Network Security Protocols

Mike Freedman
COS 461: Computer Networks

http://www.cs.princeton.edu/courses/archive/spr14/cos461/

Network Security

• Application layer
  – E-mail: PGP, using a web-of-trust
  – Web: HTTP-S, using a certificate hierarchy
• Transport layer
  – Transport Layer Security/ Secure Socket Layer
• Network layer
  – IP Sec
• Network infrastructure
  – DNS-Sec and BGP-Sec

Basic Security Properties

• Confidentiality:
• Authenticity:
• Integrity:
• Availability:
• Non-repudiation:
• Access control:
Encryption and MAC/Signatures

**Confidentiality (Encryption)**

Sender:
- Compute \( C = Enc_k(M) \)
- Send \( C \)

Receiver:
- Recover \( M = Dec_k(C) \)

**Auth/Integrity (MAC / Signature)**

Sender:
- Compute \( s = \text{Sig}_k(\text{Hash}(M)) \)
- Send \(<M, s>\)

Receiver:
- Compute \( s' = \text{Ver}_k(\text{Hash}(M)) \)
- Check \( s' = s \)

These are simplified forms of the actual algorithms.

Email Security:
Pretty Good Privacy (PGP)

E-Mail Security

- **Security goals**
  - Confidentiality: only intended recipient sees data
  - Integrity: data cannot be modified en route
  - Authenticity: sender and recipient are who they say

- **Security non-goals**
  - Timely or successful message delivery
  - Avoiding duplicate (replayed) message
  - (Since e-mail doesn’t provide this anyway!)

Sender and Receiver Keys

- If the sender knows the receiver’s public key
  - Confidentiality
  - Receiver authentication

- If the receiver knows the sender’s public key
  - Sender authentication
  - Sender non-repudiation
Sending an E-Mail Securely

- **Sender digitally signs the message**
  - Using the sender’s private key
- **Sender encrypts the data**
  - Using a one-time session key
  - Sending the session key, encrypted with the receiver’s public key
- **Sender converts to an ASCII format**
  - Converting the message to base64 encoding
  - (Email messages must be sent in ASCII)

Public Key Certificate

- **Binding between identity and a public key**
  - “Identity” is, for example, an e-mail address
  - “Binding” ensured using a digital signature
- **Contents of a certificate**
  - Identity of the entity being certified
  - Public key of the entity being certified
  - Identity of the signer
  - Digital signature
  - Digital signature algorithm id

Web of Trust for PGP

- **Decentralized solution**
  - Protection against government intrusion
  - No central certificate authorities
- **Customized solution**
  - Individual decides whom to trust, and how much
  - Multiple certificates with different confidence levels
- **Key-signing parties!**
  - Collect and provide public keys in person
  - Sign other’s keys, and get your key signed by others

HTTP Security
HTTP Threat Model

- Eavesdropper  
  - Listening on conversation (confidentiality)
- Man-in-the-middle  
  - Modifying content (integrity)
- Impersonation  
  - Bogus website (authentication, confidentiality)

HTTP-S: Securing HTTP

- HTTP sits on top of secure channel (SSL/TLS)  
  - TCP port 443 vs. 80
- All (HTTP) bytes encrypted and authenticated  
  - No change to HTTP itself!
- Where to get the key???

HTTP-Secure Transport Layer

Learning a Valid Public Key

- What is that lock?
  - Securely binds domain name to public key (PK)
    - If PK is authenticated, then any message signed by that PK cannot be forged by non-authorized party
  - Believable only if you trust the attesting body
    - Bootstrapping problem: Who to trust, and how to tell if this message is actually from them?

Hierarchical Public Key Infrastructure

- Public key certificate  
  - Binding between identity and a public key
  - “Identity” is, for example, a domain name
  - Digital signature to ensure integrity
- Certificate authority  
  - Issues public key certificates and verifies identities
  - Trusted parties (e.g., VeriSign, GoDaddy, Comodo)
  - Preconfigured certificates in Web browsers
Public Key Certificate

Transport Layer Security (TLS)
Based on the earlier Secure Socket Layer (SSL) originally developed by Netscape

TLS Handshake Protocol
- Send new random value, list of supported ciphers
- Send pre-secret, encrypted under PK
- Create shared secret key from pre-secret and random
- Switch to new symmetric-key cipher using shared key

TLS Record Protocol
- Messages from application layer are:
  - Fragmented or coalesced into blocks
  - Optionally compressed
  - Integrity-protected using an HMAC
  - Encrypted using symmetric-key cipher
  - Passed to the transport layer (usually TCP)
- Sequence #5 on record-protocol messages
  - Prevents replays and reorderings of messages
Comments on HTTPS

- HTTPS authenticates server, not content
  - If CDN (Akamai) serves content over HTTPS, customer must trust Akamai not to change content

- Symmetric-key crypto after public-key ops
  - Handshake protocol using public key crypto
  - Symmetric-key crypto much faster (100-1000x)

- HTTPS on top of TCP, so reliable byte stream
  - Can leverage fact that transmission is reliable to ensure: each data segment received exactly once
  - Adversary can’t successfully drop or replay packets

IP Security

- There are range of app-specific security mechanisms
  - eg. TLS/HTTPS, S/MIME, PGP, Kerberos, ...

- But security concerns that cut across protocol layers
- Implement by the network for all applications?

Enter IPSec!

IPSec

- General IP Security framework

- Allows one to provide
  - Access control, integrity, authentication, originality, and confidentiality

- Applicable to different settings
  - Narrow streams: Specific TCP connections
  - Wide streams: All packets between two gateways
**IPSec Uses**

- If in a firewall/router:
  - Strong security to all traffic crossing perimeter
  - Resistant to bypass

- Below transport layer
  - Transparent to applications
  - Can be transparent to end users

- Can provide security for individual users

**Benefits of IPSec**

**IP Security Architecture**

- Specification quite complex
  - Mandatory in IPv6, optional in IPv4

- Two security header extensions:
  - Authentication Header (AH)
    - Connectionless integrity, origin authentication
      - MAC over most header fields and packet body
    - Anti-replay protection
  - Encapsulating Security Payload (ESP)
    - These properties, plus confidentiality

**Encapsulating Security Payload (ESP)**

- Transport mode: Data encrypted, but not header
  - After all, network headers needed for routing!
  - Can still do traffic analysis, but is efficient
  - Good for host-to-host traffic

- Tunnel mode: Encrypts entire IP packet
  - Add new header for next hop
  - Good for VPNs, gateway-to-gateway security
Replay Protection is Hard

- Goal: Eavesdropper can’t capture encrypted packet and duplicate later
  - Easy with TLS/HTTP on TCP: Reliable byte stream
  - But IP Sec at packet layer; transport may not be reliable

- IP Sec solution: Sliding window on sequence #’s
  - All IPSec packets have a 64-bit monotonic sequence number
  - Receiver keeps track of which seqno’s seen before
    - [lastest – windowsize + 1 , latest] ; windowsize typically 64 packets
  - Accept packet if
    - seqno > latest (and update latest)
    - Within window but has not been seen before
  - If reliable, could just remember last, and accept iff last + 1

Hierarchical Naming in DNS

DNS Security

- 13 root servers (see http://www.root-servers.org/)
- Labeled A through M

DNS Root Servers

- A Verisign, Dulles, VA
- C Cogent, Herndon, VA (also Los Angeles)
- D U Maryland College Park, MD
- G U US DoD Vienna, VA
- J Verisign, (11 locations)
- K RIPE London (+ Amsterdam, Frankfurt)
- L ICANN Los Angeles, CA
DoS attacks on DNS Availability

- Feb. 6, 2007
  - Botnet attack on the 13 Internet DNS root servers
  - Lasted 2.5 hours
- None crashed, but two performed badly:
  - g-root (DoD), l-root (ICANN)
  - Most other root servers use anycast

Defense: Replication and Caching

<table>
<thead>
<tr>
<th>Letter</th>
<th>Old name</th>
<th>Operator</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>ns.internic.net</td>
<td>VeriSign</td>
<td>Dulles, Virginia, USA</td>
</tr>
<tr>
<td>B</td>
<td>ns1.fei.edu</td>
<td>ISI</td>
<td>Marina Del Rey, California, USA</td>
</tr>
<tr>
<td>C</td>
<td>c pris net</td>
<td>Cogent Communications</td>
<td>distributed using anycast</td>
</tr>
<tr>
<td>D</td>
<td>tarp.ums.edu</td>
<td>University of Maryland</td>
<td>College Park, Maryland, USA</td>
</tr>
<tr>
<td>E</td>
<td>ns.ripe.net</td>
<td>NASA</td>
<td>Mountain View, California, USA</td>
</tr>
<tr>
<td>F</td>
<td>ns.ac.org</td>
<td>ISIC</td>
<td>distributed using anycast</td>
</tr>
<tr>
<td>G</td>
<td>ns.rot.dns.mil</td>
<td>U.S. DoD NIC</td>
<td>Columbus, Ohio, USA</td>
</tr>
<tr>
<td>H</td>
<td>asec.at army.mil</td>
<td>U.S. Army Research Lab</td>
<td>Aberdeen-Proving Ground, Maryland, USA</td>
</tr>
<tr>
<td>I</td>
<td>nic缜.ndu.net</td>
<td>Autonome &amp; Distribution using anycast</td>
<td></td>
</tr>
<tr>
<td>J</td>
<td>Verisign</td>
<td>distributed using anycast</td>
<td></td>
</tr>
<tr>
<td>K</td>
<td>RIPE NCC</td>
<td>distributed using anycast</td>
<td></td>
</tr>
<tr>
<td>L</td>
<td>ICANN</td>
<td>Los Angeles, California, USA</td>
<td></td>
</tr>
<tr>
<td>M</td>
<td>WIDE Project</td>
<td>distributed using anycast</td>
<td></td>
</tr>
</tbody>
</table>

source: wikipedia

Denial-of-Service Attacks on Hosts

-x40 amplification

580,000 open resolvers on Internet (Kaminsky-Shiffman’06)

Preventing Amplification Attacks

- Prevent ip spoofing
- Disable open amplifiers
DNS Integrity and the TLD Operators

- If domain name doesn’t exist, DNS should return NXDOMAIN (non-existant domain) msg
- Verisign instead creates wildcard records for all .com and .net names not yet registered
  – September 15 – October 4, 2003
- Redirection for these domain names to Verisign web portal: “to help you search”
  – And serve you ads...and get “sponsored” search
  – Verisign and online advertising companies make $$

DNS Integrity: Cache Poisoning

- Was answer from an authoritative server?
  – Or from somebody else?
- DNS cache poisoning
  – Client asks for www.evil.com
  – Nameserver authoritative for www.evil.com returns additional section for (www.cnn.com, 1.2.3.4, A)
  – Thanks! I won’t bother check what I asked for

DNS Integrity: DNS Hijacking

- To prevent cache poisoning, client remembers:
  – The domain name in the request
  – A 16-bit request ID (used to demux UDP response)
- DNS hijacking
  – 16 bits: 65K possible IDs
  – What rate to enumerate all in 1 sec? 64B/packet
    – 64*65536*8 / 1024 / 1024 = 32 Mbps
- Prevention: also randomize DNS source port
  – Kaminsky attack: this source port... wasn’t random

Let’s strongly believe the answer! Enter DNSSEC

- DNSSEC protects against data spoofing and corruption
- DNSSEC also provides mechanisms to authenticate servers and requests
- DNSSEC provides mechanisms to establish authenticity and integrity

http://unixwiz.net/techtips/guide-kaminsky-dns-vuln.html
PK-DNSSEC (Public Key)

- The DNS servers sign the hash of resource record set with its private (signature) keys
  - Public keys can be used to verify the SIGs
- Leverages hierarchy:
  - Authenticity of name server’s public keys is established by a signature over the keys by the parent’s private key
  - In ideal case, only roots’ public keys need to be distributed out-of-band

Conclusions

- Security at many layers
  - Application, transport, and network layers
  - Customized to the properties and requirements
- Exchanging keys
  - Public key certificates
  - Certificate authorities vs. Web of trust
- Next time
  - Interdomain routing security
- Learn more: take COS 432 in the fall!