IPSec
VPN:

IPSec
**IPSec**

**Layer 3.5 implementation:** applications do not have to be changed to use it - all applications automatically use it whether they like it or not.

<table>
<thead>
<tr>
<th>User</th>
<th>Applications</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>TCP</td>
</tr>
<tr>
<td></td>
<td>IPSec</td>
</tr>
<tr>
<td></td>
<td>IP</td>
</tr>
<tr>
<td></td>
<td>...</td>
</tr>
</tbody>
</table>

But the OS needs to be modified IPSec capable of authenticating but can only tell app the address of source

Operates like a firewall
- encrypts
- has policies
- authenticates address to app
**IPSec**

**Encapsulating Security Payload (ESP):**
- Provides integrity and/or encryption
- Only protects information after the IP header (HMAC for ESP header and payload)
- Typical encryption algorithms used: DES, 3-DES, Blowfish, AES

Nothing in the packet says it's encrypted
- firewalls may get stuck if they need to check ports, e.g., because they cannot reliably get that info
- only the sender and receiver know whether the ESP packets are encrypted

**Authenticating Header (AH):**
- Message integrity protection only (Message Authentication)
- IPSec computes a HMAC checksum over nearly all the fields of the IP packet, and stores it in a new AH header
IPSec

Transport Mode:
1. IPSec info between IP header and rest of packet
2. Applied end-to-end, authentication, encryption, or both
IPSec

Transport Mode (AH):

Original IPv4 Datagram:
- ver
- hlen
- TOS
- pkt len
- ID
- flgs
- frag offset
- TTL
- proto=TCP
- header cksum
- src IP address
- dst IP address
- TCP header (proto = 6)
- TCP payload

Protected by AH Auth Data

New IPv4 Datagram:
- ver
- hlen
- TOS
- pkt len + AH size
- ID
- flgs
- frag offset
- TTL
- proto=AH
- header cksum
- src IP address
- dst IP address
- next=TCP
- AH len
- Reserved
- SPI (Security Parameters Index)
- Sequence Number
- Authentication Data (usually MD5 or SHA-1 hash)
- TCP header (proto = 6)
- TCP payload
**IPSec**

**Transport Mode (ESP):**

<table>
<thead>
<tr>
<th>Original IPv4 Datagram</th>
<th>New IPv4 Datagram</th>
</tr>
</thead>
<tbody>
<tr>
<td>ver</td>
<td>hlen</td>
</tr>
<tr>
<td>ID</td>
<td>hlen</td>
</tr>
<tr>
<td>flgs</td>
<td>frag offset</td>
</tr>
<tr>
<td>TTL</td>
<td>proto=TCP</td>
</tr>
<tr>
<td>src IP address</td>
<td></td>
</tr>
<tr>
<td>dst IP address</td>
<td></td>
</tr>
<tr>
<td>TCP header (proto = 6)</td>
<td></td>
</tr>
<tr>
<td>TCP payload</td>
<td></td>
</tr>
<tr>
<td>Encrypted Data</td>
<td></td>
</tr>
<tr>
<td>Authenticated Data</td>
<td></td>
</tr>
</tbody>
</table>

ESP + Payload (variable)
**IPSec**

**Transport Mode:**
1. IPSec info between IP header and rest of packet
2. Applied end-to-end, authentication, encryption, or both

**Tunnel Mode:**
1. Keep original IP packet intact, add new IP header and IPSec information (AH or ESP)
2. Firewall-to-firewall, end-to-firewall, encrypt header & payload
IPSec

Tunnel Mode (AH):

Original IPv4 Datagram

<table>
<thead>
<tr>
<th>ver</th>
<th>hlen</th>
<th>TOS</th>
<th>pkt len</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ID</td>
<td>flgs</td>
<td>frag offset</td>
<td></td>
</tr>
<tr>
<td>TTL</td>
<td>proto=TCP</td>
<td>header cksum</td>
<td></td>
</tr>
<tr>
<td>src IP address</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>dst IP address</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

TCP header (proto = 6)

TCP payload

Protected by AH Auth Data

New IPv4 Datagram

<table>
<thead>
<tr>
<th>ver</th>
<th>hlen</th>
<th>TOS</th>
<th>pkt len + AH + IP</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ID</td>
<td>flgs</td>
<td>frag offset</td>
<td></td>
</tr>
<tr>
<td>TTL</td>
<td>proto=AH</td>
<td>header cksum</td>
<td></td>
</tr>
<tr>
<td>src IP address</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>dst IP address</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

next=IP | AH len | Reserved |
SPI (Security Parameters Index) |
Sequence Number |
Authentication Data (usually MD5 or SHA-1 hash)

<table>
<thead>
<tr>
<th>ver</th>
<th>hlen</th>
<th>TOS</th>
<th>pkt len</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ID</td>
<td>flgs</td>
<td>frag offset</td>
<td></td>
</tr>
<tr>
<td>TTL</td>
<td>proto=TCP</td>
<td>header cksum</td>
<td></td>
</tr>
<tr>
<td>src IP address</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>dst IP address</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

TCP header (proto = 6)

TCP payload
Tunnel Mode (ESP):

Original IPv4 Datagram

- ver
- hlen
- TOS
- pkt len
- ID
- flgs
- frag offset
- TTL

proto=TCP
- header checksum
- src IP address
- dst IP address
- TCP header (proto = 6)

TCP payload

- Encrypted Data
- Authenticated Data

New IPv4 Datagram

- ver
- hlen
- TOS
- pkt len
- ID
- flgs
- frag offset
- TTL

proto=ESP
- header checksum
- src IP address
- dst IP address
- SPI (Security Parameters Index)
- Sequence Number

ESP

- IP Header
- TCP Header + Payload
- Padding (variable)
- pad len
- next=IP
- Authentication Data (optional)
IPSec

Security Association:
• A cryptographically protected connection
• Each end has ≥ one key, sequence number, identity of other end
• Each end has crypto services used:
  integrity only, encryption+integrity, crypto algorithms
• Unidirectional - two connections needed for 2-way operation
• Details of an SA are kept in a database
• IPSec header has a Security Parameter Index (SPI) field that identifies the SA allowing the sender to look up necessary info in the sender's SA database.
• SPI value is chosen by the receiver.
• An SA is defined by an SPI and destination address
  (if receiver is involved in a multicast, it may not have chosen the SA – the destination address is a group address in the case of multicast)
**IPSec**

**Internet Key Exchange:**
Protocol for doing mutual authentication and establishing a shared secret key to create an IPSec SA.

**Uses:**
- long term keys (public signature-only keys, pre-shared secret keys, public encryption keys)

**Pieces:**
- ISAKMP (Internet Security Association and Key Management Protocol) framework (OAKLEY implementation)
- IKE (Internet Key Exchange) defines fields, chooses options of ISAKMP
- DOI (Domain of Interpretation) specifies particular use of ISAKMP
IPSec

Internet Key Exchange Phases:

**Phase 1:** Does mutual authentication and establishes session keys based on identities such as names, and secrets

**Phase 2:** SAs are established between two entities

**Reason for two phases:**

Different SAs may be established for different traffic flows. There might be security weaknesses if two different flows used the same key.

Phase 1 need be done once, phase 2 uses the same phase 1 session key to generate multiple SAs.

Phase 1 relies on (slower) public-key cryptography, phase 2 relies on (faster) secret keys – hence session-keys are made quickly
Possible Security Problem: (encryption w/o integrity)

C can decrypt packet sent by A to B
- Record packet from A to B and packet from C to D
- Splice encrypted part with src-dst from C to D onto A to B packet
- Forward packet to Firewall 2, Firewall decrypts, sends result to D
IPSec

Internet Key Exchange Phase 1:

Aggressive Mode: Accomplishes mutual authentication in three messages
**IPSec**

**Internet Key Exchange Phase 1:**

**Aggressive Mode:** Accomplishes mutual authentication in three messages

\[ g^a \mod p, \text{ ID, crypto prop.} \]

**Diffie-Hellman Exchange**
Internet Key Exchange Phase 1:

Aggressive Mode: Accomplishes mutual authentication in three messages

Diffie-Hellman Exchange

proof (of ID) might be a signature on some client data
**IPSec**

**Internet Key Exchange Phase 1:**

**Aggressive Mode:** Accomplishes mutual authentication in three messages

![Diagram](image)
IPSec

Internet Key Exchange Phase 1:

Aggressive Mode Problems:

1. Someone other than Server can send a refusal back to Client and Client cannot tell if it is fake (such a message should be sent encrypted).

2. Identities of the client and server are revealed
Internet Key Exchange Phase 1:

**Main Mode:** Accomplishes mutual authentication in six msgs. Includes ability to hide end-point identifiers from eavesdroppers and flexibility in negotiating crypto algorithms.

Parameter negotiation
IPSec

Internet Key Exchange Phase 1:

Main Mode: Accomplishes mutual authentication in six msgs. Includes ability to hide end-point identifiers from eavesdroppers and flexibility in negotiating crypto algorithms.

Parameter negotiation
**IPSec**

**Internet Key Exchange Phase 1:**

**Main Mode:** Accomplishes mutual authentication in six msgs. Includes ability to hide end-point identifiers from eavesdroppers and flexibility in negotiating crypto algorithms.

![Diffie-Hellman exchange diagram](attachment:image.png)

\[ g^a \mod p, \text{ non}1 \]
IPSec

Internet Key Exchange Phase 1:

Main Mode: Accomplishes mutual authentication in six msgs. Includes ability to hide end-point identifiers from eavesdroppers and flexibility in negotiating crypto algorithms.

Diffie-Hellman exchange

Client

\[ g^b \mod p, \text{non2} \]

Server
**IPSec**

**Internet Key Exchange Phase 1:**

**Main Mode:** Accomplishes mutual authentication in six msgs. Includes ability to hide end-point identifiers from eavesdroppers and flexibility in negotiating crypto algorithms.

K\{ID, proof of ID,[cert]\}

\[ K = f(g^{ab} \mod p, \text{non1}, \text{non2}) \]

authenticate, encrypted
nonces allow same Diffie-Hellman private value for many transactions
proof of ID: signature on a hash of ID, DH values, nonces, crypto choices
**IPSec**

**Internet Key Exchange Phase 1:**

**Main Mode:** Accomplishes mutual authentication in six msgs. Includes ability to hide end-point identifiers from eavesdroppers and flexibility in negotiating crypto algorithms.
Internet Key Exchange Phase 1:

Proof of Identity: Some hash of the key associated with the identity, the Diffie-Hellman values, nonces, cryptographic choices, and cookies.

Problem: choice of cryptographic suite by server is not encrypted. A man-in-the-middle might actually replace a good choice with a poor (crackable) choice then decrypt and impersonate server from then on.

Stateless cookies? No, must remember crypto proposals - they are included in the hash used in the proof of identity - sender could repeat this info later in the handshake, but IPSec does not allow this

Duplicate connection identifiers? Possible to have two connections with the same crypto parameters
Internet Key Exchange Phase 1:

Regarding key types:
Phase 1 might be based on a public-key pair for encryption, public-key pair for signing, or pre-shared secret.
A more recent possibility is encrypting a randomly chosen secret key with the other side's public key and using that to encrypt all remaining fields.
Hence there are 8 choices for Phase 1 handshakes:
4 authentication methods × (main + aggressive modes)
IPSec

Internet Key Exchange Phase 1:

Crypto Parameters:

1. Encryption algorithm (DES, 3DES, IDEA)
2. Hash algorithm (MD5, SHA)
3. Authentication method (RSA signature, DSS...)
4. Diffie-Hellman group ((g,p), elliptic curves)

Any combination is allowed – could result in more than 100 crypto suite choices in the proposal.

Optionally, an expiration time/date can be specified in the proposal.
IPSec

Internet Key Exchange Phase 1:

Certificates: Client nor Server can ask the other side for a certificate. If they do not know the other side's public key they cannot use the protocol. If certificates are sent in first two messages then identities are revealed.
IPSec
Internet Key Exchange Phase 1:

Main Mode Revised: requires a single private key operation on either side.

Parameter negotiation
Starts out as before
Internet Key Exchange Phase 1:
Main Mode Revised:

Parameter negotiation
No change yet
IPSec

Internet Key Exchange Phase 1: Main Mode Revised:

\[ K_1 = \text{hash}(\text{non1, cookie1}) \]

- \[ K_1\{g^a \mod p\}, \]
- \[ K_1\{\text{ID}\}, K_1\{\text{[certificate]}\}, \]
- \[ \text{ServerPublicKey}\{\text{non1}\} \]

Diffie-Hellman exchange
Server uses private key to decrypt non1 then determines K1 then decrypts ID, and everything else
Internet Key Exchange Phase 1: Main Mode Revised:

\[ K_2 = \text{hash}(\text{non2}, \text{cookie2}) \]

\[ K_2\{ g^b \mod p \}, \]

\[ K_2\{ \text{ID} \}, \]

\[ \text{ClientPublicKey}\{ \text{non2} \} \]

Diffie-Hellman exchange
IPSec
Internet Key Exchange Phase 1:
Main Mode Revised:

Client

\[ K = f(g^{ab} \mod p, \text{nons, cooks}) \]

K\{proof of ID\}

authenticate, encrypted

Server
Internet Key Exchange Phase 1:

**Shared Secret Main Mode:** Only required protocol. Requires Client and Server to already share a secret - intended for laptops trying to get into a firewall at work while on the road.

Parameter negotiation
IPSec

Internet Key Exchange Phase 1:

Shared Secret Main Mode: Only required protocol. Requires Client and Server to already share a secret - intended for laptops trying to get into a firewall at work while on the road?

Parameter negotiation
**IPSec**

**Internet Key Exchange Phase 1:**

**Shared Secret Main Mode:** Only required protocol. Requires Client and Server to already share a secret - intended for laptops trying to get into a firewall at work while on the road?

\[ g^a \mod p, \text{ non} 1 \]

**Diffie-Hellman**
IPSec

Internet Key Exchange Phase 1:

**Shared Secret Main Mode:** Only required protocol. Requires Client and Server to already share a secret - intended for laptops trying to get into a firewall at work while on the road?

\[
g^b \text{ mod } p, \text{ non2}
\]

Diffie-Hellman
IPSec

Internet Key Exchange Phase 1:

**Shared Secret Main Mode:** Only required protocol. Requires Client and Server to already share a secret - intended for laptops trying to get into a firewall at work while on the road?

\[
K = f(J, g^{ab} \mod p, \text{nons, coks})
\]

\[
K \{ \text{ID, proof(ID)} \}
\]
Internet Key Exchange Phase 1:

**Shared Secret Main Mode:** Only required protocol. Requires Client and Server to already share a secret - intended for laptops trying to get into a firewall at work while on the road?

**IPSec**

Client

![Diagram showing shared secret and authentication process]  

Server

- $K = f(J, g^{ab} \mod p, \text{nons, coks})$
- $K\{\text{ID, proof(ID)}\}$

authentication
IPSec

Internet Key Exchange Phase 1:

Problems:

1. Identities *must* be IP addresses - so cannot seriously be used in road warrior application

2. Client sends identity in message 5 encrypted with key K which is a function of shared secret J. Server cannot decrypt that message to find out who the Client is unless it knows J. But that means Server must know who the Client is in the first place! This is why identities are IP addresses.
Internet Key Exchange Phase 1:

Problems:

1. Identities must be IP addresses - so cannot seriously be used in road warrior application

2. Client sends identity in message 5 encrypted with key K which is a function of shared secret J. Server cannot decrypt that message to find out who the Client is unless it knows J. But that means Server must know who the Client is in the first place! This is why identities are IP addresses.

Fix:

Do not make K a function of J. OK since J is included in the hash which is proof of identity.
**IPSec**

**Internet Key Exchange Phase 1:**

Negotiating Cryptographic Parameters:
- encryption algorithm: DES, 3DES, IDEA
- hash: MD5, SHA
- authentication method: pre-shared-keys, RSA signing, RSA encryption, DSS
- Diffie-Hellman type: p, g

Session Keys:
- Two established: integrity, encryption for protecting the last phase 1 transaction (both directions) and all the phase 2 transactions

Keys are: hashes of Diffie-Hellman values, nonces, cookies,..
Internet Key Exchange Phase 2:

Setting up IPSec SAs: All messages are encrypted with Phase 1 SA's encryption key $K_1$ and integrity protected with phase 1 SA's integrity key $K_2$.

\[ x, y, c_p, \text{traffic}, \text{SPI-1}, \text{nonce}_1, [g^a \text{ mod } p] \]

- $X$ is a pair of cookies from phase 1
- $Y$ is a 32 bit number chosen to distinguish this setup from others that may be setup simultaneously in phase 1.
- $X$ and $Y$ are unencrypted. (vulnerable to replay)
Internet Key Exchange Phase 2:

Setting up IPSec SAs: All messages are encrypted with Phase 1 SA's encryption key K1 and integrity protected with phase 1 SA's integrity key K2.

X, Y, CP, traffic, SPI-1, nonce1, [g^a mod p]

Rest of message: authenticator using K2, crypto parameters, optional Diffie-Hellman values, optionally a description of traffic.
Internet Key Exchange Phase 2:

Setting up IPSec SAs: All messages are encrypted with Phase 1 SA's encryption key $K_1$ and integrity protected with phase 1 SA's integrity key $K_2$.

Encryption: Initialization vector is the final ciphertext block of last message of phase 1 hashed with $Y$. 
Internet Key Exchange Phase 2:

Setting up IPSec SAs:

Encryption: Initialization vector is the final ciphertext block of last message of previous phase 2 message hashed with Y.
**IPSec**

**Internet Key Exchange Phase 2:**

Setting up IPSec SAs:

- **Phase 1 SA**
  - X,Y,ack

Encryption: Initialization vector is the final ciphertext block of last message of previous phase 2 message hashed with Y.
IPSec

Internet Key Exchange Phase 2:

Problems:

1. It is vulnerable to replay:
   a. Since Y is "random" instead of based on a sequence #, to detect a replay attack one must remember all Y's generated.
   b. Headers and session keys are the same in both directions so attacker can replay easily.

2. Vulnerable to reflection attack.
IPSec

Internet Key Exchange Phase 2:

Problems:

1. It is vulnerable to replay:
   a. Since Y is "random" instead of based on a sequence #, to detect a replay attack one must remember all Y's generated.
   b. Headers and session keys are the same in both directions so attacker can replay easily.

2. Vulnerable to reflection attack.

Fix:

Use different keys in different directions or by reversing the order of the cookies so the recipient's cookie appears first. Use sequence numbers instead of message IDs.