IPSec, TLS, and DNSSEC

Three Major Secure Protocols

- Last lecture presented mechanisms
- This lecture presents 3 examples of their use
  - Layer 3: IPSec
  - Layer 4: TLS
  - Layer 7: DNSSEC

IPSec Overview

- Layer 3: between hosts, covers both IPv4 and IPv6 [RFC 4301]
- AH: IP Authentication Header (MAY, [RFC 4302])
- ESP: Encapsulated Security Payload (MUST, [RFC 4303])
- Very comprehensive: this lecture will only cover some of the basics (no multicast, combined ESP+AH, IPv6, etc.)

IPSec Operation [RFC 4301]

- Hosts can use IPSec directly (“transport mode”)
- Security gateways can tunnel traffic through IPSec (“tunnel mode”)
- Security Associations (SAs) specify security services for traffic in a half-duplex “connection”
  - Bi-directional traffic requires two SAs
  - Security Parameters Index (SPI) field specifies SA in unicast traffic
- Security Association Database (SAD) maintained at each endpoint
  - Packets processed based on SA, src/dest IP address
  - SAD managed “semi-manually”

Transport Mode vs. Tunneling Mode

- Transport mode operates directly on top of IP
  - Next header is TCP, UDP, etc.
  - IPSec header interposes between IP and transport header
- Tunneling mode encapsulates entire IP packet
  - Next header is IP
  - Separate source, destination addresses

Encapsulating Security Payload [RFC 4303]

- Provides confidentiality, integrity, or both
- Next header field specifies payload
Transport vs. Tunneling

ESP Algorithm Support Complications
- Some algorithms require an initialization vector (IV), e.g. CBC
- Some algorithms integrate confidentiality and integrity ("combined mode algorithms")
  - If confidentiality is required for integrity, need to repeat SPI and sequence number
- Algorithm can specify payload substructure (append/prepend data)

ESP details
- Must avoid replays
  - Keep counter for 64-bit sequence number
  - Receiver must accept some packets out of order (e.g., up to 32)
  - Only low 32 bits of sequence number in actual packet
    (would be bad if you lost 4 billion packets)
- Support for traffic flow confidentiality (TFC)
  - Can pad packets to fixed length
  - Can send dummy packets
- Support for encryption without MAC... Bummer!
  - Rationale: App might be SSL, which has MAC-only mode
  - But then attacker can mess with destination address!

Internet Key Exchange (IKEv2, [RFC 4306])
- Can establish SAs for IPSec
- UDP port 500, designed to work over NATs
- All messages are request/response exchanges, use Diffie-Hellman
  - Alice and Bob have secrets $a$, $b$, public values $g$, $p$
  - Alice computes $A = g^a \mod p$, Bob $B = g^b \mod p$
  - Exchange $A$ and $B$, Alice computes $s = B^a \mod p$, Bob $s = A^b \mod p$
  - Both $s$ are $g^{ab} \mod p$ shared secret

IPSec Complication: NAT
- Transport mode can encrypt transport header, integrity covers transport header
- NAT needs to rewrite transport header!
- NAT-T [RFC 3948], tunnel IPSec in UDP

ESP details
- Must avoid replays
  - Keep counter for 64-bit sequence number
  - Receiver must accept some packets out of order (e.g., up to 32)
  - Only low 32 bits of sequence number in actual packet
    (would be bad if you lost 4 billion packets)
- Support for traffic flow confidentiality (TFC)
  - Can pad packets to fixed length
  - Can send dummy packets
- Support for encryption without MAC... Bummer!
  - Rationale: App might be SSL, which has MAC-only mode
  - But then attacker can mess with destination address!

Internet Key Exchange (IKEv2, [RFC 4306])
- Can establish SAs for IPSec
- UDP port 500, designed to work over NATs
- All messages are request/response exchanges, use Diffie-Hellman
  - Alice and Bob have secrets $a$, $b$, public values $g$, $p$
  - Alice computes $A = g^a \mod p$, Bob $B = g^b \mod p$
  - Exchange $A$ and $B$, Alice computes $s = B^a \mod p$, Bob $s = A^b \mod p$
  - Both $s$ are $g^{ab} \mod p$ shared secret
IKEv2 Exchanges

- **IKE_SA_INIT:** negotiate crypto algorithms, establish a shared secret
- **IKE_AUTH:** authenticate INIT messages, exchange certificates, establish first SA
- **CREATE_CHILD_SA:** create a new SA, renegotiate keys for an SA
- **INFORMATIONAL:** Notification, Delete, and Configuration

IPSec Overview

- Layer 3 security, transport and tunneling mode
- Tunneling mode supports security gateways
- Transport mode has trouble with NATs
- Security specified by SPI, SAs established manually or through IKE

SSL/TLS [RFC 5246] Overview

- SSL offers security for HTTP protocol
  - That’s what the padlock means in your web browser
- Authentication of server to client
- Optional authentication of client to server
  - Incompatibly implemented in different browsers
  - CA infrastructure not in widespread use
- Confidentiality of communications
- Integrity protection of communications

TLS/SSL

SSL offers security for HTTP protocol
- That’s what the padlock means in your web browser

Purpose in more detail

- Authentication based on certification authorities (CAs)
  - Certifies who belongs to a public key (domain name and real name of company)
  - Example: Verisign

- What SSL Does Not Address
  - Privacy
  - Traffic analysis
  - Trust management

Ciphersuites: Negotiating ciphers

- Server authentication algorithm (RSA, DSS)
- Key exchange algorithm (RSA, DHE)
- Symmetric cipher for confidentiality (RC4, DES, AES)
- MAC (HMAC-MD5, HMAC-SHA)
Overview of SSL Handshake

Client

- Supported ciphers, client random
- Chosen cipher, server random, certificate
- Compute keys
- MAC of handshake messages

Server

- Supported ciphers, client random
- Chosen cipher, server random, certificate
- Compute keys
- MAC of handshake messages

Compute keys

Encrypted pre-master secret

MAC of handshake messages

From “SSL and TLS” by Eric Rescorla

SSL Handshake

- Client and server negotiate on cipher selection
- Cooperatively establish session keys
- Use session keys for secure communication
- Details
  - Multiple messages per stage
  - Get an idea of protocol in action:
    ```
    openssl s_client -connect www.paypal.com:443
    ```

SSL Client Certificate

Client

- Supported ciphers, client random
- Chosen cipher, server random, certificate
- Compute keys
- MAC of handshake messages
- Certificate request
- Certificate, cert verify

Server

- Certificate requested
- Compute keys
- MAC of handshake messages
- Certificate, cert verify

From “SSL and TLS” by Eric Rescorla

Client Authentication Handshake

- Server requests that client send its certificate
- Client signs a signed digest of the handshake messages

Establishing a Session Key

- Server and client both contribute randomness.
- Client sends server a “pre-master secret” encrypted with server’s public key
- Use randomness and pre-master secret to create session keys:
  - Client MAC
  - Server MAC
  - Client Write
  - Server Write
  - Client IV
  - Server IV

Establishing a Session Key

- Client random
- Pre-master secret
- Server random

Key block

- Client MAC
- Server MAC
- Key verify
- AI verify
- AI verify
- AI verify
- Server IV

From “SSL and TLS” by Eric Rescorla
Session Resumption

- Problem: Public key crypto expensive
- New TCP connection, reuse master secret.
  - Avoids unnecessary public key cryptography.
- Combines cached master secret with new randomness to generate new session keys.
- Works even when the client IP changes (servers cache on session ID, clients cache on server hostname).

Example cross-layer issue

- TLS puts message format on top of TCP (control, application)
- Handshake through control messages before application data
- Virtual hosts: single web server that responds differently for different requested hosts (in HTTP request)
- What's the problem?

2-minute break

Example cross-layer issue

- TLS puts message format on top of TCP (control, application)
- Handshake through control messages before application data
- Virtual hosts: single web server that responds differently for different requested hosts (in HTTP request)
- What's the problem?

DNS Review

- Domain Name System: resource records (RRs) bind values to names
- Designed to be highly scalable, distributed administration

Structure of a DNS message [RFC 1035]

<table>
<thead>
<tr>
<th>Header</th>
</tr>
</thead>
<tbody>
<tr>
<td>Question</td>
</tr>
<tr>
<td>Answer</td>
</tr>
<tr>
<td>Authority</td>
</tr>
<tr>
<td>Additional</td>
</tr>
</tbody>
</table>

- Same message format for queries and replies
  - Query has zero RRs in Answer/Authority/Additional sections
  - Reply includes question, plus has RRs
- Authority allows for delegation
- Additional for glue + other RRs client might need
DNS software architecture

- Two types of query
  - Recursive
  - Non-Recursive
- Apps make recursive queries to local DNS server (1)
- Local server queries remote servers non-recursively (2, 4, 6)
  - Aggressively caches result
  - E.g., only contact root on first query ending .umass.edu

DNS Vulnerabilities

- Results cached: cache poisoning
- UDP, no session to guess
- Responses spoofable, can structure content to generate many opportunities
- Chance, but networks are always faster and one poisoned cache can be disastrous

DNSSEC [RFC 4034]

- Security extensions for DNS (new RRs):
  - DNSKEY: public key for zone
  - RRSIG: signature for a set of RRs
  - NSEC: next authoritative name
  - DS: identifying digest of DNSKEY (stored in parent zone)
- Provides integrity and authentication, not confidentiality

DNSSEC Example: www.berkeley.edu

- dig . DNSKEY | grep -Ev ' ^($|;)'> root.keys
- dig +sigchase +trusted-key=./root.keys www.berkeley.edu

Proof of Non-Existence

- Problem: spoofed host name, maybe don’t want to sign RR
- NSEC record allows a client to verify a node does not exist
- Contains next valid name
- dig +sigchase +trusted-key=./root.keys wws.berkekey.edu
- dig +sigchase +trusted-key=./root.keys wuz.berkekey.edu
- See a problem?

Hashed Denial of Existence [RFC 5155]

- NSEC record allows an adversary to enumerate a domain
- NSEC3 record: rather than next host name, contains hash of next host name
- Couldn’t find anyone serving NSEC3...
Question 1: How do resolvers validate root DNSKEY?

Question 2: Who has the root zone private key?

Certificates

How to get a Verisign certificate
• Get DBA license from city call ($20)
  - No on-line check for name conflicts...can I do business as Microsoft?
• Letterhead from company ($0)
• Notarized document (need driver’s license) ($0)
• E.g., pay Verisign ($399–$1,499/year)
• Conclusions:
  - Easy to get a fraudulent certificate
  - Maybe not so easy to avoid prosecution afterwards
• But that’s only Verisign’s policy
  - Many CAs can issue certificates

What does CA mean by certificate?
• That a public key belongs to someone authorized to represent a hostname?
• That a public key belongs to someone who is associated in some way with a hostname?
• That a public key belongs to someone who has lots of paper trails associated to a company related to a hostname?
• That the CA has no liability, or $100,000, or $250,00?
• >100-page Certification Practice Statement (CPS)

So many CAs...
CA Convenience vs. Security

- **How convenient is a Verisign certificate?**
  - Need fee + cooperation from Stanford IT to get one here
  - Good for credit cards, but shuts out many other people

- **How trustworthy is a Verisign certificate?**
  - In mid-March 2001, VeriSign, Inc., advised Microsoft that on January 29 and 30, 2001, it issued two… [fraudulent] certificates…. The common name assigned to both certificates is “Microsoft Corporation.”

  VeriSign has revoked the certificates…. However… it is not possible for any browser’s CRL-checking mechanism to locate and use the VeriSign CRL.

  – Microsoft Security Bulletin MS01-017

How Do You Revoke a Certificate?