# IPSec-3

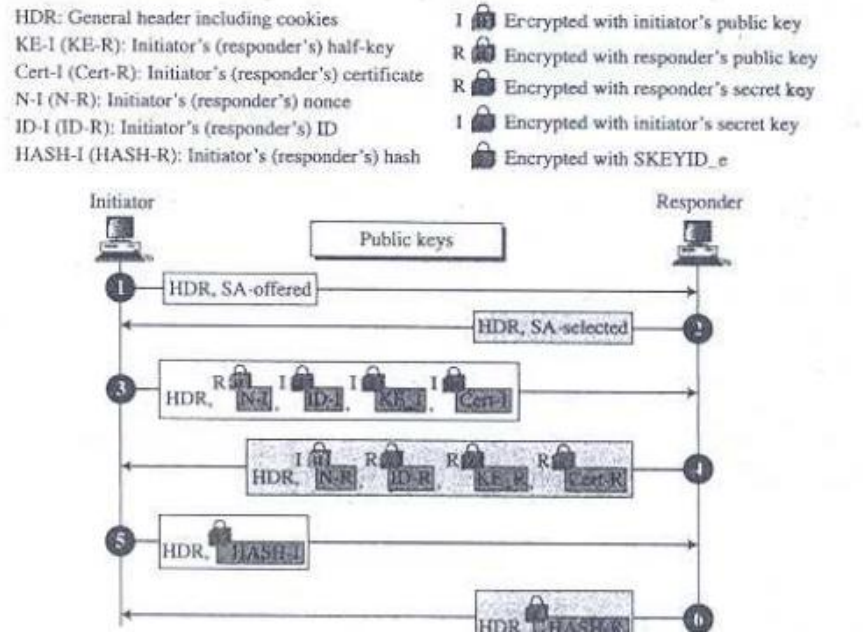
G.P. Biswas

Prof/CSE, IIT, Dhanbad

# Revised Public Key Method

The original public-key method has some drawbacks. First, two instances of public-key encryption/decryption place a heavy load on the initiator and responder. Second, the initiator cannot send its certificate encrypted by the public key of the responder, since anyone could do this with a false certificate. The method was revised so that the public key is used only to create a temporary secret key, as shown in Figure 18.22.

**Figure 18.22** *Main mode, revised public-key method*



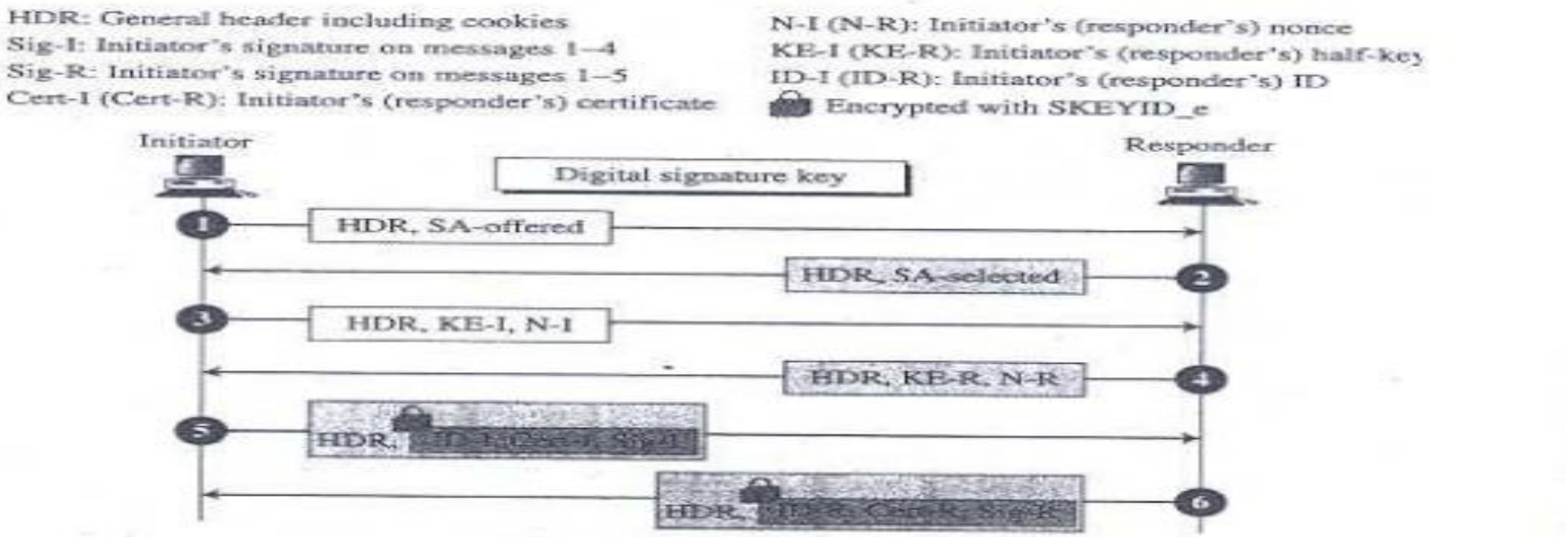
- Note that two temporary secret keys are created from a hash of nonces and cookies. The initiator uses the public key of the responder to send its nonce. The responder decrypts the nonces and calculates the initiator's temporary secret keys. After that the half keys, the ID and optional certificate can be decrypted. The two temporary secret keys, K-I and K-R are calculated as

$$K\text{-}I = prf(N\text{-}I, Cookie\text{-}I) \qquad K\text{-}R = prf(N\text{-}R, Cookie\text{-}R)$$

***Digital Signature Method***

In this method, each party shows that it possesses the certified private key related to a digital signature. Figure 18.23 shows the exchanges in this method. It is similar to the preshared-key method except for the SKEYID calculation.

**Figure 18.23** *Main mode, digital signature method*



Note that in this method the sending of the certificates is optional. The certificate can be sent here because it can be encrypted with SKEYID_e, which does not depend on the signature key. In message 5, the initiator signs all the information exchanged in messages 1 to 4 with its signature key. The responder verifies the signature using the public key of the initiator, which authenticates the initiator. Likewise, in message 6, the responder signs all the information exchanged with its signature key. The initiator verifies the signature.

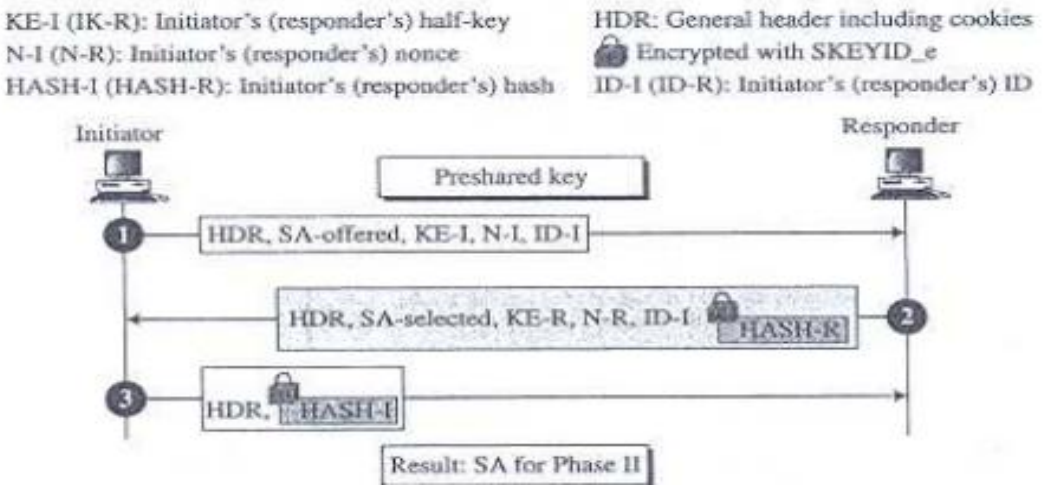
## Phase I: Aggressive Mode

Each **aggressive mode** is a compressed version of the corresponding main mode. Instead of six messages, only three are exchanged. Messages 1 and 3 are combined to make the first message. Messages 2, 4, and 6 are combined to make the second message. Message 5 is sent as the third message. The idea is the same.

### *Preshared-Key Method*

Figure 18.24 shows the preshared-key method in the aggressive mode. Note that after receiving the first message, the responder can calculate SKEYID and consequently, HASH-R. But the initiator cannot calculate SKEYID until it receives the second message. HASH-I in the third message can be encrypted.

**Figure 18.24** *Aggressive mode, preshared-key method*



### *Original Public-Key Method*

Figure 18.25 shows the exchange of messages using the original public-key method in the aggressive mode. Note that the responder can calculate the SKEYID and HASH-R after receiving the first message, but the initiator must wait until it receives the second message.

### *Revised Public-Key Method*

Figure 18.26 shows the revised public-key method in the aggressive mode. The idea is the same as for the main mode, except that some messages are combined.

### *Digital Signature Method*

Figure 18.27 shows the digital signature method in the aggressive mode. The idea is the same as for the main mode, except that some messages are combined.

**Figure 18.25** *Aggressive mode, original public-key method*

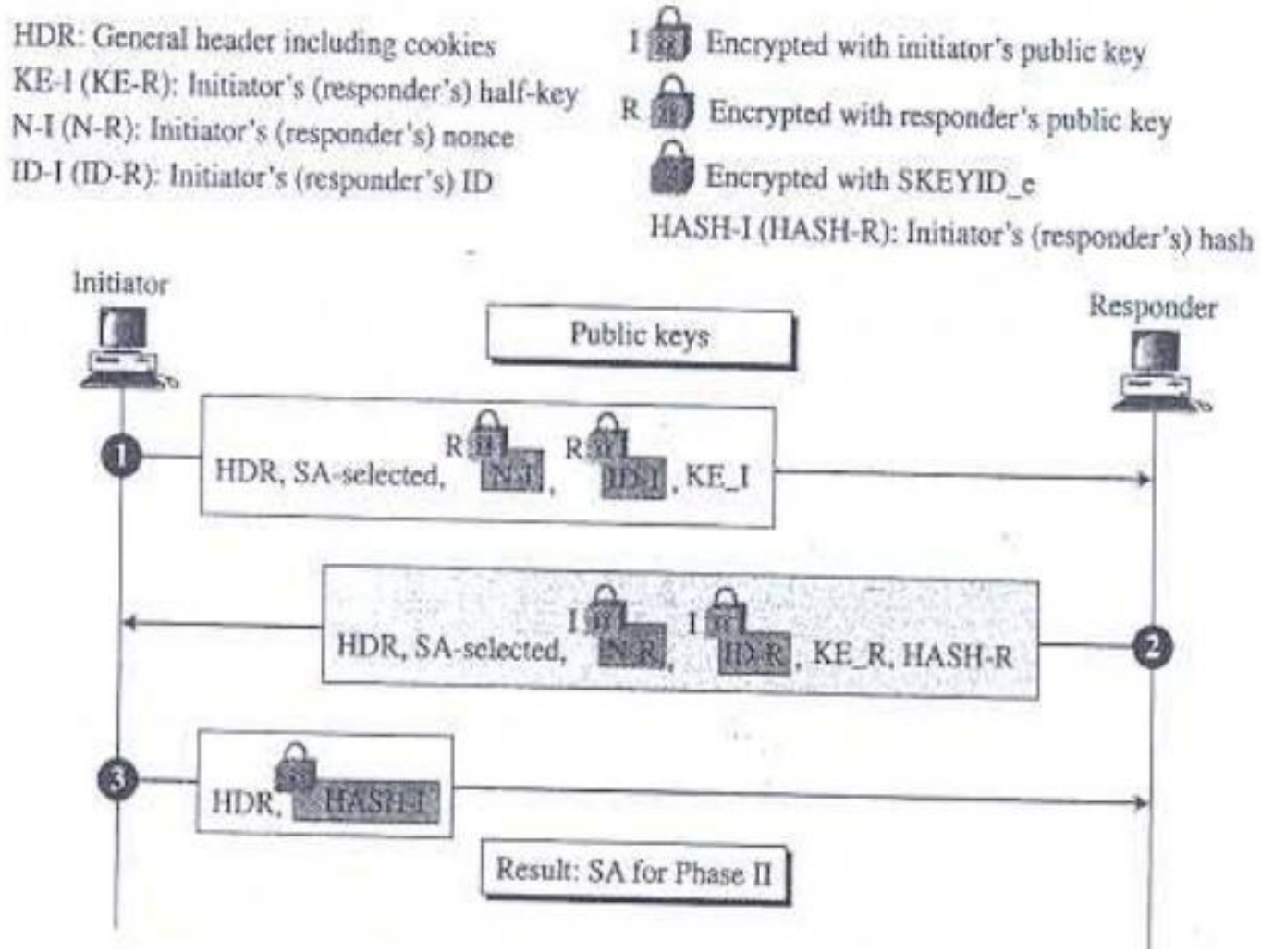


**Figure 18.26** *Aggressive mode, revised public-key method*

HDR: General header including cookies
KE-I (KE-R): Initiator's (responder's) half-key
Cert-I (Cert-R): Initiator's (responder's) certificate
N-I (N-R): Initiator's (responder's) nonce
ID-I (ID-R): Initiator's (responder's) ID
HASH-I (HASH-R): Initiator's (responder's) hash

I Encrypted with initiator's public key
R Encrypted with responder's public key
R Encrypted with responder's secret key
I Encrypted with initiator's secret key
Encrypted with SKEYID_e

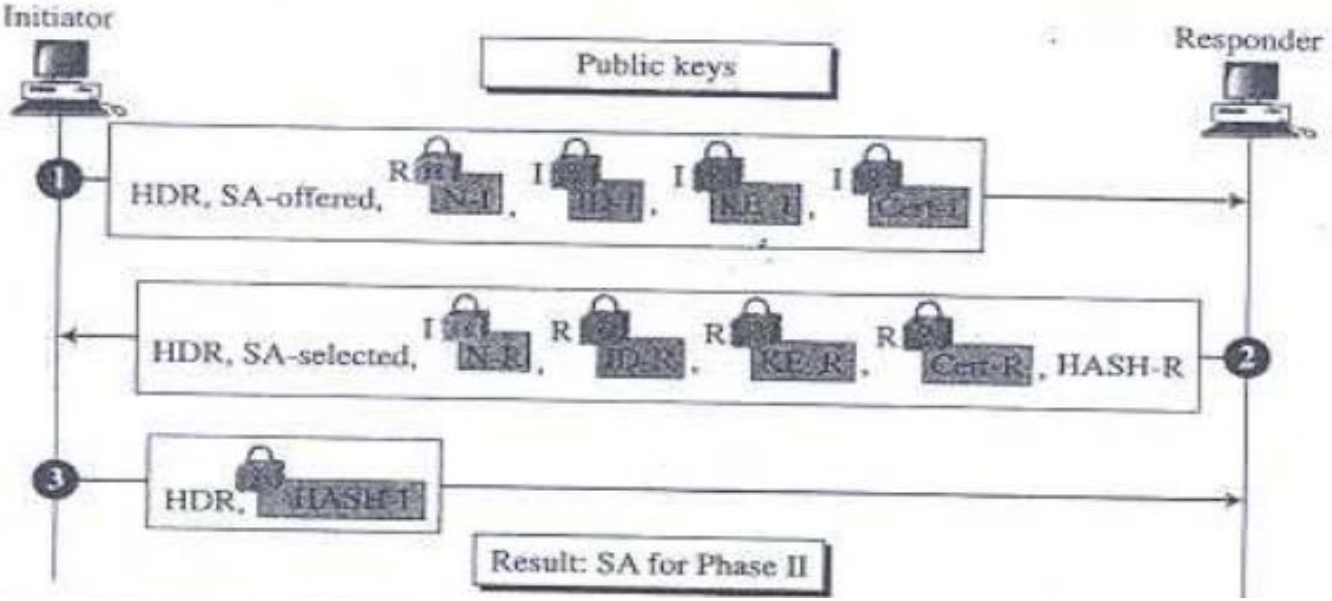
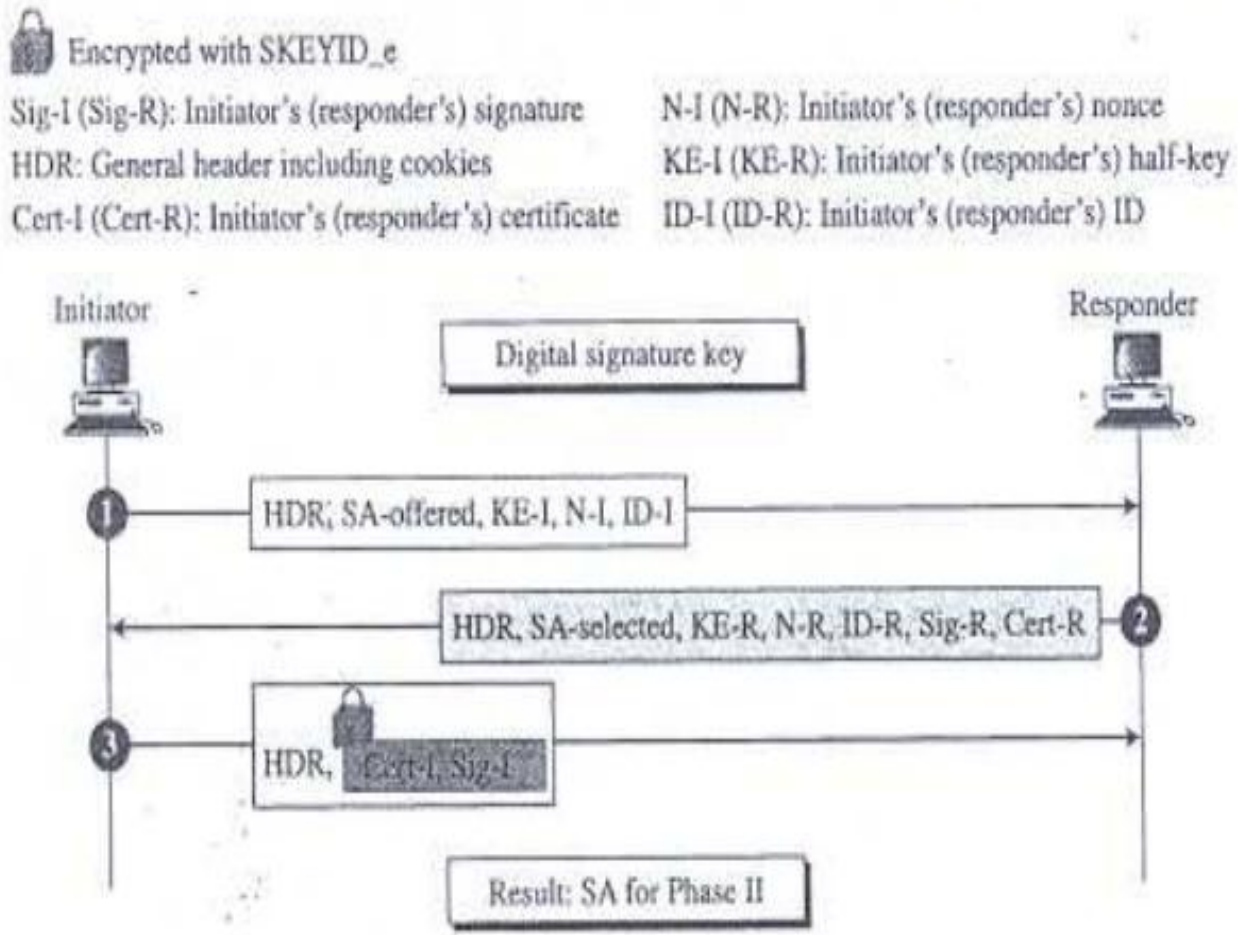


**Figure 18.27** *Aggressive mode, digital signature method*



The first two messages are the same as in the previous method. In the third message, the initiator sends its half-key, the nonce, and the ID. In the fourth message, the responder does likewise. However, the nonces and IDs are encrypted by the public key of the receiver and decrypted by the private key of the receiver. As can be seen from Figure 18.21, the nonces and IDs are encrypted separately, because, as we will see later, they are encoded separately from separate payloads.

One difference between this method and the previous one is that the IDs are exchanged with the third and fourth messages instead of the fifth and sixth messages. The fifth and sixth messages just carry the HASHs.

The calculation of SKEYID in this method is based on a hash of the nonces and the symmetric key. The hash of the nonces is used as the key for the keyed-HMAC function. Note that here we use a double hash. Although SKEYID, and consequently, the hashes are not directly dependent on the secret that each party possesses, they are related indirectly. SKEYID depends on the nonces and the nonces can only be decrypted by the private key (secret) of the receiver. So if the calculated hashes match those received, it is proof that each party is who it claims to be.

### *Original Public-Key Method*

Figure 18.25 shows the exchange of messages using the original public-key method in the aggressive mode. Note that the responder can calculate the SKEYID and HASH-R after receiving the first message, but the initiator must wait until it receives the second message.

### *Revised Public-Key Method*

Figure 18.26 shows the revised public-key method in the aggressive mode. The idea is the same as for the main mode, except that some messages are combined.

### *Digital Signature Method*

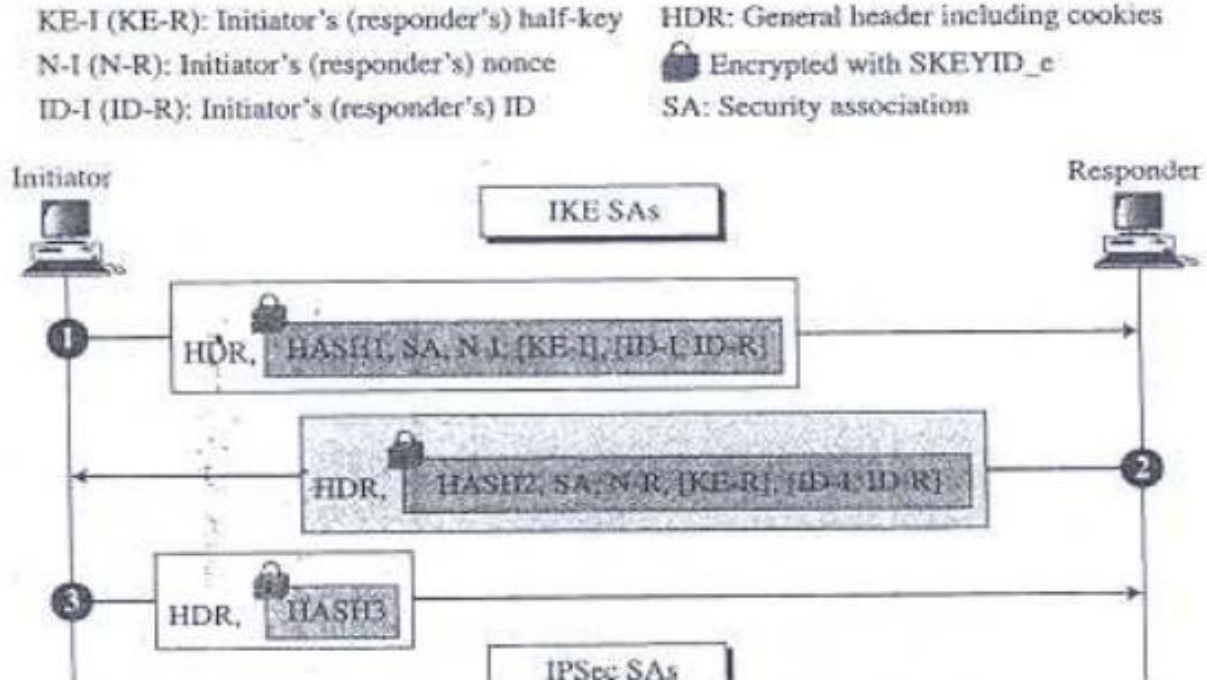
Figure 18.27 shows the digital signature method in the aggressive mode. The idea is the same as for the main mode, except that some messages are combined.

# Phase-II: Quick Mode

After SAs have been created in either the main mode or the aggressive mode, phase II can be started. There is only one mode defined for phase II so far, the *quick mode*. This mode is under the supervision of the IKE SAs created by phase I. However, each quick-mode method can follow any main or aggressive mode.

The quick mode uses IKE SAs to create IPSec SAs (or SAs for any other protocol). Figure 18.28 shows the messages exchanged during the quick mode.

**Figure 18.28** *Quick mode*



In phase II, either party can be the initiator. That is, the initiator of phase II can be the initiator of phase I or the responder of phase I.

The initiator sends the first message, which includes the keyed-HMAC HASH1 (explained later), the entire SA created in phase I, a new nonce (N-I), an optional new Diffie-Hellman half-key (KE-I), and the optional IDs of both parties. The second message is similar, but carries the keyed-HMAC HASH2, the responder nonce (N-R), and, if present, the Diffie-Hellman half-key created by the responder. The third message contains only the keyed-HMAC HASH3.

The messages are authenticated using three keyed-HMACs: HASH1, HASH2, and HASH3. These are calculated as follows:

$$\text{HASH1} = prf(\text{SKEYID\_d}, \text{MsgID} \mid \text{SA} \mid \text{N-I})$$

$$\text{HASH2} = prf(\text{SKEYID\_d}, \text{MsgID} \mid \text{SA} \mid \text{N-R})$$

$$\text{HASH3} = prf(\text{SKEYID\_d}, 0 \mid \text{MsgID} \mid \text{SA} \mid \text{N-I} \mid \text{N-R})$$

Each HMAC includes the message ID (MsgID) used in the header of ISAKMP headers. This allows multiplexing in phase II. The inclusion of MsgID prevents simultaneous creations of phase II from bumping into each other.

All three messages are encrypted for confidentiality using the SKEYID_e created during phase I.

### *Perfect Forward Security (PFS)*

After establishing an IKE SA and calculating SKEYID_d in phase I, all keys for the quick mode are derived from SKEYID_d. Since multiple phase IIs can be derived from a single phase I, phase II security is at risk if the intruder has access to SKEYID_d. To prevent this from happening, IKE allows **Perfect Forward Security (PFS)** as an option. In this option, an additional Diffie-Hellman half-key is exchanged and the resulting shared key ($g^{ir}$) is used in the calculation of key material (see the next section) for IPSec. PFS is effective if the Diffie-Hellman key is immediately deleted after the calculation of the key material for each quick mode.

### *Key Materials*

After the exchanges in phase II, an SA for IPSec is created including the key material, K, that can be used in IPSec. The value is derived as:

$$K = prf(SKEYID\_d, protocol \mid SPI \mid N\text{-}I \mid N\text{-}R) \quad \text{(without PFS)}$$
$$K = prf(SKEYID\_d, g^{ir} \mid protocol \mid SPI \mid N\text{-}I \mid N\text{-}R) \quad \text{(with PFS)}$$

If the length of K is too short for the particular cipher selected, a sequence of keys is created, each key is derived from the previous one, and the keys are concatenated to

make a longer key. We show the case without PFS; we need to add $g^{ir}$ for the case with PFS.

The key material created is unidirectional; each party creates different key material because the SPI used in each direction is different.

$K_1 = prf$ (SKEYID_d, protocol | SPI | N-I | N-R)
$K_2 = prf$ (SKEYID_d, $K_1$ | protocol | SPI | N-I | N-R)
$K_3 = prf$ (SKEYID_d, $K_2$ | protocol | SPI | N-I | N-R)
...
$K = K_1 | K_2 | K_3 | ...$

---

**The key material created after phase II is unidirectional; there is one key for each direction.**

---