Unit I
Summary

UNIT I

NETWORK LAYER SECURITY

&

TRANSPORT LAYER SECURITY

Network layer security:

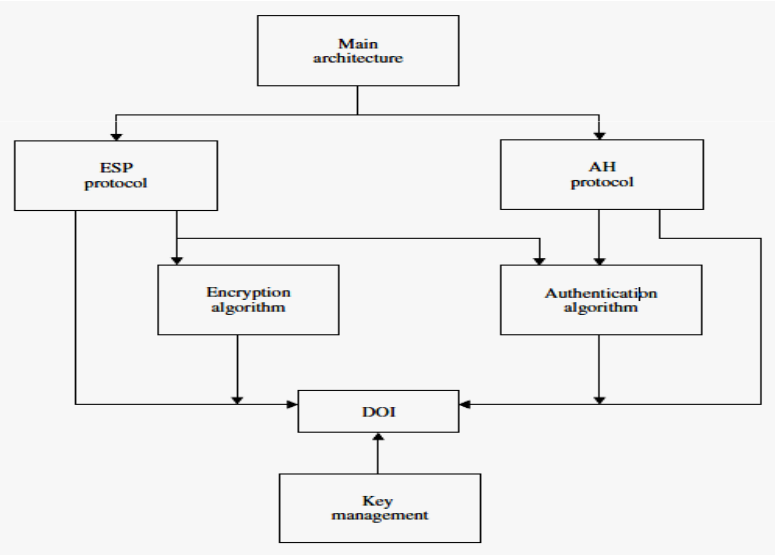
- IPSec Protocol
- IP Authentication Header
- IP ESP
- Key Management Protocol for IPSec

Transport layer Security:

- SSL protocol
- Cryptographic Computations
- TLS Protocol

IPSec Protocol Documents

- IP Security Document Roadmap - RFC 2411 by IETF - November 1998
- IPSec protocols is divided into seven groups
- Seven-group documents describes the set of IPSec protocols
  - *Architecture*
  - *ESP*
  - *AH*
  - *Encryption algorithm*
  - *Authentication algorithm*
  - *Key management*
  - *DOI :*Domain of Interpretation



Security Associations (SAs)

- An SA is uniquely identified by three parameters
  1. Security Parameters Index (SPI)
  2. IP Destination Address
  3. Security Protocol Identifier

- Two nominal databases
  - Security Policy Database (SPD)
  - Security Association Database (SAD)

    - Info in SPD indicates **"what"** to do with arriving datagram
      - specifies the policies that is to applied on all IP traffic (inbound or outbound, from host or security gateways)
    - Info in the SAD indicates **"how"** to do it

AH Format

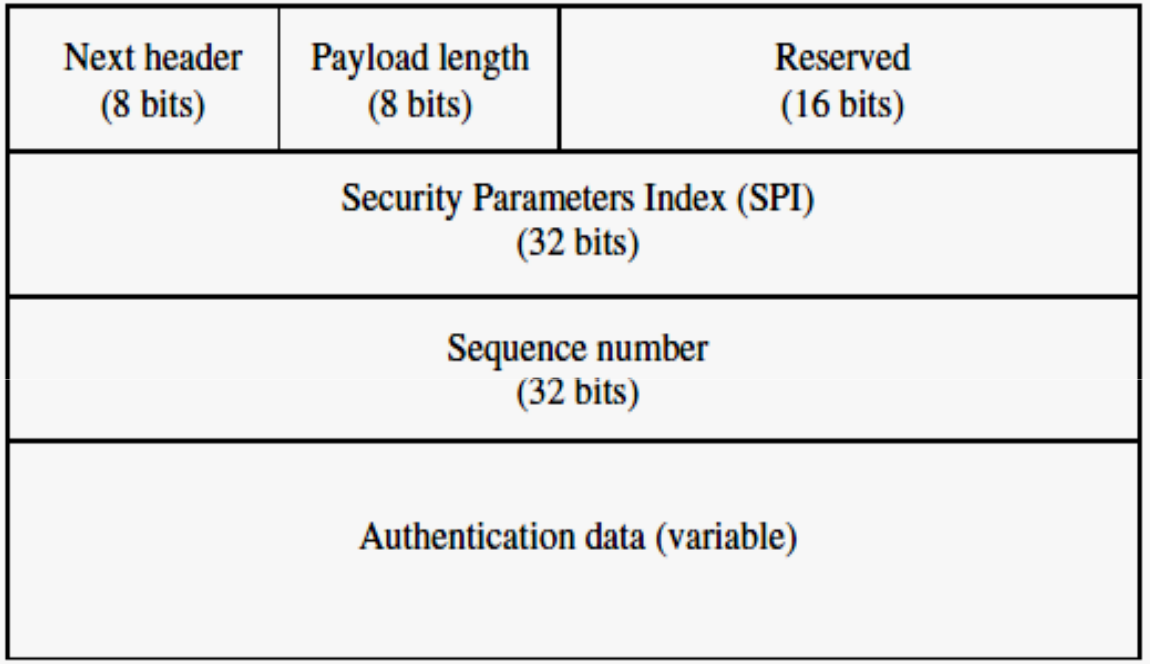
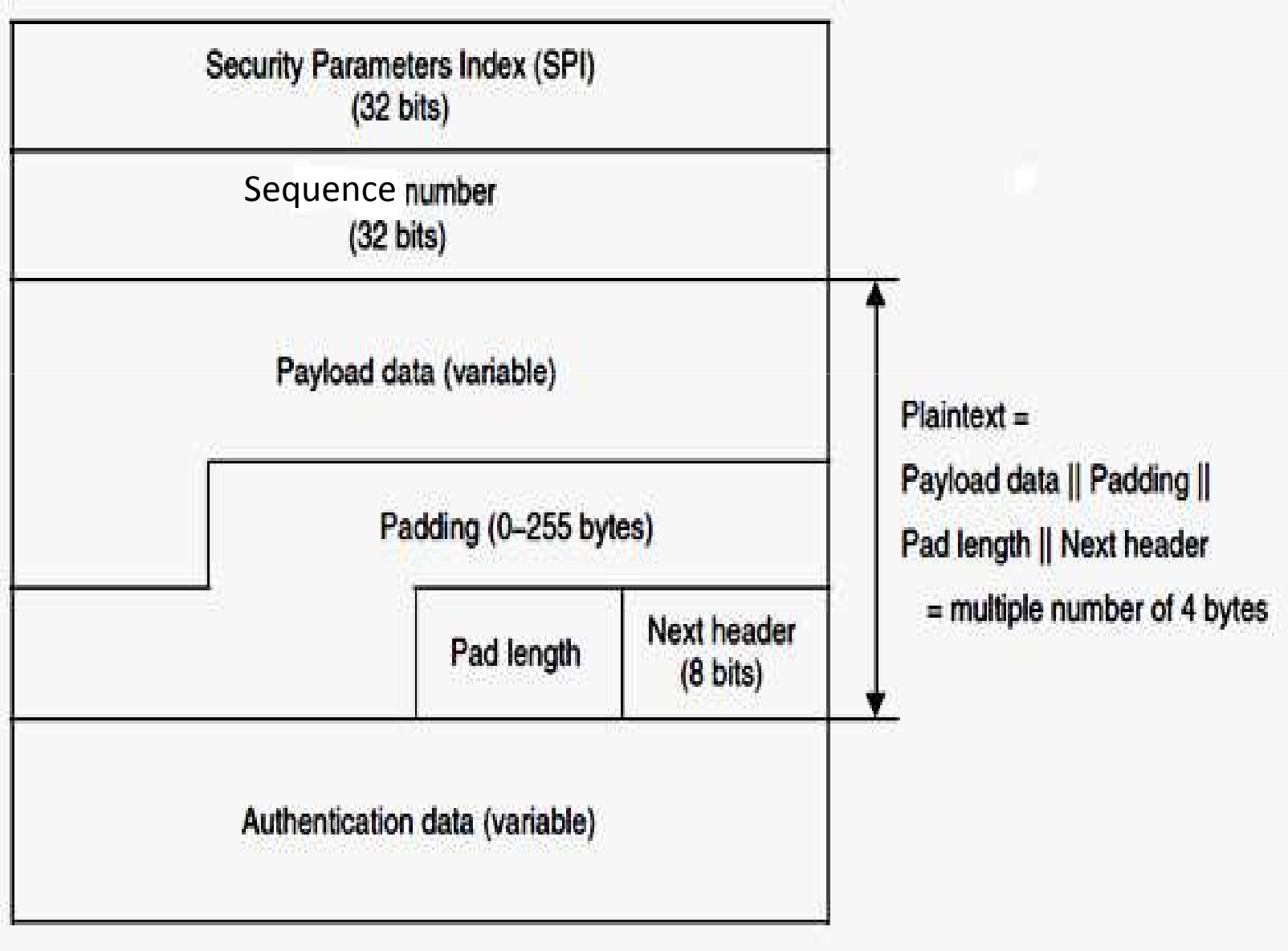


Figure 7.4 IPsec AH format.

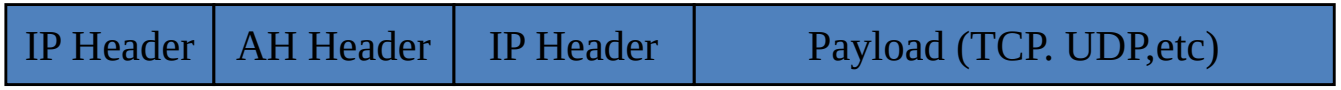
ESP Packet Format

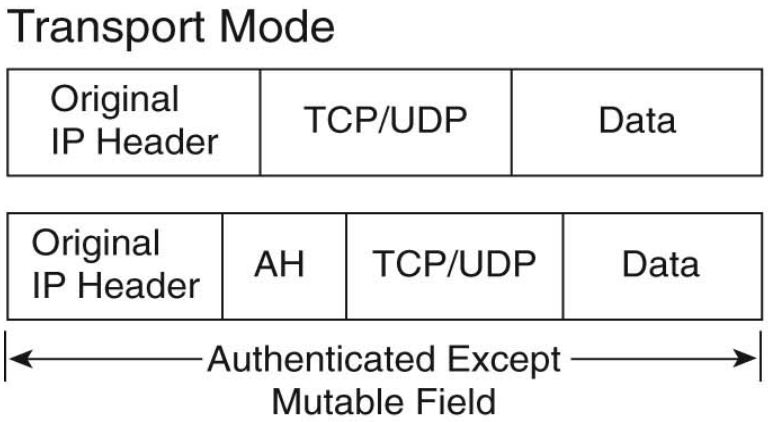


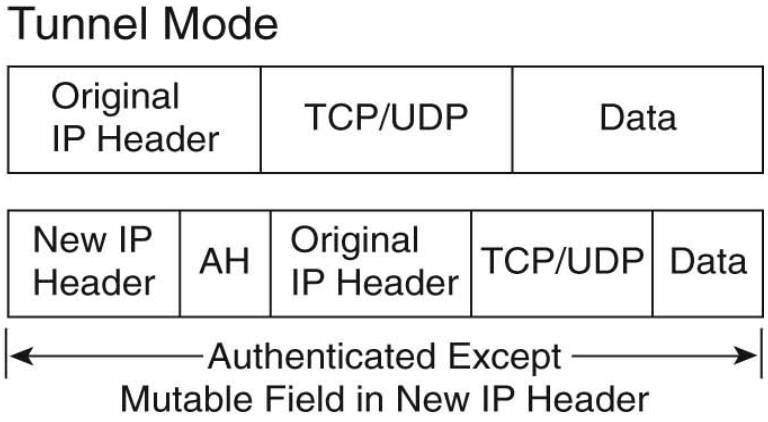
IPSec Modes of Operation-AH

Transport Packet layout

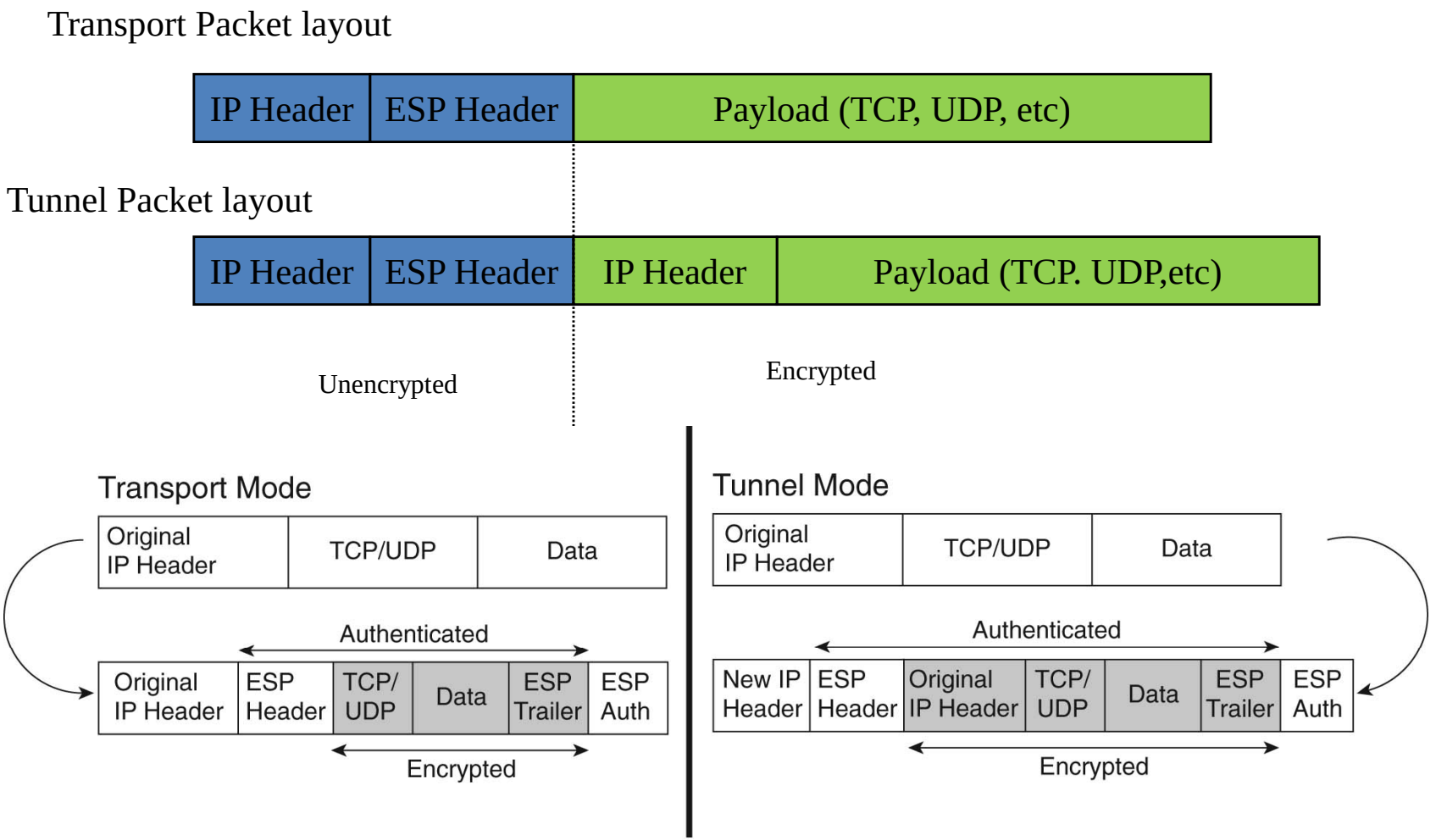
Tunnel Packet layout



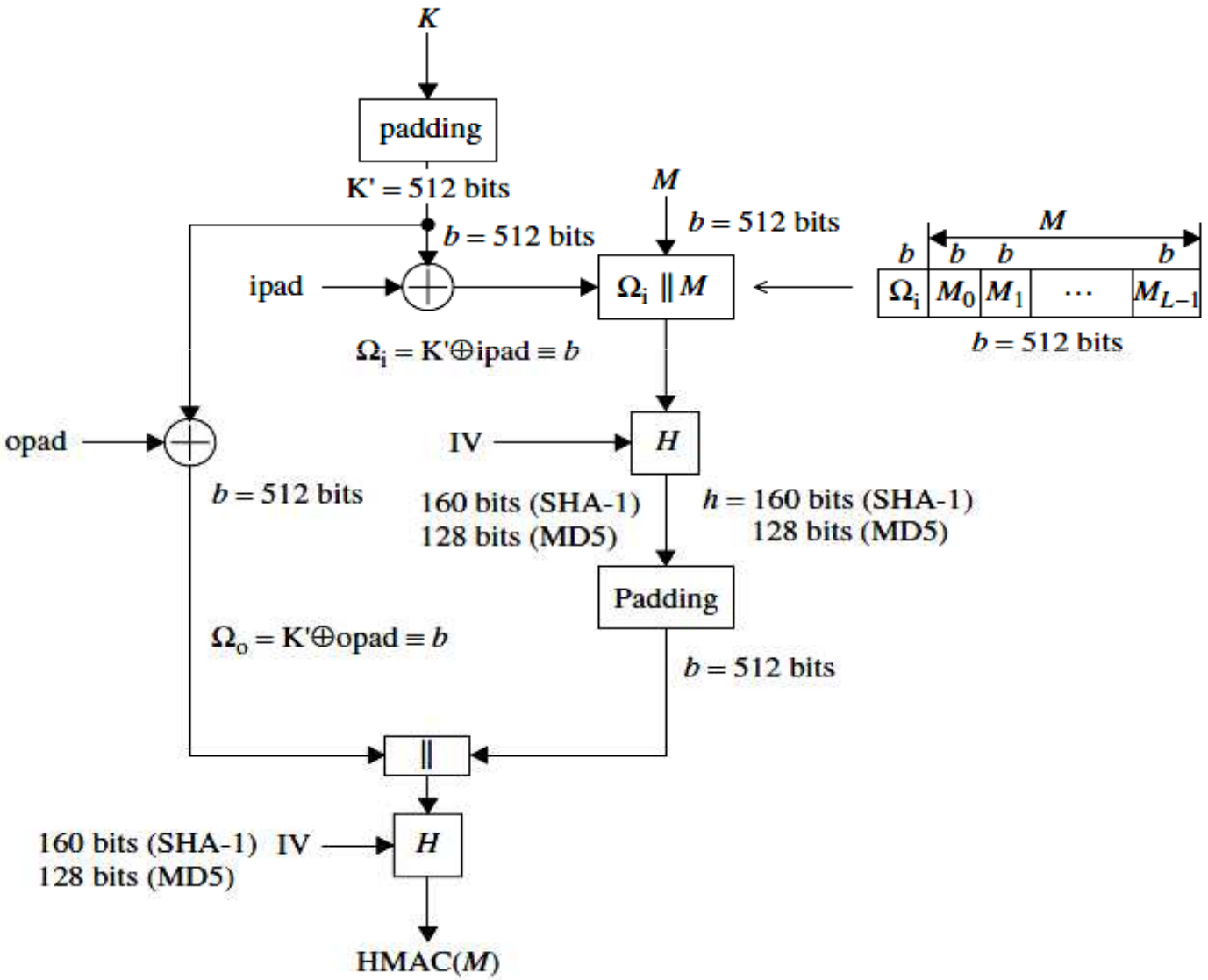




IPSec Modes of Operation - ESP



HMAC


K
padding
K' = 512 bits
b = 512 bits
M
b = 512 bits
ipad
Ωi ‖ M
Ωi = K'⊕ipad ≡ b
M
b b b b
Ωi M0 M1 ··· ML−1
b = 512 bits
opad
IV
H
b = 512 bits
160 bits (SHA-1)
128 bits (MD5)
h = 160 bits (SHA-1)
128 bits (MD5)
Padding
Ωo = K'⊕opad ≡ b
b = 512 bits
‖
160 bits (SHA-1)
128 bits (MD5)
IV
H
HMAC(M)

Encryption and Authentication Algorithms

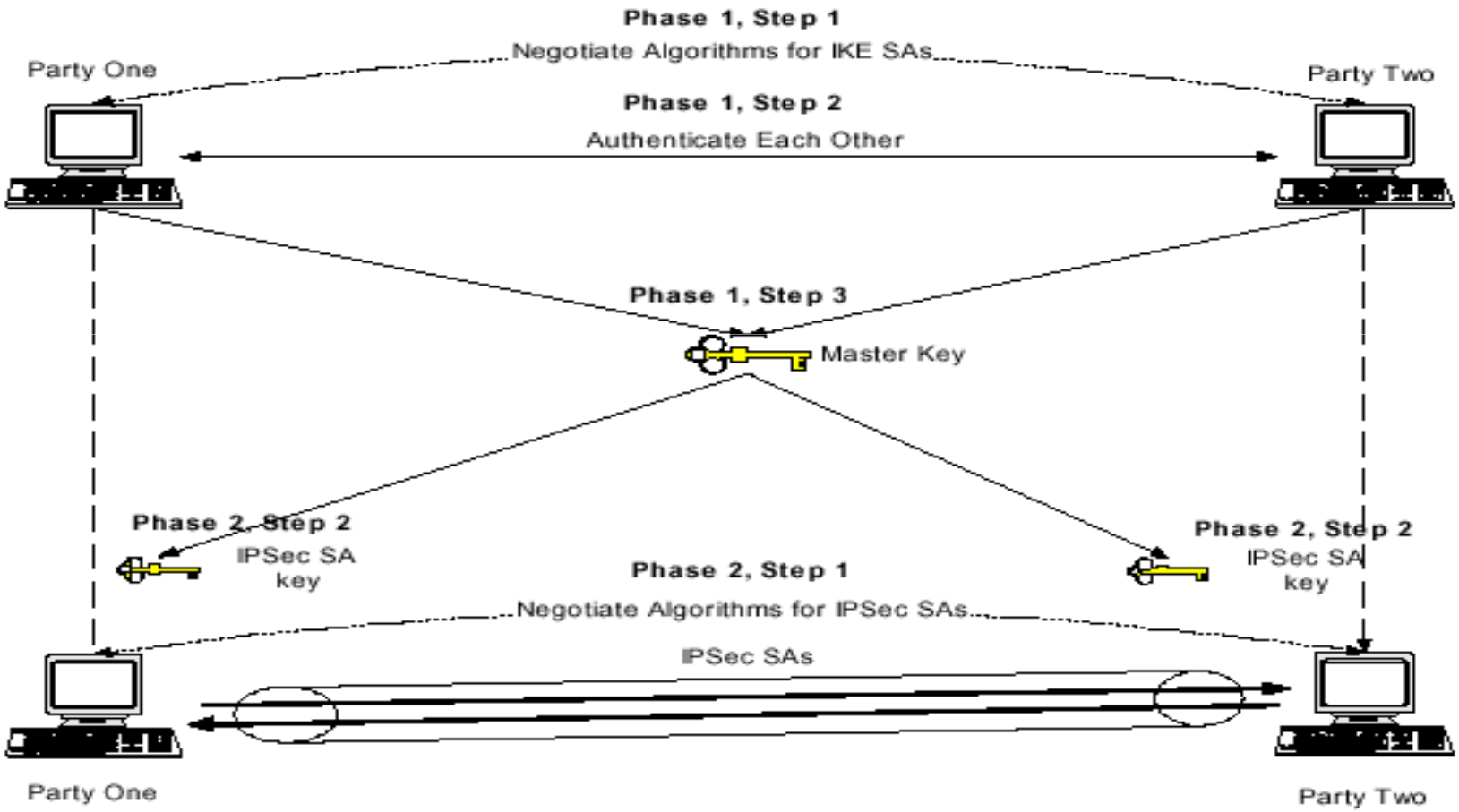
- The encryption authentication algorithm employed is specified by the SA

1. *Encryption*
2. *Decryption*
3. *Authentication*
4. *ICV*

Key management

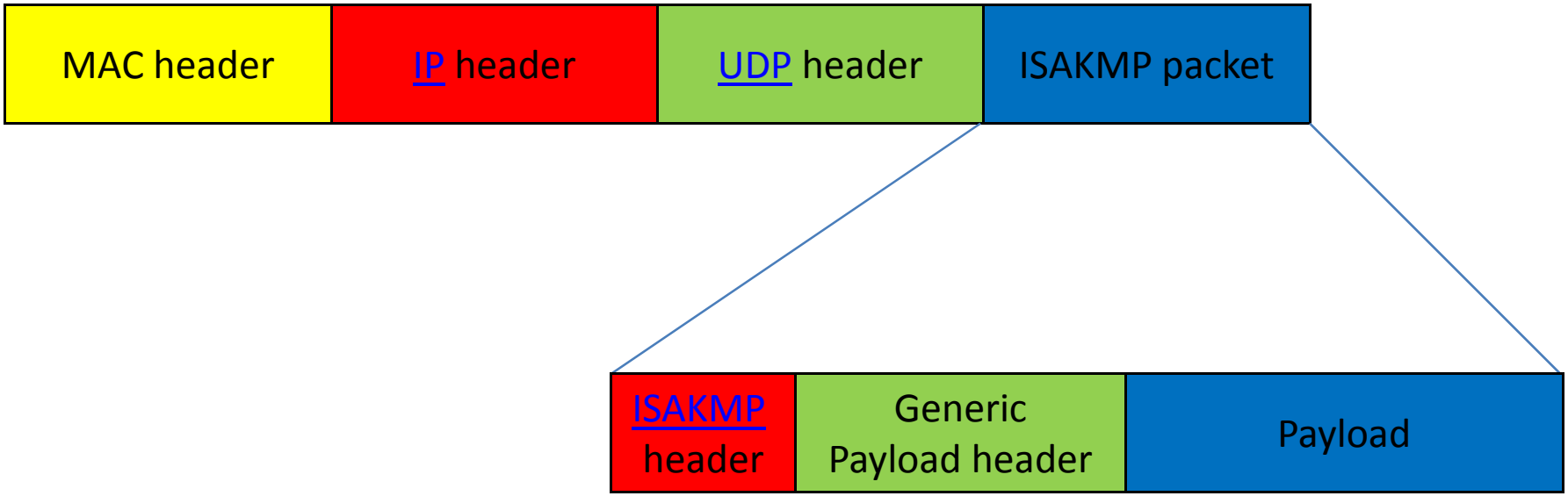
- Two types of key establishment
  - Manual
    - System administrator configures each system with the necessary keys
  - Automated
    - On-demand creation of keys for SA
- Default automated method is ISAKMP/Oakley
  - IKE = ISAKMP + OAKELY key exchange
  - Oakley key determination protocol
    - A key exchange protocol based on Diffie-Hellman
    - Provides added security (e.g., authentication)
  - ISAKMP – Internet Security Association and Key Management Protocol
    - Provides a framework for key exchange
    - Defines message formats that can carry the messages of various key exchange protocols

Key Management



(I) ISAKMP Payloads

- Provide modular building blocks for constructing ISAKMP messages
- The presence and ordering of payloads in ISAKMP is defined the Exchange Type Field in ISAKMP Header



(I) ISAKMP Payloads

- **ISAKMP Header**

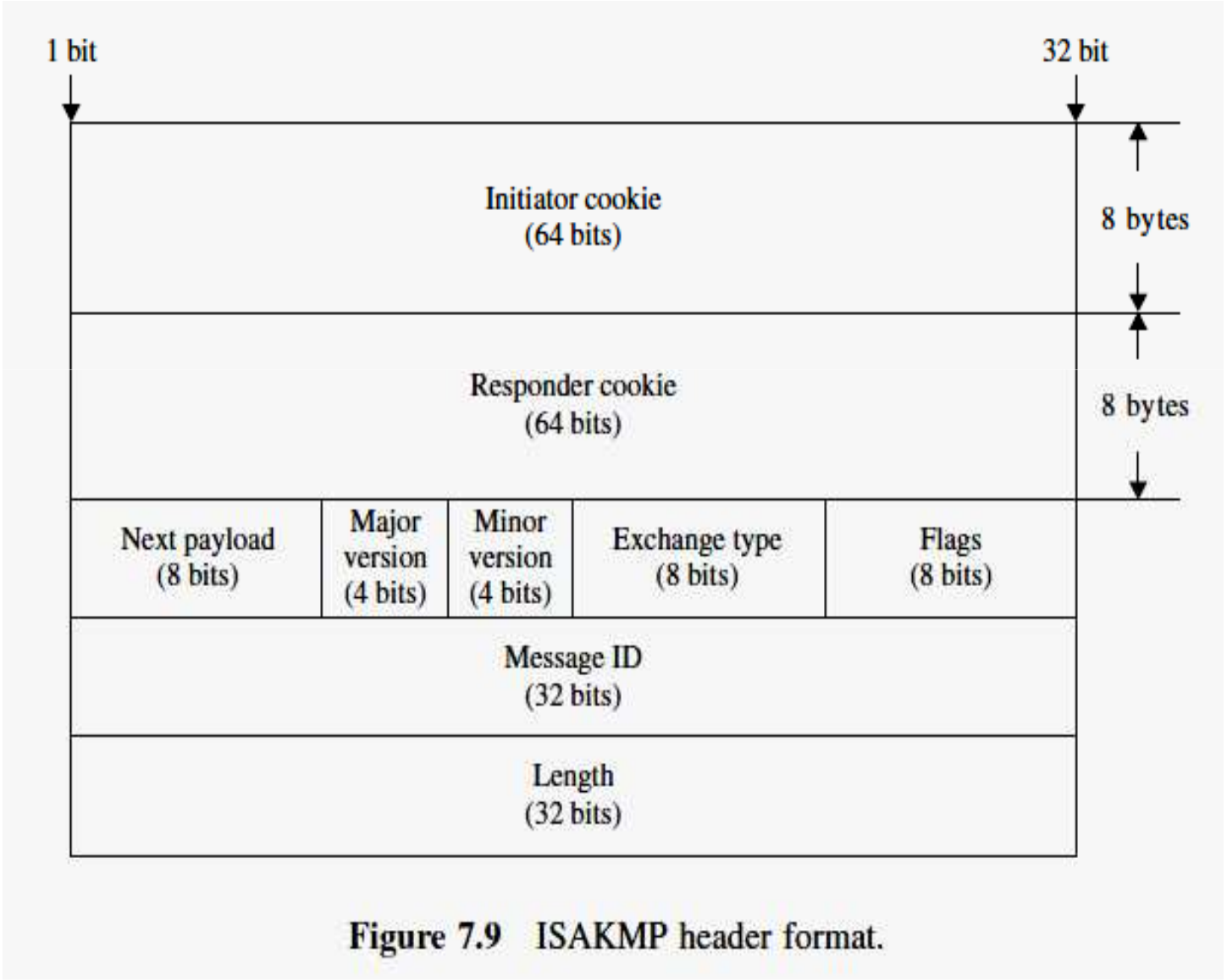
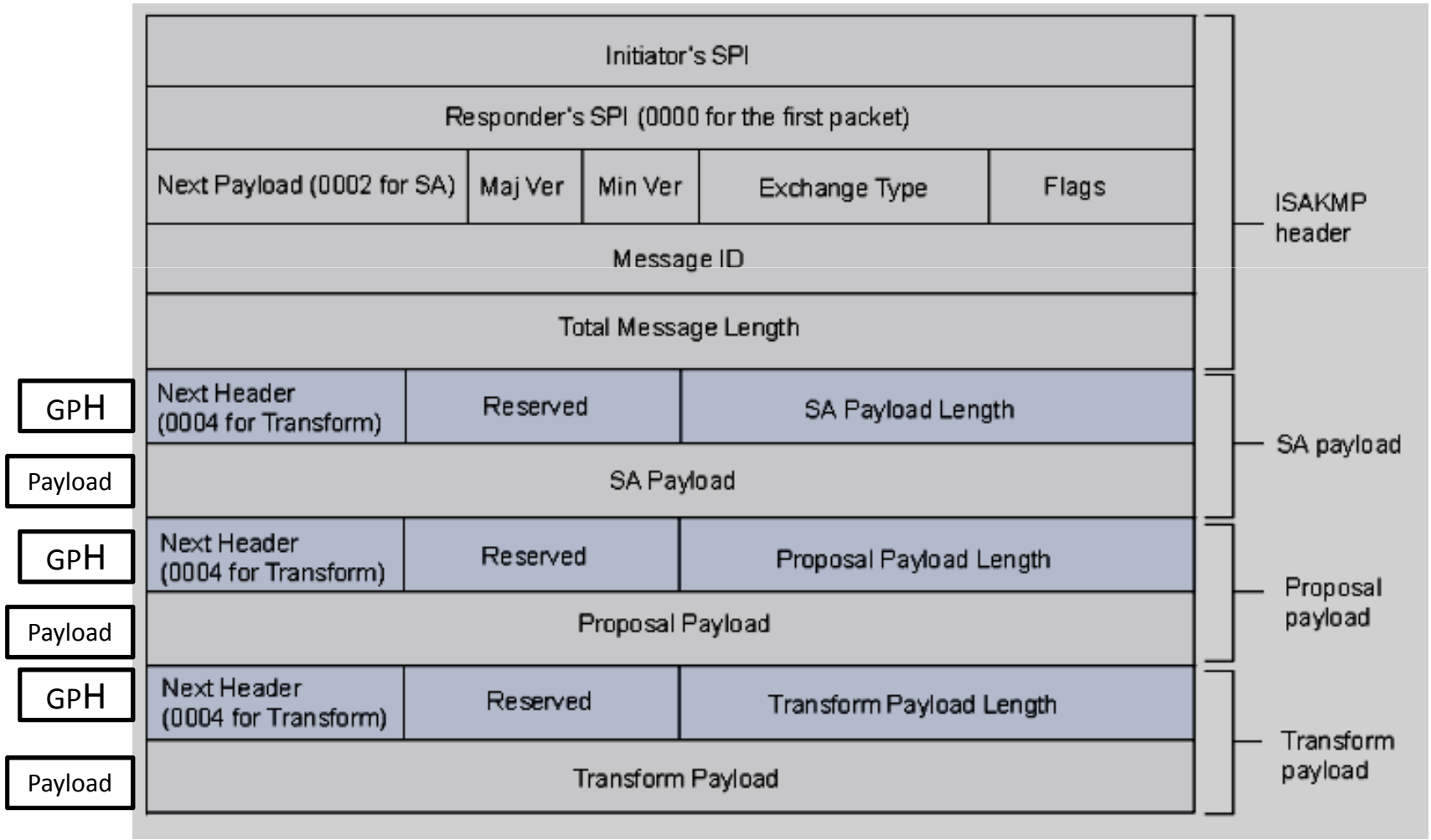


Figure 7.9 ISAKMP header format.

ISAKMP

SAKMP Header with Generic ISAKMP Payloads



ISAKMP

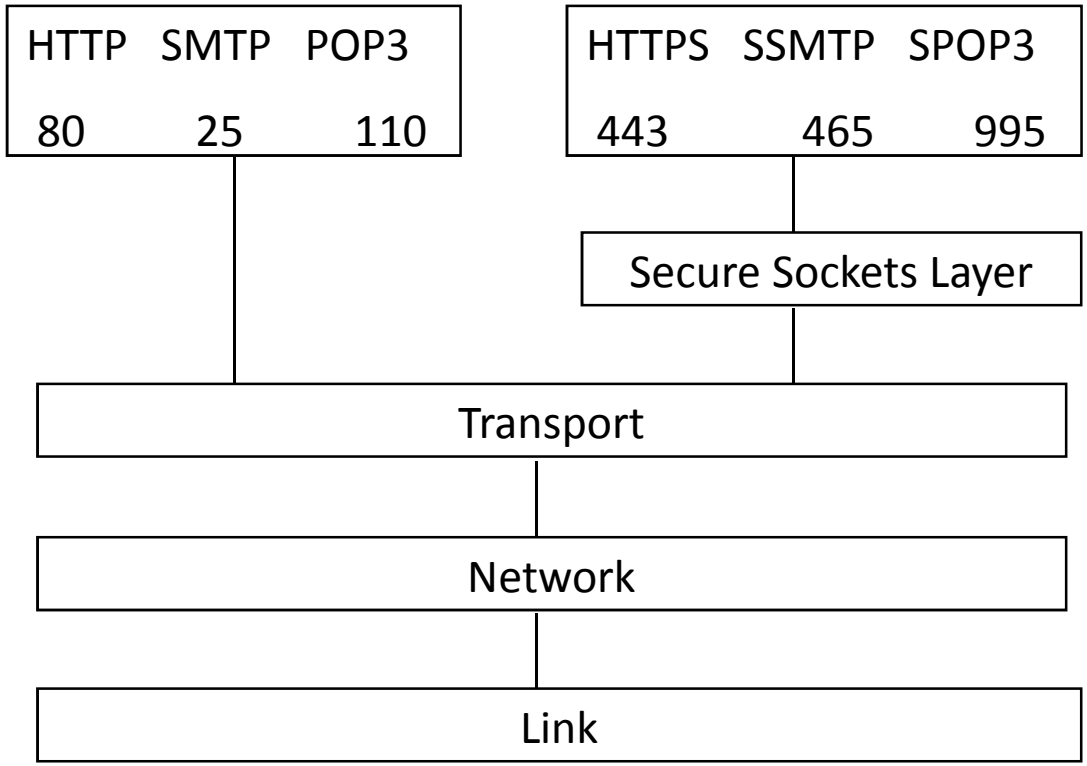
- **Payload Types for ISAKMP**
- Payloads are used to transfer information such as SA data or key exchange data in DOI-defined formats
    1. **SA :** used to begin the setup of a new SA; carries various attributes
    2. **Proposal (P):** used during SA setup; indicates protocol to be used (AH or ESP) and number of transforms
    3. **Transform (T) :** used during SA setup; indicates transform (e.g., DES, 3DES) and its attributes
    4. **IKE :** used to carry key exchange data (e.g., Oakley)
    5. **Identification (ID) :** used to exchange identification information (e.g., IP address)
    6. **Certificate Payload :** carries a public key certificate (PGP, X.509, SPKI, …)
    7. **Certificate Request Payload**
    8. **Hash (HASH)**
    9. **Signature Payload**
    10. **Nonce (NONCE)**
    11. **Notification (N) :** contains error or status information
    12. **Delete Payload :** indicates one or more SAs that the sender has deleted from its database (no longer valid)
    13. **Vendor ID**

ISAKMP

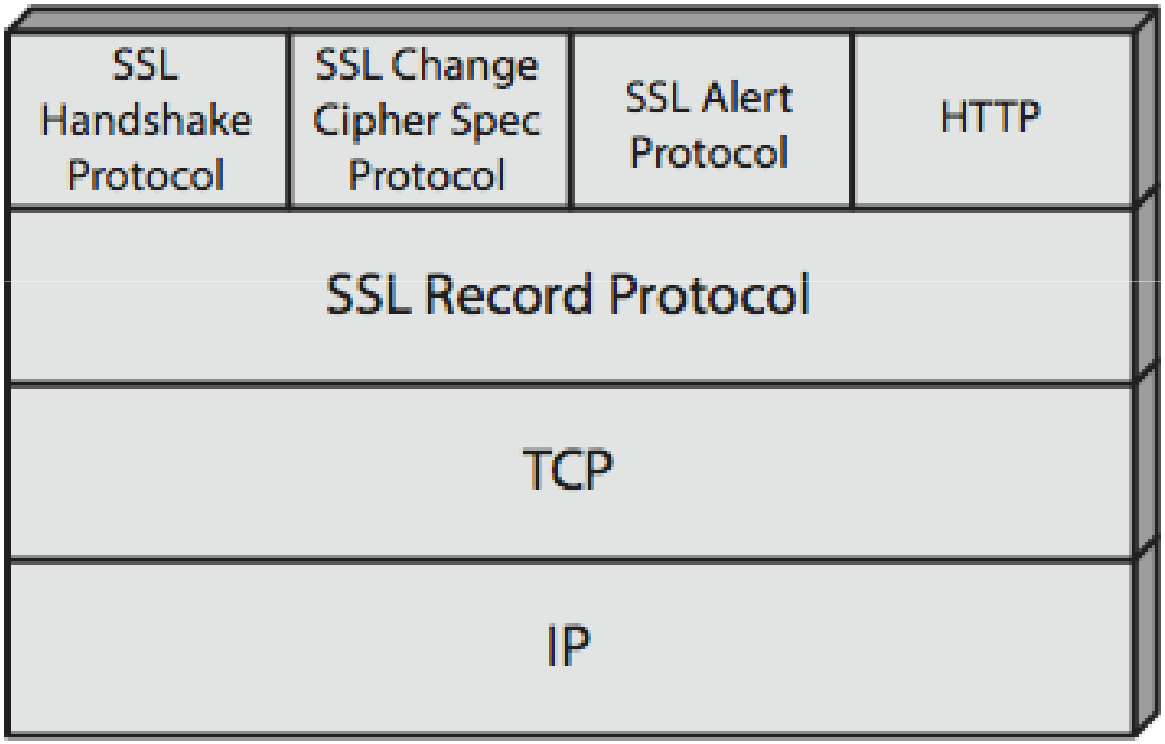
- **ISAKMP Exchanges**

  - **Base Exchange:** Key Exchange and Authentication
  - **Identity Protection Exchange:** Identity and Authentication
  - **Authentication Only Exchange**
  - **Aggressive Exchange:**the Security Association, Key Exchange and Authentication-related
  - **Informational Exchange:** information for security association management

Where SSL Fits



SSL Architecture



SSL Record Protocol Operation

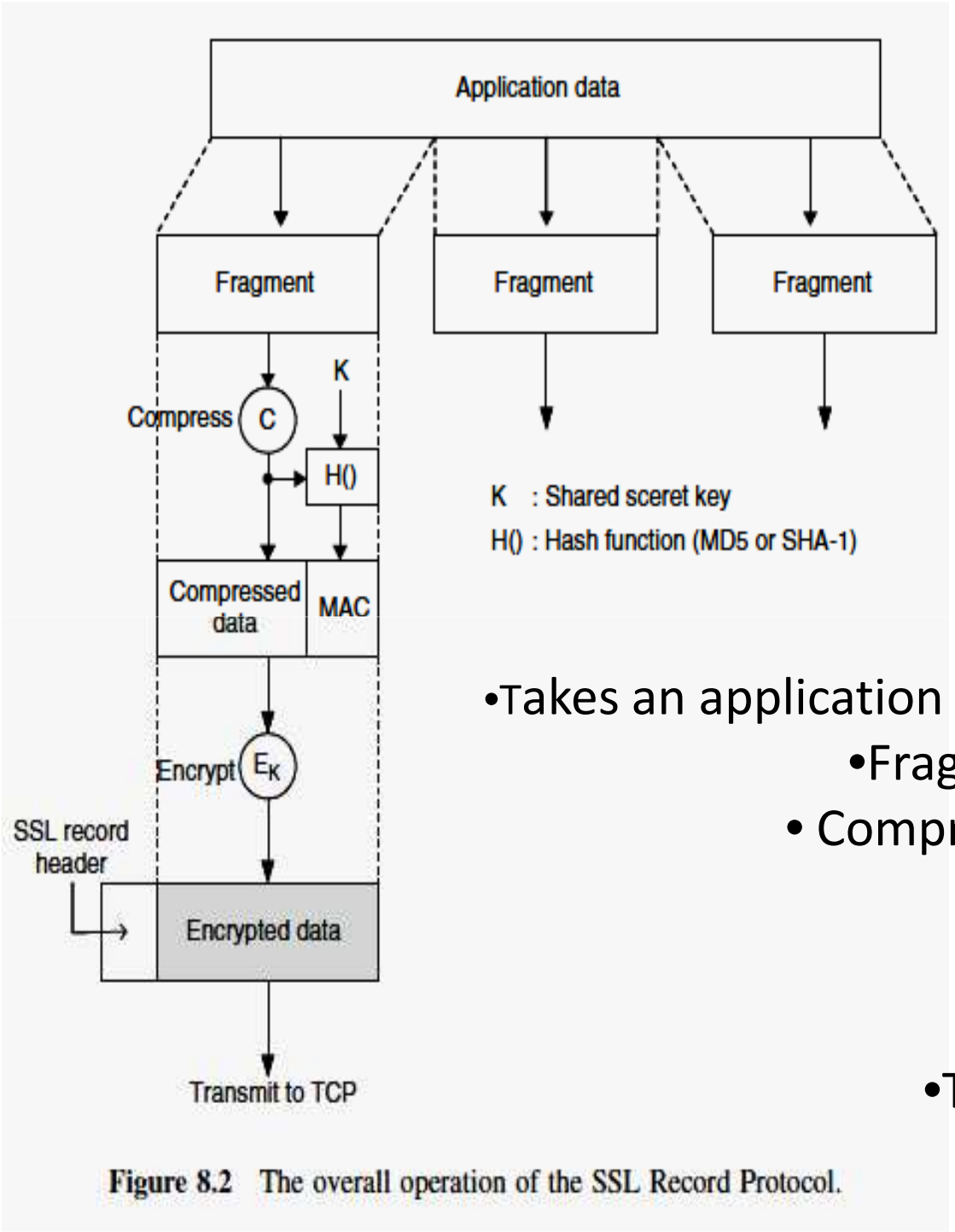


Figure 8.2 The overall operation of the SSL Record Protocol.

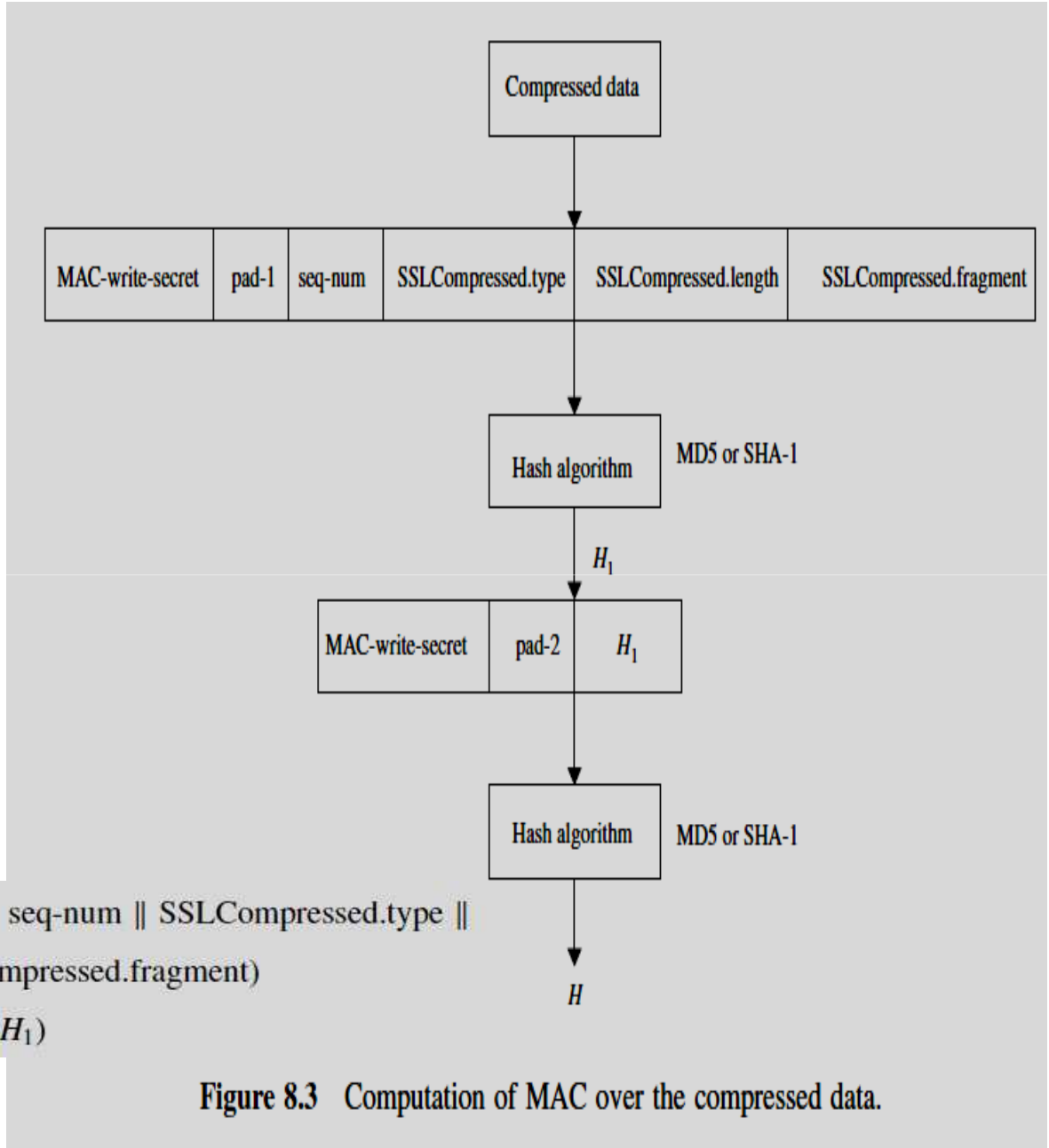
Operations:

- Takes an application message to be transmitted
- Fragments the data into blocks
- Compresses the data (optionally)
- Applies a MAC
- Encrypts
- Adds a header
- Transmits the resulting unit

SSL Record Protocol Operation

MAC:

The MAC is computed before encryption



$$H_1 = \text{hash}(\text{MAC-write-secret} \parallel \text{pad-1} \parallel \text{seq-num} \parallel \text{SSLCompressed.type} \parallel \text{SSLCompressed.length} \parallel \text{SSLCompressed.fragment})$$

$$H = \text{hash}(\text{MAC-write-secret} \parallel \text{pad-2} \parallel H_1)$$

Figure 8.3 Computation of MAC over the compressed data.

SSL Record Protocol Format

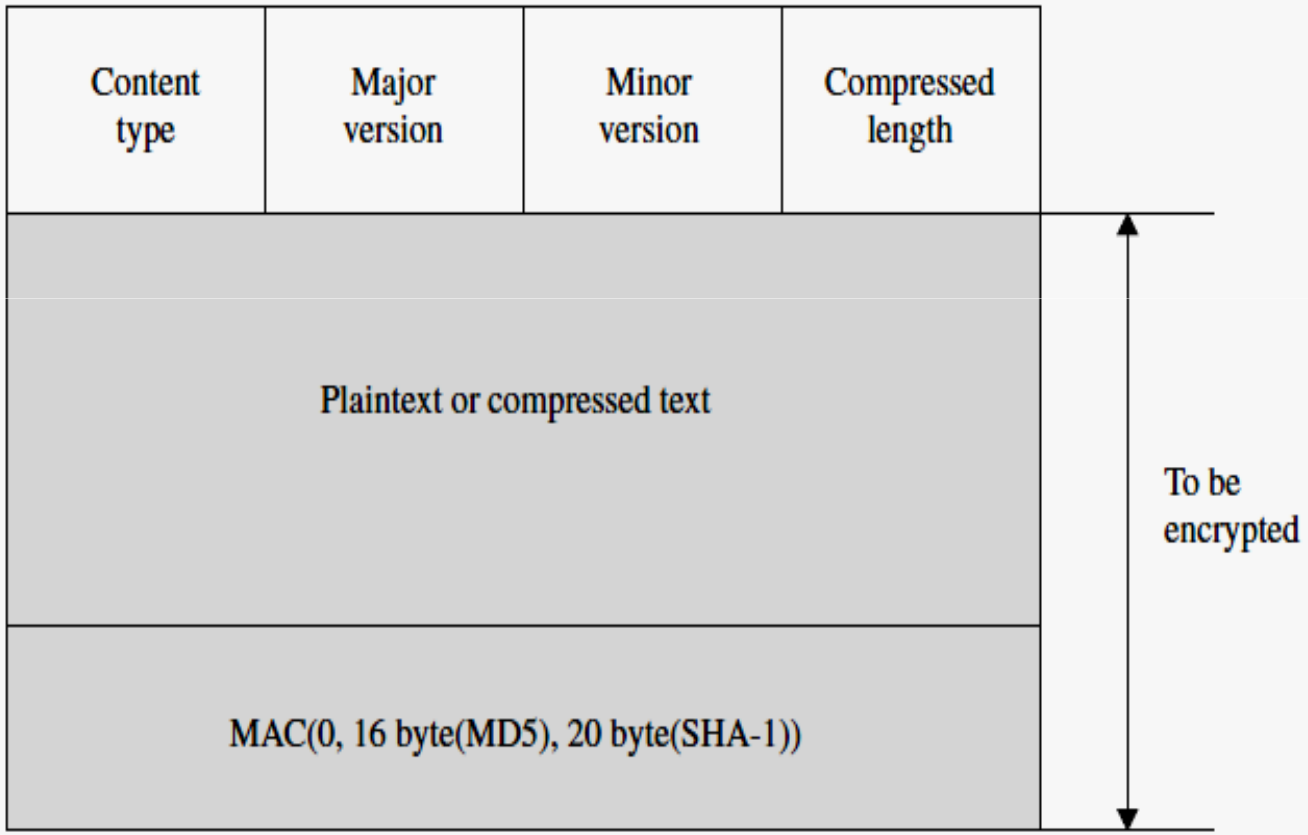


Figure 8.4 SSL Record Protocol format.

SSL-specific protocols

- Change Cipher Spec Protocol: Notify the receiving party that subsequent records will be protected under the just-negotiated CipherSpec and keys
- Alert Protocol : Convey the **severity** of the message and a **description** of the alert
- Handshake Protocol

SSL Handshake Protocol

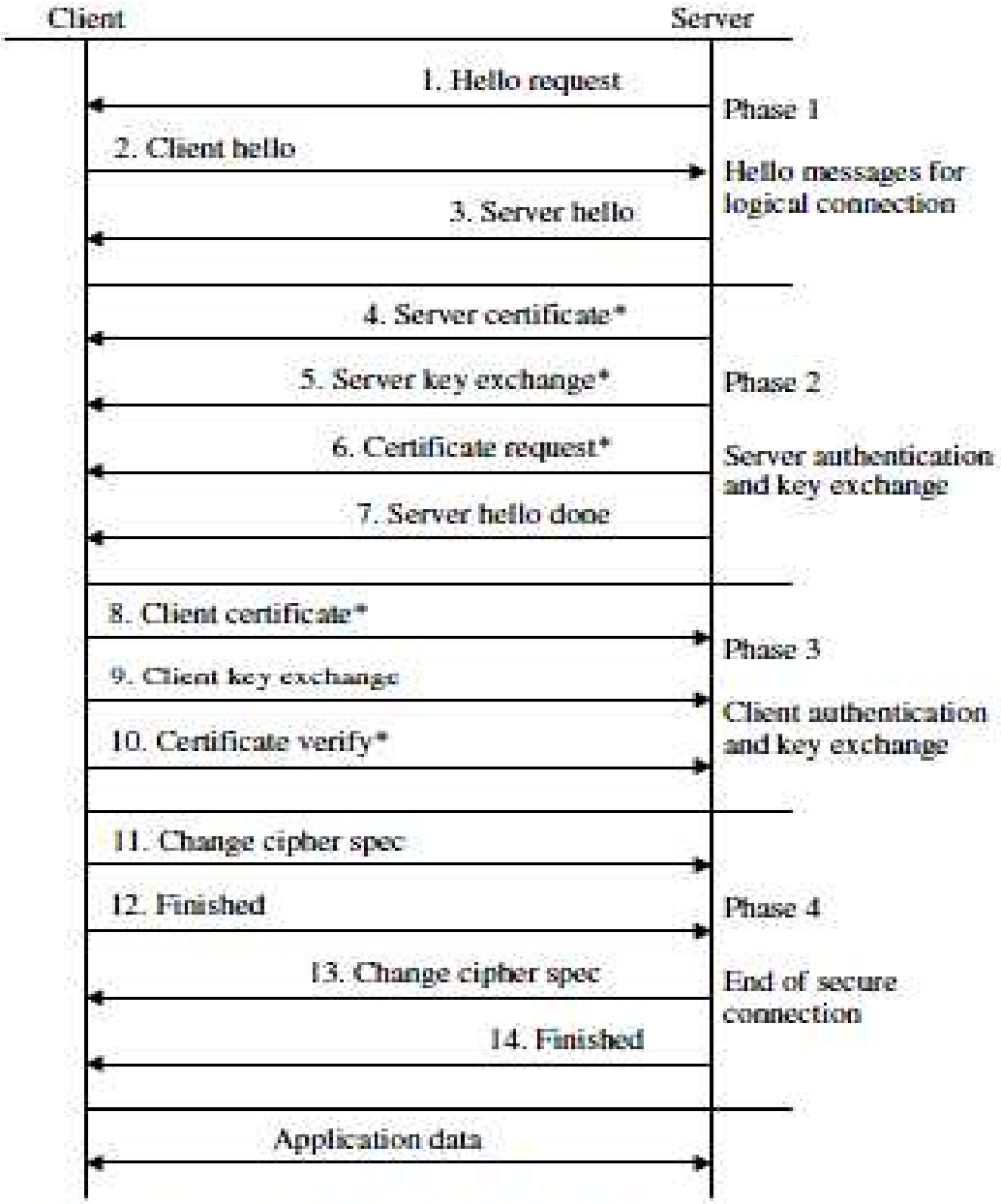


Figure 8.5 SSL Handshake Protocol.

Computing the Master Secret

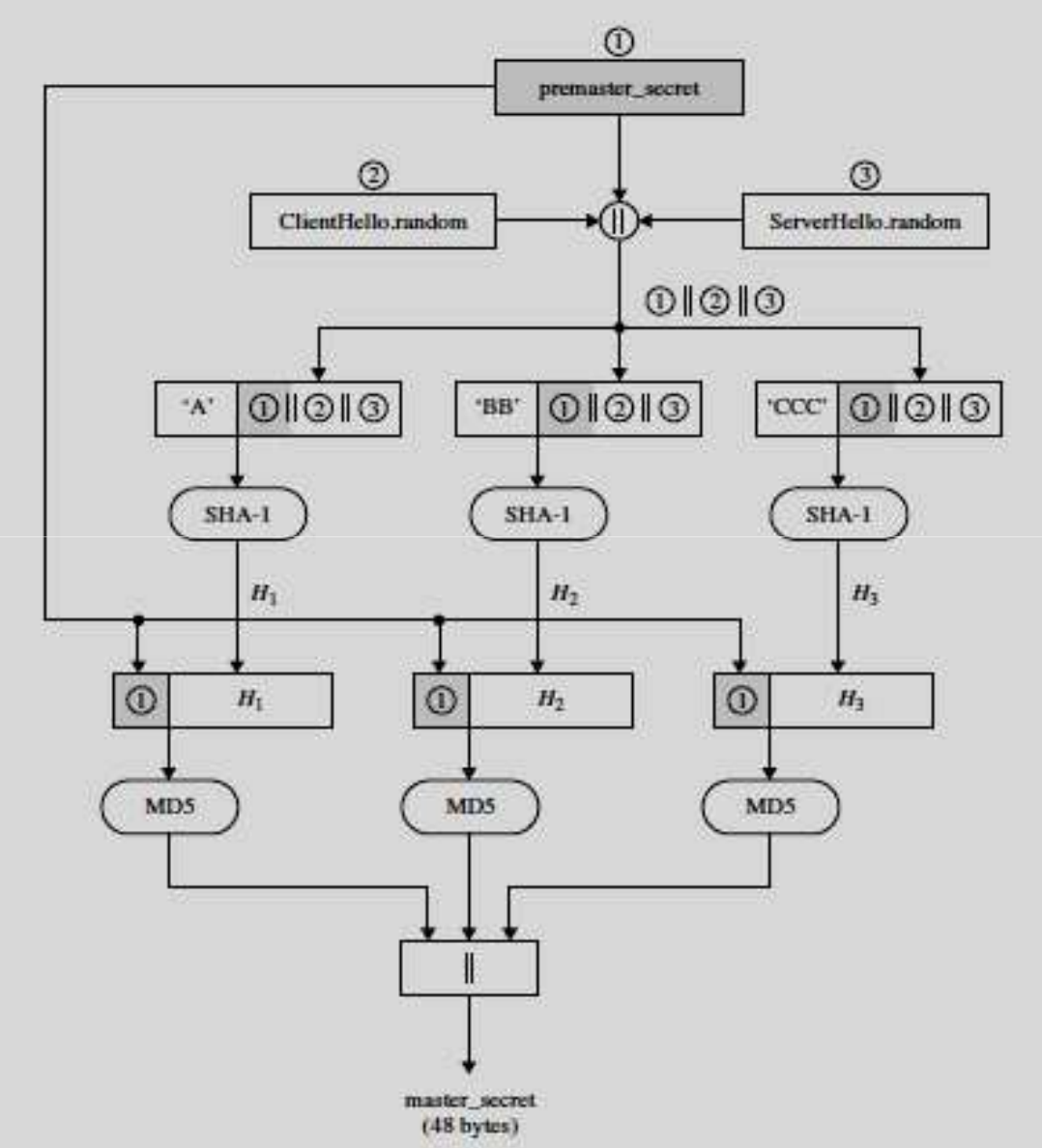


Figure 8.6 Computation of the master secret.

Converting the Master Secret into Cryptographic Parameters

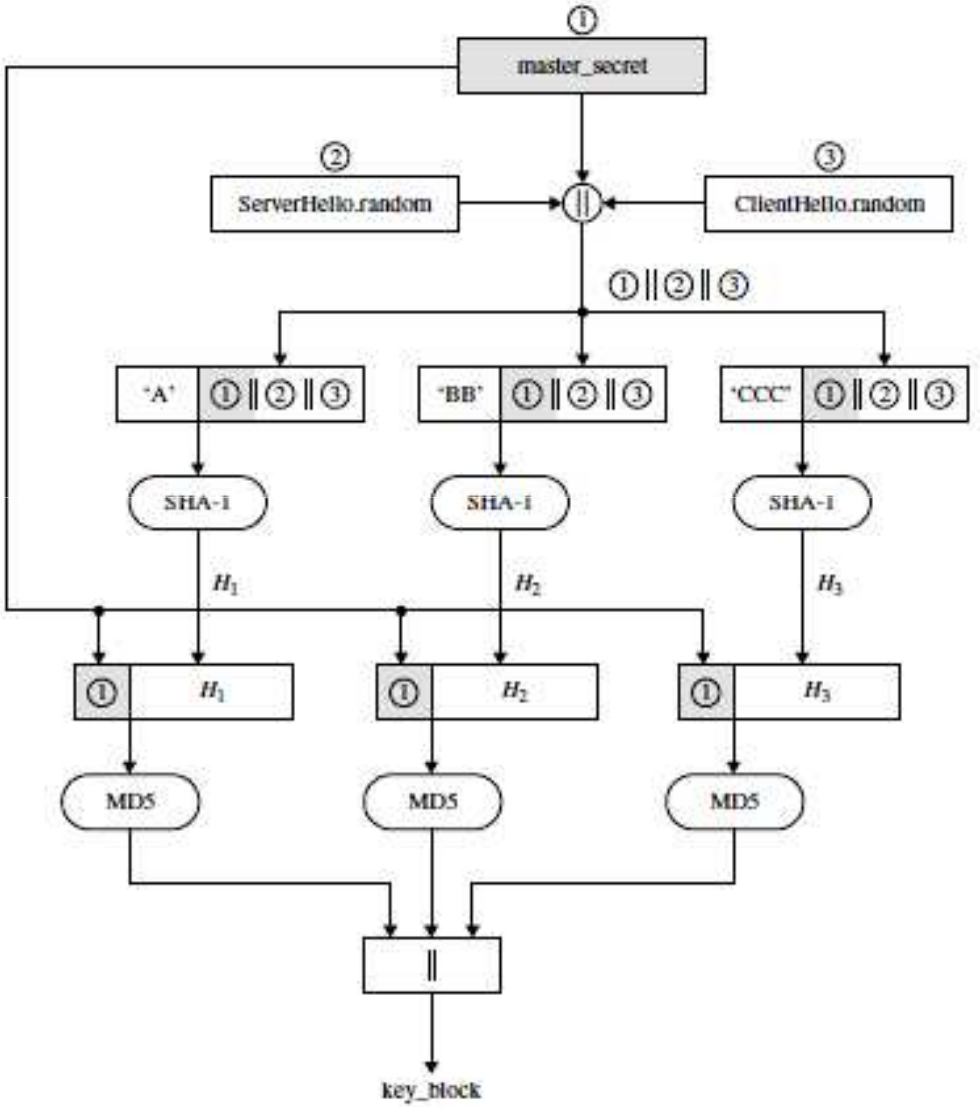


Figure 8.7 Generation of key block.

TLS Protocol Pseudo-random Function (PRF)

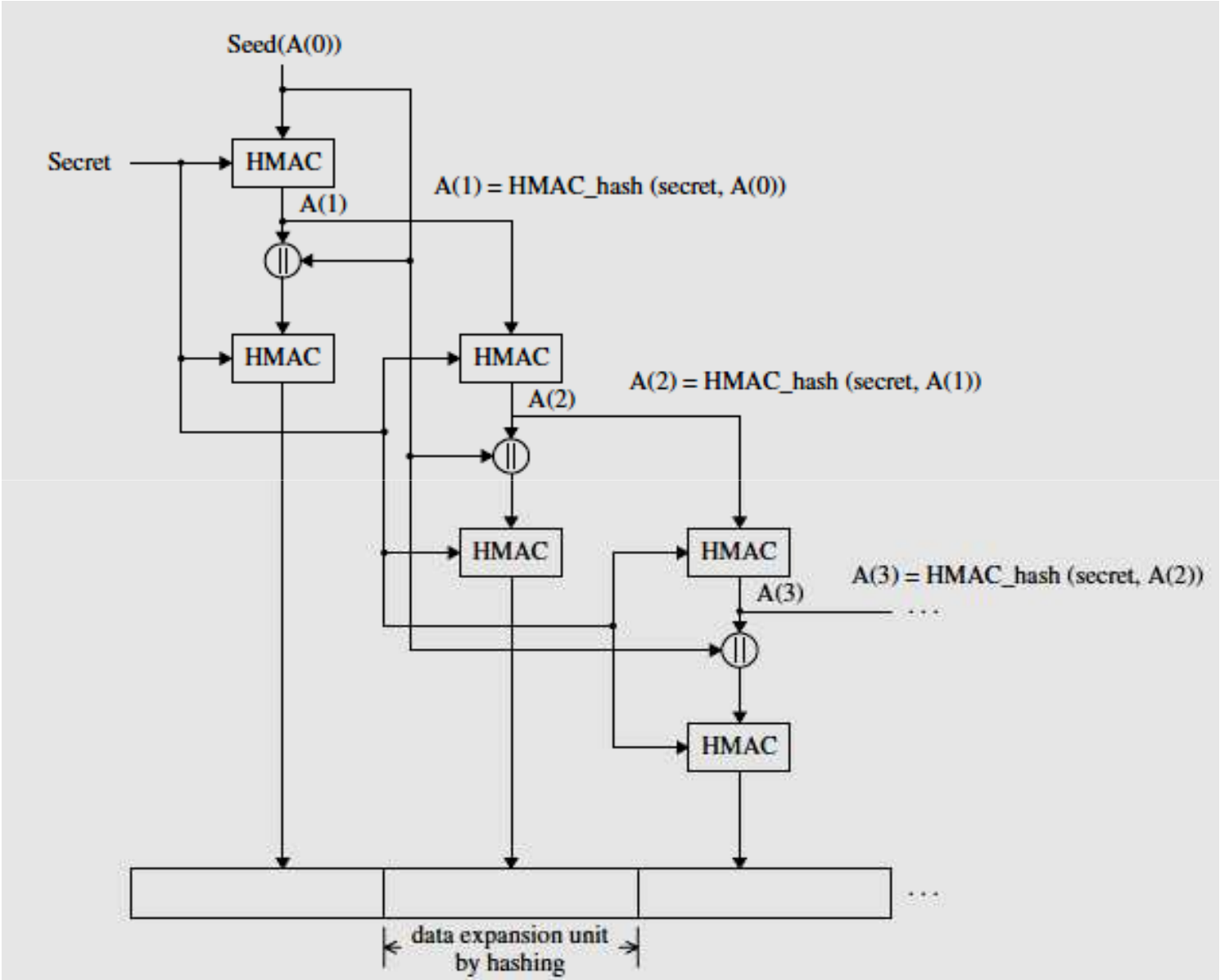
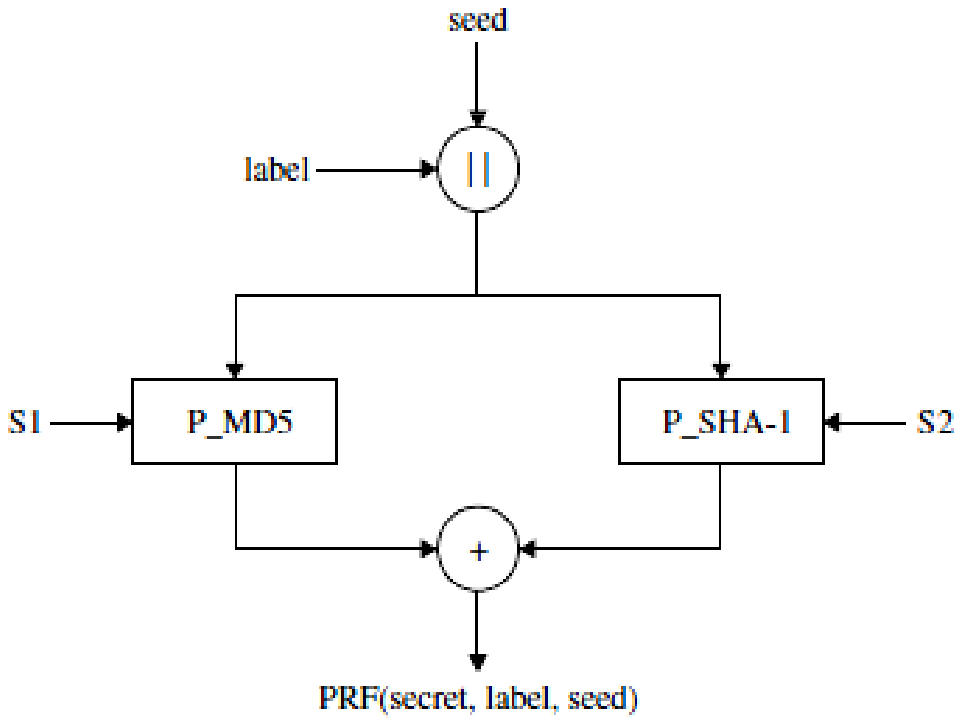


Figure 8.10 TLS data expansion mechanism using P_hash(secret,seed).

TLS Protocol

- **Pseudo-random Function (PRF)**

$$PRF(secret, label, seed) = P\_MD5(S1, label||seed) \oplus P\_SHA-1(S2, label||seed)$$



TLS Protocol

- **Error alerts**
  - TLS supports all of the error alerts defined in SSLv3 with additional alert
    - Decryption failed
    - Record overflow
    - Unknown CA
    - Access denied
    - Decode error
    - Decrypt error
    - Decrypt error:
    - Export restriction
    - Protocol version
    - Insufficient security
    - Internal error:
    - User cancelled
    - No renegotiation
- **Alert level**
  - Not explicitly specified, the sending party may determine at its discretion whether this is a fatal error or not
  - Warning is received, the receiving party may decide at its discretion whether to treat this as a fatal error or not
  - Fatal is received , all messages must be treated as fatal messages and close connection

TLS Protocol

- **Certificate Verify Message**

```
CertificateVerify.signature.md5_hash
        MD5(handshake_message)
CertificateVerify.signature.sha_hash
        SHA(handshake_message)
```

- **Finished Message**

```
PRF(master_secret, finished_label, MD5(handshake_message)||
      SHA-1(handshake_message))
```

- **Cryptographic Computations -Master secret**

```
master_secret = PRF(premaster_secret, ‘‘master secret’’,
                    ClientHello.random||ServerHello.random)
```

```
key_block = PRF(master_secret, ‘‘key expansion’’,
           SecurityParameters.server_random||
                   SecurityParameters.client_random)
```