Security

Reading: Chapter 15, [OSC]
(except Section 15.9)

Outline

• The Security Problem
• Program Threats
• System and Network Threats
• Cryptography as a Security Tool
• User Authentication
• Implementing Security Defenses
• Firewalling to Protect Systems and Networks
• Computer-Security Classifications
The Security Problem

- System **secure** if resources used and accessed as intended under all circumstances
  - Unachievable
- Intruders (**crackers**) attempt to breach security
- **Threat** is potential security violation
- **Attack** is attempt to breach security
- Attack can be accidental or malicious
- Easier to protect against accidental than malicious misuse
Security Violation Categories

- **Breach of confidentiality**: Unauthorized reading of data
- **Breach of integrity**: Unauthorized modification of data
- **Breach of availability**: Unauthorized destruction of data
- **Theft of service**: Unauthorized use of resources
- **Denial of service (DoS)**: Prevention of legitimate use
Security Violation Methods

• **Masquerading** (breach of authentication): Pretending to be an authorized user to escalate privileges

• **Replay attack**: As is or with message modification

• **Man-in-the-middle attack**: Intruder sits in data flow, masquerading as sender to receiver and vice versa

• **Session hijacking**: Intercept an already-established session to bypass authentication
Standard Security Attacks

Normal

sender

communication

receiver

attacker

Masquerading

sender

communication

receiver

attacker

Man-in-the-middle

sender

communication

attacker

communication

receiver
Security Measure Levels

• Impossible to have absolute security, but make cost to perpetrator sufficiently high to deter most intruders

• Security must occur at four levels to be effective:
  – **Physical**: Data centers, servers, connected terminals
  – **Human**: Avoid *social engineering, phishing, dumpster diving*
  – **Operating Systems**: Protection mechanisms, debugging
  – **Network**: Intercepted communications, interruption, DoS

• Security is as weak as the weakest link in the chain

• Can too much security be a problem?
Software Attacks (1)

• Many variations, many names

• **Trojan Horse**
  – Code segment that misuses its environment
  – Exploits mechanisms for allowing programs written by users to be executed by other users
  – **Spyware, pop-up browser windows, covert channels**
  – Up to 80% of spam delivered by spyware-infected systems

• **Trap Door**
  – Specific user identifier or password that circumvents normal security procedures
  – Could be included in a compiler
  – How to detect them?
Software Attacks (2)

- **Logic Bomb**: Program that initiates a security incident under certain circumstances

- **Stack and Buffer Overflow**
  - Exploits a bug in a program (overflow either the stack or memory buffers)
  - Failure to check bounds on inputs, arguments
  - Write past arguments on the stack into the return address on stack
  - When routine returns from call, returns to hacked address
    - Pointed to code loaded onto stack that executes malicious code
  - Unauthorized user or privilege escalation
C Program with Buffer Overflow

```c
#include <stdio.h>
#define BUFFER SIZE 256
int main(int argc, char *argv[])
{
    char buffer[BUFFER SIZE];
    if (argc < 2)
        return -1;
    else {
        strcpy(buffer,argv[1]);
        return 0;
    }
}
```
Layout of Typical Stack Frame
#include <stdio.h>

int main(int argc, char *argv[])
{
    execvp("\bin\sh","\bin \sh", NULL);
    return 0;
}

Modified Shell Code
Hypothetical Stack Frame

Before attack

After attack
Great Programming Required?

- For the first step of determining the bug, and second step of writing exploit code, yes
- **Script kiddies** can run pre-written exploit code to attack a given system
- Attack code can get a shell with the processes’ owners’ permissions
  - Or open a network port, delete files, download a program, etc.
- Depending on bug, attack can be executed across a network using allowed connections, bypassing firewalls
- Buffer overflow can be disabled by disabling stack execution or adding bit to page table to indicate “non-executable” state
  - Available in SPARC and x86
  - But still have security exploits
Software Attacks (3)

• **Viruses**
  – Code fragment embedded in legitimate program
  – Self-replicating, designed to infect other computers
  – Very specific to CPU architecture, operating system, applications
  – Usually borne via email or as a macro
  – Visual Basic Macro to reformat hard drive

```vba
Sub AutoOpen()
    Dim oFS
    Set oFS = CreateObject("Scripting.FileSystemObject")
    vs = Shell("c:command.com /k format c:", vbHide)
End Sub
```
Software Attacks (4)

- **Virus dropper** inserts virus onto the system
- Many categories of viruses, literally many thousands of viruses
  - File / parasitic
  - Boot / memory
  - Macro
  - Source code
  - Polymorphic to avoid having a **virus signature**
  - Encrypted
  - Stealth
  - Tunneling
  - Multipartite
  - Armored
Boot-Sector Computer Virus

1. Virus copies boot sector to unused location X
2. Virus replaces original boot block with itself
3. At system boot, virus decreases physical memory, hides in memory above new limit
4. Virus attaches to disk read-write interrupt, monitors all disk activity
5. Whenever new removable R/W disk is installed, it infects that as well
6. It blocks any attempts of other programs to write the boot sector
7. It has a logic bomb to wreak havoc at a certain date
Software Attacks (5)

• Attacks still common, still occurring
• Attacks moved over time from science experiments to tools of organized crime
  – Targeting specific companies
  – Creating botnets to use as tool for spam and DDOS delivery
  – Keystroke logger to grab passwords, credit card numbers
• Why is Windows the target for most attacks?
  – Most common
  – Everyone is an administrator
    • Licensing required?
  – Monoculture considered harmful
Network Attacks (1)

• Some systems “open” rather than secure by default
  – Reduce attack surface
  – But harder to use, more knowledge needed to administer
• Network threats harder to detect, prevent
  – Protection systems weaker
  – More difficult to have a shared secret on which to base access
  – No physical limits once system attached to internet
    • Or on network with system attached to internet
  – Even determining location of connecting system difficult
    • IP address is only knowledge
Network Attacks (2)

- **Worms**: standalone programs that self-propagate
- Morris worm (1988)
  - Exploited UNIX networking features (remote access) and bugs in `finger` and `sendmail` programs
  - Exploited trust-relationship mechanism used by `rsh` to access friendly systems without use of password
  - **Grappling hook** program uploaded main worm program
    - 99 lines of C code
  - Hooked system then uploaded main code, tried to attack connected systems
  - Also tried to break into other users accounts on local system via password guessing
  - If target system already infected, abort, except every 7th time
The Morris Worm

grappling hook

rsh attack
finger attack
sendmail attack

request for worm

worm sent

worm

target system
infected system
Network Threats (3)

• Port scanning
  – Automated attempt to connect to a range of ports on one or a range of IP addresses
  – Detection of answering service protocol
  – Detection of OS and version running on system
  – nmap scans all ports in a given IP range for a response
  – nessus has a database of protocols and bugs (and exploits) to apply against a system
  – Frequently launched from zombie systems to decrease traceability
Network Threats (4)

• Denial of Service
  – Overload the targeted computer preventing it from doing any useful work
  – Distributed denial-of-service (DDoS) come from multiple sites at once
  – Consider the start of TCP 3-way handshake (SYN)
    • How many started connections can the OS handle?
  – Consider traffic to a web site
    • How can you tell the difference between being a target and being really popular?
  – Accidental: CS students writing bad `fork()` code
  – Purposeful: extortion, punishment
Sobig.F Worm

- More modern example
- Disguised as a photo uploaded to adult newsgroup via account created with stolen credit card
- Targeted Windows systems
- Had own SMTP engine to mail itself as attachment to everyone in infect system’s address book
- Disguised with innocuous subject lines, looking like it came from someone known
- Attachment was executable program that created `WINPPR23.EXE` in default Windows system directory
- Plus the Windows Registry

```
[HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Run]
  "TrayX" = %windir%\winppr32.exe /sinc

[HKLMMSOFTWARE\Microsoft\Windows\CurrentVersion\Run]
  "TrayX" = %windir%\winppr32.exe /sinc
```
Cryptography (1)

• Broadest security tool available
  – Internal to a given computer, messages’ sources and destinations can be known, protected
    • OS creates, manages, protects process IDs, communication ports
  – Source and destination of messages on network cannot be trusted without cryptography
    • Local network: IP address? Unauthorized host added
    • WAN/Internet: how to establish authenticity? IP addresses can be spoofed!
Cryptography (2)

- Means to constrain potential senders (sources) and/or receivers (destinations) of messages
  - Based on secrets (keys)
  - Enables
    - Confirmation of source
    - Receipt only by certain destination
    - Trust relationship between sender and receiver
Encryption (1)

- Constrains the set of possible receivers of a message
- **Encryption** algorithm consists of
  - Sets: $K$ of keys, $M$ of messages, $C$ of ciphertexts (encrypted messages)
  - A function $E : K \rightarrow (M \rightarrow C)$. That is, for each $k \in K$, $E_k$ is a function for generating ciphertexts from messages
    - Both $E$ and $E_k$ for any $k$ should be efficiently computable functions
  - A function $D : K \rightarrow (C \rightarrow M)$. That is, for each $k \in K$, $D_k$ is a function for generating messages from ciphertexts
    - Both $D$ and $D_k$ for any $k$ should be efficiently computable functions
Encryption (2)

• An encryption algorithm must provide this essential property: Given a ciphertext $c \in C$, a computer can compute $m$ such that $E_k(m) = c$ only if the computer has $k$
  
  – Thus, a computer holding $k$ can decrypt ciphertexts to the plaintexts used to produce them, but a computer not holding $k$ cannot decrypt ciphertexts
  
  – Since ciphertexts are generally exposed (e.g., sent on the network), it must be infeasible to derive $k$ from the ciphertexts
Symmetric Encryption

- Same key $k$ used to encrypt and decrypt; hence $k$ must be secret
- DES was commonly used symmetric block-encryption algorithm (created by U.S. Gov.)
  - Encrypts a block of data at a time
  - Keys too short so now considered insecure
- Triple-DES considered more secure
  - Algorithm used 3 times using 2 or 3 keys
  - For example: $c = E_{k3}(D_{k2}(E_{k1}(m)))$
  - Keys of 128, 192, or 256 bits; works on 128 bit blocks
- RC4: most common symmetric stream cipher, but has vulnerabilities
  - Encrypts/decrypts a stream of bytes (i.e., wireless transmission)
  - Key is a input to pseudo-random-bit generator that generates an infinite keystream
Secure Communication over Insecure Medium
Asymmetric Encryption (1)

• **Public-key encryption** based on each user having two keys:
  – **Public key** – published key used to encrypt data
  – **Private key** – key known only to individual user used to decrypt data

• Must be encryption scheme that can be made public without “giving away” decryption scheme
  – Most common: **RSA** block cipher
  – Efficient algorithm for testing whether a number is prime
  – No efficient algorithm known that finds prime factors of numbers
Asymmetric Encryption (2)

• Formally, it is computationally infeasible to derive $k_{d,N}$ from $k_{e,N}$, and so $k_e$ need not be kept secret and can be widely disseminated
  – $k_e$ is the **public key**
  – $k_d$ is the **private key**
  – $N$ is the product of two large, randomly chosen prime numbers $p$ and $q$ (e.g., $p$ and $q$ are 512 bits each)
  – Encryption algorithm is $E_{ke,N}(m) = m^{ke} \mod N$, where $k_e$ satisfies $k_e k_d \mod (p - 1)(q - 1) = 1$
  – Decryption algorithm is then: $D_{kd,N}(c) = c^{kd} \mod N$
Asymmetric Encryption Example

1. Choose \( p = 7 \) and \( q = 13 \)
2. Calculate \( N = 7 \times 13 = 91 \) and \( (p - 1)(q - 1) = 72 \)
3. Select \( k_e \) relatively prime to 72 where \( k_e < 72 \), yielding 5
4. Finally, we calculate \( k_d \) such that \( k_e k_d \mod 72 = 1 \), yielding 29

- We now have our keys:
  - Public key, \( k_{e,N} = 5, 91 \)
  - Private key, \( k_{d,N} = 29, 91 \)

- Encrypting message 69 with public key yields ciphertext 62
- Ciphertext can be decoded with the private key
  - Public key can be distributed in cleartext to anyone who wants to communicate with holder of public key
Encryption using RSA Asymmetric Cryptography

1. Write message 69
2. Encrypt message using key $k_{69,91}$
3. Transmit encrypted message over insecure channel
4. Decrypt message using key $k_{29,91}$
5. Read the decrypted message 69
Cryptography (3)

• Note symmetric cryptography based on transformations, asymmetric based on mathematical functions
  – Asymmetric much more computationally intensive
  – Typically not used for bulk data encryption
Authentication (1)

- Constraining set of potential senders of a message
  - Complementary to encryption
  - Also can prove message unmodified
- Algorithm components
  - Sets $K$ of keys, $M$ of messages, $A$ of authenticators
  - A function $S : K \to (M \to A)$
    - That is, for each $k \in K$, $S_k$ is a function for generating authenticators from messages
    - Both $S$ and $S_k$ for any $k$ should be efficiently computable functions
  - A function $V : K \to (M \times A \to \{\text{true, false}\})$. For each $k \in K$, $V_k$ is a function that verifies authenticators on messages
    - Both $V$ and $V_k$ for any $k$ should be efficiently computable functions
Authentication (2)

• For a message $m$, a computer can generate an authenticator $a \in A$ such that $V_k(m, a) = \text{true}$ only if computer has $k$
• Thus, computer holding $k$ can generate authenticators on messages so that any other computer possessing $k$ can verify them
• Computer not holding $k$ cannot generate authenticators on messages that can be verified using $V_k$
• Since authenticators are generally exposed (e.g., they are sent on the network with the messages themselves), it must not be feasible to derive $k$ from the authenticators
• Practically, if $V_k(m, a) = \text{true}$ then we know $m$ has not been modified and that send of message has $k$
  – If we share $k$ with only one entity, we know message origin
Authentication – Hash Functions

• Basis of authentication
• Creates small, fixed-size block of data message digest (hash value) from \( m \)
• Hash function \( H \) must be collision resistant on \( m \) [infeasible to find \( m' \neq m \) where \( H(m') = H(m) \)]
• If \( H(m) = H(m') \), then \( m = m' \)
• Common hash functions include \textbf{MD5} and \textbf{SHA-1}, which output 128-bit and 160-bit hashes, respectively
• Not useful as authenticators.
  – \( H(m) \) can be sent with a message
  – But if \( H \) is known someone could modify \( m \) to \( m' \) and recompute \( H(m') \); modification not detected!
  – Hence, must authenticate \( H(m) \)
Authentication - MAC

• Symmetric encryption used in message-authentication code (MAC) authentication algorithm
• Cryptographic checksum generated from message using secret key
  – Can securely authenticate short values
• If used to authenticate $H(m)$ for an $H$ that is collision resistant, then obtain a way to securely authenticate long message by hashing them first
• Note that $k$ is needed to compute both $S_k$ and $V_k$, so anyone able to compute one can compute the other
Authentication – Digital Signature

- Based on asymmetric keys and digital signature algorithm
- Authenticators produced are **digital signatures**
- Very useful – *anyone* can verify authenticity of a message
- In a digital-signature algorithm, computationally infeasible to derive $k_s$ from $k_v$
  - $V$ is a one-way function
  - Thus, $k_v$ is the public key and $k_s$ is the private key
- Consider the RSA digital-signature algorithm
  - Similar to the RSA encryption algorithm, but the key use is reversed
  - Digital signature of message $S_{ks}(m) = H(m)^{ks} \mod N$
  - The key $k_s$ again is a pair $(d, N)$, where $N$ is the product of two large, randomly chosen prime numbers $p$ and $q$
  - Verification algorithm is $V_{kv}(m, a) \triangleq (a^{k_v} \mod N = H(m))$
  where $k_v$ satisfies $k_v k_s \mod (p − 1)(q − 1) = 1$
Authentication (3)

• Why is authentication a subset of encryption?
  – Fewer computations (except for RSA digital signatures)
  – Authenticator usually shorter than message
  – Sometimes want authentication but not confidentiality (e.g., signed patches)
  – Can be basis for non-repudiation
Key Distribution

• Delivery of symmetric key is huge challenge
  – Sometimes done out-of-band
• Asymmetric keys can proliferate – stored on key ring
  – Even asymmetric key distribution needs care: man-in-the-middle attack
Digital Certificates

- Proof of who or what owns a public key
- Public key digitally signed by a trusted party
- Trusted party receives proof of identification from entity and certifies that public key belongs to entity

- **Certificate authorities** are trusted parties; their public keys included with web browsers
  - They vouch for other authorities via digitally signing their keys, and so on
Man-in-the-middle Attack on Asymmetric Cryptography
Implementation of Cryptography

• Can be done at various **layers** of ISO Reference Model
  – TLS at transport layer
  – Network layer: **IPSec**
    • IKE for key exchange
    • Basis of **Virtual Private Networks (VPNs)**

• Why not just at lowest level?
  – Sometimes need more knowledge than available at low levels
    (e.g., user authentication, email delivery)

Encryption Example - TLS

- Insertion of cryptography at one layer of the ISO network model (the transport layer)
- TLS – Transport Layer Security
- Cryptographic protocol that limits two computers to only exchange messages with each other (complicated!)
- Used between web servers and browsers for secure communication (credit card numbers)
- The server is verified with a certificate assuring client is talking to correct server
- Asymmetric cryptography used to establish a secure session key (symmetric encryption) for bulk of communication during session
- Communication between each computer then uses symmetric key cryptography
User Authentication

- Crucial to identify user correctly: protection systems depend on user ID
- User identity often established through passwords, can be considered a special case of either keys or capabilities
- Passwords must be kept secret
  - Frequent change of passwords
  - History to avoid repeats
  - Use of “non-guessable” passwords
  - Log all invalid access attempts (but not the passwords themselves)
  - Unauthorized transfer
- Passwords may be encrypted or allowed to be used only once
  - Does encrypting passwords solve the exposure problem?
    - Might solve sniffing
    - Consider shoulder surfing, Trojan horse keystroke logger
    - How are passwords stored at authenticating site?
Passwords

• Encrypt to avoid having to keep secret
  – But keep secret anyway (Unix uses superuser-only file /etc/shadow)
  – Use algorithm easy to compute but difficult to invert
  – Only encrypted password stored, never decrypted
  – Add “salt” to avoid the same password being encrypted to the same value

• One-time passwords
  – Use a function based on a seed to compute a password, both user and computer
  – Hardware device / calculator / key fob to generate the password (changes frequently)

• Biometrics: a physical attribute (fingerprint, hand scan)

• Multi-factor authentication
  – Need 2+ factors for authentication (e.g., fingerprint scan and password)
Implementing Security Defenses

- **Defense in depth**: most common security theory (multiple layers of security)
- **Security policy** describes what is being secured
- Vulnerability assessment compares real state of system/network to policy
- Intrusion detection tries to detect attempted or successful intrusions
  - **Signature-based** detection spots known bad patterns
  - **Anomaly detection** spots differences from normal behavior (detects *zero-day* attacks)
  - False positives and false negatives
- Virus protection: searches all programs (at run-time) for known virus patterns or runs them in **sandboxes**
- Audit, account for, and log all system/network activities
- Practice **safe computing**: avoid sources of infection, download from only “good” sites, etc.
Firewalls: Protect Systems and Networks

• A network **firewall** is placed between trusted and untrusted hosts
  – The firewall limits network access between these two **security domains**

• Can be tunneled or spoofed
  – Tunneling allows disallowed protocol to travel within allowed protocol (i.e., telnet inside of HTTP)
  – Firewall rules typically based on host name or IP address which can be spoofed

• **Personal firewall** is software layer on given host
  – Can monitor / limit traffic to and from the host

• **Application proxy firewall** understands application protocol and can control them (i.e., SMTP)

• **System-call firewall** monitors all important system calls and apply rules to them (i.e., this program can execute that system call)
Network Security via Firewalled Domain Separation

Internet access from company's computers

DMZ access from Internet

firewall

access between DMZ and company's computers
Computer Security Classifications

• U.S. Department of Defense outlines four divisions of computer security: A, B, C, and D

• D – Minimal security

• C – Provides discretionary protection through auditing
  – Divided into C1 and C2
    • C1 identifies cooperating users with the same level of protection
    • C2 allows user-level access control

• B – All the properties of C, however each object may have unique sensitivity labels
  – Divided into B1, B2, and B3

• A – Uses formal design and verification techniques to ensure security
Thank You

Questions and comments?