Chapter 11
The IPSec Security Architecture for the Internet Protocol
The TCP/IP Protocol Suite

- **IP (Internet Protocol)**: unreliable, connectionless network protocol
- **TCP (Transmission Control Protocol)**: reliable, connection-oriented transport protocol, realized over IP
- **UDP (User Datagram Protocol)**: unreliable, connectionless transport protocol, offers an application interface to IP

Examples for application protocols:
- HTTP: Hypertext Transfer Protocol
- SMTP: Simple Mail Transfer Protocol
The IP Packet Format

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ver.</td>
<td><strong>4 bit</strong>&lt;br&gt;Currently version 4 is widely deployed&lt;br&gt;Version 6 is in deployed sparsely</td>
</tr>
<tr>
<td>IHL</td>
<td><strong>4 bit</strong>&lt;br&gt;Length of the IP header in 32-bit words</td>
</tr>
<tr>
<td>TOS</td>
<td><strong>8 bit</strong>&lt;br&gt;This field is used to indicate the traffic requirements of a packet&lt;br&gt;It is currently under review at the IETF</td>
</tr>
<tr>
<td>Length</td>
<td><strong>16 bit</strong>&lt;br&gt;The length of the packet including the header in octets&lt;br&gt;This field is, like all other fields in the IP suite, in “big endian” representation</td>
</tr>
<tr>
<td>Identification</td>
<td><strong>16 bit</strong>&lt;br&gt;Used to “uniquely” identify an IP datagram&lt;br&gt;Important for re-assembly of fragmented IP datagrams</td>
</tr>
<tr>
<td>Flags</td>
<td><strong>3 bit</strong>&lt;br&gt;Fragmentation</td>
</tr>
<tr>
<td>Fragment Offset</td>
<td><strong>13 bit</strong>&lt;br&gt;The position of this packet in the corresponding IP datagram</td>
</tr>
<tr>
<td>TTL</td>
<td><strong>8 bit</strong>&lt;br&gt;At every processing network node, this field is decremented by one&lt;br&gt;When TTL reaches 0 the packet is discarded to avoid packet looping</td>
</tr>
<tr>
<td>Protocol</td>
<td><strong>8 bit</strong>&lt;br&gt;Indicates the (transport) protocol of the payload&lt;br&gt;Used by the receiving end system to de-multiplex packets among various transport protocols like TCP, UDP, ...</td>
</tr>
<tr>
<td>IP Checksum</td>
<td><strong>16 bit</strong>&lt;br&gt;Protection against transmission errors&lt;br&gt;As this is not a cryptographic checksum, it can easily be forged</td>
</tr>
<tr>
<td>Source Address</td>
<td><strong>32 bit</strong>&lt;br&gt;The IP address of sender of this packet</td>
</tr>
<tr>
<td>Destination Address</td>
<td><strong>32 bit</strong>&lt;br&gt;The IP address of the intended receiver of this packet</td>
</tr>
<tr>
<td>IP Options</td>
<td>variable length&lt;br&gt;An IP header can optionally carry additional information</td>
</tr>
</tbody>
</table>

TCP / UDP / ... Payload
Security Problems of the Internet Protocol

- When an entity receives an IP packet, it has no assurance of:
  - **Data origin authentication / data integrity:**
    - The packet has actually been sent by the entity which is referenced by the source address of the packet
    - The packet contains the original content the sender placed into it, so that it has not been modified during transport
    - The receiving entity is in fact the entity to which the sender wanted to send the packet
  - **Confidentiality:**
    - The original data was not inspected by a third party while the packet was sent from the sender to the receiver
Security Objectives of IPSec

- IPSec aims to ensure the following security objectives:
  - *Data origin authentication / connectionless data integrity*: It is not possible to send an IP datagram with neither a masqueraded IP source nor destination address without the receiver being able to detect this.
  - It is not possible to modify an IP datagram in transit without the receiver being able to detect the modification.
  - *Replay protection*: it is not possible to later replay a recorded IP packet without the receiver being able to detect this.
  - *Confidentiality*: it is not possible to eavesdrop on the content of IP datagrams (and limited traffic flow confidentiality).

- Security policy:
  - Sender, receiver and intermediate nodes can determine the required protection for an IP packet according to a local security policy.
  - Intermediate nodes and the receiver will drop IP packets that do not meet these requirements.
Overview of the IPSec Standardization

IPSec-Architecture
RFC 2401

Encapsulating Security Payload
RFC 2406

- DES-CBC
RFC 2405

CBC Mode Cipher Algorithms
RFC 2451

Authentication Header
RFC 2402

- HMAC-MD5
RFC 2403

- HMAC-SHA-1
RFC 2404

- HMAC-RIPMED-160
RFC 2857

Key Management

- Photuris
RFC 2522

- SKIP (expired Internet Draft)

ISAKMP
RFCs 2407, 2408

- Internet Key Exchange
RFC 2409

- Oakley Key Mgmt. Protocol
RFC 2412

uses

consists of

[NetSec], WS 2006/2007 11.6
Overview of the IPSec Architecture (1)

- RFC 2401 defines the basic architecture of IPSec:
  - **Concepts:**
    - Security association (SA), security association database (SADB)
    - Security policy, security policy database (SPD)
  - **Fundamental IPSec Protocols:**
    - Authentication Header (AH)
    - Encapsulating Security Payload (ESP)
  - **Protocol Modes:**
    - Transport Mode
    - Tunnel Mode
  - Use of various cryptographic primitives with AH and ESP:
    - Encryption: DES-CBC, CBC mode cipher algorithms
    - Integrity: HMAC-MD5, HMAC-SHA-1, HMAC-RIPEMD-160
  - **Key Management Procedures:**
    - ISAKMP, IKE
A **security association (SA)** is a simplex “connection” that provides security services to the traffic carried by it.

- Security services are provided to one SA by the use of either AH or ESP, but not both.
- For bi-directional communication two security associations are needed.
- An SA is uniquely identified by a triple consisting of a **security parameter index (SPI)**, an IP destination address, and a security protocol identifier (AH / ESP).

An SA can be set up between the following peers:

- **Host ↔ Host**
- **Host ↔ Gateway (or vice versa)**
- **Gateway ↔ Gateway**

There are two conceptual databases associated with SAs:

- **The security policy database (SPD)** specifies, what security services are to be provided to which IP packets and in what fashion.
- **The security association database (SADB)**
Overview of the IPSec Architecture (3)

- Protocol modes – An SA is always of one of the following types:
  - **Transport mode** can only be used between end-points of a communication
  - **Tunnel mode** can be used with arbitrary peers

- The difference between the two modes is, that:
  - Transport mode just adds a security specific header (+ trailer):
    
    | IP header | IPSec header | protected data |
    |------------|--------------|----------------|

  - Tunnel mode encapsulates IP packets:
    
    | IP header | IPSec header | IP header | protected data |
    |------------|--------------|-----------|----------------|

Encapsulation of IP packets allows for a gateway protecting traffic on behalf of other entities (e.g. hosts of a subnetwork, etc.)
Overview of the IPSec Architecture (4)

- **Authentication header (AH)**
  - Provides data origin authentication and replay protection
  - Is realized as a header which is inserted between the IP header and the data to be protected

![Diagram](https://via.placeholder.com/150)

- **Encapsulating security payload (ESP)**
  - Provides data origin authentication, confidentiality and replay protection
  - Is realized with a header and a trailer encapsulating the data to be protected

![Diagram](https://via.placeholder.com/150)
Overview of the IPSec Architecture (4)

- Setup of security associations is realized with:
  - Internet Security Association Key Management Protocol (ISAKMP)
    - Defines generic framework for key authentication, key exchange and negotiation of security association parameters [RFC2408]
    - Does not define a specific authentication protocol, but specifies:
      - Packet formats
      - Retransmission timers
      - Message construction requirements
    - Use of ISAKMP for IPSec is further detailed in [RFC2407]
  - Internet Key Exchange (IKE)
    - Defines an authentication and key exchange protocol [RFC2409]
    - Is conformant to ISAKMP and may be used for different applications
    - Setup of IPSec SAs between two entities is realized in two phases:
      - Establishment of an IKE SA (defines how to setup IPSec SAs)
      - Setup of IPSec SAs
Both AH- and ESP-protected IP packets carry a sequence number which realizes a replay protection:

- When setting up an SA this sequence number is initialized to zero
- The sequence number is increased with every IP packet sent
- A new session key is needed before a wrap-around occurs
- The receiver of an IP packet checks, if the sequence number is contained in a window of acceptable numbers

The window size has to be at least 32 in practice.

Packet with sequence number \( N \) can still be accepted when

- \( N + 7 \) arrives
- \( N + 16 \) arrives

Packet “\( N + 17 \)” arrives
IPSec Replay Protection (2)

- If a received packet has a sequence number which:
  - is to the left of the current window ⇒ the receiver rejects the packet
  - is inside the current window ⇒ the receiver accepts the packet
  - is to the right of the current window ⇒ the receiver accepts the packet and advances the window

Of course IP packets are only accepted if they pass the authentication verification and the window is never advanced before this verification.

- The minimum window size is 32 packets (64 packets is recommended)

Sliding window of received packets

Packet with sequence number N can no longer be accepted
IPSec Implementation Alternatives: Host Implementation

- Advantages of IPSec implementation in end systems:
  - Provision of end-to-end security services
  - Provision of security services on a per-flow basis
  - Ability to implement all modes of IPSec

- Two main integration alternatives:

<table>
<thead>
<tr>
<th>OS integrated</th>
<th>“Bump” in the stack</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application</td>
<td>Application</td>
</tr>
<tr>
<td>Transport</td>
<td>Transport</td>
</tr>
<tr>
<td>Network + IPSec</td>
<td>Network</td>
</tr>
<tr>
<td>Data Link</td>
<td>IPSec</td>
</tr>
<tr>
<td></td>
<td>Data Link</td>
</tr>
</tbody>
</table>

True OS integration is the method of choice, as it avoids duplication of functionality.

If the OS cannot be modified, IPSec is inserted above the data link driver.
IPSec Implementation Alternatives: Router Implementation

- Advantages of IPSec implementation in routers:
  - Ability to secure IP packets flowing between two networks over a public network such as the Internet:
    - Allows to create *virtual private networks (VPNs)*
    - No need to integrate IPSec in every end system
  - Ability to authenticate and authorize IP traffic coming in from remote users

- Two main implementation alternatives:

<table>
<thead>
<tr>
<th>Router integrated</th>
<th>“Bump” in the wire</th>
</tr>
</thead>
<tbody>
<tr>
<td>RA</td>
<td>RA_B</td>
</tr>
<tr>
<td>RB</td>
<td>RA_C</td>
</tr>
<tr>
<td>RC</td>
<td>Internet</td>
</tr>
<tr>
<td>Internet</td>
<td>Physical Link</td>
</tr>
<tr>
<td>Physical Link</td>
<td>Internet</td>
</tr>
<tr>
<td></td>
<td>Physical Link</td>
</tr>
</tbody>
</table>
When to use which IPSec Mode? (1)

- Transport mode is used when the “cryptographic endpoints” are also the “communication endpoints” of the secured IP packets
  - Cryptographic endpoints: the entities that generate / process an IPSec header (AH or ESP)
  - Communication endpoints: source and destination of an IP packet

- In most cases, communication endpoints are hosts (workstations, servers), but this is not necessarily the case:
  - Example: a gateway being managed via SNMP by a workstation
When to use which IPSec Mode? (2)

- Tunnel mode is used when at least one “cryptographic endpoint” is not a “communication endpoint” of the secured IP packets
  - This allows for gateways securing IP traffic on behalf of other entities

### Diagram

- **IP packet flow**
  - A (Src = RA, Dst = RB) to B (Src = A, Dst = B)
  - Through RA and SA_{RA,RB} in the Internet

### Packet Structure

<table>
<thead>
<tr>
<th>IP header</th>
<th>IPSec header</th>
<th>IP header</th>
<th>protected data</th>
</tr>
</thead>
<tbody>
<tr>
<td>Src = RA</td>
<td></td>
<td>Src = A</td>
<td></td>
</tr>
<tr>
<td>Dst = RB</td>
<td></td>
<td>Dst = B</td>
<td></td>
</tr>
</tbody>
</table>

[NetSec], WS 2006/2007
When to use which IPSec Mode? (3)

- The above description of application scenarios for tunnel mode includes the case in which only one cryptographic endpoint is not a communication endpoint:
  - Example: a security gateway ensuring authentication and/or confidentiality of IP traffic between a local subnetwork and a host connected via the Internet (“road warrior scenario”)

```
++- IP packet flow 
|   Internet   |
++- packet structure

<table>
<thead>
<tr>
<th>IP header</th>
<th>IPSec header</th>
<th>IP header</th>
<th>protected data</th>
</tr>
</thead>
<tbody>
<tr>
<td>Src = A</td>
<td></td>
<td>Dst = RB</td>
<td></td>
</tr>
<tr>
<td>Dst = RB</td>
<td></td>
<td>Src = A</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Source = A</td>
<td>Dst = B</td>
<td></td>
</tr>
</tbody>
</table>
```

[NetSec], WS 2006/2007 11.18
Nesting of Security Associations (1)

- Security associations may be nested:
  - Example: Host A and gateway RB perform data origin authentication and gateways RA and RB perform subnetwork-to-subnetwork confidentiality

![Diagram of nesting of security associations]

**IP packet flow**

**Packet structure**

<table>
<thead>
<tr>
<th>IP header</th>
<th>IP header</th>
<th>IP header</th>
<th>protected data</th>
</tr>
</thead>
<tbody>
<tr>
<td>Src = RA</td>
<td>Src = A</td>
<td>Src = A</td>
<td></td>
</tr>
<tr>
<td>Dst = RB</td>
<td>Dst = RB</td>
<td>Dst = B</td>
<td></td>
</tr>
</tbody>
</table>
However, one has to take care when nesting SAs that there occurs no “incorrect bracketing” of SAs, like “[()]”

One example of valid SA nesting:

Packet Structure

<table>
<thead>
<tr>
<th>IP header</th>
<th>IPSec header</th>
<th>IP header</th>
<th>IPSec header</th>
<th>IP header</th>
<th>protected data</th>
</tr>
</thead>
<tbody>
<tr>
<td>Src = RB</td>
<td>Src = RA</td>
<td>Src = A</td>
<td>Dst = D</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Dst = RC</td>
<td>Dst = RD</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Nesting of Security Associations (3)

- One example of invalid SA nesting:

  A --- RA --- RB --- RC --- RD --- D

  Tunnel 1

  Tunnel 2

  Packet Structure

<table>
<thead>
<tr>
<th>IP header</th>
<th>IPSec header</th>
<th>IP header</th>
<th>IPSec header</th>
<th>IP header</th>
<th>protected data</th>
</tr>
</thead>
<tbody>
<tr>
<td>Src = RB</td>
<td>Src = RA</td>
<td>Src = A</td>
<td>Dst = D</td>
<td>Dst = RC</td>
<td>Dst = RC</td>
</tr>
<tr>
<td>Dst = RD</td>
<td>Dst = RC</td>
<td>Dst = A</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- As the packet is tunneled from RB to RD the gateway RC can not process the inner IPSec header
- A possible result of this faulty configuration could be that the packet is routed back to RC
Basic Scheme of IPSec Processing: Outgoing Packets

- In order to support IPSec it has to perform the following steps:
  1. Determine if and how the outgoing packet needs to be secured:
     - This is realized by performing a lookup in the SPD
     - If the policy specifies “discard” then drop the packet ⇒ done
     - If the packet does not need to be secured, then send it ⇒ done
  2. Determine which SA should be applied to the packet:
     - If there is not yet an appropriate SA established, then ask the key management demon to perform an IKE
  3. Look up the determined (and eventually freshly created) SA in the SADB
  4. Perform the security transform determined by the SA by using the algorithm, its’ parameters and the key as specified in the SA
     - This results in the construction of an AH or an ESP header
     - Eventually also a new (outer) IP header will be created (tunnel mode)
  5. Send the resulting IP packet ⇒ done
In order to support IPSec it has to perform the following steps:

1. Determine if the packet contains an IPSec header this entity is supposed to process:
   - If there is such an IPSec header then look up the SA in the SADB which is specified by the SPI of the IPSec header and perform the appropriate IPSec processing
   - If the SA referenced by the SPI does not (yet) exist, drop the packet

2. Determine if and how the packet should have been protected:
   - This is again realized by performing a lookup in the SPD, with the lookup being performed by evaluating the inner IP header in case of tunneled packets
   - If the policy specifies “discard” then drop the packet
   - If the protection of the packet did not match the policy, drop the packet

3. If the packet had been properly secured, then deliver it to the appropriate protocol entity (network / transport layer)
IPSec Security Policy Selection

- The following selectors to be extracted from the network and transport layer headers allow to select a specific policy in the SPD:
  - *IP source address*:
    - Specific host, network prefix, address range, or wildcard
  - *IP destination address*:
    - Specific host, network prefix, address range, or wildcard
    - In case of incoming tunneled packets the inner header is evaluated
  - *Name*:
    - DNS name, X.500 name or other name types as defined in IPSec domain of interpretation of a protocol for setting up SAs
    - Only used during SA negotiation
  - *Protocol*:
    - The protocol identifier of the transport protocol for this packet
    - This may not be accessible when a packet is secured with ESP
  - *Upper layer ports*:
    - If accessible, the upper layer ports for session oriented policy selection
IPSec Security Policy Definition

- Policy selectors are used to select specific policy definitions
- A policy definition specifies:
  1. How to perform setup of an IKE SA between two nodes:
     - *Phase I mode*: main mode or aggressive mode
     - *Protection suite(s)*: specify how IKE authentication is performed
  2. Which and how security services should be provided to IP packets:
     - *Selectors*, that identify specific flows
     - *Security attributes* for each flow:
       - *Security protocol*: AH or ESP
       - *Protocol mode*: transport or tunnel mode
       - *Security transforms*: cryptographic algorithms and parameters
       - *Other parameters*: SA lifetime, replay window
     - *Action*: discard, secure, bypass
- If an SA is already established with a corresponding security endpoint, it is referenced in the SPD
The Encapsulating Security Payload (1)

- ESP constitutes a generic security protocol that provides to IP packets replay protection and one or both of the following security services:
  - Confidentiality, by encrypting encapsulated packets or just their payload
  - Data origin authentication, by creating and adding MACs to packets

- The ESP definition is divided up into two parts:
  - The definition of the base protocol [RFC2406]:
    - Definition of the header and trailer format
    - Basic protocol processing
    - Tunnel and transport mode operation
  - The use of specific cryptographic algorithms with ESP:
    - Encryption: DES-CBC [RFC2405], use of other ciphers [RFC2451]
    - Authentication: HMAC-MD5-96 [RFC2403], HMAC-SHA-96 [RFC2404]
The Encapsulating Security Payload (2)

- The ESP header immediately follows an IP header or an AH header.
- The next-header field of the preceding header indicates “50” for ESP.
The Encapsulating Security Payload (3)

ESP Outbound Processing

Mode?

Tunnel

Prepare Tunnel Mode Header

no

Encrypt?

yes

Encrypt Payload

no

Sign?

yes

Compute MAC

Compute Checksum of Outer IP header
The Encapsulating Security Payload (4)

Prepare Tunnel Mode Header

- Put ESP Header Before IP Header
- ESP.nextHeader = IP
- Fill Other ESP Header Fields
- Put New IP Header Before ESP Header
- NewIP.nextHeader = ESP
  - NewIP.src = this.IP
  - NewIP.dest = tunnelEnd.IP

Prepare Transport Mode Header

- Insert ESP Header After IP Header
- ESP.nextHeader = IP.nextHeader
- IP.nextHeader = ESP
- Fill Other ESP Header Fields
The Encapsulating Security Payload (5)

ESP Inbound Processing (1)

All Fragments Available?

- no
  - Wait for Fragments

- yes
  - Does SA for SPI Exist?
    - no
      - Discard Packet
    - yes
      - Is this a Replay?
        - no
          - Discard Packet
        - yes
          - Discard Packet
      - Packet Authentic?
        - no
          - Discard Packet
        - yes
          - Advance Replay Window & Continue Processing
The Encapsulating Security Payload (6)

ESP Inbound Processing (2)

1. Decrypt Packet

Tunnel Mode?

- Strip Outer IP Header
- Strip ESP Header

Transport Mode?

- IP.nextHeader = ESP.nextHeader
- Strip ESP Header
- Re-Compute IP Checksum

Does Packet Conform to SAs Policy?

- no → Discard Packet
- yes → Deliver Packet
The Authentication Header (1)

- AH constitutes a generic security protocol that provides to IP packets:
  - Replay protection
  - Data origin authentication, by creating and adding MACs to packets
- Like with ESP the AH definition is divided up into two parts:
  - The definition of the base protocol [RFC2402]:
    - Definition of the header format
    - Basic protocol processing
    - Tunnel and transport mode operation
  - The use of specific cryptographic algorithms with AH (for authentication):
    - HMAC-MD5-96 [RFC2403], HMAC-SHA-96 [RFC2404]

- If both ESP and AH are to be applied by one entity, then ESP is always applied first:
  - This results in AH being the outer header
  - Advantage: the ESP header can also be protected by AH
  - Remark: two SAs (one for each AH, ESP) are needed for each direction
In tunnel mode the payload constitutes a complete IP packet.
Although AH also protects the outer IP header, some of its’ fields must not be protected as they are subject to change during transit:
- This also applies to mutable IPv4 options or IPv6 extensions
- Such fields are assumed being zero when computing the MAC

All immutable fields, options and extensions (gray) are protected
The Authentication Header (4)

AH Outbound Processing

Tunnel Mode?

Prepare Tunnel Mode Header

Compute MAC

Transport Mode?

Prepare Transport Mode Header

Compute Checksum of Outer IP header
The Authentication Header (5)

Prepare Tunnel Mode Header

- Put AH Header Before IP Header
- AH.nextHeader = IP
- Fill Other AH Header Fields
- Put New IP Header Before AH Header
  - NewIP.nextHeader = AH
  - NewIP.src = this.IP
  - NewIP.dest = tunnelEnd.IP

Prepare Transport Mode Header

- Insert AH Header After IP Header
- AH.nextHeader = IP.nextHeader
- IP.nextHeader = AH
- Fill Other AH Header Fields
The Authentication Header (6)

AH Inbound Processing (1) →

All Fragments Available?

no → Wait for Fragments

yes →

Does SA for SPI Exist?

no → Discard Packet

yes →

Is this a Replay?

no → Discard Packet

yes →

Packet Authentic?

no → Discard Packet

yes →

Advance Replay Window & Continue Processing
The Authentication Header (7)

AH Inbound Processing (2)

Mode?

Tunnel

- Strip Outer IP Header
- Strip AH Header

Transport

- IP.nextHeader = AH.nextHeader
- Strip AH Header
- Re-Compute IP Checksum

Does Packet Conform to SAs Policy?

- no → Discard Packet
- yes → Deliver Packet
Establishment of Security Associations

- Prior to any packet being protected by IPSec, an SA has to be established between the two “cryptographic endpoints” providing the protection.

- SA establishment can be realized:
  - Manually, by proprietary methods of systems management
  - Dynamically, by a standardized authentication & key management protocol
  - Manual establishment is supposed to be used only in very restricted configurations (e.g. between two encrypting firewalls of a VPN) and during a transition phase

- IPSec defines a standardized method for SA establishment:
  - *Internet Security Association and Key Management Protocol (ISAKMP)*
    - Defines protocol formats and procedures for security negotiation
  - *Internet Key Exchange (IKE)*
    - Defines IPSec’s standard authentication and key exchange protocol
The IETF has adopted two RFCs on ISAKMP for IPSec:
- RFC 2408, which defines the ISAKMP base protocol
- RFC 2407, which defines IPSec’s “domain of interpretation” (DOI) for ISAKMP further detailing message formats specific for IPSec

The ISAKMP base protocol is a generic protocol, that can be used for various purposes:
- The procedures specific for one application of ISAKMP are detailed in a DOI document
- Other DOI documents have been produced:
  - Group DOI for secure group communication (Internet Draft, Sep. 2000)
  - MAP DOI for use of ISAKMP to establish SAs for securing the Mobile Application Protocol (MAP) of GSM (Internet Draft, Nov. 2000)

ISAKMP defines two fundamental categories of exchanges:
- Phase 1 exchanges, which negotiate some kind of “Master SA”
- Phase 2 exchanges, which use the “Master SA” to establish other SAs
ISAKMP – Limited Denial of Service Protection

- The initiator and responder cookies also serve as a protection against simple denial of service attacks:
  - Authentication and key exchange often requires “expensive” computations, e.g. exponentiation (for Diffie-Hellman key exchange)
  - In order to avoid, that an attacker can easily flood an ISAKMP entity with bogus messages from forged source addresses and cause these expensive operations, the following scheme is used:
    - The initiating ISAKMP entity generates an initiator cookie:
      \[ CKY-I = H(\text{Secret}_{\text{Initiator}}, \text{Address}_{\text{Responder}}, t_{\text{Initiator}}) \]
    - The responder generates his own cookie:
      \[ CKY-R = H(\text{Secret}_{\text{Responder}}, \text{Address}_{\text{Initiator}}, t_{\text{Responder}}) \]
    - Both entities always include both cookies, and always check their own cookie before performing any expensive operation
    - The attack mentioned above will, therefore, not be successful as the attacker needs to receive a response from the attacked system in order to obtain a cookie from it
  - ISAKMP does not specify the exact cookie generation method
The proposal payload provides the initiating entity with the capability to present to the responding entity the security protocols and associated security mechanisms for use with the security association being negotiated.

If the SA establishment negotiation is for a combined protection suite consisting of multiple protocols, then there must be multiple proposal payloads each with the same proposal number.

These proposals must be considered as a unit and must not be separated by a proposal with a different proposal number.
This first example shows an ESP AND AH protection suite:

- The first protocol is presented with two transforms supported by the proposing entity, ESP with:
  - Transform 1 as 3DES; Transform 2 as DES
  - The responder must select from the two transforms proposed for ESP
- The second protocol is AH and is presented with a single transform:
  - Transform 1 as SHA

The resulting protection suite will be either:

- 3DES and SHA, or
- DES and SHA

In this case, the SA payload will be followed by the following payloads:

- [Proposal 1, ESP, (Transform 1, 3DES, ...), (Transform 2, DES)]
- [Proposal 1, AH, (Transform 1, SHA)]

Please remark, that this will result in two SAs per direction!
This second example shows a proposal for two different protection suites:

- The first protection suite is presented with:
  - one transform (MD5) for the first protocol (AH), and
  - one transform (3DES) for the second protocol (ESP)

- The second protection suite is presented with two transforms for a single protocol (ESP):
  - 3DES, or
  - DES

Please note, that it is not possible to specify that transform 1 and transform 2 have to be used for one instance of a protocol specification.

- In this case, the SA payload will be followed by the following payloads:
  - [Proposal 1, AH, (Transform 1, MD5, ...)]
  - [Proposal 1, ESP, (Transform 1, 3DES, ...)]
  - [Proposal 2, ESP, (Transform1, 3DES, ...), (Transform 2, DES, ...)]

Please note, that proposal 1 results in two SAs per direction.
ISAKMP – SA Negotiation (4)

- When responding to a security association payload, the responder must send a Security Association payload with the selected proposal, which may consist of multiple proposal payloads and their associated transform payloads.
- Each of the proposal payloads must contain a single transform payload associated with the protocol.
- The responder should retain the Proposal # field in the proposal payload and the Transform # field in each transform payload of the selected proposal.
  - Retention of proposal and transform numbers should speed the initiator's protocol processing by avoiding the need to compare the responder's selection with every offered option.
  - These values enable the initiator to perform the comparison directly and quickly.
- The initiator must verify that the SA payload received from the responder matches one of the proposals sent initially.
IKE – Introduction

- Whereas ISAKMP defines the basic data formats and procedures to negotiate arbitrary SAs, the *Internet Key Exchange* specifies the standardized protocol to negotiate IPSec SAs.

- IKE defines five exchanges:
  - Phase 1 exchanges for establishment of an IKE SA:
    - *Main mode exchange* which is realized by 6 exchanged messages
    - *Aggressive mode exchange* which needs only 3 messages
  - Phase 2 exchange for establishment of IPSec SAs:
    - *Quick mode exchange* which is realized with 3 messages
  - Other exchanges:
    - *Informational exchange* to communicate status and error messages
    - *New group exchange* to agree upon private Diffie-Hellman groups
IKE – Computation of IKE Session Keys

IKE establishes four different keys with an authentication exchange:

- **SKEYID** is a string derived from secret material known only to the active players in the exchange and it serves as a “master key”
  - The computation of SKEYID depends on the authentication method
- **SKEYID_d** is the keying material used to derive keys for non-IKE SAs
  - \( SKEYID_d = H(SKEYID, g^{xy}, CKY-I, CKY-R, 0) \)
    with \( g^{xy} \) denoting the shared Diffie-Hellman secret
- **SKEYID_a** is the keying material used by the IKE SA to authenticate its messages
  - \( SKEYID_a = H(SKEYID, SKEYID_d, g^{xy}, CKY-I, CKY-R, 1) \)
- **SKEYID_e** is the keying material used by the IKE SA to protect the confidentiality of its messages
  - \( SKEYID_e = H(SKEYID, SKEYID_a, g^{xy}, CKY-I, CKY-R, 2) \)

If required, keys are expanded by the following method:

- \( K = (K_1, K_2, ...) \) with \( K_i = H(SKEYID, K_{i-1}) \) and \( K_0 = 0 \)
IKE – Authentication Methods

- Phase 1 IKE exchanges are authenticated with the help of two hash values Hash-I and Hash-R, created by the initiator and responder:
  - Hash-I = H(SKEYID, g^x, g^y, CKY-I, CKY-R, SA-offer, ID-I)
  - Hash-R = H(SKEYID, g^y, g^x, CKY-R, CKY-I, SA-offer, ID-R)

  where g^x, g^y denote the exchanged public Diffie-Hellman values
  ID-I, ID-R denote the identity of the initiator and the responder
  SA-offer denotes the payloads concerning SA negotiation

- IKE supports four different methods of authentication:
  - Pre-shared key:
    - SKEYID = H(K_{Initiator,Responder}, r_{Initiator}, r_{Responder})
  - Two different forms of authentication with public-key encryption:
    - SKEYID = H(H(r_{Initiator}, r_{Responder}), CKY-I, CKY-R)
  - Digital Signature:
    - SKEYID = H(r_{Initiator}, r_{Responder}, g^{xy})
    - As in this case SKEYID itself provides no authentication, the values
      Hash-I and Hash-R are signed by the initiator / responder
The following descriptions list the exchanged ISAKMP- and IKE-payloads when performing different “flavors” of IKE authentication:

- **Initiator**
  - $Header, SA$
  - $Header, KE, N_i$
  - $Header, \{ID_i, Hash-I\}_{SKEYID_e}$

- **Responder**
  - $Header, SA$
  - $Header, KE, N_r$
  - $Header, \{ID_r, Hash-R\}_{SKEYID_e}$

where: $N_i, N_r$ denote $r_{\text{Initiator}}, r_{\text{Responder}}$ (IKE notation)
$ID_i, ID_r$ denote the identity of the initiator and the responder
$KE$ denotes the public values of a DH-exchange

- Please note that $Hash-I$ and $Hash-R$ need not to be signed, as they already “contain an authentic secret” (pre-shared key)
IKE – Main Mode Exchange with Signatures

- **Initiator**
  - `Header, SA`
  - `Header, KE, Ni (, CertReq)`
  - `Header, \{ID_i, (Cert_i, ) I[Hash-I]\}_{SKEYID_e}`

- **Responder**
  - `Header, SA`
  - `Header, KE, Nr (, CertReq)`
  - `Header, \{ID_r, (Cert_R, ) R[Hash-R]\}_{SKEYID_e}`

- where: `(m)` denotes that `m` is optional
  - `I[m]` denotes that `I` signs `m`

- Please note that `Hash-I` and `Hash-R` need to be signed, as they do not contain anything which is known to be authentic
IKE – Main Mode Exchange with Public Key Encryption

Initiator

Header, SA

Header, KE, \{ID\}_+K_R, \{N\}_+K_R

Header, \{Hash-I\}_{SKEYID_e}

Responder

Header, SA

Header, KE, \{ID\}_+K_I, \{N\}_+K_I

Header, \{Hash-R\}_{SKEYID_e}

- where: \{m\}_+K_I denotes that \( m \) is encrypted with the public key +K_I

- Please note that Hash-I and Hash-R need not to be signed, as they “contain” the exchanged random numbers \( N_I \) or \( N_R \), respectively
  - So, every entity proves his authenticity by decrypting the received random number (\( N_I \) or \( N_R \)) with its’ private key
IKE – Main Mode Exchange with Public Key Encryption 2

Initiator

Header, SA

Header, \( \{N_i\}^{+K_R} \), \( \{KE\}_{K_i} \), \( \{ID_i\}_{K_i} \), \( \{Cert_i\}_{K_i} \)

Header, \( \{Hash-I\}_{SKEYID_e} \)

Responder

Header, SA

Header, \( \{N_r\}^{+K_i} \), \( \{KE\}_{K_r} \), \( \{ID_r\}_{K_r} \), \( \{Cert_r\}_{K_r} \)

Header, \( \{Hash-R\}_{SKEYID_e} \)

where:

- \( \{m\}^{+K_i} \) denotes that \( m \) is encrypted with the public key \( +K_i \)
- \( \{m\}_{K_i} \) denotes that \( m \) is encrypted with the symmetric key \( K_i \) with \( K_i = H(N_i, CKY-I) \)
- \( K_r = H(N_r, CKY-R) \)

- Please note that all schemes described so far provide protection of identity against eavesdroppers in the Internet, as the IDs and certificates are not send in the clear:
  - However, the IP addresses of exchanged packets are always readable...
IKE – Aggressive Mode Exchange with Pre-Shared Key

As the identity of the initiator and the responder have to be send before a session key can be established, the aggressive mode exchange can not provide identity protection against eavesdroppers.

There are similar aggressive mode variants for authentication with:
- Digital signature
- Public key encryption
Further Issues with IPSec

- **Compression:**
  - If encryption is used, then the resulting IP packets cannot be compressed in the link layer, e.g., when connecting to an ISP via Modem.
  - Therefore, the *IP payload compression protocol (PCP)* has been defined.
  - PCP can be used with IPSec:
    - IPSec policy definition allows to specify PCP.
    - IKE SA negotiation allows to include PCP in proposals.

- **Interoperability problems of end-to-end security with header processing in intermediate nodes:**
  - Interoperability with firewalls:
    - End-to-end encryption conflicts with the firewalls’ need to inspect upper layers protocol headers in IP packets.
  - Interoperability with network address translation (NAT):
    - Encrypted packets do not permit analysis nor change of addresses.
    - Authenticated packets will be discarded if source or destination address is changed.
Conclusion

- IPSec is IETF’s security architecture for the Internet Protocol
- It provides the following security services to IP packets:
  - Data origin authentication
  - Replay protection
  - Confidentiality
- It can be realized in end systems or intermediate systems:
  - End system implementation: OS integrated or “bump in the stack”
  - Gateway implementation: Router integrated or “bump in the wire”
- Two fundamental security protocols have been defined:
  - Authentication header (AH)
  - Encapsulating security payload (ESP)
- SA negotiation and key management is realized with:
  - Internet security association key management protocol (ISAKMP)
  - Internet key exchange (IKE)
**Additional References**


