SSL and IPSec

CS461/ECE422

Fall 2009

Reading

- Chapter 11 in Computer Science: Art and Science
- Stallings book
- Can also look at Standards Documents

SSL

- Transport layer security
  - Provides confidentiality, integrity, authentication of endpoints
  - Developed by Netscape for WWW browsers and servers
- Internet protocol version: TLS
  - Compatible with SSL
  - Standard rfc2712

Working at Transport Level

- Data link, Network, and Transport headers sent unchanged
- Original transport header can be protected if tunneling

Ethernet Frame Header	IP Header	TCP Header	TCP data stream Encrypted/authenticated Regardless of application

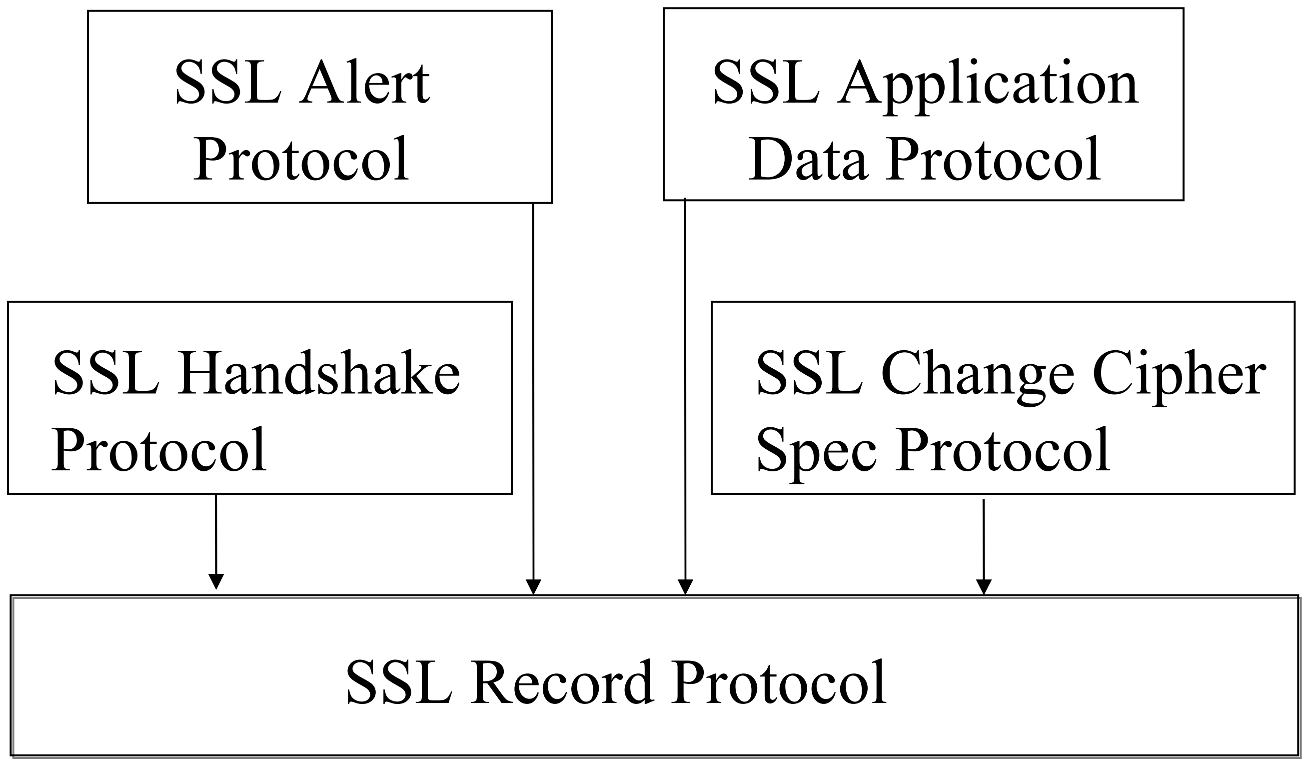
SSL Session

- Association between two peers
  - May have many associated connections
  - Information for each association:
    - Unique session identifier
    - Peer's X.509v3 certificate, if needed
    - Compression method
    - Cipher spec for cipher and MAC
    - "Master secret" shared with peer
      - 48 bits

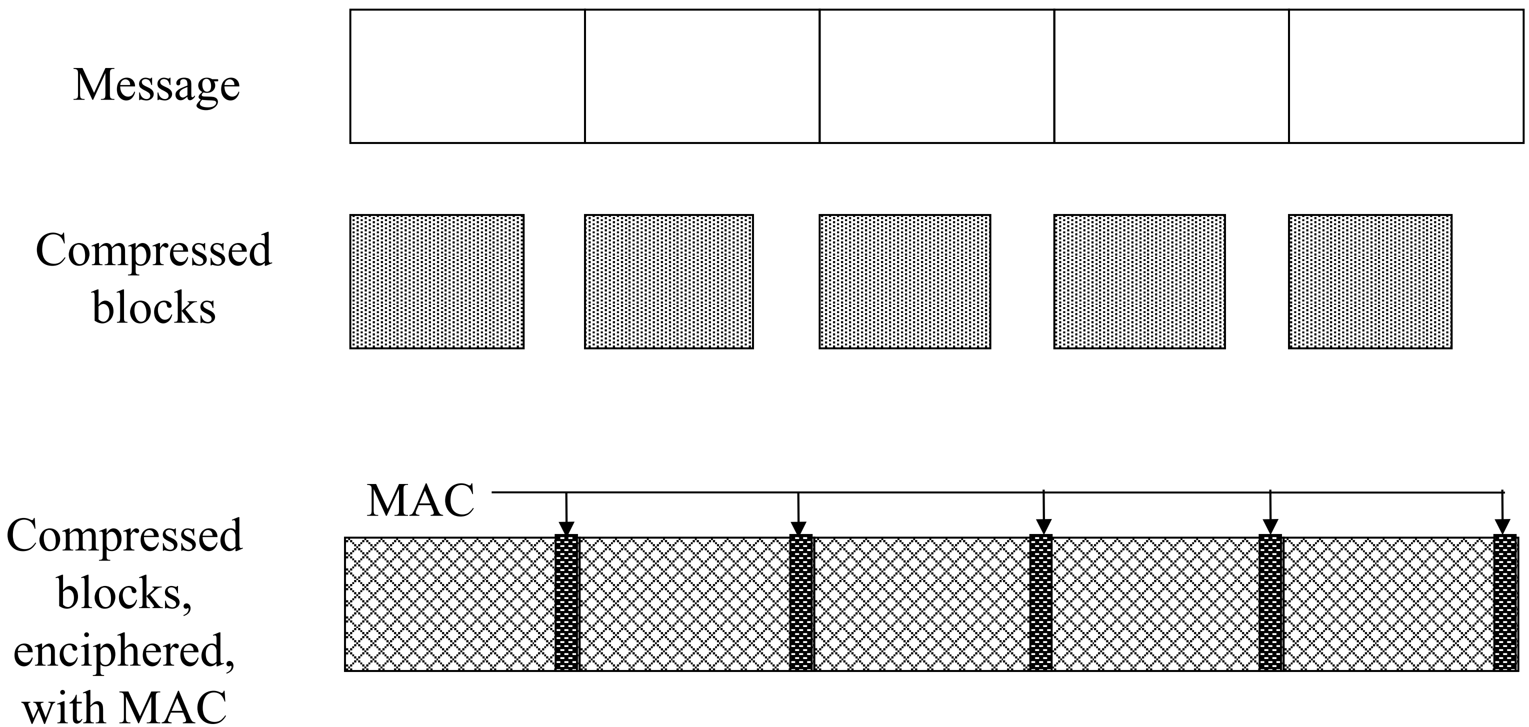
SSL Connection

- Describes how data exchanged with peer
- Information for each connection
  - Random data
  - Write keys (used to encipher data)
  - Write MAC key (used to compute MAC)
  - Initialization vectors for ciphers, if needed
  - Sequence numbers

Structure of SSL



SSL Record Layer



Record Protocol Overview

- Lowest layer, taking messages from higher
  - Max block size 16,384 bytes
  - Bigger messages split into multiple blocks
- Construction
  - Block b compressed; call it b_c
  - MAC computed for b_c
    - If MAC key not selected, no MAC computed
  - b_c, MAC enciphered
    - If enciphering key not selected, no enciphering done
  - SSL record header prepended

SSL MAC Computation

- Symbols
  - *h* hash function (MD5 or SHA)
  - k_w write MAC key of entity
  - *ipad* = 0x36, *opad* = 0x5C
    - Repeated to block length (from HMAC)
  - *seq* sequence number
  - *SSL_comp* message type
  - *SSL_len* block length
- MAC

$$h(k_w || opad || h(k_w || ipad || seq || SSL\_comp || SSL\_len || block))$$

SSL Handshake Protocol

- Used to initiate connection
  - Sets up parameters for record protocol
  - 4 rounds
- Upper layer protocol
  - Invokes Record Protocol
- Note: what follows assumes client, server using RSA as interchange cryptosystem

Overview of Rounds

- Create SSL connection between client, server
- Server authenticates itself
- Client validates server, begins key exchange
- Acknowledgments all around

Handshake Round 1

$$\{ v_C \,\|\, r_1 \,\|\, s_1 \,\|\, ciphers \,\|\, comps \}$$

Client → Server

$$\{ v \,\|\, r_2 \,\|\, s_1 \,\|\, cipher \,\|\, comp \}$$

Client ← Server

v_C	Client's version of SSL
v	Highest version of SSL that Client, Server both understand
r_1, r_2	nonces (timestamp and 28 random bytes)
s_1	Current session id (0 if new session)
ciphers	Ciphers that client understands
comps	Compression algorithms that client understand
cipher	Cipher to be used
comp	Compression algorithm to be used

Handshake Round 2

Client ← $\{certificate\}$ ← Server

Client ← $\{mod \,\|\, exp \,\|\, \mathrm{Sig}_S(h(r_1 \,\|\, r_2 \,\|\, mod \,\|\, exp))\}$ ← Server

Client ← $\{ctype \,\|\, gca\}$ ← Server

Client ← $\{er2\}$ ← Server

Note: if Server not to authenticate itself, only last message sent; third step omitted if Server does not need Client certificate

k_S Server's private key

ctype Certificate type requested (by cryptosystem)

gca Acceptable certification authorities

er2 End round 2 message

Handshake Round 3

Client $\xrightarrow{\{\ client\_cert\ \}}$ Server

Client $\xrightarrow{\{\ pre\ \}\text{Pub}_S}$ Server

Both Client, Server compute master secret *master*:

$$master = \text{MD5}(pre \parallel \text{SHA}(\text{'A'} \parallel pre \parallel r_1 \parallel r_2) \parallel$$
$$\text{MD5}(pre \parallel \text{SHA}(\text{'BB'} \parallel pre \parallel r_1 \parallel r_2) \parallel$$
$$\text{MD5}(pre \parallel \text{SHA}(\text{'CCC'} \parallel pre \parallel r_1 \parallel r_2)$$

Client $\xrightarrow{\{\ h(master \parallel opad \parallel h(msgs \parallel master \mid ipad))\ \}}$ Server

msgs	Concatenation of previous messages sent/received this handshake
opad, *ipad*	As above

Handshake Round 4

Client sends "change cipher spec" message using that protocol

Client → Server

$\{ h(master \,||\, opad \,||\, h(msgs \,||\, \mathrm{0x434C4E54} \,||\, master \,||\, ipad)) \}$

Client → Server

Server sends "change cipher spec" message using that protocol

Client ← Server

$\{ h(master \,||\, opad \,||\, h(msgs \,||\, 0x53525652 \,||\, master \,|\, ipad)) \}$

Client ← Server

msgs Concatenation of messages sent/received this handshake in *previous* rounds (does notinclude these messages)

opad, *ipad*, *master* As above

Supporting Crypto

- All parts of SSL use them
- Initial phase: public key system exchanges keys
  - Classical ciphers ensure confidentiality, cryptographic checksums added for integrity
  - Only certain combinations allowed
    - Depends on algorithm for interchange cipher
  - Interchange algorithms: RSA, Diffie-Hellman, Fortezza
  - AES added in 2002 by rfc3268

RSA: Cipher, MAC Algorithms

Interchange cipher	*Classical cipher*	*MAC Algorithm*
RSA, key ≤ 512 bits	*none*	MD5, SHA
	RC4, 40-bit key	MD5
	RC2, 40-bit key, CBC mode	MD5
	DES, 40-bit key, CBC mode	SHA
RSA	*None*	MD5, SHA
	RC4, 128-bit key	MD5, SHA
	IDEA, CBC mode	SHA
	DES, CBC mode	SHA
	DES, EDE mode, CBC mode	SHA

Diffie-Hellman: Types

- Diffie-Hellman: certificate contains D-H parameters, signed by a CA
  - DSS or RSA algorithms used to sign
- Ephemeral Diffie-Hellman: DSS or RSA certificate used to sign D-H parameters
  - Parameters not reused, so not in certificate
- Anonymous Diffie-Hellman: D-H with neither party authenticated
  - Use is "strongly discouraged" as it is vulnerable to attacks

D-H: Cipher, MAC Algorithms

Interchange cipher	*Classical cipher*	*MAC Algorithm*
Diffie-Hellman, DSS Certificate	DES, 40-bit key, CBC mode	SHA
	DES, CBC mode	SHA
	DES, EDE mode, CBC mode	SHA
Diffie-Hellman, key ≤ 512 bits RSA Certificate	DES, 40-bit key, CBC mode	SHA
	DES, CBC mode	SHA
	DES, EDE mode, CBC mode	SHA

Ephemeral D-H: Cipher, MAC Algorithms

Interchange cipher	*Classical cipher*	*MAC Algorithm*
Ephemeral Diffie-Hellman, DSS Certificate	DES, 40-bit key, CBC mode	SHA
	DES, CBC mode	SHA
	DES, EDE mode, CBC mode	SHA
Ephemeral Diffie-Hellman, key ≤ 512 bits, RSA Certificate	DES, 40-bit key, CBC mode	SHA
	DES, CBC mode	SHA
	DES, EDE mode, CBC mode	SHA

Anonymous D-H: Cipher, MAC Algorithms

Interchange cipher	*Classical cipher*	*MAC Algorithm*
Anonymous D-H, DSS Certificate	RC4, 40-bit key	MD5
	RC4, 128-bit key	MD5
	DES, 40-bit key, CBC mode	SHA
	DES, CBC mode	SHA
	DES, EDE mode, CBC mode	SHA

Fortezza: Cipher, MAC Algorithms

Interchange cipher	*Classical cipher*	*MAC Algorithm*
Fortezza key exchange	*none*	SHA
	RC4, 128-bit key	MD5
	Fortezza, CBC mode	SHA

SSL Change Cipher Spec Protocol

- Send single byte
- In handshake, new parameters considered "pending" until this byte received
  - Old parameters in use, so cannot just switch to new ones

SSL Alert Protocol

- Closure alert
  - Sender will send no more messages
  - Pending data delivered; new messages ignored
- Error alerts
  - Warning: connection remains open
  - Fatal error: connection torn down as soon as sent or received

SSL Alert Protocol Errors

- Always fatal errors:
  - unexpected_message, bad_record_mac, decompression_failure, handshake_failure, illegal_parameter
- May be warnings or fatal errors:
  - no_certificate, bad_certificate, unsupported_certificate, certificate_revoked, certificate_expired, certificate_unknown

SSL Application Data Protocol

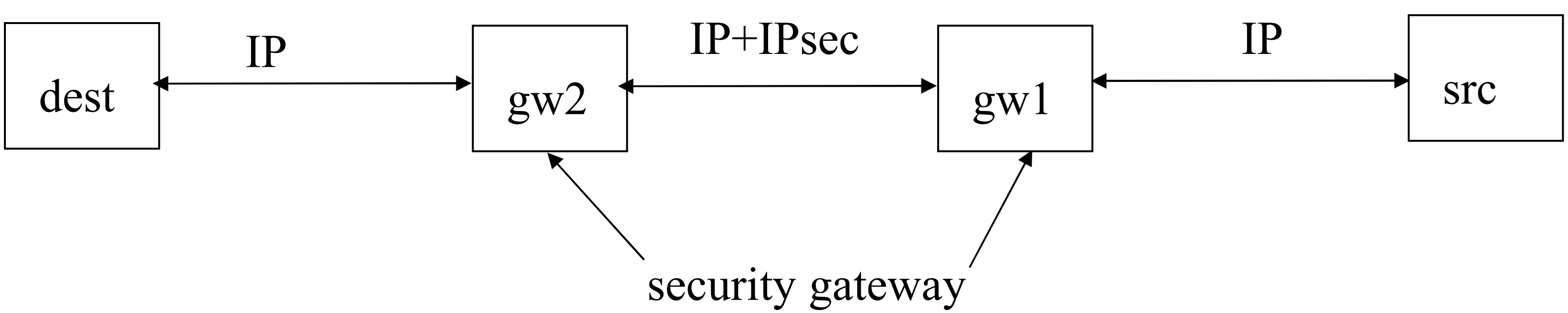
- Passes data from application to SSL Record Protocol layer

MITM Attacks

- Classic attack foiled by certificates
- More subtle attacks appear over time
  - TLS Authentication Gap
    - Interaction of TLS and HTTP
    - http://www.phonefactor.com/sslgap
- Application above SSL/TLS tends to be HTTP but does not have to be

IPsec

- Network layer security
  - Provides confidentiality, integrity, authentication of endpoints, replay detection
- Protects all messages sent along a path



Standards

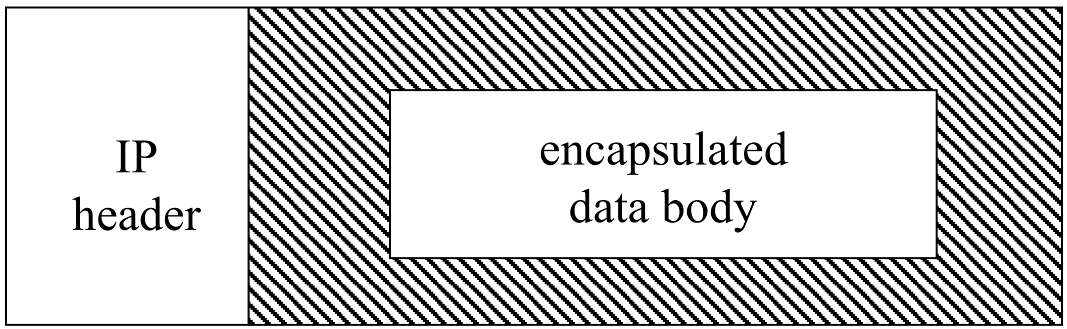
- Original RFC’s 2401-2412
- Mandatory portion of IPv6
- Bolted onto IPv4
- Newer standards
  - IKE: Standardized Key Management Protocol RFC 2409
  - NAT-T: UDP encapsulation for traversing address translation RFC 3948

Network Level Encryption

- Data link header and network header is unchanged
- With tunneling original IP header can be protected

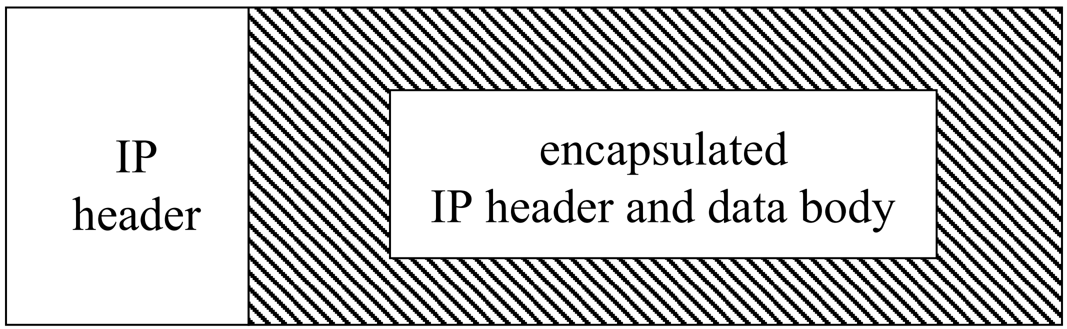
Ethernet Frame Header	IP Header	IP packet Encrypted/authenticated Regardless of application

IPsec Transport Mode



- Encapsulate IP packet data area
- Use IP to send IPsec-wrapped data packet
- Note: IP header not protected

IPsec Tunnel Mode



- Encapsulate IP packet (IP header *and* IP data)
- Use IP to send IPsec-wrapped packet
- Note: IP header protected

IPsec Protocols

- Authentication Header (AH)
  - Integrity of payload
  - Integrity of outer header
  - Anti-replay
- Encapsulating Security Payload (ESP)
  - Confidentiality of payload and inner header
  - Integrity of payload (and now header)

ESP and integrity

- Originally design, use AH to add integrity if needed.
- Bellovin showed integrity is always needed
  - So added directly to ESP
  - http://www.cs.columbia.edu/~smb/papers/

IPsec Architecture

- Security Policy Database (SPD)
  - Says how to handle messages (discard them, add security services, forward message unchanged)
  - SPD associated with network interface
  - SPD determines appropriate entry from packet attributes
    - Including source, destination, transport protocol

Example

- Goals
  - Discard SMTP packets from host 192.168.2.9
  - Forward packets from 192.168.19.7 without change
- SPD entries

```
src 192.168.2.9, dest 10.1.2.3 to 10.1.2.103, port 25, discard
src 192.168.19.7, dest 10.1.2.3 to 10.1.2.103, port 25, bypass
dest 10.1.2.3 to 10.1.2.103, port 25, apply IPsec
```

- Note: entries scanned in order
  - If no match for packet, it is discarded

IPsec Architecture

- Security Association (SA)
  - Association between peers for security services
    - Identified uniquely by dest address, security protocol (AH or ESP), unique 32-bit number (security parameter index, or SPI)
  - Unidirectional
    - Can apply different services in either direction
  - SA uses either ESP or AH; if both required, 2 SAs needed

SA Database (SAD)

- Entry describes SA; some fields for all packets:
  - AH algorithm identifier, keys
    - When SA uses AH
  - ESP encipherment algorithm identifier, keys
    - When SA uses confidentiality from ESP
  - ESP authentication algorithm identifier, keys
    - When SA uses authentication, integrity from ESP
  - SA lifetime (time for deletion or max byte count)
  - IPsec mode (tunnel, transport, either)

SAD Fields

- Antireplay (inbound only)
  - When SA uses antireplay feature
- Sequence number counter (outbound only)
  - Generates AH or ESP sequence number
- Sequence counter overflow field
  - Stops traffic over this SA if sequence counter overflows
- Aging variables
  - Used to detect time-outs

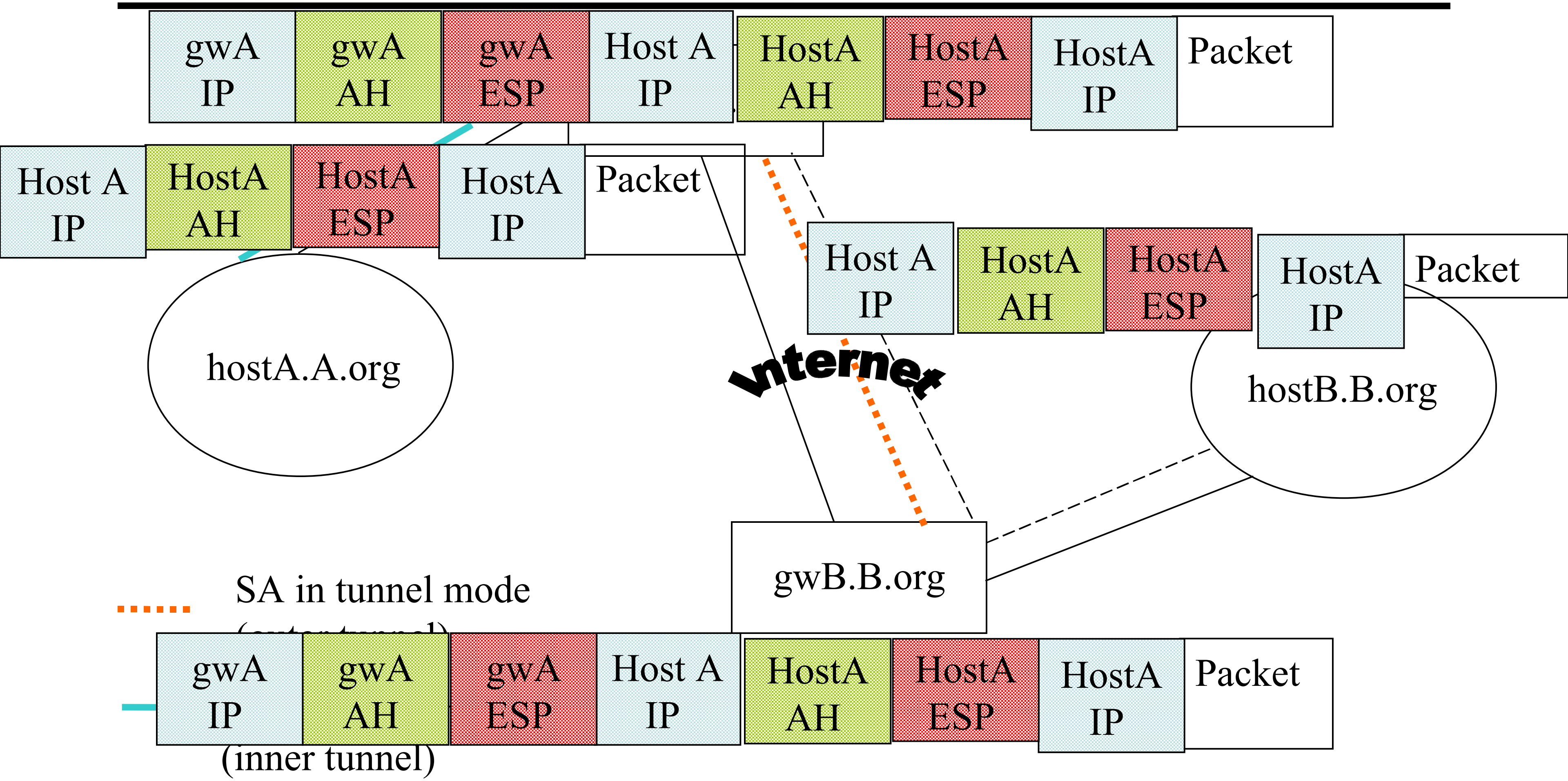
IPsec Architecture

- Packet arrives
- Look for existing SA
- Otherwise look in SPD
  - Find appropriate entry
  - Get dest address, security protocol, SPI
- Find associated SA in SAD
  - Use dest address, security protocol, SPI
  - Apply security services in SA (if any)

Example: Nested Tunnels

- Group in A.org needs to communicate with group in B.org
- Gateways of A, B use IPsec mechanisms
  - But the information must be secret to everyone except the two groups, even secret from other people in A.org and B.org
- Inner tunnel: a SA between the hosts of the two groups
- Outer tunnel: the SA between the two gateways

Example: Systems



Example: Packets

IP header from gwA	AH header from gwA	ESP header from gwA	IP header from hostA	AH header from hostA	ESP header from hostA	IP header from hostA	Transport layer headers, data

- Packet generated on hostA
- Encapsulated by hostA's IPsec mechanisms
- Again encapsulated by gwA's IPsec mechanisms
  - Above diagram shows headers, but as you go left, everything to the right would be enciphered and authenticated, *etc*.

AH Protocol

- Parameters in AH header
  - Length of header
  - SPI of SA applying protocol
  - Sequence number (anti-replay)
  - Integrity value check
- Two steps
  - Check that replay is not occurring
  - Check authentication data

Sender

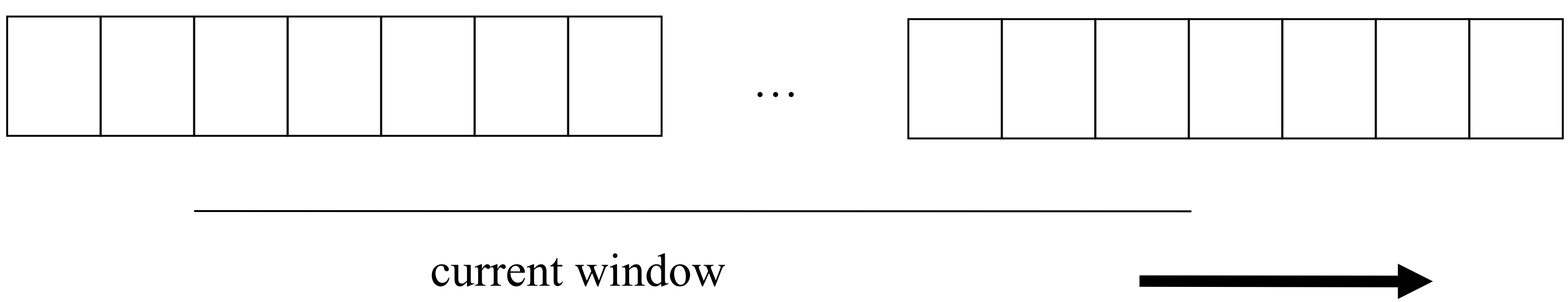
- Check sequence number will not cycle
- Increment sequence number
- Compute IVC of packet
  - Includes IP header, AH header, packet data
    - IP header: include all fields that will not change in transit; assume all others are 0
    - AH header: authentication data field set to 0 for this
    - Packet data includes encapsulated data, higher level protocol data

Recipient

- Assume AH header found
- Get SPI, destination address
- Find associated SA in SAD
  - If no associated SA, discard packet
- If antireplay not used
  - Verify IVC is correct
    - If not, discard

Recipient, Using Antireplay

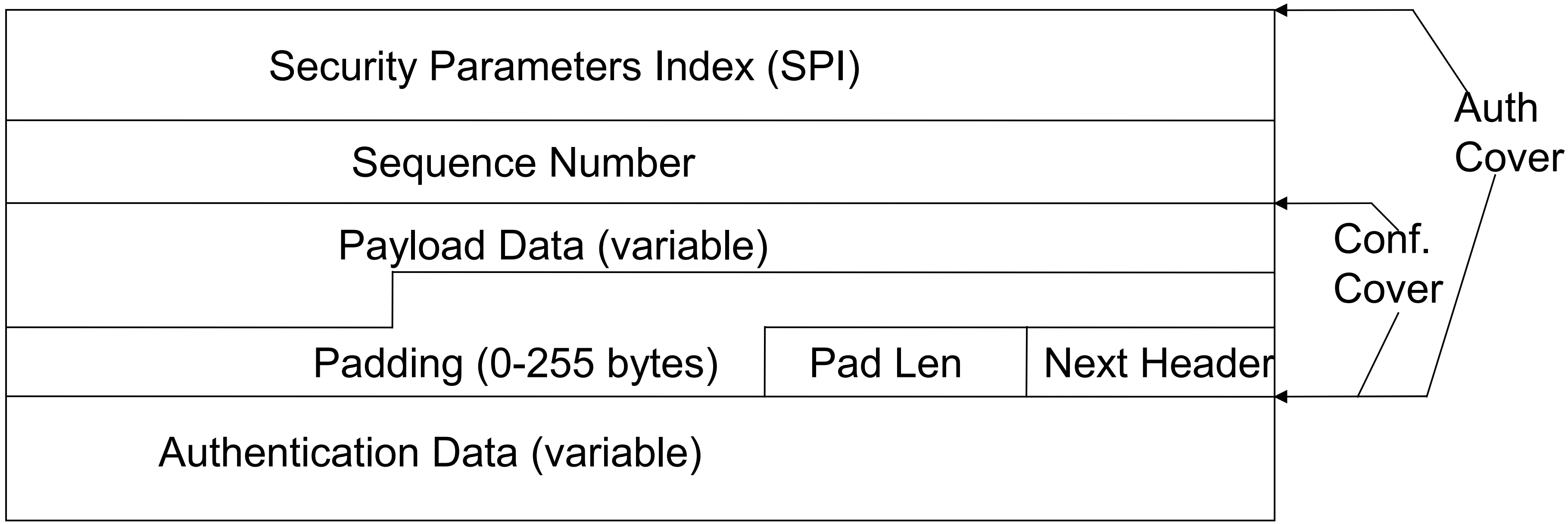
- Check packet beyond low end of sliding window
- Check IVC of packet
- Check packet's slot not occupied
  - If any of these is false, discard packet



AH Miscellany

- All implementations must support:
  HMAC_MD5
  HMAC_SHA-1
- May support other algorithms

ESP Header


Security Parameters Index (SPI)
Sequence Number
Payload Data (variable)
Padding (0-255 bytes)
Pad Len
Next Header
Authentication Data (variable)
Auth Cover
Conf. Cover

ESP Protocol

- Parameters in ESP header
  - SPI of SA applying protocol
  - Sequence number (anti-replay)
  - Generic “payload data” field
  - Padding and length of padding
    - Contents depends on ESP services enabled; may be an initialization vector for a chaining cipher, for example
    - Used also to pad packet to length required by cipher
  - Optional authentication data field

Sender

- Add ESP header
  - Includes whatever padding needed
- Encipher result
  - Do not encipher SPI, sequence numbers
- If authentication desired, compute as for AH protocol *except* over ESP header, payload and *not* encapsulating IP header

Recipient

- Assume ESP header found
- Get SPI, destination address
- Find associated SA in SAD
  - If no associated SA, discard packet
- If authentication used
  - Do IVC, antireplay verification as for AH
    - Only ESP, payload are considered; *not* IP header
    - Note authentication data inserted after encipherment, so no deciphering need be done

Recipient

- If confidentiality used
  - Decipher enciphered portion of ESP heaser
  - Process padding
  - Decipher payload
  - If SA is transport mode, IP header and payload treated as original IP packet
  - If SA is tunnel mode, payload is an encapsulated IP packet and so is treated as original IP packet

ESP Miscellany

- Must use at least one of confidentiality, authentication services
- Synchronization material must be in payload
  - Packets may not arrive in order, so if not, packets following a missing packet may not be decipherable

More ESP Miscellany

- All implementations must support (encipherment algorithms):
  - DES in CBC mode
  - NULL algorithm (identity; no encipherment)
- All implementations must support (integrity algorithms):
  - HMAC_MD5
  - HMAC_SHA-1
  - NULL algorithm (no MAC computed)
- Both cannot be NULL at the same time

Which to Use: PEM, SSL, IPsec

- What do the security services apply to?
  - If applicable to one application *and* application layer mechanisms available, use that
    - PEM for electronic mail
  - If more generic services needed, look to lower layers
    - SSL for transport layer, end-to-end mechanism
    - IPsec for network layer, either end-to-end or link mechanisms, for connectionless channels as well as connections
  - If endpoint is host, SSL and IPsec sufficient; if endpoint is user, application layer mechanism such as PEM needed

Key Points

- Key management critical to effective use of cryptosystems
  - Different levels of keys (session *vs*. interchange)
- Keys need infrastructure to identify holders, allow revoking
  - Key escrowing complicates infrastructure
- Digital signatures provide integrity of origin and content

  Much easier with public key cryptosystems than with classical cryptosystems