Session-based Security Model for SNMPv3 (SNMPv3/SBSM)

David T. Perkins

Wes Hardaker

NMRG Meeting – October 19, 2003

SNMPv3 Background

The following topics were left out due to time considerations:

- SNMPv3 message format
- SNMPv3 security and general security terminology
- The operational problems with using SNMPv3 with USM

SBSM Characteristics

- Uses existing security infrastructures for identity authentication (supports many)
- Both ends of message exchange are authenticated, and may use different mechanisms (including “anonymous” identity)
- When session establishment is initiated by a manager, the agent reveals its identity and authenticates before the manager (note that identities are encrypted)
- Has limited life time keys for message authentication and encryption

Characteristics (continued)

- Separates security mechanisms used for identity authentication from those used for message authentication and encryption
- Has no reprocessing of messages that are duplicated or replayed (reduces cost of packet loss – processing and latency)
- Operates over connection oriented and connectionless transports
- Can use unmodified VACM, or with slight modifications

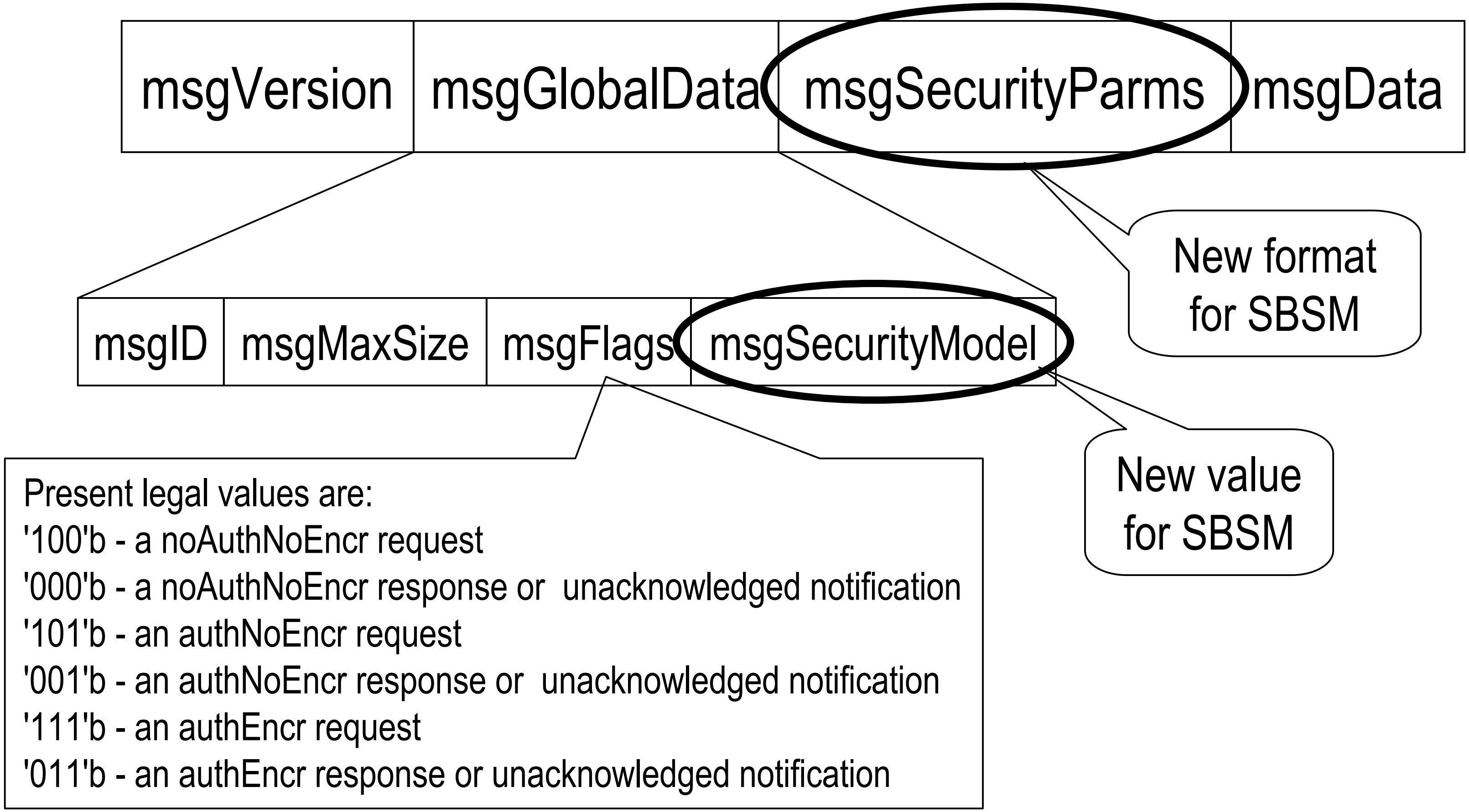
Consequences of Characteristics

- No (or low) cost to create new identities, change their authentication credentials, or delete, since provided by existing security infrastructure
- Saved encrypted messages can not be decrypted after compromised identity key

Most Important Characteristic and Consequence

- Session establishment based on SIGMA protocol, which has had extensive review
- SIGMA is “simple and efficient”, (it minimizes messages and computation)
- SIGMA protects identity of the session initiator
- SIGMA - Krawczyk, H., "SIGMA: the `SIGn-and-MAc' Approach to Authenticated Diffie-Hellman and its Use in the IKE Protocols", in Advances in Cryptography - CRYPTO 2003 Proceedings, LNCS 2729, Springer, 2003. available at: http://www.ee.technion.ac.il/~hugo/sigma.html
- Current draft of IKEv2 is draft-ietf-ipsec-ikev2-11.txt

SNMPv3 Message Format



msgVersion	msgGlobalData	msgSecurityParms	msgData

msgID	msgMaxSize	msgFlags	msgSecurityModel

New format for SBSM

New value for SBSM

Present legal values are:
'100'b - a noAuthNoEncr request
'000'b - a noAuthNoEncr response or unacknowledged notification
'101'b - an authNoEncr request
'001'b - an authNoEncr response or unacknowledged notification
'111'b - an authEncr request
'011'b - an authEncr response or unacknowledged notification

SBSM Overview

- Security based on sessions
- Three phases of a session, which are:
  - Establishment: SNMP entity identity authentication, and creation of session authentication and encryption keys
  - Running: SNMP operations
  - Termination: graceful close of session

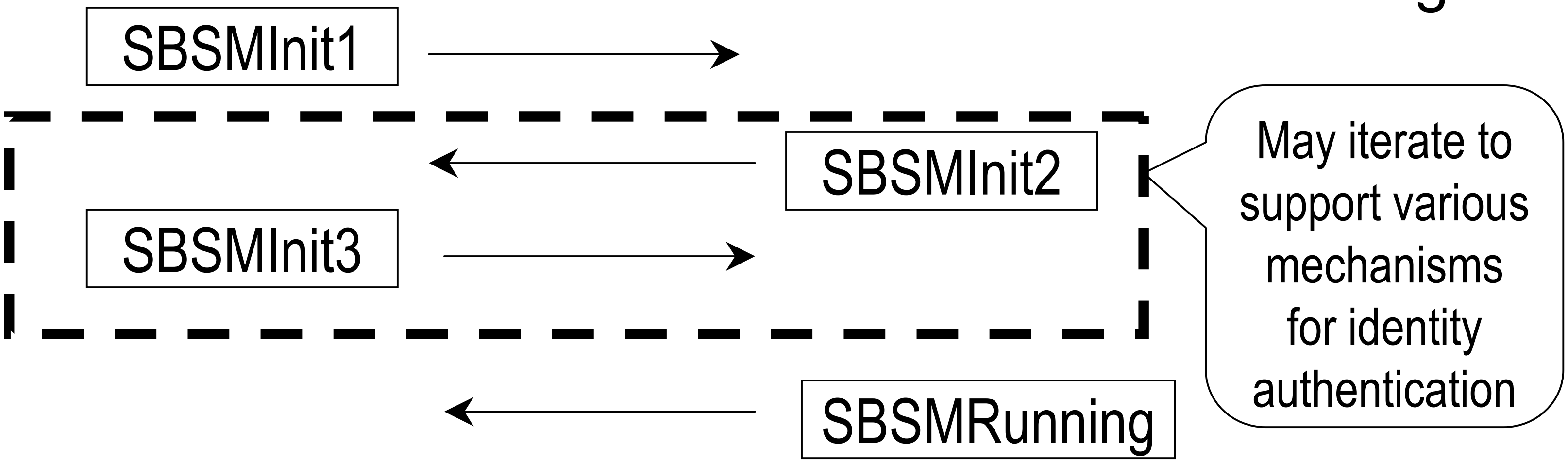
SBSM Session Establishment

Initiator

SNMP GET message

Responder

SNMP REPORT message



SBSMInit1

SBSMInit2

SBSMInit3

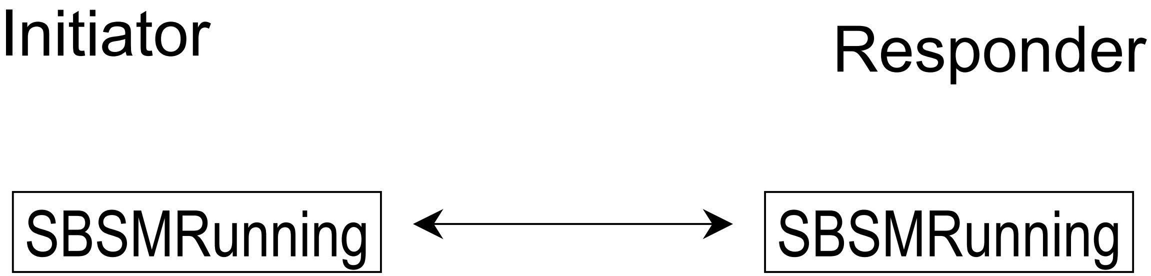
SBSMRunning

May iterate to support various mechanisms for identity authentication

… Session established …

Note: for SNMPv3 messages containing SBSMInit[1,2,or 3] messages, the value for field *msgFlags* indicates *noAuthNoEncr* security level

Session Operation



Initiator

Responder

SBSMRunning

SBSMRunning

Note: SNMPv3 messages containing SBSMRunning messages are always authenticated, and are possibly encrypted using the session auth and encr keys. Thus, the value for field *msgFlags* never indicates *noAuthNoEncr* security level.

Session Graceful Termination

… Details later …

Use With VACM

- VACM has abstract function *isAccessAllowed*, which has the following input parameters:
  - *security model ID*: the message level security model
  - *security name*: the identity for the operation
  - *security level*: one of noAuthNoEncr, authNoEncr, or authEncr
  - *operation type*: one of read, write, or notify
  - *context ID*: the context which contains the instance of management information
  - *instance ID*: the ID of the instance of management information for the operation

VACM Modification

- Abstract function *isAccessAllowed* has input *securityName* and *securityModelID*, which maps to a group name via table *vacmSecurityToGroupTable*
- Clarification:
  - SBSM is really a higher level security model that supports many combinations of endpoint identity authentication. The security model ID for VACM is the identity security model, which is called the *security sub-model*.
- Issue:
  - The “to group” table contains security names, which means that it must be updated for each new security identity, and if a system is compromised, then provides a list to help attacker.
  - Need more study to determine if another or additional mechanisms are needed to get group ID

Details on SBSM security parms

- In an SNMPv3 message, field “security parms” is an octet string, which is the BER serialization of a security model dependent ASN.1 value
- For SBSM, the ASN.1 definition of a value is:

```
SBSMSecurityParameters ::= CHOICE {
               sbsm-establishment1[0]        SBSMInit1,
               sbsm-establishment2[1]        SBSMInit2,
               sbsm-establishment3[2]        SBSMInit3,
                      sbsm-running[3]        SBSMRunning
-- other values for termination and errors
           }
```

SBSM Session Attributes

```
local-identifier                 Unsigned32,
remote-identifier                Unsigned32,
session-status                   INTEGER { init1(1),
                                           init2(2), up(3) }
diffieHelmanExponent             OCTET STRING,
outgoingSequenceNumber           Unsigned32,
incomingMinSequenceNumber        Unsigned32,
security-sub-model               Unsigned32,
securityName                     OCTET STRING,
authenticationType               OBJECT IDENTIFER,
incomingAuthenticationKey        OBJECT STRING,
outgoingAuthenticationKey        OBJECT STRING,
```

Session Attributes (continued)

```
encryptionType                   OBJECT IDENTIFER,
incomingEncryptionParameters     OCTET STRING,
outgoingEncryptionParameters     OCTET STRING,
incomingEncryptionKey            OBJECT STRING,
outgoingEncryptionKey            OBJECT STRING,
window-size                      INTEGER (1..255),
startTime                        Unsigned32,
legalSessionLength               Unsigned32,  -- seconds
remoteEngineID                   OCTET STRING (0|5..32)
 -- data cache array for replaying responses
lastIncomingInit                 OCTET STRING,
messageCacheList                 SEQUENCE (SIZE(0..255))
                                     OF SBSMMessageCache
```

SBSMInit1 Generation Results

- SBSMInit1 is used to start establishment of a session
- Causes creation of a session instance
- Generator fills in:
  - init-identifier
  - session-status
  - diffieHelmanExponent
  - outgoingEncryptionParameters

SBSMInit1 Reception Results

- Reception results in creation of a session instance with field values:
  - local-identifier
  - remote-identifier
  - authenticationType and encryptionType
  - incomingEncryptionParameters
  - outgoingEncryptionParameters
  - Incoming/outgoing Auth/Encr Key
  - startTime and legalSessionLength
  - lastIncomingInit, messageCacheList[0].message

SBSMInit2 Reception Results

- Reception results in update of the following:
  - Incoming/outgoing Auth/Encr Key
  - authenticationType and encryptionType
  - remoteEngineID
  - window-size
  - outgoingSequenceNumber and incomingMinSequenceNumber
  - session-status
  - securityName
  - startTime and legalSessionLength

SBSMInit3 Reception Results

- Reception results in update of the following:
  - window-size
  - session-status
  - securityName
  - startTime and legalSessionLength
  - remoteEngineID

What’s Next?

- Further update of I-D to polish terminology, and fill in small missing pieces
- Complete the error handling descriptions
- Work through notification generation using the model and MIB from RFC 3414 (SNMP Applications) (was RFC
- Choose a couple of Identity Authentication types, document well, and write code

Questions