SEAS-8405
Cybersecurity Architecture

Week-4

Dr. Mallarapu

Security Frameworks vs. Architectures

- Frameworks = 'What' to protect (standards, controls)
- Architectures = 'How' to implement (design, layering)

Notable Frameworks

- NIST Cybersecurity Framework
- ISO 27001
- CIS Controls
- MITRE ATT&CK

Notable Architectures

- Zero Trust Architecture (ZTA)
- Defense in Depth (DiD)
- Secure SDLC (SSDLC)
- Zero Knowledge Architecture (ZKA)
- Adaptive Security Architecture (ASA)

Why Distinguish Them?

- Frameworks alone are insufficient
- Architectural coherence is key to real security
- Combined approach yields best outcomes

MVC at a Glance

- Components: Model, View, Controller
- Widely used for web apps & UI-based systems

MVC – Security Touchpoints

- Model: Validate data, parameterized queries
- View: Output encoding against XSS
- Controller: Access control, request sanitization

Common MVC Vulnerabilities

- SQL Injection
- XSS
- Privilege Escalation

MVC in Cloud Platforms

- Model in managed DB (RDS, Azure SQL)
- Controller as microservices (Lambda, Azure Functions)
- Security emphasis: IAM roles, container security

Case Example – MVC Gone Wrong

- User input not sanitized → malicious query in Model
- Data exfiltration & pivot to broader environment

DiD Concept Overview

- Multiple layers: Network, Endpoint, Application, Data
- Goal: No single point of failure

Layer-by-Layer Breakdown

- Network: Firewalls, IDS/IPS
- Endpoint: Antivirus, EDR
- Application: Secure coding, WAF
- Data: Encryption, DLP

DiD in Modern Cloud

- AWS Examples: Security Groups, GuardDuty, etc.
- Azure Examples: NSGs, Azure Firewall, Azure Defender
- SIEM Tools for monitoring (CloudWatch, Azure Monitor)

DiD Strengths & Weaknesses

- Strengths: Redundancy, layered defense
- Weaknesses: Complexity, misconfiguration risk, cost

DiD Real-World Incidents

- Target breach: Weak segmentation
- Equifax breach: Patch management gap

Building Resilience with DiD

- Regular audits across layers
- Automated patching & monitoring
- Continual improvement & defense tuning

ZTA Core Principle

- "Never trust, always verify"
- Every access must be authenticated & authorized
- Micro-segmentation, strong IAM

ZTA Pillars

- Identity & Access Management (IAM)
- Micro-segmentation
- Encryption everywhere

ZTA vs. Traditional Perimeter

- Traditional: Trusted internal network, lesser external trust
- Zero Trust: No default trust, frequent re-auth
- Implications for user experience & device checks

ZTA in AWS / Azure

- AWS: IAM w/ MFA, VPC segmentation, AWS SSO conditional access
- Azure: Azure AD conditional policies, vNET isolation, JIT VM access

ZTA Success Factors

- Comprehensive logging & analytics
- Automated policy enforcement
- Continuous monitoring

ZTA Implementation Challenges

- Legacy integration challenges
- User friction (MFA fatigue)
- Cultural & operational shift required

SSDLC Overview

- Phases: Requirements, Design, Coding, Testing, Deployment, Maintenance
- Secure at every stage

"Shift Left" Philosophy

- Security integrated early
- SAST/DAST, Threat Modeling in design
- Cost-effective vulnerability mitigation

Key Security Activities

- Threat Modeling (STRIDE, DREAD)
- Code Scanning (SAST, DAST)
- Dependency Management (vulnerable libraries)

CI/CD Pipeline Integration

- Signed artifacts, secured build environment
- Automated tests for each commit
- Secrets management

SSDLC in the Cloud

- AWS: CodePipeline, CodeBuild + scanning (CodeGuru)
- Azure: DevOps with SAST/DAST Extensions
- Goal: Prevent pipeline compromises (like SolarWinds)

Overcoming SSDLC Challenges

- Developer resistance & false positives
- Frequent updates to libraries & frameworks
- Cultural shift with DevSecOps

ZKA Fundamentals

- Provider has zero visibility of plaintext data
- All encryption/decryption on client side
- User retains complete key control

Cryptographic Underpinnings

- Client-Side Encryption (CSE)
- Zero-Knowledge Proofs (ZKPs)
- User-managed keys (BYOK)

Practical Trade-Offs (ZKA)

- Pros: Ultimate data privacy, minimal insider threat
- Cons: Complex search/analytics, key loss = data loss

ZKA Use Cases

- Highly regulated sectors (finance, healthcare)
- Privacy-focused solutions (encrypted email, file storage)

ZKA in AWS / Azure

- AWS: S3 client-side encryption, CloudHSM, external key mgmt
- Azure: Client-side encryption + Key Vault integration
- Server-side analytics remains difficult

ASA Definition

- Adaptive Security Architecture
- Continuous monitoring, real-time response
- Behavioral analytics (UEBA)

ASA Core Loop

- 1) Collect telemetry (logs, flows)
- 2) Analyze & detect anomalies
- 3) Adapt & respond automatically
- 4) Learn & improve (feedback loop)

ASA Tools & Tech

- SIEM + SOAR solutions
- Machine Learning for anomaly detection
- Threat intelligence feeds

ASA in AWS / Azure

- AWS: GuardDuty, Macie, CloudTrail, automation via Lambda
- Azure: Microsoft Defender, Sentinel, Logic Apps for response

ASA Strengths & Weaknesses

- Strengths: Real-time resilience, rapid containment
- Weaknesses: Complexity, false positives, tuning overhead

Example: Capital One Incident

- Missed opportunity: SIEM alert not escalated in time
- ASA could automate rapid response to anomalies

Supply Chain Attacks

- Compromise at the vendor side
- Malware introduced into trusted updates

SUNBURST Attack Timeline

- Compromised build environment at SolarWinds
- Malicious code injected into Orion updates
- Signed trojan updates installed by thousands

Attack Tactics

- Stealth: Dormant backdoor evaded detection
- Selective targeting for high-value victims
- Lateral movement via AD/SSO exploitation

Architecture Failures

- Insufficient pipeline security & code-signing checks
- Weak credentials in build systems
- Late detection & monitoring gaps

Mitigation Strategies

- Secure SDLC: Signed build steps, zero-trust in pipeline
- Defense in Depth: Strong segmentation around key assets
- Adaptive Security: Automated anomaly detection in build environment

Lessons for Doctoral Research

- Supply chain complexity = major risk
- Formal verification of build processes as a frontier
- Enhanced cryptographic signing architectures

Integrating Multiple Architectures

- No single approach solves all problems
- Tailor solutions to environment & risk
- Combine DiD, ZTA, SSDLC, ASA, etc.