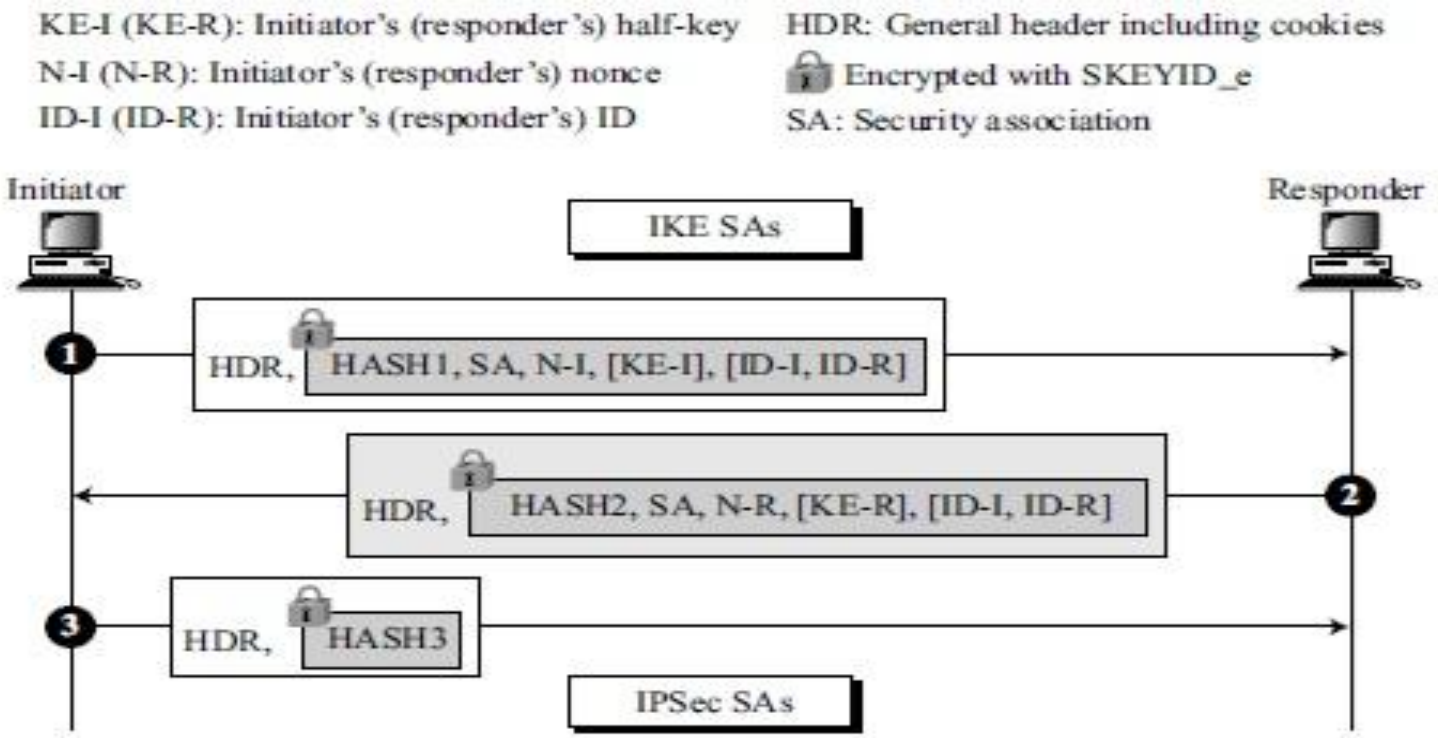
Security at the Network Layer: IPSec (Part-II)

Sachin Tripathi

IIT(ISM), Dhanbad

Phase-II Quick Mode

The quick mode uses IKE SAs to create IPSec SAs (or SAs for any other protocol).



- 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.

Note: The messages are authenticated using three keyed-HMACs: HASH1, HASH2, and HASH3.

HASH1 = *prf* (SKEYID_d, MsgID | SA | N-I)

HASH2 = *prf* (SKEYID_d, MsgID | SA | N-R)

HASH3 = *prf* (SKEYID_d, 0 | MsgID | SA | N-I | 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 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

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

The key material created is unidirectional; each party creates different key material because the SPI used in each direction is different. (Without PFS)

$K_1 = prf(SKEYID\_d, protocol \mid SPI \mid N\text{-}I \mid N\text{-}R)$
$K_2 = prf(SKEYID\_d, K_1 \mid protocol \mid SPI \mid N\text{-}I \mid N\text{-}R)$
$K_3 = prf(SKEYID\_d, K_2 \mid protocol \mid SPI \mid N\text{-}I \mid N\text{-}R)$
...
$K = K_1 \mid K_2 \mid K_3 \mid \ldots$

SA Algorithms

The algorithms that are negotiated during the first two IKE exchanges.

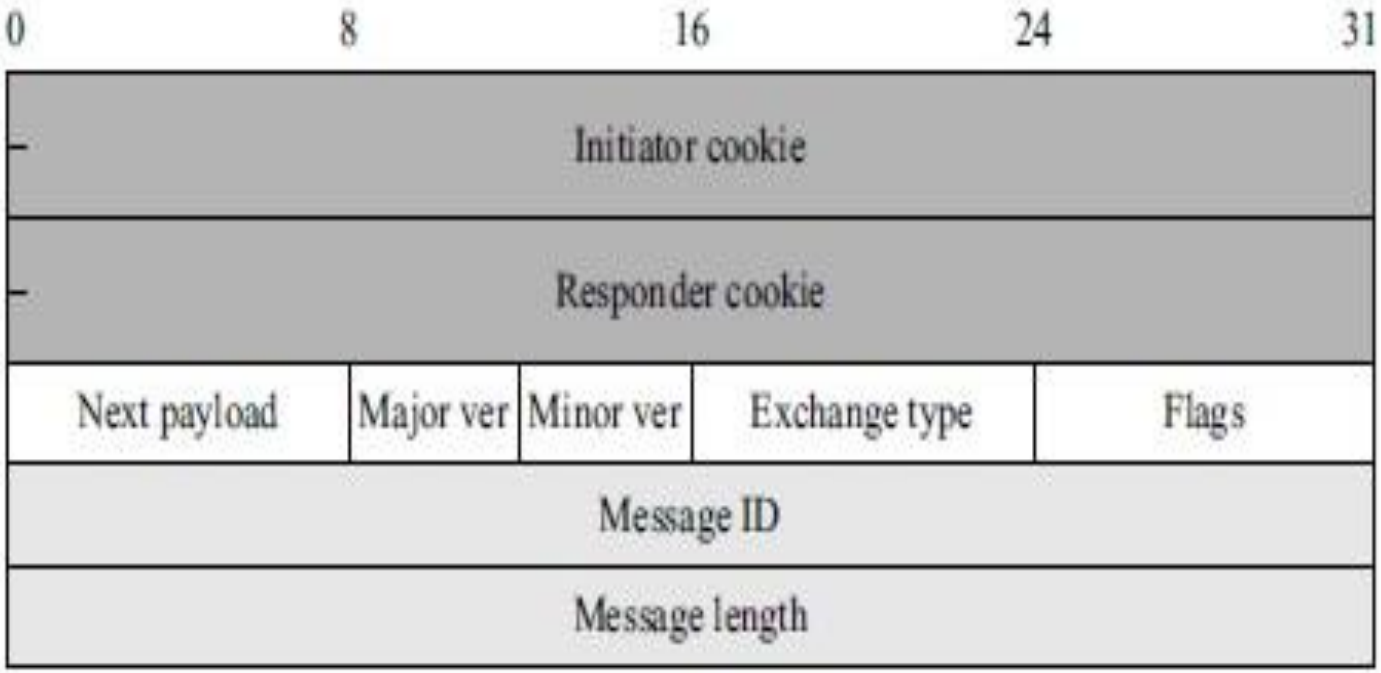
Value	*Description*
1	Modular exponentiation group with a 768-bit modulus
2	Modular exponentiation group with a 1024-bit modulus
3	Elliptic curve group with a 155-bit field size
4	Elliptic curve group with a 185-bit field size
5	Modular exponentiation group with a 1680-bit modulus

Value	*Description*
1	MD5
2	SHA
3	Tiger
4	SHA2-256
5	SHA2-384
6	SHA2-512

Value	*Description*
1	DES
2	IDEA
3	Blowfish
4	RC5
5	3DES
6	CAST
7	AES

ISAKMP

The ISAKMP protocol is designed to carry messages for the IKE exchange.



- Initiator cookie. This 32-bit field defines the cookie of the entity that initiates the SA establishment, SA notification, or SA deletion.
- Responder cookie. This 32-bit field defines the cookie of the responding entity. The value of this field is 0 when the initiator sends the first message.
- Next payload. This 8-bit field defines the type of payload that immediately follows the header(Different types of payload)
- Major version. This 4-bit version defines the major version of the protocol. Currently, the value of this field is 1.
- Minor version. This 4-bit version defines the minor version of the protocol. Currently, the value of this field is 0.
- Exchange type. This 8-bit field defines the type of exchange that is being carried by the ISAKMP packets.

- Flags. This is an 8-bit field in which each bit defines an option for the exchange. So far only the three least significant bits are defined.

  - The encryption bit, when set to 1, specifies that the rest of the payload will be encrypted using the encryption key and the algorithm defined by SA.
  - The commitment bit, when set to 1, specifies that encryption material is not received before the establishment of the SA.

  - The authentication bit, when set to 1, specifies that the rest of the payload, though not encrypted, is authenticated for integrity.

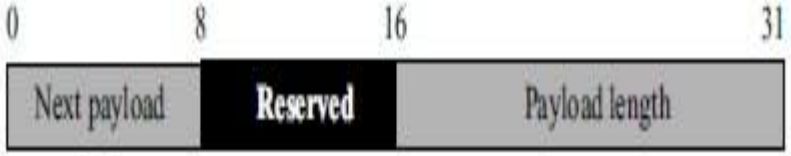
- Message ID. This 32-bit field is the unique message identity that defines the protocol state. This field is used only during the second phase of negotiation and is set to 0 during the first phase.

❑ Message length. Because different payloads can be added to each packet, the length of a message can be different for each packet. This 32-bit field defines the length of the total message, including the header and all payloads

Payloads

Types	*Name*	*Brief Description*
0	None	Used to show the end of the payloads
1	SA	Used for starting the negotiation
2	Proposal	Contains information used during SA negotiation
3	Transform	Defines a security transform to create a secure channel
4	Key Exchange	Carries data used for generating keys
5	Identification	Carries the identification of communication peers
6	Certification	Carries a public-key certificate
7	Certification Request	Used to request a certificate from the other party
8	Hash	Carries data generated by a hash function
9	Signature	Carries data generated by a signature function
10	Nonce	Carries randomly generated data as a nonce
11	Notification	Carries error message or status associated with an SA
12	Delete	Carries one more SA that the sender has deleted
13	Vendor	Defines vendor-specification extensions

Each payload has a generic header and some specific fields.



❑ Next payload. This 8-bit field identifies the type of the next payload. When there is no next payload, the value of this field is 0. Note that there is no type field for the current payload. The type of the current payload is determined by the previous payload or the general header (if the payload is the first one).

❑ Payload length. This 16-bit field defines the length of the total payload (including the generic header) in bytes.

SA Payload

The SA payload is used to negotiate security parameters

- However, these parameters are not included in the SA payload; they are included in two other payloads (proposal and transform)
- An SA payload is followed by one or more proposal payloads, and each proposal payload is followed by one or more transform payloads.
- The SA payload just defines the domain of interpretation field and the situation field.

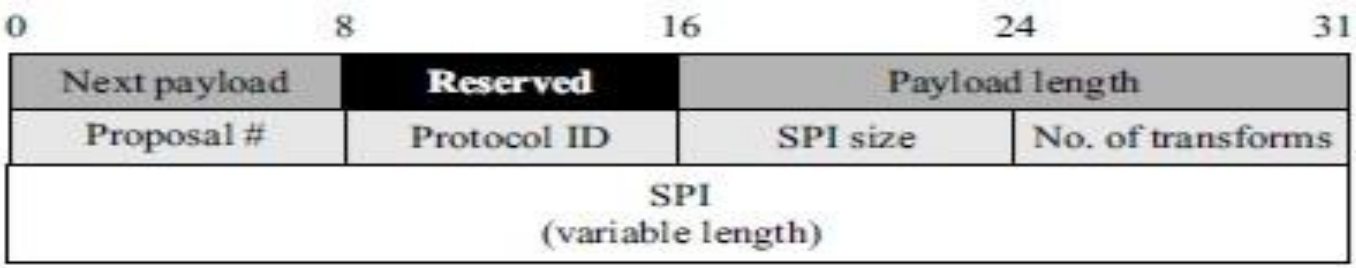

0
8
16
31
Next payload
Reserved
Payload length
DOI
Situation
(variable length)

The fields in the generic header have been discussed. The descriptions of the other fields follow:

- Domain of interpretation (DOI). This is a 32-bit field. For phase I, a value of 0 for this field defines a generic SA; a value of 1 defines IPSec.
- Situation. This is a variable-length field that defines the situation under which the negotiation takes place.

Proposal Payload

- The proposal payload initiates the mechanism of negotiation. Although by itself it does not propose any parameters, it does define the protocol identification and the SPI.
- The parameters for negotiation are sent in the transform payload that follows.
- Each proposal payload is followed by one or more transform payloads that give alternative sets of parameters.


0
8
16
24
31
Next payload
Reserved
Payload length
Proposal #
Protocol ID
SPI size
No. of transforms
SPI
(variable length)

The fields in the generic header have been discussed. The descriptions of the other fields follow:

- Proposal #. The initiator defines a number for the proposal so that the responder can refer to it. Note that an SA payload can include several proposal payloads. If all of the proposals belong to the same set of protocols, the proposal number must be the same for each protocol in the set. Otherwise, the proposals must have different numbers.

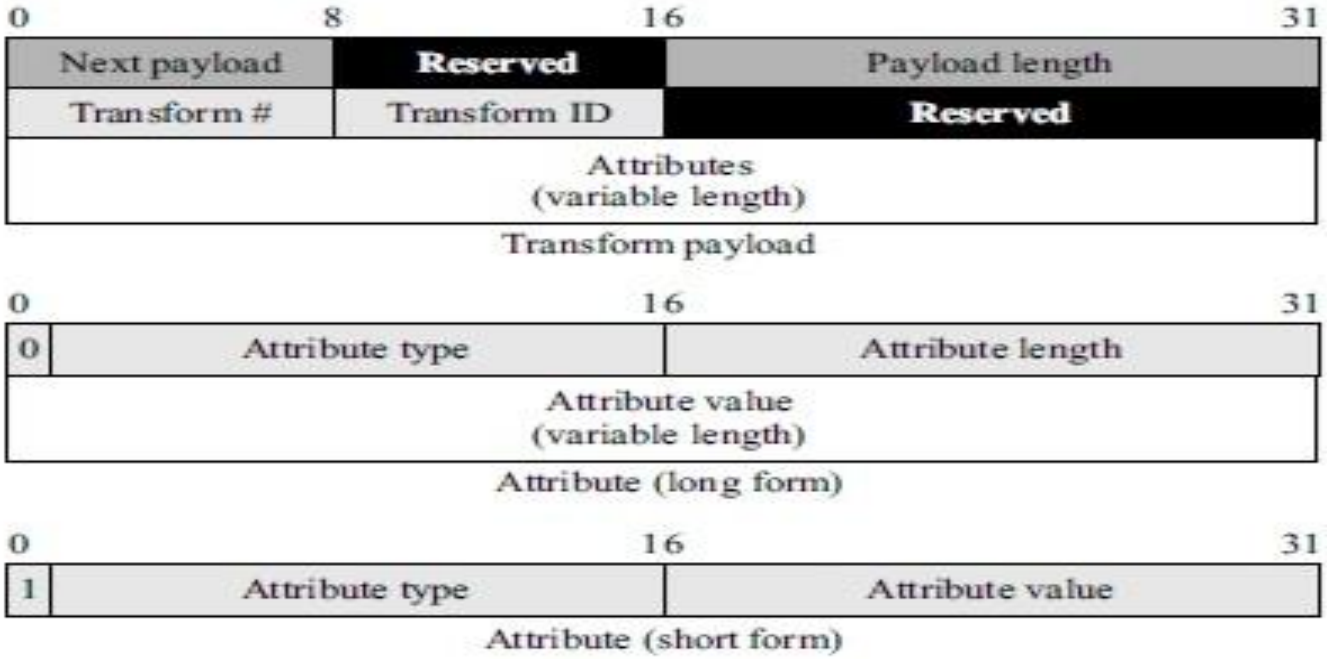
- Protocol ID. This 8-bit field defines the protocol for the negotiation. For example, IKE phase1 = 0, ESP = 1, AH = 2, etc.
- SPI size. This 8-bit field defines the size of the SPI in bytes.
- Number of Transforms. This 8-bit field defines the number of transform payloads that will follow this proposal payload.
- SPI. This variable-length field is the actual SPI. Note that if the SPI does not fill the 32-bit space, no padding is added.

Transform Payload

- The transform payload actually carries attributes of the SA negotiation

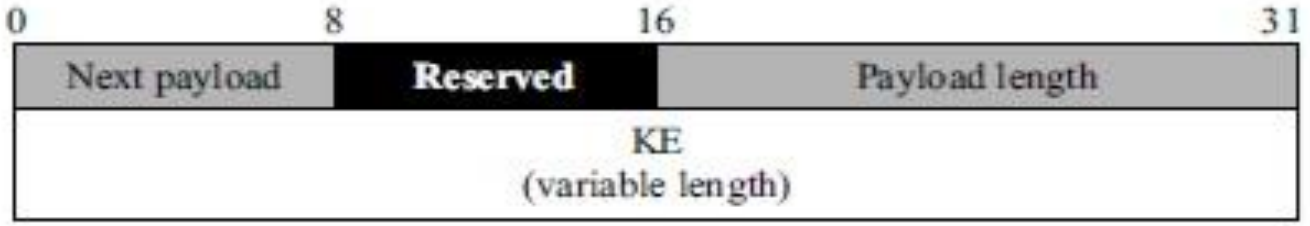
- The transform payload actually carries attributes of the SA negotiation. The fields in the generic header have been discussed. The descriptions of the other fields follow:

- Transform #. This 8-bit field defines the transform number. If there is more than one transform payload in a proposal payload, then each must have its own number.
- Transform ID. This 8-bit field defines the identity of the payload.
- Attributes. Each transform payload can carry several attributes. Each attribute itself can have three or two subfields . The attribute type subfield defines the type of attribute as defined in the DOI. The attribute length subfield, if present, defines the length of the attribute value. The attribute value field is two bytes in the short form or of variable-length in the long form.


0
8
16
31
Next payload
Reserved
Payload length
Transform #
Transform ID
Reserved
Attributes
(variable length)
Transform payload
0
16
31
0
Attribute type
Attribute length
Attribute value
(variable length)
Attribute (long form)
0
16
31
1
Attribute type
Attribute value
Attribute (short form)

Key-Exchange Payload

The key exchange payload is used in those exchanges that need to send preliminary keys that are used for creating session keys. For example, it can be used to send a Diffie-Hellman half-key

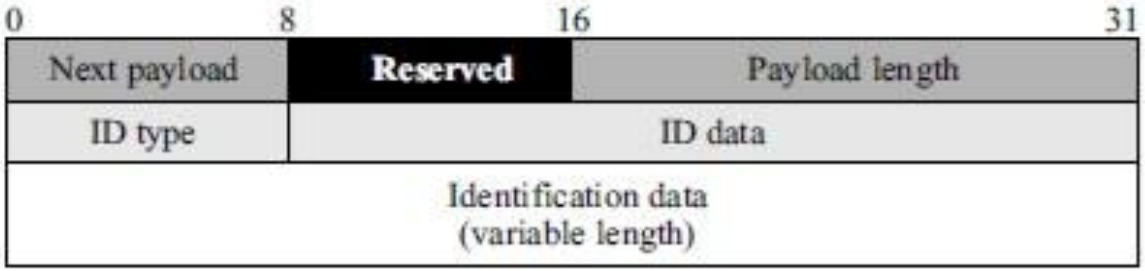


The fields in the generic header have been discussed. The description of the KE field follows:

❑ KE. This variable-length field carries the data needed for creating the session key.

Identification Payload

The identification payload allows entities to send their identifications to each other.

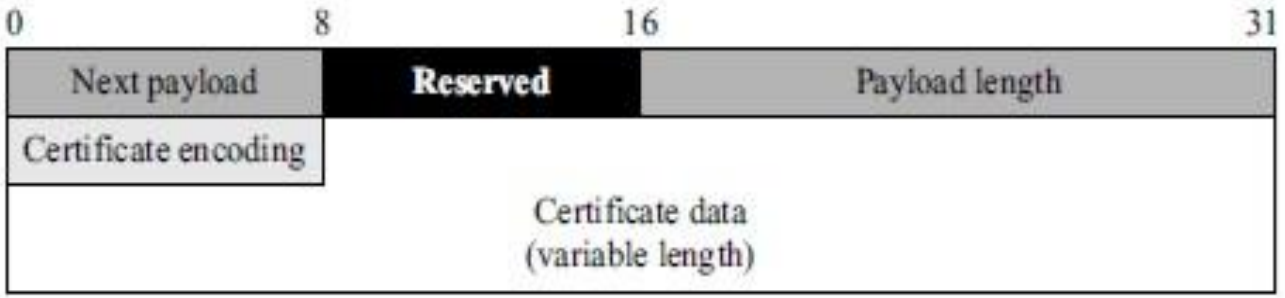


The fields in the generic header have been discussed. The descriptions of the other fields follow:

- ID type. This 8-bit field is DOI specific and defines the type of ID being used.
- ID data. This 24-bit field is usually set to 0.
- Identification data. The actual identity of each entity is carried in this variable length field.

Certification Payload

Anytime during the exchange, an entity can send its certification (for public-encryption/decryption keys or signature keys). Although the inclusion of the certification payload in an exchange is normally optional, it needs to be included if there is no secure directory available to distribute the certificates.



The fields in the generic header have been discussed. The descriptions of the other fields follow:

- ❑ Certificate encoding. This 8-bit field defines the encoding (type) of the certificate.
- ❑ Certificate data. This variable-length field carries the actual value of the certificate. Note that the previous field implicitly defines the size of this field.

Certification types

Value	*Type*
0	None
1	Wrapped X.509 Certificate
2	PGP Certificate
3	DNS Signed Key
4	X.509 Certificate —Signature
5	X.509 Certificate—Key Exchange
6	Kerberos Tokens
7	Certification Revocation List
8	Authority Revocation List
9	SPKI Certificate
10	X.509 Certificate—Attribute

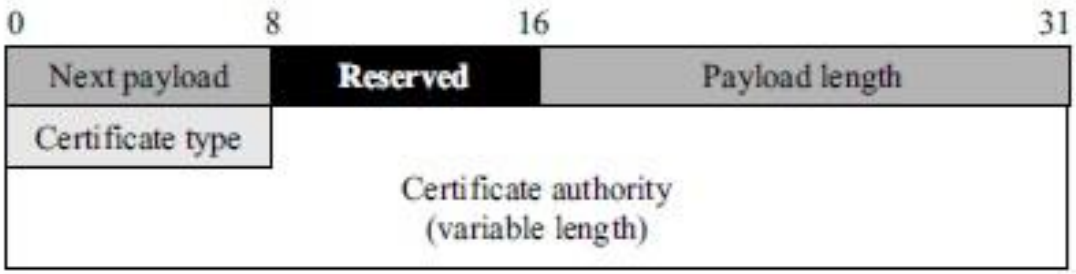
Certificate Request Payload

The fields in the generic header have been discussed. The descriptions of the other fields follow:

❑ Certificate type. This 8-bit field defines the type of certificate as previously defined in the certificate payload.

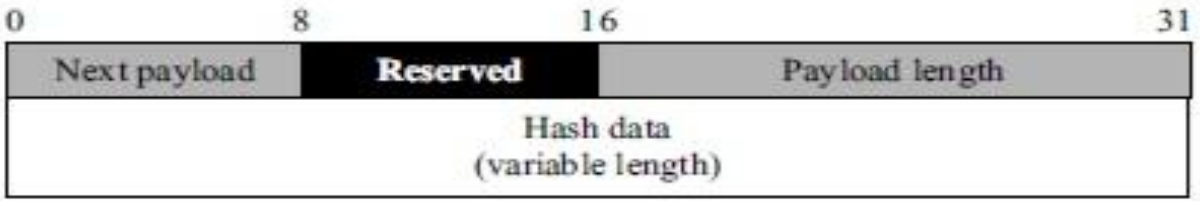
❑ Certificate authority. This is a variable-length field that defines the authority for the type of certificate issued.

Certification request payload



Hash Payload

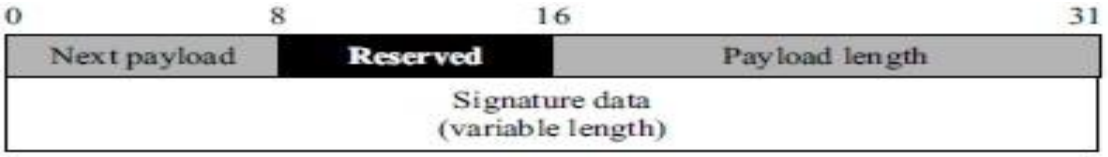
The hash payload contains data generated by the hash function as described in the IKE exchanges. The hash data guarantee the integrity of the message or part of the ISAKMP states.



- Hash data. This variable-length field carries the hash data generated by applying the hash function to the message or part of the ISAKMP states.

Signature Payload

- The signature payload contains data generated by applying the digital signature procedure over some part of the message or ISAKMP state.

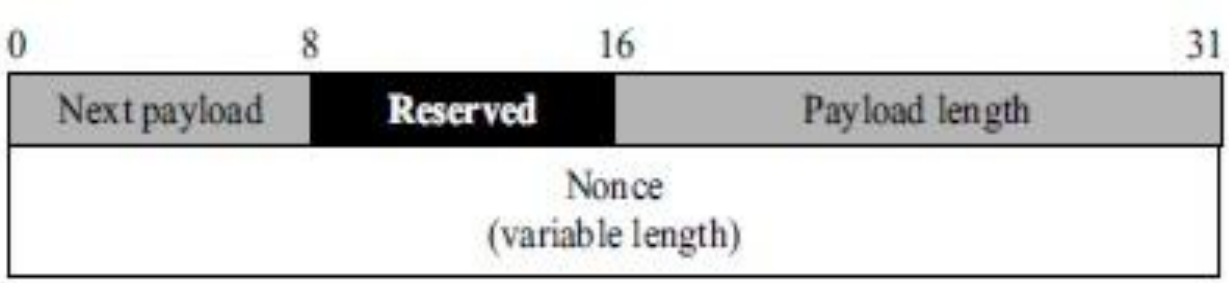


The fields in the generic header have been discussed. The description of the last field follows:

- Signature. This variable-length field carries the digest resulting from applying the signature over part of the message or ISAKMP state.

Nonce Payload

The nonce payload contains random data used as a nonce to assure liveliness of the message and to prevent a replay attack.

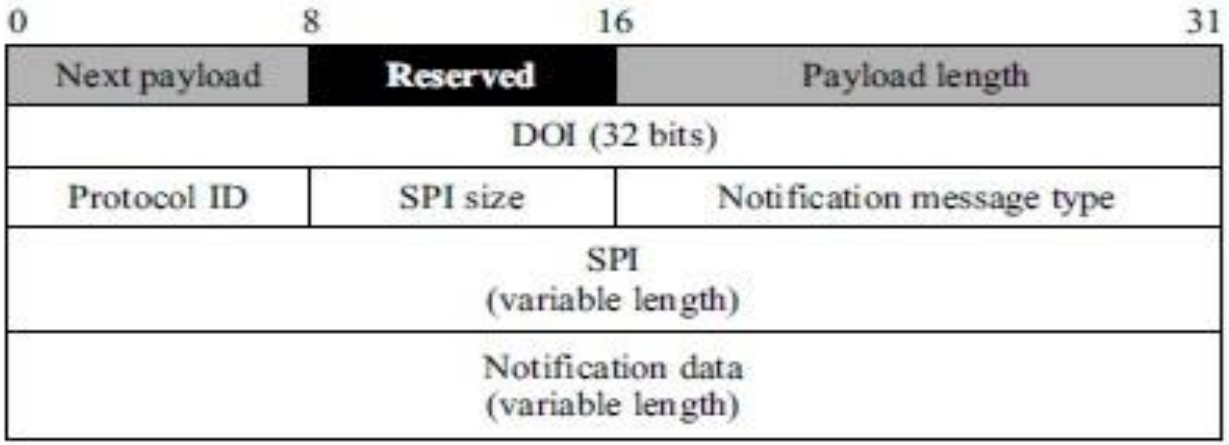


The fields in the generic header have been discussed. The description of the last field follows:

❑ Nonce. This is a variable-length field carrying the value of the nonce.

Notification Payload

- During the negotiation process, sometimes a party needs to inform the other party of the status or errors. The notification payload is designed for these two purposes



The fields in the generic header have been discussed. The descriptions of the other fields follow:

❏ DOI. This 32-bit field is the same as that defined for the Security Association payload.

❏ Protocol ID. This 8-bit field is the same as that defined for the proposal payload.

❏ SPI size. This 8-bit field is the same as that defined for the proposal payload.

❏ Notification message type. This 16-bit field specifies the status or the type of error that is to be reported. Table 18.8 gives a brief description of these types.

❏ SPI. This variable-length field is the same as that defined for the proposal payload.

❏ Notification data. This variable-length field can carry extra textual information about the status or errors.. The values 31 to 8191 are for future use and the values 8192 to 16383 are for private use

Notification types

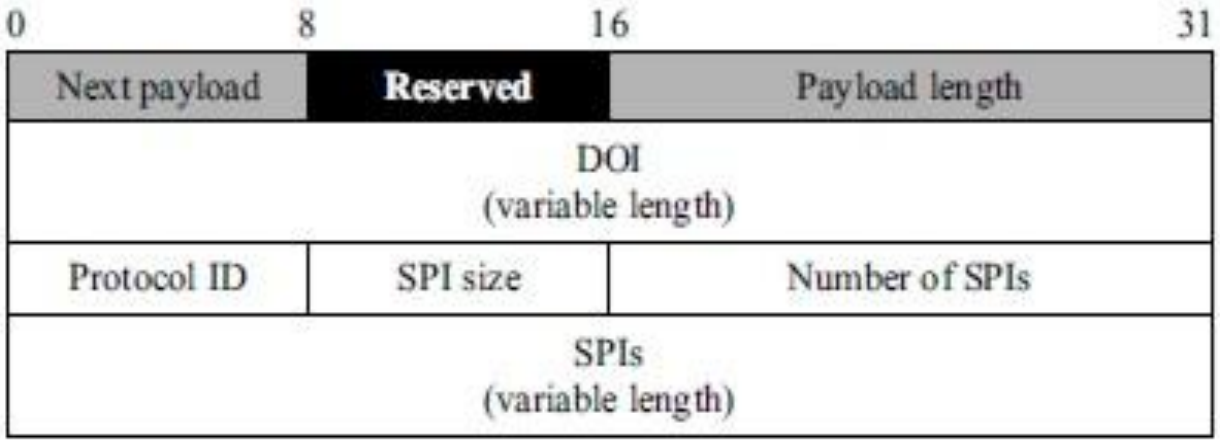
Value	*Description*	*Value*	*Description*
1	INVALID-PAYLOAD-TYPE	8	INVALID-FLAGS
2	DOI-NOT-SUPPORTED	9	INVALID-MESSAGE-ID
3	SITUATION-NOT-SUPPORTED	10	INVALID-PROTOCOL-ID
4	INVALID-COOKIE	11	INVALID-SPI
5	INVALID-MAJOR-VERSION	12	INVALID-TRANSFORM-ID
6	INVALID-MINOR-VERSION	13	ATTRIBUTE-NOT-SUPPORTED
7	INVALID-EXCHANGE-TYPE	14	NO-PROPOSAL-CHOSEN

Status Notification Values

Value	*Description*
16384	CONNECTED
24576-32767	DOI-specific codes

Delete Payload

The delete payload is used by an entity that has deleted one or more SAs and needs to inform the peer that these SAs are no longer supported.

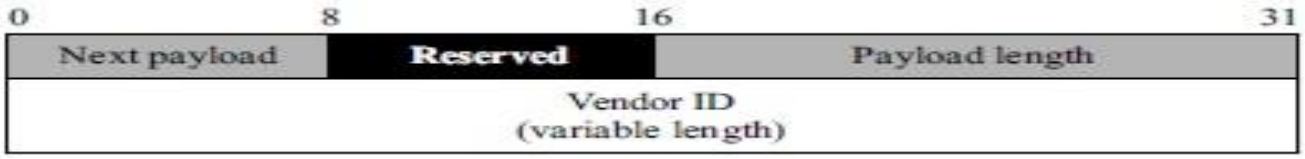


The fields in the generic header have been discussed. The descriptions of the other fields follow:

- ❑ DOI. This 32-bit field is the same as that defined for the Security Association payload.
- ❑ Protocol ID. This 8-bit field is the same as that defined for the proposal payload.
- ❑ SPI size. This 8-bit field is the same as that defined for the proposal payload.
- ❑ Number of SPIs. This 16-bit field defines the number of SPIs. One delete payload can report the deletion of several SAs.
- ❑ SPIs. This variable-length field defines the SPIs of the deleted SAs.

Vendor Payload

ISAKMP allows the exchange of information particular to a specific vendor



The fields in the generic header have been discussed. The description of the last field follows:

❑ Vendor ID. This variable-length field defines the constant used by the vendor.