**VISION INSTITUTE OF MANAGEMENT**

**COMPUTER NETWORK SECURITY**

**BCA 3rd YEAR/6th SEM**

**UNIT 3(IP Security Architecture)**

**IPSec Architecture**

**IPSec (IP Security) architecture** uses two protocols to secure the traffic or data flow. These protocols are ESP (Encapsulation Security Payload) and AH (Authentication Header). IPSec Architecture include protocols, algorithms, DOI, and Key Management. All these components are very important in order to provide the three main services:

* Confidentiality
* Authentication
* Integrity

**IP Security Architecture:**

 

1. **Architecture:**

Architecture or IP Security Architecture covers the general concepts, definitions, protocols, algorithms and security requirements of IP Security technology.

1. **ESP Protocol:**

ESP (Encapsulation Security Payload) provide the confidentiality service. Encapsulation Security Payload is implemented in either two ways:

* ESP with optional Authentication.
* ESP with Authentication.

**Packet Format:**

 

* **Security Parameter Index (SPI):**
This parameter is used in Security Association. It is used to give a unique number to the connection build between Client and Server.
* **Sequence Number:**
Unique Sequence number are allotted to every packet so that at the receiver side packets can be arranged properly.
* **Payload Data:**
Payload data means the actual data or the actual message. The Payload data is in encrypted format to achieve confidentiality.
* **Padding:**
Extra bits or space added to the original message in order to ensure confidentiality. Padding length is the size of the added bits or space in the original message.
* **Next Header:**
Next header means the next payload or next actual data.
* **Authentication Data**:
This field is optional in ESP protocol packet format.
1. **Encryption algorithm:**
Encryption algorithm is the document that describes various encryption algorithm used for Encapsulation Security Payload.
2. **AH Protocol:**
AH (Authentication Header) Protocol provides both Authentication and Integrity service. Authentication Header is implemented in one way only: Authentication along with Integrity.



Authentication Header covers the packet format and general issue related to the use of AH for packet authentication and integrity.

1. **Authentication Algorithm:**
Authentication Algorithm contains the set of the documents that describe authentication algorithm used for AH and for the authentication option of ESP.
2. **DOI (Domain of Interpretation):**
DOI is the identifier which support both AH and ESP protocols. It contains values needed for documentation related to each other.
3. **Key Management:**
Key Management contains the document that describes how the keys are exchanged between sender and receiver.

# Authentication Header (AH)

## **Definition - What does *Authentication Header (AH)* mean?**

Authentication Header (AH) is a protocol and part of the Internet Protocol Security (IPsec) protocol suite, which authenticates the origin of IP packets (datagrams) and guarantees the integrity of the data. The AH confirms the originating source of a packet and ensures that its contents (both the header and payload) have not been changed since transmission.

If security associations have been established, AH can be optionally configured to defend against replay attacks using the sliding window technique.

AH provides authentication of the IP header and next-level protocol data. This can be applied in a nested fashion, or in conjunction with the IP encapsulating security payload (ESP). Security services are initiated between two communicating hosts, between two communicating security gateways or between a security gateway and a host.

AH provides data integrity using a checksum generated by an authentication code, similar to MD5. There is a secret shared key in the AH algorithm for data origin authentication. Using a sequence number field inside the AH header, relay protection is ensured.

AH can be used in tunnel or transport mode. In transport mode, the IP header of a datagram is the outermost IP header, followed by the AH header and the datagram. This mode requires a reduced processing overhead compared to tunnel mode, which creates new IP headers and uses them in the outermost IP header of the datagram.

The fields within an AH header include:

* Next header
* Payload length
* Reserved
* Security parameters
* Sequence numbers
* Integrity check value

# Encapsulating Security Payload (ESP)

## **Definition - What does *Encapsulating Security Payload (ESP)* mean?**

**“**An Encapsulating Security Payload (ESP) is a protocol within the IPSec for providing authentication, integrity and confidentially of network packets data/payload in IPv4 and IPv6 networks. ESP provides message/payload encryption and the authentication of a payload and its origin within the IPSec protocol suite.**”**

**OR**

# “An Encapsulating Security Payload is primarily designed to provide encryption, authentication and protection services for the data or payload that is being transferred in an IP network. ESP doesn’t protect the packet header; however, in a tunnel mode if the entire packet is encapsulated within another packet as a payload/data packet, it can encrypt the entire packet residing inside another packet. Typically, in an IP network packet, the ESP header is placed after the IP header. The components of an ESP header include sequence number, payload data, padding, next header, an integrity check and sequenced numbers.”

Specified in RFC 2406, IP Encapsulating Security Payload (ESP), and the ESP Header allows IP nodes to exchange datagrams whose payloads are encrypted. The ESP Header is designed to provide several different services (some overlapping with the Authentication Header), including the following.

* Confidentiality of datagrams through encryption
* Authentication of data origin through the use of public key encryption
* Antireplay services through the same sequence number mechanism as provided by the Authentication Header
* Limited traffic flow confidentiality through the use of security gateways

The ESP Header can be used in conjunction with an Authentication Header. In fact, unless the ESP Header uses some mechanism for authentication, it is recommended that the Authentication Header be used with the ESP Header.

The ESP Header must follow any headers that need to be processed by nodes intermediate to the destination node-all data that follows the ESP Header will be encrypted, with the encrypted payload beginning directly after the last ESP Header field (see following).

ESP can be used in tunnel or transport mode, similar to the Authentication Header. In transport mode, the IP Header and any Hop-by-Hop, Routing, or Fragmentation Extension Headers precede the Authentication Header (if present), followed by the ESP Header. Any Destination Options Headers can either precede or follow the ESP Header, or even both; any Headers that follow the ESP Header are encrypted.

The result appears, in many respects, to simply be a regular IP datagram transmitted from source to destination, with an encrypted payload. This use of ESP in transport mode is appropriate in some cases, but it allows attackers to study traffic between the two nodes, noting which nodes are communicating, how much data they exchange, when they exchange it, and so forth. All this information may potentially provide the attacker with some information that helps defeat the communicating parties.

An alternative is to use a security gateway, much as just described for the Authentication Header. A security gateway can operate directly with a node or can link to another security gateway. A single node can use ESP in tunnel mode by encrypting all outbound packets and encapsulating them in a separate stream of IP datagrams that are sent to the security gateway. That gateway then can decrypt the traffic and resend the original datagrams to their destinations.

When tunnelling, the ESP Header encapsulates the entire tunnelled IP datagram and is an extension to the IP Header directing that datagram to a security gateway. It is also possible to combine ESP Headers with Authentication Headers in several different ways; for example, the tunnelled datagram may have a Transport-Mode Authentication Header.

The following ESP Header format (taken from RFC 2406) includes the Next Header field, which appears near the end of the ESP Header and indicates the presence (and identity) of any other headers (such as AH) that may follow. The rest of the ESP Header consists of the following-

**Security Parameter Index (SPI)** This is the same 32-bit value referred to in the section on the Authentication Header. This value is used by the communicating nodes to refer to a security association, which can be used to determine how the data should be encrypted.

**Sequence Number** This 32-bit value is set to zero to start and is incremented by one with each datagram sent. As just described for the Authentication Header, the sequence number can be used to protect against replay attacks, and a new security association must be set up before this value cycles through all 2 32 values.

**Payload Data** This is a variable-length fi eld and actually contains the encrypted portion of the datagram, along with any supplementary data necessary for the encryption algorithm (e.g., initialization data). The payload begins with an initialization vector , a value that must be sent in plaintext; encryption algorithms need this value to decrypt the protected data.

**Padding** The encrypted portion of the header (the payload) must end on the appropriate boundary, so padding may be necessary.

**Padding** Length This field indicates how much padding has been added to the payload data.

**Next Header** This field operates as it normally does with other IPv6 extension headers; it just appears near the end of the header (where it can be given confidentiality protection) rather than at the beginning so that the next layer protocol can be hidden from any unauthorized third parties.

**Authentication Data** This is an Integrity Check Value (ICV) calculated on the entire ESP Header (except for the authentication data). This authentication calculation is optional.

**Key Management**

In cryptography it is a very tedious task to distribute the public and private key between sender and receiver. If key is known to the third party (forger/eavesdropper) then the whole security mechanism becomes worthless. So, there comes the need to secure the exchange of keys.

There are 2 aspects for Key Management:

1. Distribution of public keys.
2. Use of public-key encryption to distribute secret.

**Distribution of Public Key:**

Public key can be distributed in 4 ways: Public announcement, publicly available directory, Public-key authority, and Public-key certificates. These are explained as following below-

1. **Public Announcement:**
Here the public key is broadcasted to everyone. Major weakness of this method is forgery. Anyone can create a key claiming to be someone else and broadcast it. Until forgery is discovered can masquerade as claimed user.



1. **Publicly Available Directory:**
In this type, the public key is stored at a public directory. Directories are trusted here, with properties like Participant Registration, access and allow to modify values at any time, contains entries like {name, public-key}.

Directories can be accessed electronically still vulnerable to forgery or tampering.

1. **Public Key Authority:**
It is similar to the directory but, improve security by tightening control over distribution of keys from directory. It requires users to know public key for the directory. Whenever the keys are needed, a real-time access to directory is made by the user to obtain any desired public key securely.
2. **Public Certification:**
This time authority provides a certificate (which binds identity to the public key) to allow key exchange without real-time access to the public authority each time. The certificate is accompanied with some other info such as period of validity, rights of use etc. All of this content is signed by the trusted Public-Key or Certificate Authority (CA) and it can be verified by anyone possessing the authority’s public-key.