Lecture 4 - Malicious Software

Networks and System Security

Two-Factor Authentication (2FA) Recap

Definition & Purpose

- **2FA**: Authentication using two distinct forms of evidence from different categories
- **Goal**: Add second defense layer - even if password is stolen, unauthorized access is harder
- **MFA**: Multi-Factor Authentication extends to three or more factors

Three Authentication Factors

1. **Something you know** - Password, PIN, security question (knowledge-based)
2. **Something you have** - Smartphone, smartcard, hardware token (possession-based)
3. **Something you are** - Fingerprint, Face ID, retina scan (inherence-based)

Common 2FA Methods

1. SMS-based codes

- One-time code sent to registered phone
- Vulnerable to SIM swapping and interception

2. App-based (TOTP)

- Time-based One-Time Password
- Apps: Google Authenticator, Authy, Microsoft Authenticator
- Offline, more secure than SMS

3. Push notifications

- "Approve login" prompts (Duo, Okta, Microsoft)
- Convenient but risk of push fatigue

4. Hardware tokens

- Physical devices: YubiKey, RSA SecurID
- Cryptographically strong authentication

5. Biometric 2FA

- Fingerprint/facial recognition paired with password/PIN
- Privacy and device limitations may apply

How TOTP Works

- Formula: **Shared Secret + Current Time → HMAC-SHA1 → Truncate → 6-digit Code**
- Based on shared secret key between client and server
- Uses current time and key to generate one-time 6-digit code
- Valid for short intervals (typically 30 seconds)
- Standards: RFC 6238 (TOTP) and RFC 4226 (HOTP)

Advantages of 2FA

- Dramatically reduces unauthorized access from stolen passwords
- Simple, low-cost protection
- Builds user confidence
- Compatible with most major platforms

Limitations & Risks

- Phishing kits can capture both password and OTP
- Man-in-the-middle attacks may intercept login flow
- SIM swapping makes SMS 2FA unreliable
- User fatigue → careless approvals
- Device loss may lock out legitimate users

Best practice: Prefer app-based or hardware-based 2FA; avoid SMS when possible

Lecture 4 - Malicious Software

Real-World Examples

- **Google (2018)**: Mandated 2FA for employees; zero account takeovers afterwards
- **GitHub (2023)**: Made 2FA mandatory for all contributors
- Banks & healthcare: Use token/app-based MFA due to regulations

SMS-Based 2FA Deep Dive

Security Model Assumption

- SMS assumed to provide "separate channel" from internet
- Relies on telecommunications network as trusted intermediary
- Problem: Mobile network and SIM system not designed for high-security authentication

Attack Vectors:

1. **SIM Swapping**
    - Attacker convinces carrier to transfer victim's number to their SIM
    - All SMS messages (including OTPs) sent to attacker
    - Attack surface: Social engineering, weak carrier verification
    - Example: Twitter CEO Jack Dorsey (2019) compromised via SIM swapping
2. **Man-in-the-Middle (MitM) Phishing**
    - Attacker tricks user into logging in on fake website
    - Site proxies credentials and SMS code to legitimate site in real time
    - Tools: Evilginx, Modlishka
    - Both password and OTP stolen during session
3. **SS7 Network Exploits**
    - Signaling System 7 underpins SMS delivery
    - Vulnerabilities allow SMS traffic interception at telecom level

Lecture 4 - Malicious Software

- Exploited repeatedly since 2014

Usability vs. Security Trade-off:

Aspect	Advantage	Weakness
Accessibility	Works on any phone, no app needed	Relies on insecure SMS protocol
User adoption	Simple and familiar	High risk from phishing and SIM fraud
Cost	Cheap to deploy	Expensive to mitigate breaches

Critical Assessment:

- SMS 2FA improves over password-only systems
- Provides false sense of strong protection
- Vulnerable to human and infrastructural manipulation
- Inadequate against targeted attacks
- **Recommendation**: Use SMS 2FA only as temporary/fallback; migrate to app-based or hardware-based

Beyond 2FA: Future Authentication

- **FIDO2 / WebAuthn**: Public-key cryptography and hardware tokens
- **Passkeys**: Modern, phishing-resistant alternative to passwords
- Combine biometrics and hardware authentication

Malicious Software (Malware)

Core Terminology

Self-Replicating:

Lecture 4 - Malicious Software

- **Virus**: Attaches to executable file, replicates when infected code runs. Requires host program and user execution
- **Worm**: Complete, independent program that self-propagates across network without needing host

Deception & Concealment:

- **Trojan Horse**: Malicious software disguised as legitimate program. Relies on deception
- **Backdoor (Trapdoor)**: Hidden mechanism to bypass security checks
- **Rootkit**: Hacker tools to conceal compromise and maintain privileged access

Triggered Malice:

- **Logic Bomb**: Dormant program that executes when predefined condition is met (date, file deletion, user action)

System Compromise:

- **Exploit**: Code written to leverage specific vulnerability
- **Keylogger**: Records every keystroke, capturing passwords

Attack Infrastructure:

- **Zombie/Bot**: Infected machine activated to launch attacks, part of botnet

Delivery & Installation:

- **Downloader**: Program that installs other malicious items
- **Auto-rooter**: Tool to break into machines remotely and gain admin access
- **Kit (Virus Generator)**: Tools for automatic generation of new viruses

Monetization & Tracking:

- **Spyware**: Collects information about user activity, transmits without knowledge
- **Adware**: Displays/downloads unwanted advertising

Denial of Service:

- **Flooder (DoS/DDoS Tool)**: Generates massive traffic to overwhelm resources
- **Spammer Programs**: Send extremely large volumes of unsolicited email

Lecture 4 - Malicious Software

Platform-Agnostic:

- **Mobile Code**: Platform-independent software (scripts, macros) that executes across different OS

Malware Classification

By Propagation Mechanism:

1. Infection of existing executable by viruses
2. Exploit of software vulnerabilities by worms or drive-by-downloads
3. Social engineering attacks (trojans, phishing)

Traditional Distinctions:

- Need host program (parasitic) vs. independent programs
- Does not replicate vs. does replicate

By Payload Actions:

- Corruption of system/data files
- Theft of service (make system zombie agent, botnet)
- Theft of information (keylogging, spyware)
- Stealthing (hide presence)
- **Blended attack**: Uses multiple methods for maximum speed and severity

Attack Kits & Crimeware

- Early malware required considerable technical skill
- Changed with virus-creation toolkits (early 1990s) and attack kits (2000s)
- Known as **crimeware**
- Include propagation mechanisms and payload modules
- Even novices can combine, select, and deploy
- Easily customized with latest vulnerabilities
- Greatly enlarged population of attackers

Attack Sources Evolution

Lecture 4 - Malicious Software

- Changed from individuals to organized groups:
    - Politically motivated attackers
    - Criminals and organized crime
    - Organizations selling services to companies and nations
    - National government agencies
- Led to development of large underground economy
- Sale of attack kits, compromised hosts, stolen information

Advanced Persistent Threat (APT)

- Well-resourced, persistent application of intrusion technologies
- Target: Usually business or political
- Characteristics:
    - **Advanced**: Components carefully selected for chosen target
    - **Persistent**: Determined application over extended period
    - **Threats**: Organized, capable, well-funded attackers
- Examples: Aurora, RSA, APT1, Stuxnet
- Differs from other attacks by careful target selection and stealthy intrusion efforts

Viruses

Definition

- Parasitic software fragments that attach to existing executable content
- Can infect other programs and modify them
- Modification includes injecting code to make copies
- Dominated early malware scene due to lack of user authentication and access controls on PCs

Virus Structure Components

Lecture 4 - Malicious Software

1. **Infection Mechanism**: How virus spreads/propagates (infection vector)
2. **Trigger**: Event/condition that activates payload (logic bomb)
3. **Payload**: What virus does besides spreading (damage or noticeable activity)

Virus Phases

1. **Dormant phase**: Virus is idle, will be activated by some event (not all viruses have this)
2. **Propagation phase**: Places copy of itself onto other programs or system areas on disk
3. **Triggering phase**: Activated to perform intended function (caused by system events)
4. **Execution phase**: Function is performed

Virus Vulnerability

- Initially infects single program
- Once executed, spreads to other files depending on permissions
- Only guaranteed prevention: complete blocking (almost impossible)
- Virus can hide inside any external software
- Nearly every system is vulnerable

Key Limitation: Deny standard users permission to modify existing programs

Virus Classification by Target

- **Boot sector infector**: Infects master boot record, spreads when system boots from infected disk
- **File infector**: Infects files OS/shell considers executable
- **Macro virus**: Infects files with macro/scripting code interpreted by application
- **Multipartite virus**: Infects files in multiple ways

Virus Classification by Concealment Strategy

Encrypted virus:

- Creates random encryption key

- Encrypts remainder of virus
- Different key for each instance = no constant bit pattern

Stealth virus:

- Explicitly designed to hide from antivirus software
- Entire virus hidden, not just payload

Polymorphic virus:

- Mutates with every infection
- Makes signature detection impossible

Metamorphic virus:

- Mutates with every infection
- Rewrites itself completely at each iteration
- May change behavior as well as appearance
- Most difficult to detect

Macro and Scripting Viruses

Why Threatening:

- Platform independent
- Infect documents, not executable code
- Easily spread through normal document sharing
- Traditional file system access controls of limited use (infect user documents, not system programs)

Worms

Definition

- Program that actively seeks out more machines to infect
- Upon activation, may replicate and propagate again
- Replicates using various access mechanisms:

- Email or instant messenger
- File sharing
- Remote execution capability
- Remote file access/transfer
- Remote login capability

Worm Phases

Same as virus: Dormant → Propagation → Triggering → Execution

Propagation Phase Functions:

1. Search for appropriate access mechanisms (host tables, address books, buddy lists, trusted peers)
2. Use access mechanisms to transfer copy to remote system
3. Cause copy to be run

Target Discovery Strategies

Scanning/Fingerprinting: Function to search for other systems to infect

Network Scanning Strategies:

1. **Random**
    - Each host probes random IP addresses, different seed
    - Produces high volume of Internet traffic
    - May cause generalized disruption
2. **Hit list**
    - Attacker compiles long list of vulnerable machines
    - Each infected machine gets portion of list
    - Results in very short scanning period
    - Difficult to detect
3. **Topological**
    - Uses information on infected machine to find more hosts

- Leverages existing network relationships

4. **Local subnet**
    - If host infected behind firewall, looks for targets in local network
    - Uses subnet address structure
    - Targets otherwise protected by firewall

The Morris Worm (1988)

- Released by Robert Morris
- Designed to spread on UNIX systems
- Discovery: Found other hosts known to infected host
- **Exploitation methods:**
    1. Attempted to log on as legitimate user
    2. Exploited bug in UNIX finger protocol
    3. Exploited trapdoor in debug option of mail process

Modern Worm Technology

Multiplatform: Attack variety of platforms

Multi-exploit: Penetrate systems in variety of ways (Web servers, browsers, email, file sharing, etc.)

Ultrafast spreading: Optimize rate of spread to maximize vulnerable machine discovery

Polymorphic: Each copy has new code generated on-the-fly using functionally equivalent instructions and encryption

Metamorphic: Change appearance AND have repertoire of behavior patterns unleashed at different propagation stages

Transport vehicles: Ideal for spreading wide variety of malicious payloads

Zero-day exploit: Exploit unknown vulnerability discovered only when worm is launched

---

Mobile Code

Lecture 4 - Malicious Software

Definition

- Programs shipped unchanged to heterogeneous platforms
- Execute with identical semantics
- Transmitted from remote to local system
- Executed without user's explicit instruction

Popular Vehicles:

- Java applets
- ActiveX
- JavaScript
- VBScript

Malicious Operations

- Cross-site scripting
- Interactive and dynamic Web sites
- Email attachments
- Downloads from untrusted sites/software

Client-Side Vulnerabilities

Drive-by-Downloads

- Exploits browser vulnerabilities
- When user views attacker-controlled Web page, code exploits browser bug
- Downloads and installs malware without user's knowledge/consent
- Does not actively propagate (unlike worm)
- Waits for unsuspecting users to visit malicious page

Watering-hole attacks:

- Variant of drive-by-download for highly targeted attacks

Lecture 4 - Malicious Software

- Attacker researches intended victims
- Identifies Web sites victims likely to visit
- Scans sites for vulnerabilities
- Compromises sites with drive-by-download

Malvertising:

- Place malware on Web sites without compromising them
- Attacker pays for advertisements likely placed on target sites
- Advertisements incorporate malware

Clickjacking (UI Redress Attack)

- Vulnerability to collect infected user's clicks
- Force user to do various things:
    - Adjust computer settings
    - Unwittingly send to malicious sites
- Uses multiple transparent/opaque layers
- Tricks user into clicking button/link on different page
- Can also hijack keystrokes
- Uses stylesheets, iframes, text boxes
- User believes typing password into legitimate site but typing into invisible frame

---

Spam

Definition & Impact

- Unsolicited bulk email
- Imposes costs on network infrastructure and users
- Most recent spam sent by botnets using compromised systems
- Significant carrier of malware
- Used in phishing attacks

Lecture 4 - Malicious Software

Note: Requires user's active choice (view email, attached document, install program) for compromise

Trojan Horses

Definition

- Useful or apparently useful program containing hidden malicious code
- When invoked, performs unwanted/harmful function
- Accomplishes functions attacker could not do directly

Three Models:

1. Continue performing original function + separate malicious activity
2. Continue performing original function but modified to perform malicious activity or disguise other malicious activity
3. Perform malicious function that completely replaces original function

Payload Types

System Corruption

Actions:

- Data destruction when trigger conditions met
- Display unwanted messages/content
- **Ransomware**: Encrypt user data, demand payment for decryption key
- Inflict real-world damage on system
- Rewrite BIOS code
- Target specific industrial control system software
- **Logic bomb**: Code set to "explode" when conditions met

Attack Agent

- Subverts computational and network resources

Bot (robot, zombie, drone):

- Secretly takes over Internet-attached computer
- Uses computer to launch/manage attacks difficult to trace
- **Botnet**: Collection of bots acting in coordinated manner

Bot Uses:

- Distributed denial-of-service (DDoS) attacks
- Spamming
- Sniffing traffic
- Keylogging
- Spreading new malware
- Installing advertisement add-ons and browser helper objects (BHOs)
- Attacking IRC networks
- Manipulating online polls/games

Remote Control Facility:

- Distinguishes bot from worm
- Worm propagates and activates itself; bot is controlled from central facility
- Typical implementation: IRC server
- Recent botnets: Covert communication via HTTP
- Distributed control: Peer-to-peer protocols (avoid single point of failure)
- Control module can activate bots and issue update commands

Information Theft

Keylogger: Captures keystrokes to monitor login/password credentials

Spyware: Subverts machine to monitor wide range of activity, compromising personal information

Phishing: Exploits social engineering, masquerades as communication from trusted source

Lecture 4 - Malicious Software

Spear-phishing: Email claiming to be from trusted source, but recipients carefully researched and each email crafted specifically for recipient

Stealthing

Backdoor (Trapdoor):

- Secret entry point into program
- Allows access without going through usual security procedures
- Code recognizes special input sequence or triggered by certain user ID
- Usually implemented as network service on nonstandard port
- Attacker can connect and issue commands

Rootkit:

- Programs installed to maintain covert access with administrator privileges
- Hides evidence of presence
- Alters host's standard functionality in malicious and stealthy way
- Attacker has complete control: add/change programs, monitor processes, send/receive traffic, backdoor access
- Hides by subverting monitoring mechanisms

Rootkit Classification:

- **Persistent**: Activates each time system boots
- **Memory-based**: No persistent code, cannot survive reboot
- **User mode**: Intercepts calls to APIs, modifies returned results
- **Kernel mode**: Intercepts calls to native APIs in kernel mode
- **Virtual machine based**: Installs lightweight VM monitor, runs OS in virtual machine above it
- **External mode**: Malware located outside normal operation mode (BIOS, system management mode), direct hardware access

---

Countermeasures

Lecture 4 - Malicious Software

Prevention Elements

1. Vulnerability Mitigation:

- Ensure all systems current with all patches applied
- Reduces number of exploitable vulnerabilities

2. Policy:

- Set appropriate access controls on applications and data
- Reduce number of files any user can access
- Limits potential infection/corruption from executing malware

3. Awareness:

- Counter social engineering with user awareness and training

Threat Mitigation (if prevention fails)

Technical mechanisms support:

- Detection
- Identification
- Removal

Requirements for Effective Countermeasures:

- Generality
- Timeliness
- Resiliency
- Minimal denial-of-service costs
- Transparency
- Global and local coverage

Malware Detection & Defense

Host-Based Scanners

Lecture 4 - Malicious Software

Four Generations of Antivirus:

1st Generation - Simple scanners:

- Requires malware signature to identify malware

2nd Generation - Heuristic scanners:

- Uses heuristic rules to search for probable malware instances
- Includes integrity checking

3rd Generation - Activity traps:

- Memory-resident programs
- Identify malware by actions rather than structure

4th Generation - Full-feature protection:

- Packages consisting of variety of antivirus techniques used in conjunction

Host-Based Behaviour-Blocking Software

- Integrates with OS
- Monitors program behaviour in real-time for malicious actions
- Blocks potentially malicious actions before they affect system
- **Advantage**: Can block suspicious software in real-time (advantage over fingerprinting/heuristics)
- **Limitation**: Malicious code must run before all behaviors identified; can cause harm before detection/blocking

Perimeter Scanning Approaches

Ingress Monitors:

- Located at border between enterprise network and Internet
- Part of ingress-filtering software (border router, external firewall) or separate passive monitor

Egress Monitors:

- Located at egress point of individual LANs or at enterprise network border
- Designed to catch source of malware attack

- Monitor outgoing traffic for signs of scanning or suspicious behavior

Used on:

- Organization's firewall and IDS
- Email and Web proxy services
- Traffic analysis component of IDS

Perimeter Worm Countermeasures

Six Classes of Worm Defense:

(Class A) Signature-based worm scan filtering:

- Generates worm signature
- Prevents worm scans from entering/leaving network/host

(Class B) Filter-based worm containment:

- Similar to Class A
- Focuses on worm content rather than scan signature

(Class C) Payload-classification-based worm containment:

- Network-based techniques
- Examine packets to see if they contain worm

(Class D) Threshold Random Walk (TRW) scan detection:

- Exploits randomness in picking destinations
- Detects if scanner is in operation

(Class E) Rate limiting:

- Limits rate of scan-like traffic from infected host

(Class F) Rate halting:

- Immediately blocks outgoing traffic when threshold exceeded
- Threshold based on outgoing connection rate or diversity of connection attempts

---

Key Takeaways

Lecture 4 - Malicious Software

Malware Evolution:

- From individual attackers to organized crime and nation-states
- From requiring technical skill to easy-to-use attack kits
- Increasingly sophisticated (APTs, zero-day exploits)

Defense Strategy:

- **Prevention**: Patches, access controls, user training
- **Detection**: Multiple generations of scanning technology
- **Containment**: Perimeter defenses, behavior monitoring
- **Response**: Identification and removal capabilities

Best Practices:

- Keep systems patched and current
- Implement strong access controls
- Use multi-layered defenses (host-based and perimeter)
- Train users on social engineering risks
- Monitor both ingress and egress traffic
- Prefer hardware/app-based authentication over SMS