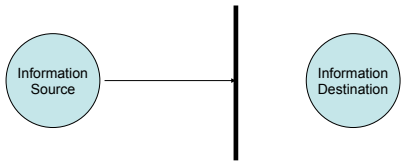
Security

Week 8

Threat: Interruption



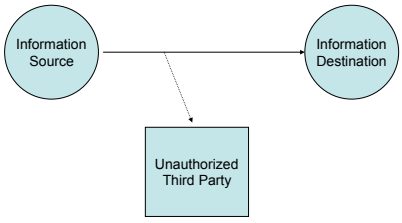
Interruption Threats

- Denial of Service
- Prevents source from sending information to the receiver
- Or receiver from sending requests to source
- A threat to availability

How do Interruption Threats Occur?

- Destruction of hardware, software, or data
- Interference with a communications channel
- Overloading a shared source

Threat: Interception



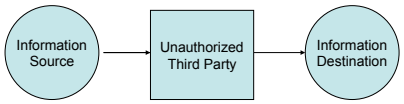
Interception Threats

- Data or services are provided to an unauthorized party
- Threat to secrecy
- Threat to exclusivity

How Do Interception Threats Occur?

- Eavesdropping
- Masquerading
- Break-ins
- Illicit data copying

Threat: Modification



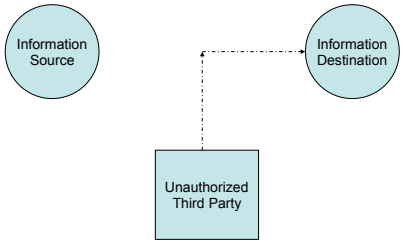
Modification Threats

- Unauthorized parties modify the data
- Either on the way to the users
- Or permanently at the servers
- A threat to integrity

How Do Modification Threats Occur?

- Interception of data requests/replies
- Masquerading
- Break-ins
- Flaws in applications allowing unintended modifications
- Other forms of illicit access to servers and their services

Threat: Fabrication



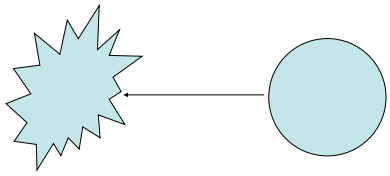
Fabrication Threats

- Unauthorized parties insert counterfeit objects into the system
- Causing improper changes in data
- Or improper use of system resources
- Bad behavior
- Threat to integrity and exclusivity

How Do Fabrication Threats Occur?

- Masquerading
- Bypassing protection mechanisms
- Duplication of legitimate requests/ responses

Destruction Threats



Destruction Threats

- Destroy data, hardware, messages, or software
- Often easier to destroy something than usefully modify it
- Often (but not always) requires physical access

Active vs. Passive Threats

- Passive threats: eavesdroppping
  - No modification, injections of requests, etc
- Active threats: aggressive
- Passive threats are mostly to secrecy
- Active threats are to all properties

Social Engineering

- Best security practices are subverted by bad human practicies
  - Ex: Give passwords over the phone to anyone who asks
- Social engineering are cheap, easy, effective

Social Engineering Example

- Phishing
- Attackers send plausible email requesting you to visit a web site
- To update your information
- Typicall a bank, website, etc
- Attacker controls the site and uses to obtain your credit card, SSN, etc

Authentication

- Determine the ID of some entity
  - Process
  - Machine
  - Human User
- Requires notion of identity
- And some degree of proof of identity

Proving Identity in Physical World

- Face, voice, body, fingerprint, retina
- By recommendation
- Credentials
- Knowledge
- Location

Cyber Identification

- Can't do certain things well like face recognition
- But fast on math computations

Authentication Mechanisms

- Passwords
- Smart cards or tokens
- Biometrics
- Role

Firewalls

- Firewall: a computer that keeps the bad guys out
  - Machine that sits between a LAN and the Internet
  - Regulates network traffic
  - Form of perimeter defense

Kernels and Sandboxes

- Kernel: Protected instructions
- Sandboxes: anything that goes wrong inside the sandbox won't disturb things outside the sandbox

Backups

- If your machine is compromised, restore from a backup!
- Backup methods
  - TimeMachine from Leopard
  - Rsync
  - Manual
- Issues
  - Location: Off-site
  - Frequency
  - Media

Intrusion Detection

- Assumes that sooner or later that your security measures will fail
- Detect improper behavior of intruder
- Inform administrator to take action

Security Policies

- Security is a policy
  - E.g., "no unauthorized user may access this file"
- Protection is a mechanism
  - E.g., "the system checks user identity against permissions"
- Protection mechanisms implement security policies

Why Intrusion Detection?

- If we can detect bad things, can't we simply prevent them?
- Possibly not:
  - Too expensive
  - Too many separate operations
  - Involve Things We Don't foresee

Basics of Intrusion Detection

- Watch what's going on in the system
- Try to detect behavior that characterizes intruders
- While avoiding improper detection of legitimate access
- Hopefully at a reasonable cost

Encryption

- Symmetric: encrypter and decrypter share a secret key used for both encrypting and decrypting
- Asymmetric: encrypter has different key than decrypter
- How do we share keys?

Public Key

- Encrypter and decrypter have different keys
- Keys are created in pairs
- One key is kept secret by the owner
- The other is made public to the world
- If you want to send an encrypted message to someone, encrypt with his public key
  - Only he has private key to decrypt

Authentication with Public Keys

- If I want to "sign" a message, encrypt it with my private key
- Only I know private key, so no one else could create that message
- Everyone knows my public key, so everyone can check my claim directly

Laboratory

- Split up into pairs
- One machine will be the ssh server
- One machine will be the ssh client
- We want to set up passwordless authentication

Laboratory

- Setup SSH server
  - Login as root
  - Set up your ssh server keys
    - Go to /etc/ssh
    - ssh-keygen –t rsa1 –f ssh_host_key –C '' –N ''
    - ssh-keygen –t rsa –f ssh_host_rsa_key –C '' –N ''
    - ssh-keygen –t dsa –f ssh_host_dsa_key –C '' –N ''
    - chmod 600 all private keys
    - chmod 644 all public keys
  - Start server: /usr/sbin/sshd

Laboratory

- Create a new user: easiest way is to use the user manager GUI
- ssh into your server machine to see if your machine is running
  - ssh user@IPAddress

Laboratory

- Generate your client keys
  - ssh-keygen –t dsa –f ~/.ssh/id_dsa "user@IPAddress"
- Copy the public address key to the ~/.ssh/ authorized_keys file of the ssh server
  - cat ~/.ssh/id_dsa.pub | ssh user@IPAddress 'cat - >> ~/.ssh/authorized_keys'

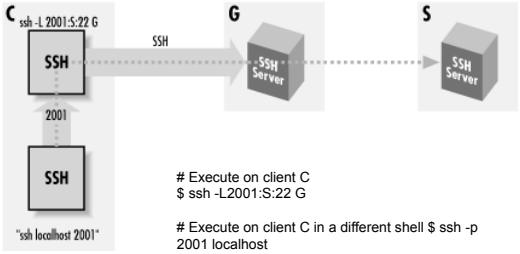
Laboratory

- Verify that DSA authentication works
  - ssh user@IPAddress
- Set up passwordless authentication
  - ssh-agent xterm
  - From the spawned xterm window
    - Add the key: ssh-add ~/.ssh/id_dsa
    - Test it: ssh user@IPaddress

Things to Think About

- If we don't use a password, how did we ensure proper authentication?
- We copied the public key to our remote server, was that safe?

Port Forwarding



# Execute on client C
$ ssh -L2001:S:22 G

Execute on client C in a different shell $ ssh -p 2001 localhost

Advantages

- You just encrypted all traffic!
- Makes unaccessible machines accessible

Forwarding X-Windows

- ssh –X user@IPAddress
  - Will allow all X-window sessions to display on your screen instead of the remote server