Security Architecture

Objectives:

- 3.1 - Compare and contrast security implications of different architecture models
- 4.1 - Given a scenario, apply common security techniques to computing resources

- **Security Architecture**
  - *Security Architecture*
    - Design, structure, and behavior of an organization's information security environment
  - On-Premise vs. Cloud Deployment
    - On-Premise
      - Traditional local infrastructure setup
    - Cloud
      - Delivery of computing services over the internet
  - Cloud Security Considerations
    - Shared Physical Server Vulnerabilities
    - Inadequate Virtual Environment Security
    - User Access Management
    - Lack of Up-to-date Security Measures
    - Single Point of Failure
    - Weak Authentication and Encryption Practices
    - Unclear Policies and Data Remnants

- Virtualization and Containerization
  - Different virtualization types
  - Containerization benefits and risks
  - Vulnerabilities like vm escape and resource reuse
- *Serverless Computing*
  - Cloud provider manages server allocation
  - Developers focus solely on writing code
- *Microservices Architecture*
  - Collection of small, autonomous services
  - Each performs a specific business process
- *Software-Defined Network (SDN)*
  - Dynamic, programmatically efficient network configuration
  - Improves network performance and monitoring
- *Infrastructure as Code (IaC)*
  - Automation of managing and provisioning technology stack
  - Software-driven setup instead of manual configuration
- Centralized vs. Decentralized Architectures
  - Benefits and risks of centralized and decentralized setups
- *Internet of Things (IoT)*
  - Network of physical devices with sensors and connectivity
  - Enables data exchange among connected objects
- ICS and SCADA
  - *Industrial Control Systems (ICS)*
    - For industrial production
  - *Supervisory Control and Data Acquisition (SCADA)*
    - Subset of ICS

- *Embedded Systems*
  - Dedicated computer system designed for specific functions
  - Part of a complete device system with hardware components

- **On-premise versus the Cloud**
  - *Cloud Computing*
    - Delivery of computing services over the internet, including servers, storage, databases, networking, software, analytics, and intelligence
    - Advantages
      - Faster innovation
      - Flexible resources
      - Economies of scale
  - *Responsibility Matrix*
    - Outlines the division of responsibilities between the cloud service provider and the customer
  - *Third-Party Vendors*
    - Provides specialized services to enhance functionality, security, and efficiency of cloud solutions
  - *Hybrid Solutions*
    - Combined on-premise, private cloud, and public cloud services, allowing workload flexibility
    - Considerations
      - Sensitive data is protected
      - Regulatory requirements are met
      - Systems can communicate with each other
      - The solution is cost-effectiveness

- *On-Premise Solutions*
  - Computing infrastructure physically located on-site at a business
- Key Considerations in Cloud Computing
  - Availability
    - System's ability to be accessed when needed
  - Resilience
    - System's ability to recover from failures
  - Cost
    - Consider both upfront and long-term costs
  - Responsiveness
    - Speed at which the system can adapt to demand
  - Scalability
    - System's ability to handle increased workloads
  - Ease of Deployment
    - Cloud services are easier to set up than on-premise solutions
  - Risk Transference
    - Some risks are transferred to the provider, but customers are responsible for security
  - Ease of Recovery
    - Cloud services offer easy data recovery and backup solutions
  - Patch Availability
    - Providers release patches for vulnerabilities automatically
  - Inability to Patch
    - Compatibility issues or lack of control can hinder patching

        - Power
            - Cloud provider manages infrastructure, including power supply
            - Reduces customer costs and eliminates power management concerns
        - Compute
            - Refers to computational resources, including CPUs, memory, and storage
            - Cloud providers offer various compute options to suit different needs
    - Remember
        - Cloud computing offers flexibility, scalability, and cost-effectiveness
        - On-premise solutions provide control and security but can be expensive and challenging to maintain
        - Hybrid solutions offer flexibility and control but require considerations of security, compliance, interoperability, and cost

- **Cloud Security**
    - Shared Physical Server Vulnerabilities
        - In cloud environments, multiple users share the same physical server
            - Compromised data from one user can potentially impact others on the same server
        - Mitigation
            - Implement strong isolation mechanisms (e.g., hypervisor protection, secure multi-tenancy)
            - Perform regular vulnerability scanning, and patch security gaps
    - Inadequate Virtual Environment Security
        - Virtualization is essential in cloud computing
            - Inadequate security in the virtual environment can lead to unauthorized access and data breaches

- Mitigation
    - Use secure VM templates
    - Regularly update and patch VMs
    - Monitor for unusual activities
    - Employ network segmentation to isolate VMs
- User Access Management
    - Weak user access management can result in unauthorized access to sensitive data and systems
    - Mitigation
        - Enforce strong password policies
        - Implement multi-factor authentication
        - Limit user permissions (Principle of Least Privilege)
        - Monitor user activities for suspicious behavior
- Lack of Up-to-date Security Measures
    - Cloud environments are dynamic and require up-to-date security measures
        - Failure to update can leave systems vulnerable to new threats
    - Mitigation
        - Regularly update and patch software and systems
        - Review and update security policies
        - Stay informed about the latest threats and best practices
- Single Point of Failure
    - Cloud services relying on specific resources or processes can lead to system-wide outages if they fail
    - Mitigation
        - Implement redundancy and failover procedures
        - Use multiple servers, data centers, or cloud providers

        - Regularly test failover procedures
    - Weak Authentication and Encryption Practices
        - Weak authentication and encryption can expose cloud systems and data
        - Mitigation
            - Use multi-factor authentication
            - Strong encryption algorithms
            - Secure key management practices
    - Unclear Policies
        - Unclear security policies can lead to confusion and inconsistencies in implementing security measures
        - Mitigation
            - Develop clear, comprehensive security policies covering data handling, access control, incident response, and more
            - Regularly review and update policies and provide effective communication and training
    - Data Remnants
        - *Data Remnants*
            - Residual data left behind after deletion or erasure processes
            - In a cloud environment, data may not be completely removed, posing a security risk
        - Mitigation
            - Implement secure data deletion procedures
            - Use secure deletion methods
            - Manage backups securely
            - Verify data removal after deletion
    - Remember that cloud security is a shared responsibility

- **Virtualization and Containerization**
  - *Virtualization*
    - Emulates servers, each with its own OS within a virtual machine
  - *Containerization*
    - Lightweight alternative, encapsulating apps with their OS environment
    - Key Benefits
      - Efficiency and Speed
      - Portability
      - Scalability
      - Isolation
      - Consistency
  - Hypervisors
    - Two Types of Hypervisors
      - *Type 1 (Bare Metal)*
        - Runs directly on hardware (e.g., Hyper-V, XenServer, ESXi)
      - *Type 2 (Hosted)*
        - Operates within a standard OS (e.g., VirtualBox, VMware)
  - Virtualization Vulnerabilities
    - *Virtual Machine (VM) Escape*
      - Attackers break out of isolated VMs to access the hypervisor
    - *Privilege Elevation*
      - Unauthorized elevation to higher-level users
    - *Live VM Migration*
      - Attacker captures unencrypted data between servers

        - *Resource Reuse*
            - Improper clearing of resources may expose sensitive data
    - Containerization Technologies
        - Docker, Kubernetes, Red Hat OpenShift are popular containerization platforms
        - Revolutionized application deployment in cloud environments
    - Securing Virtual Machines
        - Regularly update OS, applications, and apply security patches
        - Install antivirus solutions and software firewalls
        - Use strong passwords and implement security policies
        - Secure the hypervisor with manufacturer-released patches
        - Limit VM connections to physical machines and isolate infected VMs
        - Distribute VMs among multiple servers to prevent resource exhaustion
        - Monitor VMs to prevent "Virtualization Sprawl”
        - Enable encryption of VM files for data safety and confidentiality

- **Serverless**
    - What is Serverless?
        - Serverless computing doesn't mean no servers; it shifts server management away from developers
        - Relies on cloud service providers to handle server management, databases, and some application logic
        - *Functions as a Service (FaaS) Model*
            - Developers write and deploy individual functions triggered by events
    - Benefits of Serverless
        - Reduced operational costs
            - Pay only for compute time used, no charges when code is idle

        - Automatic scaling
            - Cloud provider scales resources based on workload, ensuring optimal capacity
        - Focus on core product
            - Developers can concentrate on application functionality, not server management
        - Faster time to market
            - Reduced infrastructure concerns speed up application development
    - Challenges and Risks
        - Vendor Lock-in
            - Reliance on proprietary interfaces limits flexibility and may increase costs
        - Immaturity of best practices
            - Serverless is a relatively new field, and best practices are still evolving
    - Not a one-size-fits-all solution
        - Consider the specific needs and requirements of your application; serverless introduces challenges like Vendor Lock-in and service provider dependencies

- **Microservices**
    - *Microservices*
        - Architectural style for breaking down large applications into small, independent services
        - Each microservice runs a unique process and communicates through a well-defined, lightweight mechanism
        - Contrasts with traditional monolithic architecture, where all components are interconnected
            - Each service in the microservice architecture is self-contained and able to

run independently

- Advantages of Microservices
  - Scalability
    - Services can be scaled independently based on demand
  - Flexibility
    - Microservices can use different technologies and be managed by different teams
  - Resilience
    - Isolation reduces the risk of system-wide failures
  - Faster Deployments and Updates
    - Independent deployment and updates allow for agility and reduced deployment risk
- Challenges of Microservices
  - Complexity
    - Managing multiple services involves inter-service communication, data consistency, and distributed system testing
  - Data Management
    - Each microservice can have its own database, leading to data consistency challenges
  - Network Latency
    - Increased inter-service communication can result in network latency and slower response times
  - Security
    - The distributed nature of microservices increases the attack surface, requiring robust security measures

- **Network Infrastructure**
  - *Network Infrastructure*
    - Backbone of modern organizations
    - Comprises hardware, software, services, and facilities for network support and management
  - *Physical Separation*
    - Security measures to protect sensitive information
    - Often referred to as "Air Gapping”
    - Isolates a system by physically disconnecting it from all networks
    - Physical separation is one of the most secure methods of security, but it is still vulnerable to sophisticated attacks
  - *Logical Separation*
    - Establishes boundaries within a network to restrict access to certain areas
    - Implemented using firewalls, VLANs, and network devices
  - Comparison
    - Physical Separation (Air-Gapping)
      - High security, complete isolation
    - Logical Separation
      - More flexible, easier to implement
      - Less secure if not configured properly

- **Software-defined Network (SDN)**
  - *Software-Defined Network (SDN)*
    - Revolutionary approach to network management
    - Enables dynamic, programmatically efficient network configuration
    - Improves network performance and monitoring

        - Reduces complexity in static and inflexible network architectures
        - Provides a centralized view of the entire network
    - *SDN Architecture*
        - Decouples network control and forwarding functions
        - Three Distinct Planes
            - *Data Plane (Forwarding Plane)*
                - Responsible for handling data packets
                - Makes decisions based on protocols like IP and Ethernet
                - Concerned with sending and receiving data
            - *Control Plane*
                - Centralized decision-maker in SDN
                - Dictates traffic flow across the entire network
                - Replaces traditional, distributed router control planes
                - Increases network manageability and flexibility
            - *Application Plane*
                - Hosts all network applications that interact with the SDN controller
                - Applications instruct the controller on network management
                - Controller manipulates the network based on these instructions

- **Infrastructure as Code (IaC)**
    - *Infrastructure as Code (IaC)*
        - Modern approach to IT infrastructure management
        - Automates provisioning and management through code
        - Used in DevOps and with cloud computing

- IaC Method
  - Developers and ops teams manage infrastructure through code
  - Code files are versioned, tested, and audited
  - High-level languages like YAML, JSON, or domain-specific languages (e.g., HCL) used
  - Idempotence ensures identical environments
    - *Idempotence*
      - Operation consistently produces the same results
      - Crucial for consistency and reliability in multiple environments
- Benefits of IaC
  - Speed and Efficiency
  - Consistency and Standardization
  - Scalability
  - Cost Savings
  - Auditability and Compliance
- Challenges
  - Learning Curve
    - New skills and mindset required
    - Teams learn to write, test, and maintain infrastructure code
  - Complexity
    - Infrastructure code can become complex
    - Mitigated with modularization and documentation
  - Security Risks
    - Sensitive data exposure in code files
    - Insecure configurations may be introduced

- **Centralized vs Decentralized Architectures**
  - *Centralized Architecture*
    - All computing functions managed from a single location or authority
    - Components
      - Central Server
      - Mainframe
      - Data Center
    - Data and applications stored in one place, accessed via a network
    - Benefits
      - Efficiency and Control
        - High resource control and efficient resource allocation
      - Consistency
        - Ensures uniform and accurate data across the organization
      - Cost-effective
        - Reduced maintenance and infrastructure costs
    - Risks
      - Single Point of Failure
        - Server failure can disrupt the entire network
      - Scalability Issues
        - Struggles to handle growth, leading to performance problems
      - Security Risks
        - Attractive targets for cybercriminals; compromised server risks data and app security
  - *Decentralized Architecture*
    - Computing functions distributed across multiple systems or locations

        - No single point of control; each node operates independently
        - Benefits
            - Resilience
                - Can continue functioning despite individual node failures
            - Scalability
                - Easily scales with organization growth by adding new nodes
            - Flexibility
                - Supports remote work and distributed teams
        - Risks
            - Security Risks
                - Vulnerable to security threats, especially in remote work scenarios
            - Management Challenges
                - Complex management, coordinating multiple nodes
            - Data Inconsistency
                - Potential issues with data consistency and synchronization
    - Considerations for Choosing Architecture
        - Choice depends on the organization's specific needs and context
        - Centralized systems for
            - Data accuracy and resource management priorities
        - Decentralized systems for
            - Resilience, flexibility, and rapid scaling needs

- **Internet of Things (IoT)**
    - *Internet of Things (IoT)*
        - Network of physical devices with sensors, software, and connectivity
        - Enables data exchange among connected objects

- *Hub/Control System*
  - Central component connecting IoT devices
  - Collects, processes, analyzes data, and sends commands
  - Can be a physical device or software platform
- *Smart Devices*
  - Everyday objects enhanced with computing and internet capabilities
  - Sense environment, process data, and perform tasks autonomously
- *Wearables*
  - Subset of smart devices worn on the body
  - Monitor health, provide real-time information, and offer hands-free interface
- *Sensors*
  - Detect changes in environment, convert into data
  - Measure various parameters (temperature, motion, etc.)
  - Enable interaction and autonomous decisions in smart devices
- IoT Risks
  - Weak Default Settings
    - Common security risk
    - Default usernames/passwords are easy targets for hackers
    - Changing defaults upon installation is essential
  - Poorly Configured Network Services
    - Devices may have vulnerabilities due to open ports, unencrypted communications
    - Unnecessary services can increase attack surface
    - Keeping IoT devices on a separate network is recommended

- **ICS and SCADA**
  - *Industrial Control Systems (ICS)*
    - Systems used to monitor and control industrial processes, found in various industries like electrical, water, oil, gas, and data
    - *Distributed Control Systems (DCS)*
      - Used in control production systems within a single location
    - *Programmable Logic Controllers (PLCs)*
      - Used to control specific processes such as assembly lines and factories
  - *Supervisory Control and Data Acquisition (SCADA) Systems*
    - Type of ICS designed for monitoring and controlling geographically dispersed industrial processes
    - Common in industries like
      - Electric power generation, transmission, and distribution systems
      - Water treatment and distribution systems
      - Oil and gas pipeline monitoring and control systems
  - Risks and Vulnerabilities
    - Unauthorized Access
      - Unauthorized individuals can manipulate system operations without proper protection
    - Malware Attacks
      - Vulnerable to disruptive malware attacks
    - Lack of Updates
      - Running outdated software with unpatched vulnerabilities
    - Physical Threats
      - Susceptible to damage to hardware or infrastructure

- Securing ICS and SCADA Systems
  - Implement Strong Access Controls
    - Strong passwords
    - Two-factor authentication
    - Limited access to authorized personnel only
  - Regularly Update and Patch Systems
    - Keep systems updated to protect against known vulnerabilities
  - Use Firewall and Intrusion Detection Systems
    - Detect and prevent unauthorized access
  - Conduct Regular Security Audits
    - Identify and address potential vulnerabilities through routine assessments
  - Employee Training
    - Train employees on security awareness and response to potential threats

- **Embedded Systems**
  - *Embedded Systems*
    - Specialized computing components designed for dedicated functions within larger devices
    - They integrate hardware and mechanical elements and are essential for various daily-use devices
  - *Real-Time Operating System (RTOS)*
    - Designed for real-time applications that process data without significant delays
    - Critical for time-sensitive applications like flight navigation and medical equipment

- Risks and Vulnerabilities in Embedded Systems
  - Hardware Failure
    - Prone to failure in harsh environments
  - Software Bugs
    - Can cause system malfunctions and safety risks
  - Security Vulnerabilities
    - Vulnerable to cyber-attacks and unauthorized access
  - Outdated Systems
    - Aging software and hardware can be more susceptible to attacks
- Key Security Strategies for Embedded Systems
  - *Network Segmentation*
    - Divide the network into segments to limit potential damage in case of a breach
  - *Wrappers (e.g., IPSec)*
    - Protect data during transfer by hiding data interception points
  - *Firmware Code Control*
    - Manage low-level software to maintain system integrity
  - Challenges in Patching
    - Updates face operational constraints; OTA updates demand meticulous planning and security measures
      - *Over-the-Air (OTA) Updates*
        - Patches are delivered and installed remotely