Two types of data center security issues

- “Standard” IT issues
  - Protecting data center machines (web servers, management tools, etc.) from unauthorized access
- Web service issues
  - Protecting the web services and users of web services
- This lecture deals primarily with the second set of issues
Basic “common sense” IT issues

- Firewall protect web servers
  - Aggressively block everything but limited set of web service ports
- Put management systems behind firewall
  - At worst allow VPN access
- Disable all processes except those related to web services and their management

Strong firewall protection

These filter on a very limited set of Server applications

Firewall / Intrusion Detection System

Internet

This allows nothing out, only manager access to management systems in
Web service security requirements

- User authentication by server
  - Not just machine authentication
- Server authentication by user
  - This is service authentication
- Privacy
  - Credit card numbers, etc.
- Integrity
  - Nobody can modify contents in transit

Basic approach

- User authenticates service with SSL and X.509 certificates
  - SSL = Secure Sockets Layer
- SSL encrypts session
  - This provides integrity and privacy
  - Wait until sensitive part of application---minimize use of (expensive) SSL
- User authentication happens within encrypted SSL session
  - User name and password
Identification

- User identifies service via URL
- Service identifies user by either
  - Cookie stored in browser
  - User name typed in by user
  - User name somehow supplied by browser
    - Anybody know more about this?
    - Is it anything more than standard form entry completion?

Service authentication by user

- Based on public key cryptography
- Based on X.509 Certificates
- Based on a small number of Certificate Authorities (CA)
  - Verisign, Thawte Consulting, Comodo, Baltimore, etc.
  - Not sure how many, but the whole HTTPS infrastructure is based on a small number of CAs!
Public key encryption

- Each participant has a public key and a private key
- Something encrypted with a private key can be decrypted with the public key
  - And vice versa

Public key encryption for authentication

- I want everybody to be sure something came from me:
  - Disseminate my public key widely
    - ex: PGP key in all my emails
  - Sign contents with my private key
    - That is, encrypt the hash of the contents
  - If you trust my public key, you can be sure I originated the contents
**Public key encryption for privacy**

- I want only you to see my content
  - I encrypt content twice: once with my private key and once with your public key
  - You decrypt twice: once with your private key and once with my public key
    - Both keys needed to decrypt, so only you can do it
  - Again need to trust the public keys!

---

**Public key trust in SSL**

1. Root CA gives its public key to browser manufacturers

![Diagram showing trust relationships between Root CAs and browser makers: Baltimore, Thawte, Verisign, Opera, Microsoft, Netscape.]

---
Public key trust in SSL

2. Browser maker puts the Root CA public keys in the browser, and gives the browser to you.

3. Web sites get their own public keys, in a cert signed by CA private key.
Public key trust in SSL

4. Web site gives user Cert inside SSL. User verifies it using the CA public key inside the browser.

What if CA private key is compromised?

- A rogue website could pretend to be a different website
- Though they would have to hack DNS too
  - This is not necessarily all that hard
- Note that other Root CAs can be imported into IE, Netscape, etc.
  - Those that don’t come pre-installed
What if CA private key is compromised?

- Browsers will have to be patched
- Web sites will need new certs
- A pain, but probably not a complete disaster
  - Not so different from responding to the latest worm or virus…
- Note that Root CA public keys have expired in the past
  - Typically not a problem because browser version upgrades contain more recent public keys

SSL (a.k.a. TLS)

- Transport-layer security
- Runs above TCP
- Encrypts everything above TCP
- Different applications can run over SSL
  - HTTP, telnet, FTP, LDAP
  - Each requires a separate port number to run over SSL
- TLS (Transport Layer Security) is IETF version of SSL
  - SSL still used in practice
SSL Overview

- Establish a session
  - Agree on algorithms
    - DES, 3DES, RC2, RC4, IDEA bulk encryption
    - MAC is SHA-1 or MD5 (Message Authentication Code)
  - Share secrets (RSA or Diffie-Hellman)
  - Perform authentication (Certs)
- Transfer application data
  - Ensure privacy and integrity

SSL Data Operation

1. Message to be transmitted
2. Fragment into blocks
3. Compress (optional)
4. Apply a MAC
5. Encrypt
6. Transmit
SSL “Alternatives”

- S-HTTP: secure HTTP protocol, shttp://
  - Predates SSL, never caught on
- IPSec: secure IP
  - Wrong features
- SET: Secure Electronic Transaction
  - Protocol and infrastructure for bank card payments
  - More than just a secure pipe
  - Hasn’t caught on
- SASL: Simple Authentication and Security Layer (RFC 2222)
  - Framework for selecting authentication and security
  - Encompasses lots of protocols
  - Not sure how much deployed and used

Why not IPsec (instead of SSL)?

- Only mutual authentication
  - Server doesn’t need to authenticate user until later
- Limitations due to NAT
  - One IPsec session at a time
  - IETF working to fix this
- Can’t control by port number
  - HTTPS has a well-known port (443)
  - IPsec would require separate IP address
- Dependent on user IP address
  - Secure session can’t span reconnects
Why not mutual authentication via SSL?

- In theory it is possible... requires that the user have a Cert and key
- Problem is, Certs are not easily portable across machines
  - Certainly humans can’t remember them!
  - As such, cert effectively becomes machine authentication, not user authentication

Denial of Service (DoS)

- Various forms
  - Simply overwhelm target system
    - Distributed DoS (DDoS)
    - Smurf attack
  - Consume resources on target system
    - SYN attack
    - Email bomb
  - Exploit bug in target system to crash it (usually some buffer overflow)
    - Ping of Death
    - Code Red
    - SQL Slammer
Smurf Attack

- Attacker sends ping to target network
  - Destination address is broadcast
    - Net number + all-ones
  - Spoofed source address is victim host
- Router on target network broadcasts the packet
- All recipients reply to ping, flood victim system
  - Victim need not be on target network

Smurf countermeasures

- Configure routers not to forward broadcast packets from off-net
- Configure hosts not to respond to pings to broadcast address

Still, if you are the victim on some different network, not much you can do but filter incoming ICMP pings
SYN attack

- Attacker sends many TCP SYN
  - With spoofed source address
  - So that it looks like lots of different sources
- Victim allocates TCP record for each one
- Eventually exhausts pool of records, legitimate TCP requests are ignored

SYN attack countermeasures

- Shorter time-outs on half-open connections
  - Or, dynamically shrink time-out when many half-open connections detected
- Put a SYN verifier in front of server
  - SYN verifier responds to SYN, if gets SYN ACK, then knows SYN is legitimate, and bridges connection to server
  - Has lots of buffers
SYN Verifier difficulties

These will be different sequence numbers. So verifier has to translate seq numbers thereafter.

Must buffer data until bridged connection is established.
SYN Cookie is cleaner approach

- Client
  - SYN
  - SYN ACK + cookie (+ challenge)
  - ACK + cookie (+ response)
  - data

Of course, the problem is that this required changes in Client software!

- SYN Verifier
  - Create “TCP record” but return it in (hashed) cookie. Optionally return a challenge that is computationally hard to compute.
  - If cookie (and response) validate, then now allocate real TCP record.

Login blocking

- Some systems will block a login after a few failed attempts
- Attacker simply writes a script that does incorrect logins for every user
  - Can even to lock out root!
Simple old buffer overflow attacks

- Ping of Death (overlarge ICMP packet)
  - Crashes victim
- NewTear, Newtear2, Bonk, Boink
  - Exploited bug in Windows 9x/NT
  - Locksup or crashes victim
- These generally fixed now

---

Always more buffer overflow attacks…

- Exploit buffer overflow to insert own code into call stack
  - Code Red worm
  - Recent SQL Slammer worm
- These will always exist
  - Search for “security advisory” on cisco.com generated 1100 hits
    - Casual inspection indicated that many were of this sort
  - Install system patches and firewall filters
New security systems bring new attack possibilities

- For instance, Blackice Intrusion Detection System (IDS)
- Operates as process that runs on each host
- Had a bug that it allowed it to accept too many TCP connections
- Attacker could consume up to 400MBytes of system memory

New security systems have their own holes

- When firewall filters for strings, must search for all encodings
  - ASCII, UTF (%xx%xx), or plain hex (%xx)
- Microsoft IIS includes a new encoding that is not an HTTP standard (%u)
- Cisco IDS was not aware of this encoding
- Thus, attacker could bypass IDS by using new encoding
  - Ex: CodeRed worm used the .ida buffer overflow
  - Attacker could encode “.ida” in %u (GET /himom.id%u0061 HTTP/1.0)
Distributed Denial-of-Service

- trin00 (WinTrinoo)
- Tribe Flood Netowrk (TFN) (TFN2k)
- Shaft
- stacheldraht
- Mstream

Trin00

- Affects Windows and many Unix OS’s
- Attacker scans for exploits, gains root, and downloads Trin00 programs.
- Attacker->Master->Daemon hierarchy
  - (One -> More -> Many)
- Attacker can telnet into a Master to initiate commands, which are distributed among its Daemons.
Trin00

- Communication between Master->Daemon through a password-protected cleartext UDP-based protocol.
  - In other variants, Internet Relay Chat is used as the means of communicating with Daemons
- Daemons attack the target with a UDP or TCP packet bombardment.
- Used in the February 2000 attacks on eBay, Amazon, CNN, etc.

Other DDoS

- TFN(2k)
  - Smurf attack, ICMP flood, SYN flood, UDP flood, simultaneous
- Stackeldraht
  - Smurf attack, ICMP flood, SYN flood, UDP flood
- Shaft
  - ICMP flood, SYN flood, UDP flood, simultaneous
Intrusion Detection Systems (IDS)

- Broad range of systems that monitor activity, attempt to flag unusual behavior
  - Changes in volume of traffic
  - Changes in protocols or ports
  - Unusual traffic patterns for a given application
  - Known exploits

- Host based
  - Look through host log files
  - Check integrity of file systems

- Network based (snoop traffic)
  - Either at host or as network monitor

- “Honeypots” (pretend to be exploitable systems, attract hackers)