Managing Long-Term Keys for Routing Protocols

November 10, 2009
Russ Housley
Tim Polk
Drafts

• Database of Long-Lived Cryptographic Keys
  – <draft-housley-saag-crypto-key-table-00.txt>

• Routing Authentication Using A Database of Long-Lived Cryptographic Keys
  – <draft-polk-saag-rtg-auth-keytable-00.txt>

• The former defines a conceptual model, the latter describes the model’s application to routing protocols
Fundamental Concept

• Manual key management is today’s reality in routing protocols
  – Future key establishment protocols must co-exist with manual keying
• Key establishment will remain separate protocols, not a handshake in routing protocols
• Modeled as a database or table of shared keys that are available to the routing protocols
• Accommodate textual description of database entries
Database

• Database is characterized as a table, with a row for each key
• Identifies 11 columns for the key and its attributes
• Describes rollover between long-lived keys
Database Columns (1 of 2)

• **LocalKeyID**
  – A 16-bit integer in hexadecimal, unique in the context of the database. The high order bit differentiates pairwise and group keys.

• **PeerKeyID**
  – For pairwise keys, the peerKeyID field is a 16 bit integer in hexadecimal provided by the peer or "unknown" if the peer has not yet provided this value.
  – For group keying, the PeerKeyID field is set to "group", which easily accommodates group keys generated by a third party.

• **KDF**
  – Indicates which key derivation function (KDF) is used to generate short-lived keys (or "none" when the long-term key is used directly).

• **KDFInputs**
  – Used when supplemental public or private data is supplied to the KDF.

• **AlgID**
  – Indicates which cryptographic algorithm to be used with the security protocol.
Database Columns (2 of 2)

- **Key**
  - A hexadecimal string representing a long-lived symmetric cryptographic key.
- **KeyDirection**
  - Indicates whether this key may be used for inbound traffic, outbound traffic, or both.
- **NotBefore**
  - Specifies the earliest date and time at which this key should be considered for use.
- **NotAfter**
  - Specifies the latest date and time at which this key should be considered for use.
- **Peers**
  - Identifies a peer system or set of peer systems
- **Protocol**
  - Identifies the security protocol where this key is to be used to provide cryptographic protection.
The Overall Model

- Manual Key Installation
- Automated Key Mgmt. Protocol

*Long-Lived Crypto Keys*

*Short-Lived Crypto Session Keys*
Initiator’s View

- Long-Lived Crypto Keys
  - Lookup Keys by Peer and Protocol
  - Select Key by Policy
    - Session Key Derivation
    - Authentication Mechanism
  - Initiate Session with Peer
Receiver’s View

- Long-Lived Crypto Keys
  - Lookup Keys by KeyID
    - Receive Data from Peer
  - Session Key Derivation
    - Authentication Mechanism
KeyID Mapping

- Database specification mandates a 16-bit KeyID
- KeyID in the table may not be the KeyID used on the wire
  - Need to support more than just one security protocol
  - Allow translation to any needed format or size
  - Overlapping ranges may unnecessarily limit the total number of keys that can be maintained
- Mapping can resolve size mismatch and overcome overlapping range issues
  - Only applicable to local KeyID values
  - Peer’s KeyIDs are not unique in the context of the table
Initiator’s View with Mapping

- Long-Lived Crypto Keys
  - Lookup Keys by Peer and Protocol
  - Select Key by Policy
    - Session Key Derivation and KeyID Mapping
  - Initiate Session with Peer
  - Authentication Mechanism
Questions?