Overview

- Three facets of web security
  - Securing the server
  - Securing information in transit
  - Securing the end user system
- Infrastructure Protocols
  - DNS
  - TCP/IP
- Web Protocols
  - Basic
  - Advanced

Three Facets of Web Security

- Securing the Server
  - Insure the server will operate without interruption (A)
  - That the information on the server has integrity (I)
  - That access to the information on the server is controlled (C)
- Securing the transmission path
  - Secure information transmitted from the user
  - Secure information transmitted to the user
- Secure the end-users computer
  - Insure the user platform is malware free
  - Make sure personal data is minimized or secured
Securing a Web Server

- Secure the physical computer
  - This ranges from the very simple matters of physical access to matters such as reducing other services to a bare minimum – e.g. no other services – mail, echo, ftp, etc.
- Secure the web server software
  - This involves a detailed and documented configuration of the web server – apache or IIS to be sure it is configured as desired.
- Secure any interactions between the operating system and the web server software
  - Make sure any documented vulnerabilities are closed and any known problems that develop are addressed

Assuring Information Integrity

- Once information leaves a web server, it is almost impossible to guarantee integrity
  - Screens can be photographed
  - Audio can be recorded
  - This can lead to altered or degraded copies and can reduce the ability of the provider to measure access
  - Digital watermarks and other invisible tagging techniques can be used

Securing Information in Transit

- This could be done by various means (securing the network, etc.), but the only practical means is encryption.
- The primary mechanism for encryption of information in transit is SSL – secure sockets layer – the SSL3.0 protocol is interoperable with transport layer security (TLS) which is considered its successor
- While SSL can insure information transmitted is secure, it does not guarantee the ability to transmit information – as a result of DOS or DDOS
Securing End-User Systems

- The ability to secure end user systems is the most problematic
- Some have suggested that education of the end user would solve the problem
- The ability of hackers to overcome the best education (sending email and worms with the email addresses of known users)
- Today, most are banking on advanced technologies and automated updates

Risk Analysis and Best Practices

- In the last analysis, a tightly secured system may not be optimal for the business goals of an organization
- Risk analysis offers a compromise, but it is difficult to implement for computer security
  - Not all risks are known
  - The likelihood of a given risk may change
  - The impact of failures may change
- Best practices provides an alternative to risk analysis, but there are problems here as well:
  - Best practices are different for different classes of sites
  - Best practices need to be constantly updated

Protocol Common Sense (1)

- There are a number of possible hardware configurations -- for our purposes, consider everything is on an Ethernet bus.
- The connected machines all have an address and operate on a CSMA/CD basis -- that is:
  - Each machine has a hardware enforced address -- called a medium access control (MAC) address
  - The network connections use carrier sense multiple access(CSMA) with a provision for collision detection(CD)
- Basically, the design says that all machines on an Ethernet bus can see all packets
  - Take a look at Wireshark
Protocol Common Sense (2)

- Using the MAC address, messages can be sent to a particular machine.
- If each machine is also given a logical address or name, we can now add a couple protocols:
  - A protocol can be added which identifies the source and destination logical address (for us this will be IP)
  - A protocol can be added which “resolves” or maps the logical address to the MAC address (for us this will be the ARP – Address Resolution Protocol)
  - A protocol can be added which “resolves” or maps the route between multiple networks (for us this will be RIPv – Routing Information Protocol, but it could be the Open Shortest Path First (OSPF) protocol or the Border Gateway Protocol (BGP))

Protocol Common Sense (3)

- We can overlay a high level protocol to add names to machines and translate them to IP addresses (DNS)
- We can add a protocol to operate on sets of packets, guaranteeing ordered and reliable delivery of multi-packet messages (TCP)

Hosts and Gateways

- Any machine capable of sending or receiving packets is a host.
- Machines that have multiple network connections allow data to be moved from one network to another:
  - Repeaters extend the physical length of a network
  - Bridges are selective repeaters
  - Routers know where to send packets
  - Gateways translate data from one protocol to another
  - Firewalls filter data allowed into or out of a network
The internet

- The internet is a packet switched network that provides a level of robustness compared with circuit switched networks.
- Packets contain not only information for the endpoints of communication, but information about the endpoints.
- The “most basic” protocol for the internet is the internet protocol (IP) which describes how packets are moved from point to point.
- At the current time, multiple versions of IP are being considered which relate to address space size and security of packets, i.e.:
  - IPV6 versus IPV4 differ in address spaces.
  - IPV4 versus IPV4Sec differ in how they are secured.

Connecting to the Internet

- The end point access for a computer to the internet could be:
  - A connection to a local area network – hardwired or wireless.
  - A connection to an Internet Service Provider (ISP) via a modem using the Point to Point Protocol.
  - A connection to an ISP using a cable modem or router.

IP Addresses and the DNS

- IP addresses (version 4) are 32 bit binary numbers – (about 4 billion).
- There is a “dotted decimal” form of the address which most people are aware of – such as 132.142.116.68.
- There is also a “domain name service” which equates given addresses with mnemonic addresses such as spring.sis.pitt.edu.
- While there is often a relationship between addresses and domain names, there need not be.
Domain Name Service

- When a machine is connected to the internet, it is also provided with the address of the Domain Name Server associated with the ISP – either manually or automatically
- The local domain name server, knows some local information and at least the name of the root name server.
- The root name servers know the names of the top level nameservers
- The top level name servers know the names of the domain name servers underneath them

How DNS Works

- The Domain Name Service (DNS) is a mapping of IP addresses to host names
  - Originally every internet host had a file called hosts.txt that stored all these mappings
  - Today, the DNS protocol begins with each host knowing the address of at least one name server
  - DNS servers either answer requests or route them for name resolution with the hierarchy of DNS servers
- The problem is that the DNS protocol can be spoofed

IP/IPSec

- The Internet Protocol (IP) suffers security issues related to replaying and spoofing
- IPSec provides a mechanism to overcome these issues
- IPSec has various features and modes of operation:
  - Authentication Header (AH) provides Integrity + Authentication
  - Encapsulating Security Payload (ESP) provides Integrity + Authentication + Confidentiality
  - In Transport Mode only the payload of the IP packet is encrypted and/or authenticated.
  - In Tunnel Mode the entire IP packet (data plus the message headers) is encrypted and/or authenticated
Some IPSec “Issues”

- Encryption decryption processes are computational intensive
- Key management in large networks is difficult
- IPSec is not implemented in TCP/IP stack, needs a client installed
- Firewalls need to be modified to use IPSec (you have to create rules with filter lists and actions and then add these to a policy, and then distribute them and)
- Scheduling causes packet loss in real time applications
- Packets have single destination addresses, so it is difficult to manage multicast traffic such as SPI (IPSec has multicast option but is not enough for all occasions.)
- Some applications using streaming multimedia assign port numbers dynamically, so IPSec policy becomes difficult to assign
- Incompatibility with NAT (RFC 3715) – When NAT changes the IP addresses or ports in the IP header, IPSec cannot re-calculate the hash

DNS Sec

- DNSSec protects clients from forged DNS data, such as that created by DNS cache poisoning.
- DNS responses under DNSSec are digitally signed.
- The signature, allows the client to check if the information came from an authoritative DNS server.

Issues with DNSSec

- DNSSEC has faced been hampered by several issues:
  - It must operate in a backward-compatible fashion
  - It must be widely deployed on servers and resolvers (clients)
  - There is disagreement among implementers over who should own the top-level domain root keys
Packet Structure Review

- As always, reality is more complex, but for our purposes, we look at four frames that will carry our messages.
- Ethernet packets are the biggest envelop and carry our messages from point to point on the physical media.
- IP packets are the addressing packets over the large logical network. They exist as Ethernet packet data.
- TCP packets are the data in IP packets and they contain additional data for message structuring.
- Your application protocol exists as TCP data and they do the work you intend.

Ethernet Packet

<table>
<thead>
<tr>
<th>Preamble</th>
<th>Ethernet Destination Address</th>
<th>Ethernet Source Address</th>
<th>Packet Length</th>
<th>Message Data</th>
<th>CRC</th>
</tr>
</thead>
<tbody>
<tr>
<td>64 bits</td>
<td>48 bits</td>
<td>48 bits</td>
<td>16 bits</td>
<td>368-12K</td>
<td>32 bits</td>
</tr>
</tbody>
</table>

- Preamble is a sequence of bits used to sync clocks.
- Ethernet addresses are MAC addresses.
- MAC addresses are bound to network interface hardware.
- CRC is a data error checking code.
- The message is normally an IP packet.

IP Packet

<table>
<thead>
<tr>
<th>Ver</th>
<th>IHL</th>
<th>Type of Service</th>
<th>Total Length (in octets)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Flags</td>
<td>Fragment Offset</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Time to Live</td>
<td>Protocol</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Source Address</td>
<td>Header Checksum</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Destination Address</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Options and Padding</td>
<td>(to meet IHL spec)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Data/payload</td>
<td></td>
</tr>
</tbody>
</table>

- Ver
- IHL
- Type of Service
- Total Length (in octets)
- Flags
- Fragment Offset
- Time to Live
- Protocol
- Source Address
- Destination Address
- Options and Padding (to meet IHL spec)
- Data/payload
**IP Packet Notes**

- **Ver** – IP Version (1-4)
- **IHL** – IP Header Length (# 32 bit words)
- **Type Service**
  - bits (0-2) network control, Inet control, Flash, Override;
b-3 = delay (norm, high); bit-4 = throughput (norm, high); bit-5 = reliability (norm, high); bit-6/7 = reserved
- **Identifier, Flags, Offset** – used to reconstruct fragments
- **TTL** – normally max number of hops
- **Protocol** – UDP=17, TCP=6, etc.

---

**UDP Packet**

<table>
<thead>
<tr>
<th></th>
<th>16</th>
</tr>
</thead>
<tbody>
<tr>
<td>Source Port</td>
<td></td>
</tr>
<tr>
<td>Destination Port</td>
<td></td>
</tr>
<tr>
<td>Length</td>
<td></td>
</tr>
<tr>
<td>Checksum</td>
<td></td>
</tr>
<tr>
<td>Data</td>
<td></td>
</tr>
</tbody>
</table>

---

**TCP Packet**

<table>
<thead>
<tr>
<th></th>
<th>16</th>
</tr>
</thead>
<tbody>
<tr>
<td>Source Port</td>
<td></td>
</tr>
<tr>
<td>Destination Port</td>
<td></td>
</tr>
<tr>
<td>Sequence Number (32)</td>
<td></td>
</tr>
<tr>
<td>ACK Number (32)</td>
<td></td>
</tr>
<tr>
<td>Offset</td>
<td></td>
</tr>
<tr>
<td>Rsvd</td>
<td></td>
</tr>
<tr>
<td>UAPRSF</td>
<td></td>
</tr>
<tr>
<td>Window</td>
<td></td>
</tr>
<tr>
<td>Checksum</td>
<td></td>
</tr>
<tr>
<td>Urgent Pointer</td>
<td></td>
</tr>
<tr>
<td>Options and padding</td>
<td></td>
</tr>
<tr>
<td>Data (n)</td>
<td></td>
</tr>
</tbody>
</table>

Sequence Number for each octet sent

**Flags**

- **U** - Urgent Pointer is significant
- **A** - ACK number field is significant
- **P** - Push Function (send all queued data)
- **R** - Reset the connection
- **S** - SYN - synchronize sequence numbers. When SYN Flag set = Initial Sequence Number (ISN)
- **F** - FIN - No more data from sender

**Window** - Number of octets receiver will accept before an ACK

**Checksum** - On parts of the header (pseudo-header)
TCP Setup Handshake

- Client
  - Send SYN, seq=x
  - Receive SYN
  - Send SYN, seq=x, ACK x+1
  - Receive SYN+ACK
  - Send ACK y+1
  - Receive ACK

Normal TCP Window

- Client
  - Send data 1-1000
  - Send data 1001-2000
  - Send data 2001-2500
  - ACK for 1000
  - ACK for 2000
  - ACK for 2500
  - Send data 2501-3500

- Server
  - Advertise window=2500
  - ACK to 1000, window 1500
  - ACK to 2000, window 500
  - ACK to 2500, window 0
  - Some Data Processed
  - ACK to 2500, window 0

  - ACK to 2500, window 2000

Loss of ACK Timeout

- Client
  - Loss timeout

- Server
  - X