Applying Security Techniques to Computing Resources

Security Fundamentals Course

March 10, 2025

Introduction: Securing Computing Resources in the Modern World

- Computing resources face continuous and evolving threats in today's interconnected environment.
- Security must be integrated into every aspect of information technology, not added as an afterthought.
- Effective security requires a balanced approach that addresses technical, administrative, and physical controls.
- This lecture covers fundamental security techniques applicable across various computing environments.

Key Challenge

Organizations must balance security needs with usability, performance, and cost constraints while addressing an ever-expanding threat landscape.

Security Fundamentals: Why Establish Secure Baselines?

- **Secure baselines** provide a consistent, documented starting point for deploying secure systems.
- Baselines ensure that security is applied systematically rather than haphazardly across an organization.
- They establish a measurable standard against which compliance can be regularly verified.
- Baselines reduce attack surface and minimize common configuration vulnerabilities.

Benefits of Secure Baselines

- Predictable security posture
- Easier troubleshooting
- Streamlined auditing
- Simplified compliance

Establishing Secure Baselines: Key Principles

- Begin by identifying critical security requirements and compliance standards relevant to your organization.
- **Principle of least privilege** ensures users and systems have only the access necessary to perform their functions.
- Document all configuration decisions, including rationale and exceptions to standard practice.
- Collaborate across security, operations, and business units to develop realistic, implementable baselines.

Example Baseline Components

System hardening specifications, approved software lists, required security controls, patch management requirements, and authentication standards.

Deploying Secure Baselines: Implementation Strategies

- Create a staged deployment plan with testing phases to identify potential compatibility issues.
- Utilize automation tools to ensure consistent application of baseline configurations.
- Establish verification procedures to confirm that systems meet baseline requirements.
- Document exceptions with formal risk assessment and management approval.

Deployment Method	**Best Use Case**
Golden Images	New system deployment
Configuration Management	Existing system maintenance
Group Policy	Windows environment control
Scripted Configuration	Cross-platform standardization

Table: Common Baseline Deployment Methods

Maintaining Secure Baselines: Ongoing Requirements

- Regularly review baselines to ensure they address current threats and organizational needs.
- **Configuration drift** occurs when systems gradually deviate from their secure baseline state.
- Implement continuous monitoring to detect unauthorized changes or configuration drift.
- Establish a formal change management process for baseline updates to maintain security and stability.

The Baseline Lifecycle

Create → Deploy → Monitor → Update → Redeploy → Verify → Monitor

Hardening Techniques: Overview and Importance

- **System hardening** refers to the process of securing a system by reducing its attack surface.
- Hardening involves removing unnecessary software, disabling unneeded services, and applying secure configurations.
- Different types of systems require specialized hardening techniques based on their function and exposure.
- Hardening should be applied in layers as part of a defense-in-depth security strategy.

Critical Hardening Categories

Operating system hardening, network hardening, application hardening, database hardening, and physical hardening all work together to protect systems.

Mobile Device Hardening: Common Vulnerabilities & Solutions

- Mobile devices face unique security challenges due to their portability, connectivity options, and diverse app ecosystems.
- **Mobile hardening** includes enforcing strong authentication, encryption, and remote wipe capabilities.
- Limit application installations to trusted sources and implement application vetting procedures.
- Configure automatic updates and establish secure connectivity requirements.

Key Mobile Hardening Controls:

- Screen locks with biometrics or strong passcodes
- Full-device encryption
- OS and app update enforcement
- Controlled app permissions

Workstation Hardening: Protecting the User Environment

- Workstations are primary targets for attackers as they are direct user interfaces to organizational resources.
- Implement **application whitelisting** to prevent execution of unauthorized programs on workstations.
- Restrict administrative privileges and apply the principle of least privilege for user accounts.
- Configure host-based firewalls and enable disk encryption to protect data at rest.

Example: Windows 11 Hardening

Use AppLocker for application control, BitLocker for disk encryption, Windows Defender for malware protection, and Group Policy to enforce secure configurations.

Network Hardware Security: Hardening Switches & Routers

- Network devices manage the flow of all organizational data and require specialized hardening techniques.
- Disable unnecessary services and protocols on network devices to reduce attack surface.
- Implement **access control lists (ACLs)** to filter traffic based on security policies.
- Secure the management interfaces with strong authentication and encrypted connections.

Essential Switch Hardening Steps

1. Disable unused ports
2. Implement port security
3. Secure spanning tree protocol
4. Configure VLAN segregation

Cloud Infrastructure Hardening: Securing Virtual Environments

- Cloud environments introduce shared responsibility models where security duties are split between providers and customers.
- Apply **infrastructure as code (IaC)** principles to ensure consistent, secure deployments.
- Implement strong identity and access management with multi-factor authentication for cloud resources.
- Monitor cloud configurations for drift and unauthorized changes using cloud security posture management tools.

Cloud Security Misconception

Cloud providers secure the infrastructure, but customers remain responsible for securing their data, access controls, and application configurations.

Server Hardening: Best Practices and Critical Controls

- Servers host critical applications and data, making them high-value targets requiring robust protection.
- Install only necessary components and remove or disable unused services, roles, and features.
- Implement **file integrity monitoring (FIM)** to detect unauthorized modifications to critical system files.
- Apply regular patches and updates through a formal testing and deployment process.

Server Type	**Special Hardening Considerations**
Web Servers	Web application firewalls, TLS configuration
Database Servers	Query restrictions, data encryption
Email Servers	Content filtering, anti-spoofing
Domain Controllers	Privileged access management

ICS/SCADA Security: Protecting Critical Infrastructure

- **Industrial Control Systems (ICS)** and **SCADA** environments control physical processes and critical infrastructure.
- These systems often use specialized protocols and legacy components with unique security requirements.
- Implement network segmentation to isolate ICS/SCADA networks from corporate IT networks.
- Apply security patches cautiously with extensive testing due to potential operational impacts.

ICS/SCADA Security Challenges

Long lifecycles (20+ years), real-time requirements, proprietary protocols, and physical safety implications create unique security considerations.

Embedded Systems & RTOS Hardening: Special Considerations

- **Embedded systems** are specialized computing devices with dedicated functions built into larger mechanical or electrical systems.
- **Real-Time Operating Systems (RTOS)** have strict timing requirements that security measures must not disrupt.
- Resource constraints (memory, processing power) limit the security mechanisms that can be implemented.
- Secure the boot process to prevent unauthorized firmware modifications.

Security vs. Performance Trade-offs

Security controls for embedded systems must be carefully balanced against performance requirements, especially in time-critical applications.

IoT Device Security: Challenges and Solutions

- **Internet of Things (IoT)** devices often combine limited computing resources with network connectivity.
- Many IoT devices lack basic security features like strong authentication or encryption capabilities.
- Default credentials present a major vulnerability—always change manufacturer passwords.
- Network segmentation is critical to contain potential compromises of vulnerable IoT devices.

IoT Security Best Practices:

- Maintain an inventory of all connected devices
- Implement dedicated IoT networks
- Disable unnecessary features
- Apply firmware updates regularly

Wireless Device Security: The Foundation of Mobile Computing

- Wireless networks extend connectivity beyond physical boundaries, creating additional attack vectors.
- Implement **defense in depth** with multiple security layers for wireless environments.
- Secure both the wireless infrastructure (access points, controllers) and client devices.
- Regular wireless scanning helps identify unauthorized access points and potential attacks.

Wireless Attack Types

Rogue access points, evil twin attacks, deauthentication attacks, and wireless packet sniffing represent common threats to wireless networks.

Site Surveys: Mapping Your Wireless Environment

- **Site surveys** are methodical assessments of wireless environments to optimize coverage and security.
- Physical site surveys identify potential signal obstructions, interference sources, and optimal access point locations.
- Predictive site surveys use software modeling to simulate wireless coverage before hardware deployment.
- Post-implementation validation surveys confirm that the deployed wireless network meets design requirements.

Site Survey Components

Signal strength measurements, interference detection, channel utilization analysis, and coverage overlap assessment.

Heat Maps: Visual Analysis of Wireless Coverage

- **Heat maps** provide color-coded visualizations of wireless signal strength throughout a physical space.
- They help identify coverage gaps, areas of signal overlap, and potential interference zones.
- Heat maps support capacity planning by showing high-density usage areas that may require additional access points.
- Regular heat map analysis helps detect unauthorized access points and potential security threats.

[This is where a sample heat map image would be displayed, showing different signal strengths represented by color variations]

Figure: Example Wireless Heat Map Showing Signal Coverage

Mobile Device Management (MDM): Controlling the Mobile Ecosystem

- **Mobile Device Management (MDM)** provides centralized control over mobile devices in an organization.
- MDM enables remote configuration, policy enforcement, application management, and security monitoring.
- Core MDM capabilities include device enrollment, inventory management, and compliance verification.
- Advanced MDM solutions incorporate mobile application management and content management features.

MDM Security Capabilities

Remote lock and wipe, encryption enforcement, application blacklisting/whitelisting, jailbreak/root detection, and certificate management.

BYOD vs. COPE vs. CYOD: Selecting the Right Deployment Model

- **Bring Your Own Device (BYOD)** allows employees to use personal devices for work, reducing hardware costs but increasing security challenges.
- **Corporate-Owned, Personally Enabled (COPE)** provides organization-owned devices with limited personal use allowed.
- **Choose Your Own Device (CYOD)** lets employees select from approved device options that are then purchased by the organization.
- Each model represents different balances between user satisfaction, cost, and security control.

Factor	BYOD	COPE	CYOD
Cost to Organization	Low	High	Medium
User Satisfaction	High	Medium	High
Security Control	Low	High	High
Support Complexity	High	Low	Medium

Mobile Connection Methods: Cellular, Wi-Fi & Bluetooth Security

- Mobile devices typically support multiple connection methods, each with distinct security considerations.
- **Cellular connections** use carrier-managed infrastructure with built-in encryption but vary in security by generation (3G/4G/5G).
- **Wi-Fi connections** offer higher bandwidth but require careful security configuration to prevent eavesdropping.
- **Bluetooth** enables short-range device communications but has historically suffered from numerous security vulnerabilities.

Securing Mobile Connections

Use VPNs for sensitive communications, disable auto-connect features for Wi-Fi and Bluetooth, and implement connection policies through MDM.

Wi-Fi Protected Access 3 (WPA3): Features and Implementation

- **WPA3** is the latest Wi-Fi security protocol that addresses vulnerabilities in previous standards like WPA2.
- WPA3 implements **Simultaneous Authentication of Equals (SAE)** to replace the vulnerable Pre-Shared Key (PSK) method.
- Forward secrecy in WPA3 ensures that captured data cannot be decrypted even if the password is compromised later.
- WPA3-Enterprise adds 192-bit security mode for organizations requiring higher security levels.

WPA2 vs. WPA3 Key Improvements

Protection against offline dictionary attacks, improved protection for weak passwords, and simplified secure configuration for devices without displays.

AAA/Remote Authentication Dial-In User Service (RADIUS): Authentication, Authorization, and Accounting

- **AAA** framework provides Authentication (identity verification), Authorization (access control), and Accounting (usage tracking).
- **RADIUS** is a network protocol that implements the AAA framework, particularly for network access authentication.
- RADIUS centralizes authentication for network devices, wireless networks, and VPN connections.
- Integration with directory services like Active Directory enables consistent identity management.

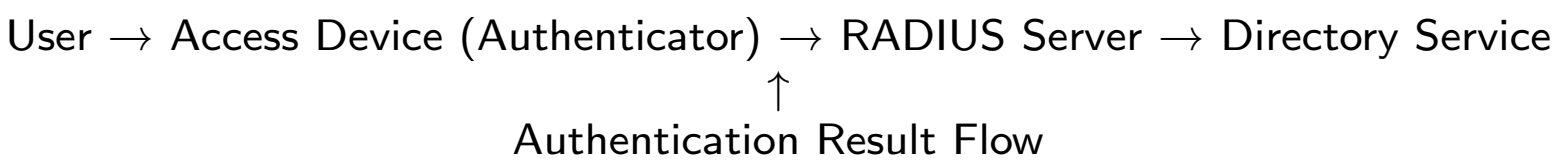

Figure: Simplified RADIUS Authentication Flow

Cryptographic Protocols: Ensuring Data Privacy in Transit

- **Cryptographic protocols** establish secure communications channels over potentially insecure networks.
- **Transport Layer Security (TLS)** secures web traffic, email, and many other application protocols.
- **Internet Protocol Security (IPsec)** provides security at the network layer for VPN implementations.
- Wireless networks use specific protocols like CCMP (Counter Mode CBC-MAC Protocol) for WPA2/WPA3 encryption.

Key Protocol Selection Factors

Consider performance requirements, compliance standards, supported algorithms, key management capabilities, and compatibility with existing systems.

Authentication Protocols: Verifying Identity Securely

- **Authentication protocols** standardize the process of verifying claimed identities across systems.
- **LDAP** (Lightweight Directory Access Protocol) provides directory services for storing and retrieving identity information.
- **Kerberos** uses ticket-based authentication to verify identities without transmitting passwords.
- **SAML** and **OAuth** enable single sign-on and delegated authorization across different systems.

Multi-Factor Authentication (MFA)

Combines two or more authentication factors: something you know (password), something you have (token), and something you are (biometric).

Application Security: The First Line of Defense

- Applications are primary attack targets because they often have direct access to sensitive data.
- **Application security** involves building security into applications from the design phase through deployment and maintenance.
- Secure software development lifecycle (SSDLC) integrates security activities throughout development.
- Defense-in-depth for applications includes input validation, authentication, authorization, and error handling.

Common Application Vulnerabilities

Injection attacks, broken authentication, sensitive data exposure, XML external entities, broken access control, security misconfigurations, and cross-site scripting (XSS).

Input Validation: Preventing Injection Attacks

- **Input validation** ensures that application data meets expected formats and constraints before processing.
- Server-side validation is critical—client-side validation can be easily bypassed by attackers.
- Implement both "allow list" (explicitly approve valid input) and "deny list" (reject known bad input) approaches.
- Proper input validation helps prevent SQL injection, command injection, and cross-site scripting attacks.

Input Validation Techniques

Data type checking, range validation, format validation, length restrictions, and parameterized queries for database operations.

Secure Cookies: Protecting Session Data

- **Cookies** store session information and user preferences in web applications.
- **Secure cookies** are transmitted only over encrypted HTTPS connections to prevent interception.
- The HttpOnly flag prevents JavaScript access to cookies, protecting against cross-site scripting attacks.
- SameSite attribute restricts cookie transmission to same-site requests, preventing cross-site request forgery.

Cookie Attribute	**Security Function**
Secure	Transmission over HTTPS only
HttpOnly	Blocks JavaScript access
SameSite	Controls cross-origin behavior
Expires/Max-Age	Limits cookie lifetime

Static Code Analysis: Finding Vulnerabilities Before Deployment

- **Static code analysis** examines source code without executing it to identify potential security vulnerabilities.
- Static analysis tools can detect issues like buffer overflows, SQL injection, and insecure cryptographic implementations.
- Analysis should be integrated into the development pipeline for early detection and remediation.
- False positives require human review to determine actual security impact and appropriate response.

Static Analysis Benefits

Scales to large codebases, finds vulnerabilities early in development, enforces coding standards, and provides consistent application of security rules.

Code Signing: Ensuring Software Integrity

- **Code signing** uses digital signatures to verify the authenticity and integrity of software.
- Signed applications provide assurance that code has not been modified since it was signed by the developer.
- Code signing relies on public key infrastructure (PKI) and trusted certificate authorities.
- Operating systems and browsers use code signatures to verify application trustworthiness.

Code Signing Process

1. Developer obtains a code signing certificate from a trusted authority
2. Developer creates a cryptographic hash of the software
3. Hash is encrypted with the developer's private key
4. Signature is verified using the developer's public key

Sandboxing: Isolating Applications for Security

- **Sandboxing** creates isolated environments where applications can run with limited access to system resources.
- Sandboxes contain potential damage from malicious or vulnerable applications by restricting their capabilities.
- Modern web browsers sandbox each tab to prevent malicious websites from affecting the system or other tabs.
- Containerization technologies like Docker implement sandbox-like isolation for applications.

Sandbox Implementation Methods

Virtual machines, containers, application virtualization, browser sandboxes, and operating system security features like seccomp on Linux.

Conclusion: Building a Comprehensive Security Strategy

- Security must be approached holistically, considering all computing resources and their interconnections.
- Defense in depth implements multiple security layers to protect against various threats and attack vectors.
- Balance security requirements with operational needs, user experience, and resource constraints.
- Effective security requires ongoing vigilance: regularly review, test, and update security controls as threats evolve.

Key Takeaway

Security is not a product but a process—it requires continuous improvement, adaptation to new threats, and integration into all aspects of computing resource management.