CSC 474 Network Security

Topic 8.2 Internet Key Management
Outline

• Key Management
  – Security Principles

• Internet Key Management
  – Manual Exchange
  – SKIP
  – Oakley
  – ISAKMP
  – IKE
Key Management

• Why do we need Internet key management
  – AH and ESP require encryption and authentication keys

• Process to negotiate and establish IPsec SAs between two entities
Security Principles

• Basic security principle for session keys
  – Compromise of a session key
    • Doesn’t permit reuse of the compromised session key.
    • Doesn’t compromise future session keys and long-term keys.
Security Principles (Cont’d)

• Perfect forward secrecy (PFS)
  – Compromise of current keys (session key or long-term key) doesn’t compromise past session keys.
  – Concern for encryption keys but not for authentication keys.
  – Not really “perfect” in the same sense as perfect secrecy for one-time pad.
Internet Key Management

• Manual key management
  – Mandatory
  – Useful when IPsec developers are debugging
  – Keys exchanged offline (phone, email, etc.)
  – Set up SPI and negotiate parameters
Internet Key Management (Cont’d)

• Automatic key management
  – Two major competing proposals
  – Simple Key Management for Internet Protocols (SKIP)
  – ISAKMP/OAKLEY
    • Photuris
      – Ephemeral D-H + authentication + Cookie
      – The first to use cookie to thwart DOS attacks
    • SKEME (extension to Photuris)
    • Oakley (RFC 2412)
    • ISAKMP (RFC 2408)
    • ISAKMP/OAKLEY → IKE (RFC 2409)
A Note about IKE

- IKE v2 was introduced in RFC 4306 (December 2005)
- IKE v2 does not interoperate with IKE v1
  - Both version can unambiguously run over the same UDP port
- IKE v2 combines the contents of previously separate documents
  - ISAKMP
  - IKE v1
  - DOI
  - NAT
  - ...
Automatic Key Management

• Key *establishment* and *management* combined
  – SKIP

• Key *establishment* protocol
  – Oakley
    • focus on key exchange

• Key *management*
  – Internet Security Association & Key Management Protocol (ISAKMP)
    • Focus on SA and key management
    • Clearly separated from key exchange.
• Idea
  – IP is connectionless in nature
  – Using security association forces a pseudo session layer underneath IP
  – Proposal: use sessionless key establishment and management
    • Pre-distributed and authenticated D-H public key
    • Packet-specific encryption keys are included in the IP packets
Two types of keys:
1. KEK
2. Packet key

Certificate repository

Bob’s certificate

Alice’s certificate

Alice → Bob

K_p encrypted with KEK. Payload encrypted with K_p.
SKIP (Cont’d)

• KEK should be changed periodically
  – Minimize the exposure of KEK
  – Prevent the reuse of compromised packet keys

• SKIP’s approach
  – KEK = h \( (K_{AB}, n) \), where \( h \) is a one-way hash function, \( K_{AB} \) is the the long term key between A and B, and \( n \) is a counter.
SKIP (Cont’d)

• Limitations
  – No Perfect Forward Secrecy
    • Can be modified to provide PFS, but it will lose the sessionless property.
  – No concept of SA; difficult to work with the current IPsec architecture

• Not the standard, but remains as an alternative.
Oakley

- Oakley is a refinement of the basic Diffie-Hellman key exchange protocol.

- Why need refinement?
  - Resource clogging attack
  - Replay attack
  - Man-in-the-middle attack
  - Choice of D-H groups
Resource Clogging Attack

- Many bogus requests with false source IPs
- Stopping requests is difficult
  - We need to provide services.
- Ignoring requests is dangerous
  - Denial of service attacks
Resource Clogging Attack (Cont’d)

• Counter measure
  – If we cannot stop bogus requests, at least we should know from where the requests are sent.
  – Cookies are used to thwart resource clogging attack
    • Thwart, not prevent
Resource Clogging Attack (Cont’d)

• Cookie
  – Each side sends a pseudo-random number, the cookie, in the initial message, which the other side acknowledges.
  – The acknowledgement must be repeated in the following messages.
  – Do not begin D-H calculation until getting acknowledgement for the other side.
Requirements for cookie generation

• The cookie must depend on the specific parties.
  – Prevent an attacker from reusing cookies.

• Impossible to forge
  – Use secret values

• Efficient

• Cookies are also used for key naming
  – Each key is uniquely identified by the initiator’s cookie and the responder’s cookie.
Replay Attack

- **Counter measure**
  - Use **nonce**

1. Cookie exchange
2. Later exchange
3. Replay
4. Busy computing

Observe
Man-In-The-Middle Attack

- Counter measure
  - Authentication
  - Depend on other mechanisms.
    - Pre-shared key.
    - Public key certificates.
Oakley Groups

- How to choose the DH groups?
  - 0  no group (placeholder or non-DH)
  - 1  MODP, 768-bit modulus
  - 2  MODP, 1024-bit modulus
  - 3  MODP, 1536-bit modulus
  - 4  EC2N over GF(2^{155})
  - 5  EC2N over GF(2^{185})
Ephemeral Diffie-Hellman

- Session key is computed on the basis of short-term DH public-private keys.
- Exchange of these short-term public keys requires authentication and integrity.
  - Digital signatures.
  - Keyed message digests.
- The only protocol known to support Perfect Forward Secrecy.
Ephemeral Diffie-Hellman

• Question: What happens if the long term key is compromised?
ISAKMP

• Oakley
  – Key exchange protocol
  – Developed to use with ISAKMP

• ISAKMP
  – Security association and key management protocol
  – Defines procedures and packet formats to establish, negotiate, modify, and delete security associations.
  – Defines payloads for security association, key exchange, etc.
ISAKMP Message

• Fixed format header
  – 64 bit initiator and responder cookies
  – Exchange type (8 bits)
  – Next payload type (8 bits)
  – Flags: encryption, commit, authentication, etc.
  – 32 bit message ID
    • Resolve multiple phase 2 SAs being negotiated simultaneously
  – Variable number of payloads
    • Each has a generic header with
      – Payload boundaries
      – Next payload type (possible none)
ISAKMP Formats

(a) ISAKMP Header

(b) Generic Payload Header
ISAKMP Phases

• Phase 1
  – Establish ISAKMP SA to protect further ISAKMP exchanges
  – Or use pre-established ISAKMP SA
  – ISAKMP SA identified by initiator cookie and responder cookie

• Phase 2
  – Negotiate security services in SA for target security protocol or application.
ISAKMP

- Disadvantage
  - Additional overhead due to 2 phases

- Advantages
  - Same ISAKMP SA can be used to negotiate phase 2 for multiple protocols
  - ISAKMP SA can be used to facilitate maintenance of SAs.
  - ISAKMP SA can simplify phase 2.
ISAKMP Domain Of Interpretation (DOI)

• DOI defines
  – Payload format
  – Exchange types
  – Naming conventions for security policies, cryptographic algorithms

• DOI for IPsec has been defined.
ISAKMP Exchange Types

• 0  none
• 1  base
• 2  identity protection
• 3  authentication only
• 4  aggressive
• 5  informational
• 6-31  reserved
• 32-239  DOI specific use
• 240-255  private use
ISAKMP Exchange Types

- Base exchange
  - reveals identities

- Identity protection exchange
  - Protects identities at cost of extra messages.

- Authentication only exchange
  - No key exchange

- Aggressive exchange
  - Reduce number of message, but reveals identity

- Informational exchange
  - One-way transmission of information.
ISAKMP Payload Types

- 0 none
- 1 SA security association
- 2 P proposal
- 3 T transform
- 4 KE key exchange
- 5 ID identification
- 6 CERT certificate
- 7 CR certificate request
ISAKKMP Payload Types

- 8 H hash
- 9 SIG signature
- 10 NONCE nonce
- 11 N notification
- 12 D delete
- 13 VID vendor ID
- 14-127 reserved
- 128-255 private use
# ISAKMP Payload Types

<table>
<thead>
<tr>
<th>Type</th>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Security Association (SA)</td>
<td>Domain of Interpretation, Situation</td>
<td>Used to negotiate security attributes and indicate the DOI and Situation under which negotiation is taking place.</td>
</tr>
<tr>
<td>Proposal (P)</td>
<td>Proposal #, Protocol-ID, SPI Size, # of Transforms, SPI</td>
<td>Used during SA negotiation; indicates protocol to be used and number of transforms.</td>
</tr>
<tr>
<td>Transform (T)</td>
<td>Transform #, Transform-ID, SA Attributes</td>
<td>Used during SA negotiation; indicates transform and related SA attributes.</td>
</tr>
<tr>
<td>Key Exchange (KE)</td>
<td>Key Exchange Data</td>
<td>Supports a variety of key exchange techniques.</td>
</tr>
<tr>
<td>Identification (ID)</td>
<td>ID Type, ID Data</td>
<td>Used to exchange identification information.</td>
</tr>
<tr>
<td>Certificate (CERT)</td>
<td>Cert Encoding, Certificate Data</td>
<td>Used to transport certificates and other certificate-related information.</td>
</tr>
<tr>
<td>Certificate Request (CR)</td>
<td># Cert Types, Certificate Types, # Cert Auths, Certificate Authorities</td>
<td>Used to request certificates; indicates the types of certificates requested and the acceptable certificate authorities.</td>
</tr>
<tr>
<td>Hash (HASH)</td>
<td>Hash Data</td>
<td>Contains data generated by a hash function.</td>
</tr>
<tr>
<td>Signature (SIG)</td>
<td>Signature Data</td>
<td>Contains data generated by a digital signature function.</td>
</tr>
<tr>
<td>Nonce (NONCE)</td>
<td>Nonce Data</td>
<td>Contains a nonce.</td>
</tr>
<tr>
<td>Notification (N)</td>
<td>DOI, Protocol-ID, SPI Size, Notify Message Type, SPI, Notification Data</td>
<td>Used to transmit notification data, such as an error condition.</td>
</tr>
<tr>
<td>Delete (D)</td>
<td>DOI, Protocol-ID, SPI Size, # of SPIs, SPI (one or more)</td>
<td>Indicates an SA that is no longer valid.</td>
</tr>
</tbody>
</table>
## ISAKMP Exchanges

**Basic Exchange**

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>( \text{I} \rightarrow \text{R}: \text{SA}; \text{NONCE} )</td>
<td>• Begin ISAKMP-SA negotiation</td>
</tr>
<tr>
<td>2.</td>
<td>( \text{R} \rightarrow \text{I}: \text{SA}; \text{NONCE} )</td>
<td>• Basic SA agreed upon</td>
</tr>
<tr>
<td>3.</td>
<td>( \text{I} \rightarrow \text{R}: \text{KE}; \text{ID}_I; \text{AUTH} )</td>
<td>• Key generated; Initiator id verified by responder</td>
</tr>
<tr>
<td>4.</td>
<td>( \text{R} \rightarrow \text{I}: \text{KE}; \text{ID}_R; \text{AUTH} )</td>
<td>• Responder id verified by initiator; key generated; SA established</td>
</tr>
</tbody>
</table>
ISAKMP Exchanges (Cont’d)

Identity Protection Exchange

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>I→R: SA</td>
<td><strong>Begin ISAKMP-SA negotiation</strong></td>
</tr>
<tr>
<td>2.</td>
<td>R→I: SA</td>
<td><strong>Basic SA agreed upon</strong></td>
</tr>
<tr>
<td>3.</td>
<td>I→R: KE; NONCE</td>
<td><strong>Key generated</strong></td>
</tr>
<tr>
<td>4.</td>
<td>R→I: KE; NONCE</td>
<td><strong>Key generated</strong></td>
</tr>
<tr>
<td>5.</td>
<td>I→R: IDᵢ; AUTH</td>
<td><strong>Initiator id verified by responder</strong></td>
</tr>
<tr>
<td>6.</td>
<td>R→I: IDᵣ; AUTH</td>
<td><strong>Responder id verified by initiator; SA established</strong></td>
</tr>
</tbody>
</table>

Red messages: Payload encrypted after ISAKMP header
### ISAKMP Exchanges (Cont’d)

#### Authentication Only Exchange

1. **I→R: SA; NONCE**
   - Begin ISAKMP-SA negotiation

2. **R→I: SA; NONCE; ID$_R$; AUTH**
   - Basic SA agreed upon; Responder id verified by initiator

3. **I→R: ID$_I$; AUTH**
   - Initiator id verified by responder; SA established
ISAKMP Exchanges (Cont’d)

### Aggressive Exchange

1. I → R: SA; KE; NONCE; ID<sub>I</sub>  
   - Begin ISAKMP-SA negotiation and key exchange

2. R → I: SA; KE; NONCE; ID<sub>R</sub>; AUTH  
   - Responder identity verified by responder; Key generated; Basic SA agreed upon;

3. I → R: AUTH  
   - Initiator id verified by responder; SA established

Red messages: Payload encrypted after ISAKMP header
ISAKMP Exchanges (Cont’d)

Informational Exchange

1. I→R: N/D
   - Error or status notification, or deletion.

Red message: Payload encrypted after ISAKMP header
IKE Overview

• IKE = ISAKMP + part of OAKLEY + part of SKEME
  – ISAKMP determines
    • How two peers communicate
    • How these messages are constructed
    • How to secure the communication between the two peers
    • No actual key exchange
  – Oakley
    • Key exchange protocol
  – Combining these two requires a Domain of Interpretation (DOI)
    • RFC 2407
IKE Overview (Cont’d)

• A separate RFC has been published for IKE
  – RFC 2409

• Request-response protocol
  – Initiator
  – Responder

• Two phases
  – Phase 1: Establish an IKE (ISAKMP) SA
    • Essentially the ISAKMP phase 1
    • Bi-directional
  – Phase 2: Use the IKE SA to establish IPsec SAs
    • Key exchange phase
    • Directional
IKE Overview (Cont’d)

• Several Modes
  – Phase 1:
    • Main mode: identity protection
    • Aggressive mode
  – Phase 2:
    • Quick mode
  – Other modes
    • New group mode
      – Establish a new group to use in future negotiations
      – Not in phase 1 or 2;
      – Must only be used after phase 1
    • Informational exchanges
      – ISAKMP notify payload
      – ISAKMP delete payload
IPSEC Architecture Revisited

IPSec module 1

What to establish

IPSec module 2

IKE policies (How to establish the IPsec SAs):
1. Encryption algorithm; 2. Hash algorithm;
A Clarification About PFS

• In RFC 2409:
  – When used in the memo Perfect Forward Secrecy (PFS) refers to the notion that compromise of a single key will permit access to only data protected by a single key.
  – The key used to protect transmission of data MUST NOT be used to derive any additional keys.
  – If the key used to protect transmission of data was derived from some other keying material, that material MUST NOT be used to derive any more keys.

• Is this consistent with what we discussed?
IKE Phase 1

• Four authentication methods
  – Digital signature
  – Authentication with public key encryption
  – The above method revised
  – Authentication with a pre-shared key
IKE Phase 1 (Cont’d)

• IKE Phase 1 goal:
  – Establish a shared secret SKEYID
  – With signature authentication
    • $SKEYID = \text{prf}(N_i_b | N_r_b, g^{xy})$
  – With public key encryption
    • $SKEYID = \text{prf}(\text{hash}(N_i_b | N_r_b), \text{CKY-I} \mid \text{CKY-R})$
  – With pre-shared key
    • $SKEYID = \text{prf}(\text{pre-shared-key}, N_i_b \mid N_r_b)$
  – Notations:
    • $\text{prf}$: keyed pseudo random function $\text{prf}(key, message)$
    • CKY-I/CKY-R: I’s (or R’s) cookie
    • $N_i_b/N_r_b$: the body of I’s (or R’s) nonce
IKE Phase 1 (Cont’d)

- Three groups of keys
  - Derived key for non-ISAKMP negotiations
    - \( SKEYID_d = \text{prf}(SKEYID, g^{xy} | \text{CKY-I | CKY-R | 0}) \)
  - Authentication key
    - \( SKEYID_a = \text{prf}(SKEYID, SKEYID_d | g^{xy} | \text{CKY-I | CKY-R | 1}) \)
  - Encryption key
    - \( SKEYID_e = \text{prf}(SKEYID, SKEYID_a | g^{xy} | \text{CKY-I | CKY-R | 2}) \)
IKE Phase 1 (Cont’d)

• To authenticate the established key
  – Initiator generates
    • $\text{HASH}_I = \text{prf}(\text{SKEYID}, g^{x_i} | g^{x_r} | CKY-I | CKY-R | SA_i_b | ID_{ii_b})$
  – Responder generates
    • $\text{HASH}_R = \text{prf}(\text{SKEYID}, g^{x_r} | g^{x_i} | CKY-R | CKY-I | SA_i_b | ID_{ir_b})$
  – Authentication with digital signatures
    • $\text{HASH}_I$ and $\text{HASH}_R$ are signed and verified
  – Public key encryption or pre-shared key
    • $\text{HASH}_I$ and $\text{HASH}_R$ directly authenticate the exchange.
IKE Phase 1 Authenticated with Signatures

Main Mode

Initiator: HDR, SA
Initiator: HDR, KE, Ni
Initiator: HDR*, IDii, [CERT,] SIG_I

Responder: HDR, SA
Responder: HDR, KE, Nr
Responder: HDR*, IDir, [CERT,] SIG_R
IKE Phase 1 Authenticated with Signatures

Aggressive Mode

<table>
<thead>
<tr>
<th>Initiator</th>
<th>Responder</th>
</tr>
</thead>
<tbody>
<tr>
<td>HDR, SA, KE, Ni, IDii</td>
<td>HDR, SA, KE, Nr, IDir, [CERT,] SIG_R</td>
</tr>
<tr>
<td>HDR, [CERT,] SIG_I</td>
<td>HDR, SA, KE, Nr, IDir, [CERT,] SIG_R</td>
</tr>
</tbody>
</table>
IKE Phase 1 Authenticated with Public Key Encryption

Main Mode

<table>
<thead>
<tr>
<th>Initiator</th>
<th>Responder</th>
</tr>
</thead>
<tbody>
<tr>
<td>HDR, SA</td>
<td>HDR, SA</td>
</tr>
<tr>
<td>HDR, KE, [HASH(1),]</td>
<td>HDR, KE, &lt;IDir_b&gt;PubKey_i,</td>
</tr>
<tr>
<td>&lt;IDii_b&gt;PubKey_r,</td>
<td>&lt;Nr_b&gt;PubKey_i</td>
</tr>
<tr>
<td>&lt;Ni_b&gt;PubKey_r</td>
<td>HDR*, HASH_I</td>
</tr>
<tr>
<td>HDR*, HASH_I</td>
<td>HDR*, HASH_R</td>
</tr>
</tbody>
</table>
IKE Phase 1 Authenticated with Public Key Encryption

Aggressive Mode

<table>
<thead>
<tr>
<th>Initiator</th>
<th>Responder</th>
</tr>
</thead>
<tbody>
<tr>
<td>HDR, SA,</td>
<td>HDR, SA, KE,</td>
</tr>
<tr>
<td>[HASH(1),] KE,</td>
<td>&lt;IDir_b&gt;PubKey_i,</td>
</tr>
<tr>
<td>&lt;IDii_b&gt;PubKey_r,</td>
<td>&lt;Nr_b&gt;PubKey_i,</td>
</tr>
<tr>
<td>&lt;Ni_b&gt;PubKey_r</td>
<td>HASH_R</td>
</tr>
<tr>
<td>HDR, HASH_I</td>
<td>HDR, HASH_I</td>
</tr>
</tbody>
</table>

IKE Phase 1 Authenticated with Public Key Encryption

Initiator: HDR, SA, [HASH(1)], KE, <IDii_b>PubKey_r, <Ni_b>PubKey_r
Responder: HDR, SA, KE, <IDir_b>PubKey_i, <Nr_b>PubKey_i, HASH_R

Aggressive Mode
Observations

- Authenticated using public key encryption
  - No non-repudiation
    - No evidence that shows the negotiation has taken place.
  - More difficult to break
    - An attacker has to break both DH and public key encryption
  - Identity protection is provided in aggressive mode.
  - Four public key operations
    - Two public key encryptions
    - Two public key decryptions
IKE Phase 1 Authenticated with A Revised Mode of Public Key Encryption

Main Mode

<table>
<thead>
<tr>
<th>Initiator</th>
<th>Responder</th>
</tr>
</thead>
<tbody>
<tr>
<td>HDR, SA</td>
<td>HDR, SA</td>
</tr>
<tr>
<td>HDR, [HASH(1),]</td>
<td>HDR, [HASH(1),]</td>
</tr>
<tr>
<td>&lt;Ni_b&gt;PubKey_r</td>
<td>&lt;Nr_b&gt;PubKey_i,</td>
</tr>
<tr>
<td>&lt;KE_b&gt;Ke_i</td>
<td>&lt;KE_b&gt;Ke_r,</td>
</tr>
<tr>
<td>&lt;IDii_b&gt;Ke_i,</td>
<td>&lt;IDir_b&gt;Ke_r</td>
</tr>
<tr>
<td>[&lt;Cert-I_b&gt;Ke_i]</td>
<td></td>
</tr>
</tbody>
</table>

HDR*, HASH_I  

HDR*, HASH_R
IKE Phase 1 Authenticated with A Revised Mode of Public Key Encryption

Aggressive Mode

<table>
<thead>
<tr>
<th>Initiator</th>
<th>Responder</th>
</tr>
</thead>
<tbody>
<tr>
<td>HDR, SA, [HASH(1),]</td>
<td>HDR, SA,</td>
</tr>
<tr>
<td>&lt;Ni_b&gt;PubKey_r,</td>
<td>&lt;Nr_b&gt;PubKey_i,</td>
</tr>
<tr>
<td>&lt;KE_b&gt;Ke_i,</td>
<td>&lt;KE_b&gt;Ke_r,</td>
</tr>
<tr>
<td>&lt;IDii_b&gt;Ke_i</td>
<td>&lt;IDir_b&gt;Ke_r,</td>
</tr>
<tr>
<td>[, &lt;Cert-I_b&gt;Ke_i]</td>
<td>HASH_R</td>
</tr>
<tr>
<td>HDR, HASH_I</td>
<td></td>
</tr>
</tbody>
</table>
Further Details

\[
\begin{align*}
Ne_i &= \text{prf}(Ni_i, \text{CKY-I}) \\
Ne_r &= \text{prf}(Nr_i, \text{CKY-R})
\end{align*}
\]

- Ke_i and Ke_r are taken from Ne_i and Ne_r, respectively.
IKE Phase 1 Authenticated with Pre-Shared Key

Main Mode

<table>
<thead>
<tr>
<th>Initiator</th>
<th>Responder</th>
</tr>
</thead>
<tbody>
<tr>
<td>HDR, SA</td>
<td>HDR, SA</td>
</tr>
<tr>
<td>HDR, KE, Ni</td>
<td>HDR, KE, Nr</td>
</tr>
<tr>
<td>HDR*, IDii, HASH_I</td>
<td>HDR*, IDir, HASH_R</td>
</tr>
</tbody>
</table>
IKE Phase 1 Authenticated with Pre-Shared Key (Cont’d)

• What provide the authentication?
• Why does it work?
IKE Phase 1 Authenticated with Pre-Shared Key

Aggressive Mode

<table>
<thead>
<tr>
<th>Initiator</th>
<th>Responder</th>
</tr>
</thead>
<tbody>
<tr>
<td>HDR, SA, KE, Ni, IDii</td>
<td>HDR, SA, KE, Nr, IDir, HASH_R</td>
</tr>
<tr>
<td>HDR, HASH_I</td>
<td></td>
</tr>
</tbody>
</table>
IKE Phase 2 -- Quick Mode

- Not a complete exchange itself
  - Must be bound to a phase 1 exchange
- Used to derive keying materials for IPsec SAs
- Information exchanged with quick mode must be protected by the ISAKMP SA
- Essentially a SA negotiation and an exchange of nonce
  - Generate fresh key material
  - Prevent replay attack
IKE Phase 2 -- Quick Mode (Cont’d)

- **Basic Quick Mode**
  - Refresh the keying material derived from phase 1
- **Quick Mode with optional KE payload**
  - Transport additional exponentiation
  - Provide PFS
IKE Phase 2 -- Quick Mode (Cont’d)

<table>
<thead>
<tr>
<th>Initiator</th>
<th>Responder</th>
</tr>
</thead>
<tbody>
<tr>
<td>HDR*, HASH(3)</td>
<td></td>
</tr>
</tbody>
</table>

HASH(1) = prf(SKEYID_a, M-ID | SA | Ni [, KE] [, IDci | IDcr )
HASH(2) = prf(SKEYID_a, M-ID | Ni_b | SA | Nr [, KE] [, IDci | IDcr )
HASH(3) = prf(SKEYID_a, 0 | M-ID | Ni_b | Nr_b)
IKE Phase 2 -- Quick Mode (Cont’d)

If PFS is not needed, and KE payloads are not exchanged, the new keying material is defined as

\[
\text{KEYMAT} = \text{prf(SKEYID}_d, \text{ protocol} | \text{ SPI} | \text{ Ni}_b | \text{ Nr}_b)
\]

If PFS is desired and KE payloads were exchanged, the new keying material is defined as

\[
\text{KEYMAT} = \text{prf(SKEYID}_d, g(qm)^{xy} | \text{ protocol} | \text{ SPI} | \text{ Ni}_b | \text{ Nr}_b)
\]

where \(g(qm)^{xy}\) is the shared secret from the ephemeral Diffie-Hellman exchange of this Quick Mode.

In either case, "protocol" and "SPI" are from the ISAKMP Proposal Payload that contained the negotiated Transform.
IKE V2

• Combines the contents of ISAKMP, IKE, DOI, NAT, legacy authentication, and remote address acquisition
  – More robust and clean than IKE
  – Initially published in RFC 4306
    • December 2005
  – Further clarified in RFC 4718
    • October 2006
  – Combined in RFC 5996
    • September 2010
IKEv2 Overview

- Performs mutual authentication between initiator and responder
- Establish an IKE SA
  - IKE_SA
- Establish IPsec SAs
  - CHILE_SAs
- All IKEv2 communications: pairs of messages
  - A request followed by a response
  - It’s the responsibility of the requestor to ensure reliability
  - Greatly simplifies protocol state maintenance on end nodes
IKEv2 Overview (Cont’d)

- IKEv2 exchanges
  - IKE_SA_INIT exchange
  - IKE_AUTH exchange
  - CREATE_CHILD_SA exchange
  - INFORMATIONAL exchange

- In common cases
  - A single IKE_SA_INIT exchange
  - A single IKE_AUTH exchange
  - Any number of CREATE_CHILD_SA and INFORMATIONAL exchanges
IKE_SA_INIT Exchange

- Intended to negotiate security parameters for IKE_SA
- After the exchange, both initiator and responder can generate seed for secret keys
- All messages following the exchange are encrypted except for message headers

Proposed crypto suites, $g^a \mod p$, Nonce$_i$

Selected crypto suite, $g^b \mod p$, Nonce$_r$
IKE_AUTH Exchange

- Transmit IDs, prove knowledge of the secrets corresponding to the IDs, set up the first SA for the CHILD_SA
- The recipient of a message must verify all signatures and MACs and the IDs correspond to the keys used to generate the authenticator
CREATE_CHILD_SA Exchange

• Can be initiated by either end
• Counter part of phase 2 in IKE
• Optional KE payload → PFS

Initiator

Offered crypto suite, nonce\textsubscript{i}, optional \(g^a \mod p\) in encrypted form

Responder

Crypto suite, nonce\textsubscript{r}, optional \(g^b \mod p\) in encrypted form
Handling DoS attacks

• Recall IKE uses Cookie mechanism in phase 1
  – Two extra messages

• IKEv2
  – Do not use Cookie mechanism by default
  – A responder performs detection of possible DoS attacks
    • A large number of half-open IKE-SAs?
  – When there is a DoS attack
    • The responder rejects initial IKE messages unless they have a valid Cookie
    • The responder sends an unprotected IKE message with a Cookie
    • An initiator that receives such a response can retry with the Cookie