisms required to support each of the security services and goals of the SNMP security protocols defined in this memo are as follows.

- 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.
- 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.
- 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.
- 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 independent of the particular choice of a message digest and encryption

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.

4.3 Security Services

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

Data Integrity. The provision of the property that data and data sequences have not been altered or destroyed in an unauthorized manner. (This service is not intended to contradict Goal 4, which requires *late* messages to be discarded (see Section 9).)

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

Data Confidentiality. 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

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.

4.2 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.

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.

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

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 8.3 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 8.4 presents one such strategy that is particularly suited to the demands of SNMP network management.

4.1 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 manipulate the value of objects in a managed agent, including reading their value. The modification threat is

4 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 6. It provides a data integrity service by having the originator compute a digest over an appropriate portion of a message and sending that digest to the recipient, with the message, for verification. 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 7. It protects messages from disclosure by encrypting their contents

1 Status of this Memo

This draft document will be submitted to the RFC editor as a protocol specification. Distribution of this memo is unlimited. Please send comments to the SNMP Security Developers mailing list (snmp-sec-dev@tis.com) or to the authors (James M. Galvin <galvin@tis.com>, Keith McCloghrie <kzm@hls.com>, and James R. Davin <jrd@ptt.lcs.mit.edu>).

2 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.

3 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:

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

8.1 Initial Configuration 25

8.2 Clock Distribution 28

8.3 Clock Synchronization 30

8.4 Secret Distribution 32

8.5 Crash Recovery 34

9 Security Considerations 37

9.1 Conformance 40

9.2 Protocol Correctness 42

9.2.1 Clock Monotonicity Mechanism 43

9.2.2 Data Integrity Mechanism 43

9.2.3 Data Origin Authentication Mechanism 44

9.2.4 Restricted Administration Mechanism 44

9.2.5 Ordered Delivery Mechanism 45

9.2.6 Message Timeliness Mechanism 46

9.2.7 Selective Clock Acceleration Mechanism 47

9.2.8 Confidentiality Mechanism 48

Contents

1 Status of this Memo 1

2 Acknowledgements 1

3 Abstract 1

4 Introduction 2

4.1 Threats 3

4.2 Goals and Constraints 5

4.3 Security Services 6

4.4 Mechanisms 7

4.4.1 Message Digest Algorithm 8

4.4.2 Symmetric Encryption Algorithm 9

5 SNMP Party 10

6 Digest Authentication Protocol 14

6.1 Generating a Message 17

6.2 Receiving a Message 18

7 Symmetric Privacy Protocol 20

7.1 Generating a Message 21

7.2 Receiving a Message 23

8 Clock and Secret Distribution 23

SNMP Security Protocols

James M. Galvin
Trusted Information Systems, Inc.

Keith McCloghrie
Hughes LAN Systems, Inc.

James R. Davin
MIT Laboratory for Computer Science

January 17, 1992