System and Network Architecture in Security Operations

Brendan Shea, PhD

March 23, 2025

1. Security Architecture: The Foundation of Modern Cybersecurity Operations

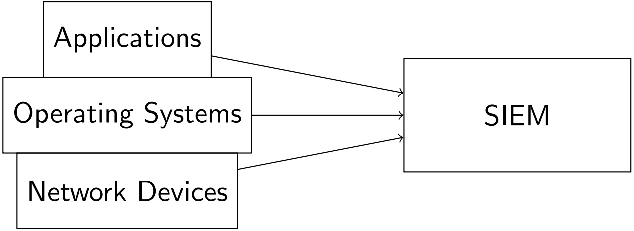
- Security architecture provides the framework for implementing protective measures across systems and networks.
- **Security architecture** is defined as the design of systems and processes that protect information assets from unauthorized access or damage.
- Effective security architecture aligns with business goals while mitigating risks through defense-in-depth strategies.
- Security operations depend on well-designed architecture to identify, respond to, and recover from security incidents.

Key Concept

Security architecture is not just about technology—it's about creating a cohesive strategy that includes people, processes, and technology working together.

2. Log Management Essentials: Capturing the Digital Footprints

- Logs record system and network events that are crucial for security monitoring and incident response.
- **Log management** involves the collection, storage, analysis, and disposal of log data from various sources.
- Effective log management enables threat detection, forensic investigations, and compliance reporting.
- Centralized log collection creates a single source of truth for security events across the environment.



3. Time Synchronization: Why Accurate Timestamps Are Critical for Security

- Time synchronization ensures all devices in a network report events with consistent timestamps.
- **Network Time Protocol (NTP)** is used to synchronize system clocks across a network to a reference time source.
- Inconsistent timestamps make it difficult to establish the sequence of events during security incidents.
- Time synchronization is essential for correlating events from different systems during an investigation.

```
# Example of logs with synchronized vs. unsynchronized time
Synchronized:
2023-10-15 14:32:45 [firewall] Connection blocked from 192.168.1.10
2023-10-15 14:32:47 [server] Failed login attempt for user admin
2023-10-15 14:32:48 [IDS] Alert: Possible brute force attack

Unsynchronized:
2023-10-15 14:32:45 [firewall] Connection blocked from 192.168.1.10
2023-10-15 14:27:12 [server] Failed login attempt for user admin
2023-10-15 14:39:22 [IDS] Alert: Possible brute force attack
```

4. Understanding Logging Levels: From Debug to Critical

- **Logging levels** define the severity and importance of logged events in a system.
- Standard logging levels typically include Debug, Info, Warning, Error, and Critical categories.
- Higher severity levels (Error, Critical) should trigger immediate alerts for security teams.
- Proper configuration of logging levels balances security visibility with storage and processing efficiency.

Level	**Priority**	**Description**
Debug	Lowest	Detailed information for debugging purposes
Info	Low	Normal system operations, successful actions
Warning	Medium	Non-critical issues that may require attention
Error	High	Runtime errors or unexpected conditions
Critical	Highest	System-critical events requiring immediate action

5. Windows Registry: The Nervous System of Your Operating System

- The **Windows Registry** is a hierarchical database that stores configuration settings and options for the operating system.
- Registry keys contain critical system and security settings that can be modified by users, applications, or malware.
- Security teams monitor registry changes to detect unauthorized modifications or malicious activity.
- Improper registry modifications can lead to system instability, security vulnerabilities, or privilege escalation.

Registry Hives

HKEY_LOCAL_MACHINE (HKLM) System-wide configuration settings

HKEY_CURRENT_USER (HKCU) Settings specific to the logged-in user

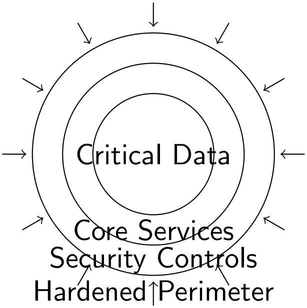
HKEY_USERS (HKU) Settings for all user profiles on the system

HKEY_CLASSES_ROOT (HKCR) File association and COM object registration

HKEY_CURRENT_CONFIG (HKCC) Current hardware profile information

6. System Hardening: Building a Digital Fortress

- **System hardening** refers to the process of securing a system by reducing its attack surface and vulnerability exposure.
- Hardening involves disabling unnecessary services, closing unused ports, and removing unneeded applications.
- Regular security updates and patches are essential components of an effective hardening strategy.
- System hardening should be guided by industry-standard benchmarks like CIS (Center for Internet Security) controls.



7. File Structure Fundamentals: Where Your System Stores Critical Data

- Operating systems organize files in hierarchical structures that determine access permissions and security contexts.
- **File systems** (like NTFS for Windows or ext4 for Linux) implement security features such as access control lists and encryption.
- Understanding file structure enables security teams to identify abnormal file placements or unauthorized access attempts.
- Critical system files and configurations are stored in protected locations that require elevated privileges to modify.

Example

Critical Windows File Locations

- `C:\Windows\System32` - Core system executables and DLLs
- `C:\Windows\System32\config` - Registry hives
- `C:\Windows\System32\drivers` - Device drivers
- `C:\Program Files` - Installed applications (64-bit)

8. Configuration Files: The Security Control Panel of Your System

- **Configuration files** store settings that determine how applications and systems operate and handle security.
- Security-critical configuration files control authentication, authorization, access control, and encryption settings.
- Misconfigurations in these files are a leading cause of security breaches and system vulnerabilities.
- Security teams should implement controls for configuration management, version control, and change detection.

Common Security Misconfigurations

Insecure default settings, excessive permissions, hardcoded credentials, unnecessary features enabled, and outdated configuration templates are frequently exploited by attackers to gain unauthorized access.

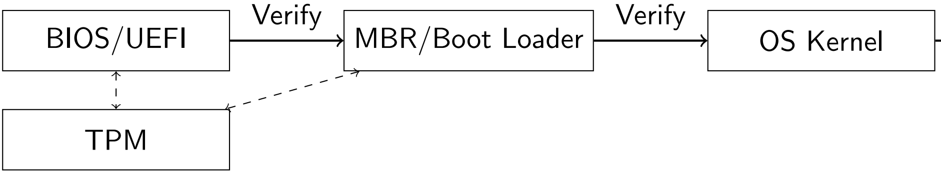
9. System Processes: Identifying Friend from Foe

- **System processes** are programs running in memory that perform essential functions for the operating system.
- Understanding normal process behavior helps security teams identify anomalous or malicious processes.
- Process attributes like parent-child relationships, resource usage, and file access patterns establish behavioral baselines.
- Advanced threats often attempt to disguise malicious processes as legitimate system processes.

```
# Windows Task Manager process list example
Name                  PID   CPU   Memory    User
System                4     0.1%   0.1 MB    SYSTEM
smss.exe              372   0.0%   0.7 MB    SYSTEM
csrss.exe             560   0.2%   3.9 MB    SYSTEM
wininit.exe           632   0.0%   1.2 MB    SYSTEM
services.exe          748   0.1%   5.6 MB    SYSTEM
svchost.exe           856   0.3%  23.4 MB    SYSTEM
svchost.exe         1345  67.2% 356.8 MB SYSTEM
explorer.exe         1424   0.5%  45.2 MB    User
```

10. Hardware Architecture Security: From BIOS to Boot Sequence

- **Hardware architecture security** involves protecting the physical components and firmware that support computing systems.
- The boot sequence represents a critical security phase where systems are vulnerable to low-level attacks.
- Secure boot technologies verify the integrity of firmware and boot loaders before the operating system loads.
- Hardware-based security features like Trusted Platform Module (TPM) provide cryptographic functions for secure key storage.



11. Serverless Security: Protecting Functions in a Cloud-Native World

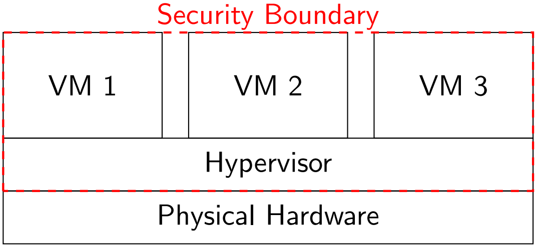
- **Serverless computing** is a cloud execution model where the cloud provider manages infrastructure, allowing developers to focus on code.
- The ephemeral nature of serverless functions creates unique security challenges for monitoring and protection.
- Security considerations include function permissions, code vulnerabilities, dependency management, and API security.
- Traditional security tools designed for persistent infrastructure may not be effective for serverless architectures.

Serverless Security Challenges

- Limited execution context makes runtime protection difficult
- Shared responsibility model shifts but doesn't eliminate security obligations
- Function permissions often follow the principle of least privilege
- Third-party dependencies can introduce vulnerabilities

12. Virtualization Security: Isolating Resources Without Isolating Protection

- **Virtualization** creates logical instances of computing resources that operate independently on shared physical hardware.
- Virtual environments require security controls at both the host and guest levels to prevent cross-VM attacks.
- Hypervisor security is critical as compromising this layer could affect all virtual machines under its management.
- Virtual network security introduces additional complexity with software-defined networking components.



13. Container Security: Protecting Microservices in Production

- **Containers** are lightweight, portable environments that package code and dependencies for consistent operation across environments.
- Container security differs from VM security because containers share the host OS kernel, creating a different isolation model.
- Security strategies include scanning container images for vulnerabilities, using minimal base images, and implementing runtime protection.
- Container orchestration platforms like Kubernetes require additional security considerations for pod communication and secrets management.

Example

Container Security Best Practices

- Use signed and verified container images from trusted repositories
- Implement network policies to control pod-to-pod communications
- Apply the principle of least privilege to container runtime permissions

14. On-Premises Architecture: Traditional Security for Traditional Infrastructure

- **On-premises architecture** refers to computing resources that are physically located within an organization's facilities.
- Traditional security models for on-premises environments often focus on perimeter-based defenses and network segmentation.
- Physical security controls complement digital security measures to protect hardware assets and data centers.
- On-premises architectures typically provide greater control but require significant capital expenditure and maintenance.

On-Premises Security Controls

Physical	Access controls, environmental monitoring, surveillance
Network	Firewalls, IDS/IPS, network segregation, DMZs
System	Host hardening, endpoint protection, patch management
Data	Encryption, access controls, data classification

15. Cloud Security Architecture: Securing Resources Beyond Your Walls

- **Cloud security architecture** addresses the protection of data, applications, and infrastructure hosted by third-party cloud providers.
- The shared responsibility model defines security obligations for both cloud providers and customers.
- Cloud-native security approaches utilize automation, infrastructure as code, and API-driven controls.
- Effective cloud security requires adaptation of traditional security principles to dynamic, scalable environments.

Cloud Security Responsibility Matrix

Table: Cloud Security Responsibility Matrix

Security Aspect	**Cloud Provider**	**Customer**
Physical Infrastructure Security	✓	
Hardware Management	✓	
Virtualization Layer	✓	
Operating Systems (IaaS)		✓
Operating Systems (PaaS/SaaS)	✓	
Application Security (IaaS/PaaS)		✓
Application Security (SaaS)	✓	
Identity and Access Management		✓
Data Protection (Encryption at Rest)	✓	✓
Data Protection (Encryption in Transit)	✓	✓
Network Security Controls (Infrastructure)	✓	
Network Security Controls (User-configured)		✓
Security Monitoring (Infrastructure-level)	✓	
Security Monitoring (Application-level)		✓
Compliance (Infrastructure Certifications)	✓	
Compliance (Industry-specific Requirements)		✓
Incident Response (Infrastructure-level)	✓	
Incident Response (Customer-level)		✓

16. Hybrid Infrastructure: Bridging Security Between Worlds

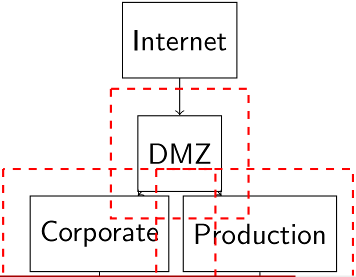
- **Hybrid infrastructure** combines on-premises systems with cloud resources, requiring integrated security strategies.
- Security challenges include maintaining consistent controls, identity management, and secure connectivity between environments.
- Data moving between on-premises and cloud environments must be protected in transit with proper encryption and authorization.
- Security monitoring and incident response processes must span both environments for comprehensive protection.

Hybrid Security Pitfalls

Organizations often struggle with security gaps when integrating cloud and on-premises environments, particularly around identity management, access controls, and security visibility across boundaries.

17. Network Segmentation: Building Digital Boundaries That Matter

- **Network segmentation** divides a network into multiple segments or subnets, each acting as a security zone with controlled access.
- Effective segmentation limits an attacker's lateral movement capabilities following an initial compromise.
- Segmentation can be implemented physically (through separate hardware) or logically (using VLANs, firewalls, and ACLs).
- The principle of least privilege should guide access controls between network segments.



18. Zero Trust Architecture: Never Trust, Always Verify

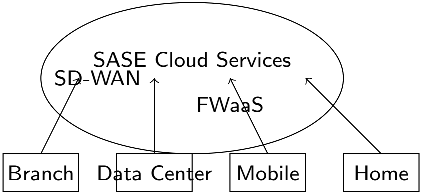
- **Zero Trust** is a security model that assumes no user or system should be inherently trusted, regardless of location or network.
- Zero Trust principles include verifying identity, validating device health, enforcing least privilege, and assuming breach.
- Implementation requires strong identity management, micro-segmentation, and continuous monitoring and validation.
- Zero Trust represents a shift from perimeter-based security to identity and data-centric approaches.

Zero Trust Core Principles

1. Verify explicitly - Always authenticate and authorize based on all available data points
2. Use least privileged access - Limit user access with Just-In-Time and Just-Enough-Access
3. Assume breach - Minimize blast radius and segment access, verify end-to-end encryption
4. Apply behavioral analytics to detect anomalies in real-time

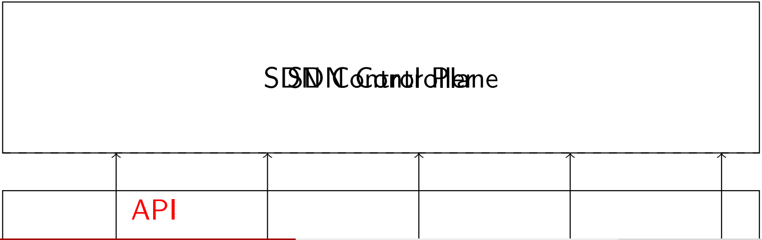
19. Secure Access Service Edge (SASE): Converging Network and Security in the Cloud Era

- **Secure Access Service Edge (SASE)** combines network security functions with WAN capabilities to support secure access for distributed organizations.
- SASE delivers security controls as cloud-based services, closer to the users and devices that need them.
- Core components include SD-WAN, Secure Web Gateway, CASB, Zero Trust Network Access, and FWaaS.
- SASE addresses the challenge of securing remote workforces and cloud applications beyond traditional network boundaries.



20. Software-Defined Networking: Security in a Programmable Network

- **Software-Defined Networking (SDN)** separates the network control plane from the data plane, enabling programmatic network management.
- SDN enhances security through centralized policy enforcement, dynamic network segmentation, and automated threat response.
- The SDN controller becomes a critical security component that must be protected from unauthorized access or manipulation.
- Security benefits include improved visibility, consistent policy application, and rapid response to detected threats.



24. Federation: Managing Identity Across Organizational Boundaries

- **Federation** is a mechanism that enables separate organizations to share identity information securely across trust boundaries.
- Federated identity allows users to use credentials from one domain to access resources in another domain without creating multiple accounts.
- Federation relies on established trust relationships between identity providers and service providers.
- Implementation typically uses standards like SAML, WS-Federation, or OAuth/OpenID Connect to securely exchange identity assertions.

Example

Federation in Action A university professor can use their university credentials (identity provider) to access an external research database (service provider) without creating a separate account. The research database trusts the university to properly authenticate the professor, while the university maintains control over the professor's authentication

21. Identity: The New Security Perimeter

- **Identity** has become the primary security boundary in modern environments where traditional network perimeters are dissolving.
- Identity and access management (IAM) systems provide the foundation for authenticating users and authorizing access to resources.
- Identity-focused security requires continuous validation of user identity, device health, and contextual risk factors.
- Identity systems must be protected as critical infrastructure since compromise can lead to widespread access across environments.

Identity as the Attack Vector

Compromised credentials are involved in over 80% of data breaches, making identity protection a critical security priority. Attack techniques like password spraying, credential stuffing, and phishing specifically target this new perimeter.

22. Multifactor Authentication: Why Passwords Alone Are Not Enough

- **Multifactor Authentication (MFA)** requires two or more verification factors: something you know, something you have, or something you are.
- MFA significantly reduces the risk of account compromise, even when passwords are leaked or stolen.
- Authentication factors vary in security strength, with biometrics and hardware tokens generally offering stronger protection than SMS codes.
- Proper MFA implementation requires balancing security requirements with user experience considerations.

Factor Type	**Examples**	**Security Level**
Knowledge	Passwords, PINs, Security questions	Low-Medium
Possession	Hardware tokens, Mobile apps, Smart cards	Medium-High
Inherence	Fingerprints, Facial recognition, Voice	High
Location	GPS, Network location	Medium
Behavior	Typing patterns, Usage patterns	Medium

23. Single Sign-On: Balancing Security and User Experience

- **Single Sign-On (SSO)** allows users to authenticate once and gain access to multiple applications without additional login prompts.
- SSO improves security by reducing password fatigue, enabling centralized authentication management, and streamlining access revocation.
- SSO implementations typically use protocols like SAML, OAuth, or OpenID Connect to securely share authentication information.
- While SSO creates a single point of entry, it should be strengthened with MFA and continuous monitoring to mitigate this risk.

=