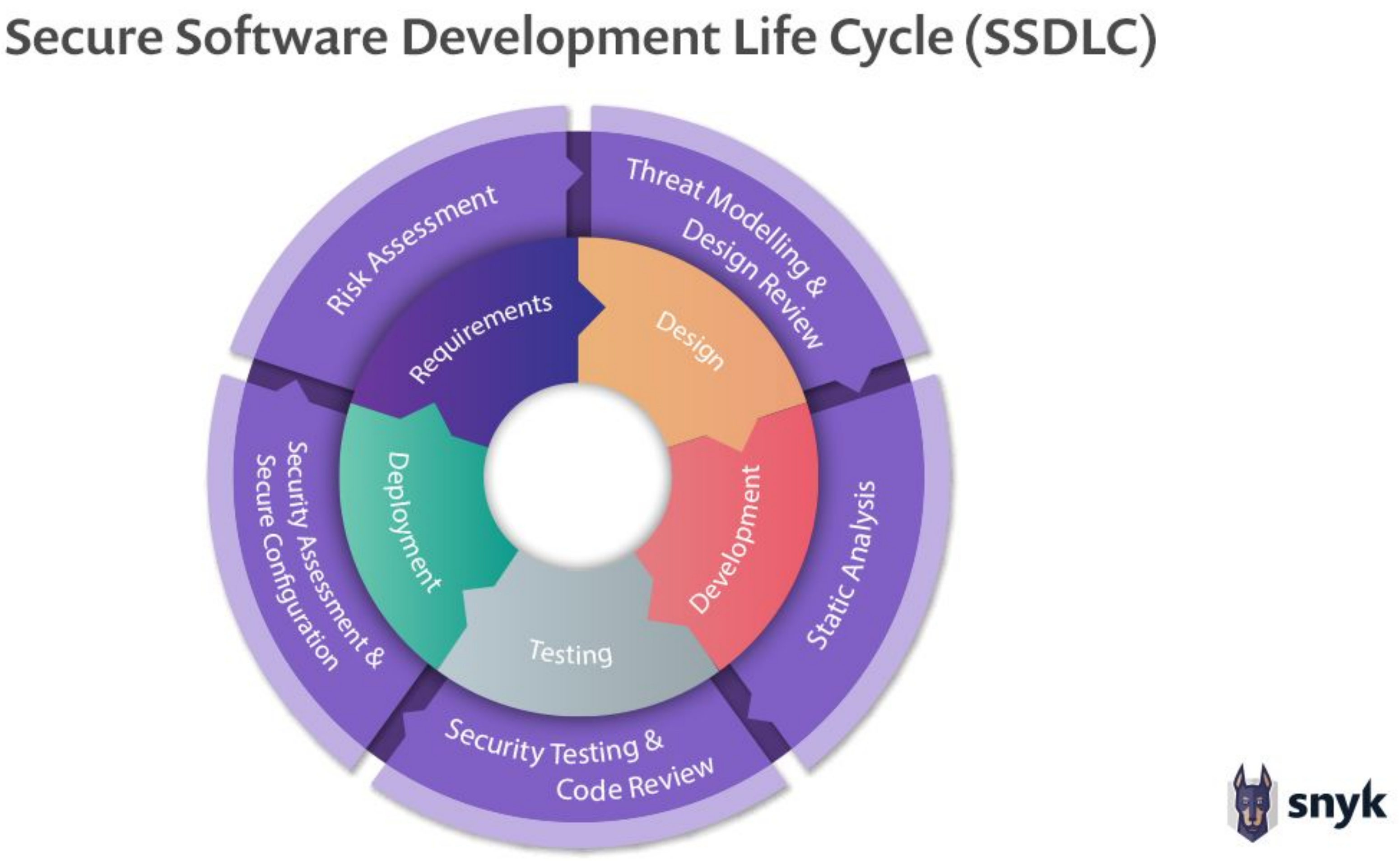

Secure Software Development Life Cycle (SSDLC)
Risk Assessment
Threat Modelling & Design Review
Static Analysis
Security Testing & Code Review
Security Assessment & Secure Configuration
Requirements
Design
Development
Testing
Deployment
snyk

Secure Software Development Life Cycle (SSDLC)

SSDLC Stages

- **Planning and Requirements Gathering**

  Define security requirements alongside functional requirements, identify threats, address regulatory compliance

- **Design**

  Integrate security features like encryption, access controls, and threat modeling

- **Development**

  Follow secure coding practices, use SAST tools to identify vulnerabilities early

- **Testing**

  Employ static and dynamic testing approaches, including penetration testing and fuzz testing

- **Deployment**

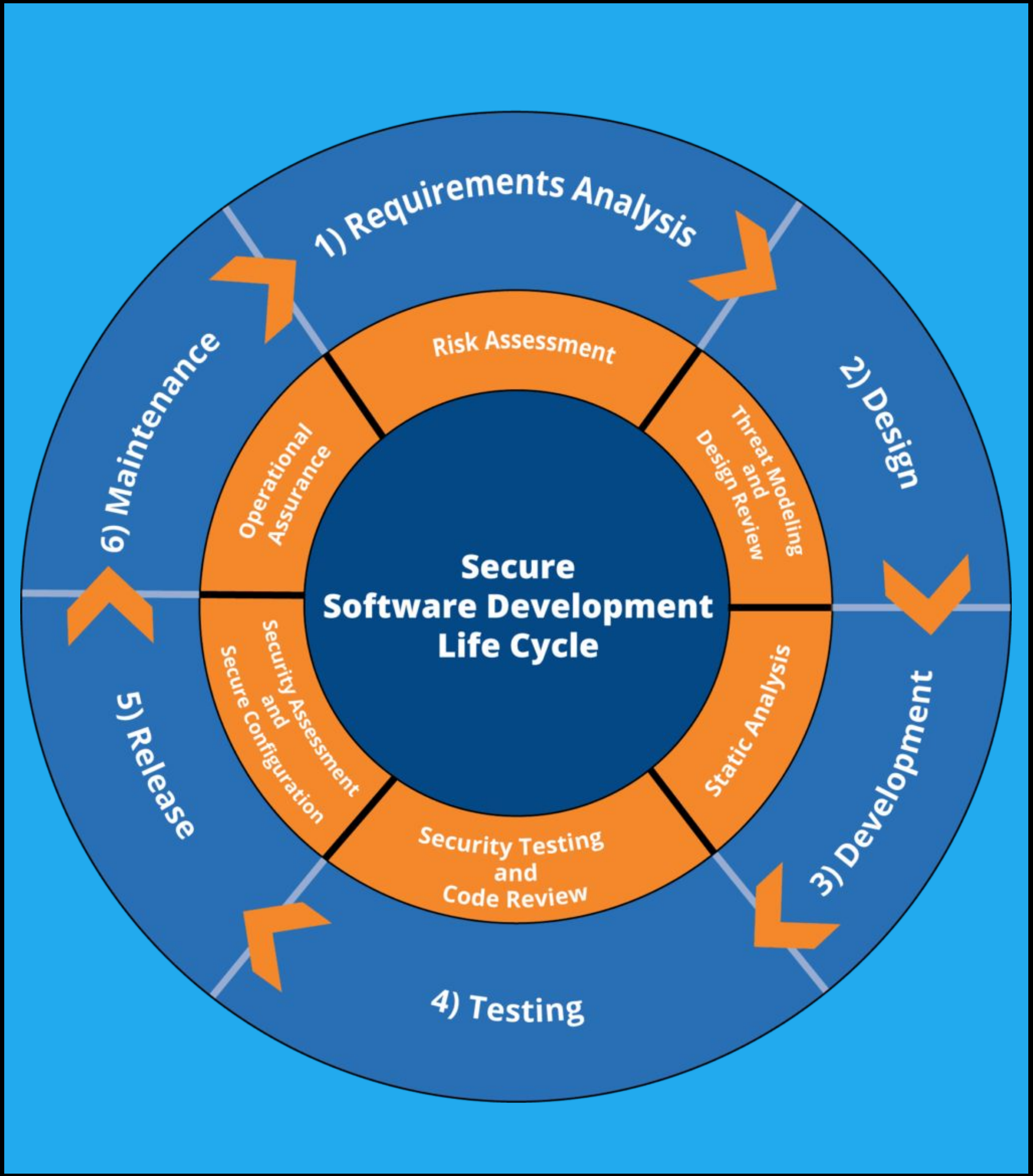
  Secure configurations, patch known vulnerabilities, and harden the application environment

- **Maintenance**

  Monitor for emerging threats, apply security patches promptly, conduct regular vulnerability assessments

- **Retirement**

  Securely decommission the application, remove sensitive data and eliminate vulnerabilities


Secure Software Development Life Cycle
1) Requirements Analysis
2) Design
3) Development
4) Testing
5) Release
6) Maintenance
Risk Assessment
Threat Modeling and Design Review
Static Analysis
Security Testing and Code Review
Security Assessment and Secure Configuration
Operational Assurance

Threat Modeling

The SSDLC involves several key stages, including planning and requirements gathering, design, development, testing, deployment, maintenance, and retirement. Security considerations are addressed at each phase to minimize risks and ensure the overall security of the application.

Pre-Deployment Testing

- **Static Application Security Testing (SAST)**

  Analyzing the source code to detect vulnerabilities such as insecure coding practices.

- **Dynamic Application Security Testing (DAST)**

  Testing the application while it is running to identify vulnerabilities like injection flaws, cross-site scripting (XSS), and authentication weaknesses.

- **Penetration Testing**

  Ethical hackers simulate real-world attacks to uncover exploitable vulnerabilities.

- **Code Reviews**

  Peer reviews of the codebase to ensure secure coding standards are followed.

Post-Deployment Testing

Vulnerability Scanning

Continuous scanning for known vulnerabilities in third-party libraries, services, and infrastructure components.

Penetration Testing

Conducted periodically to identify new attack vectors.

Red Teaming

A simulated attack on the system from the perspective of an adversary.

Incident Response Testing

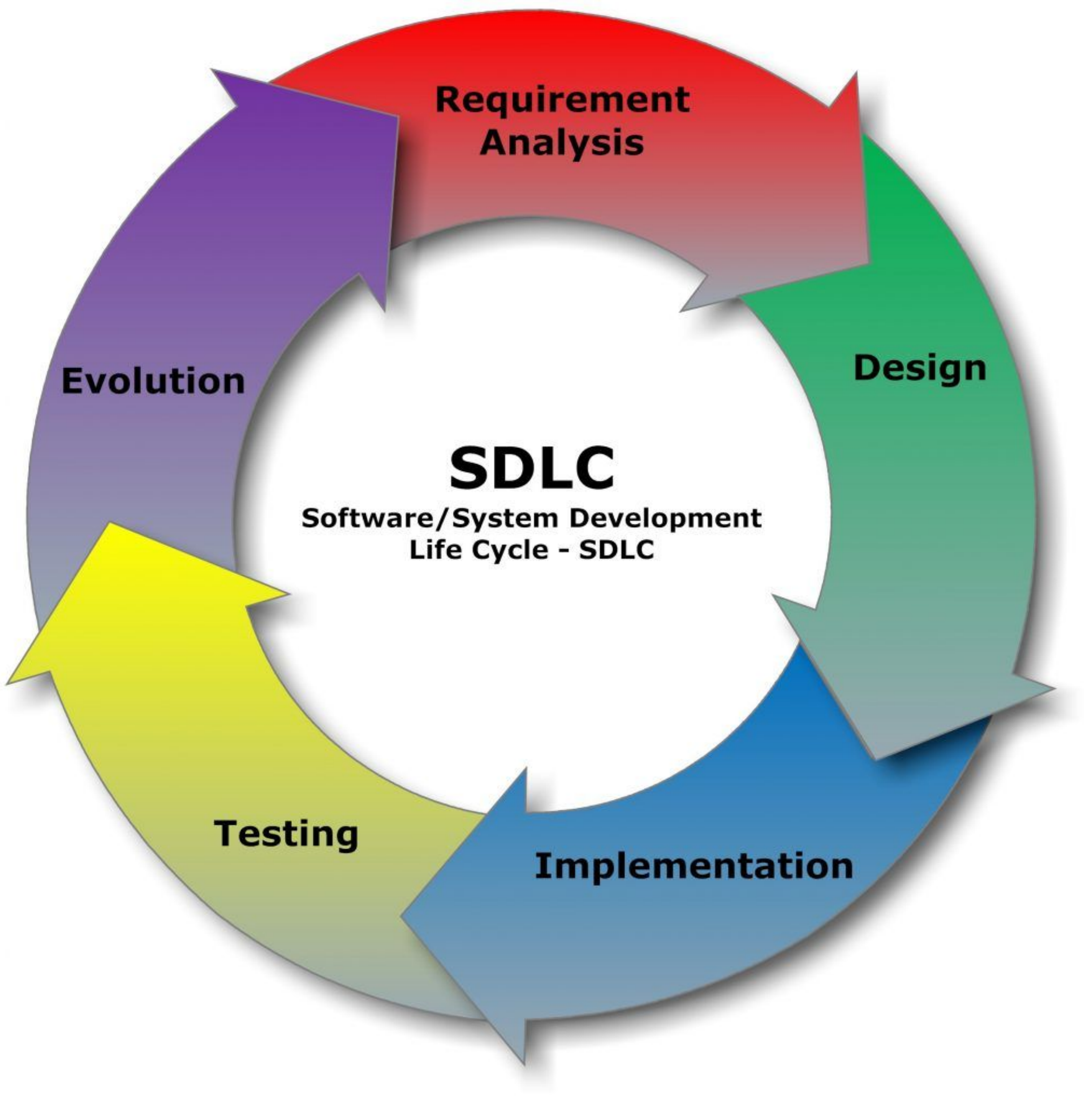
Evaluating how well the organization can detect, respond to, and recover from a security incident.

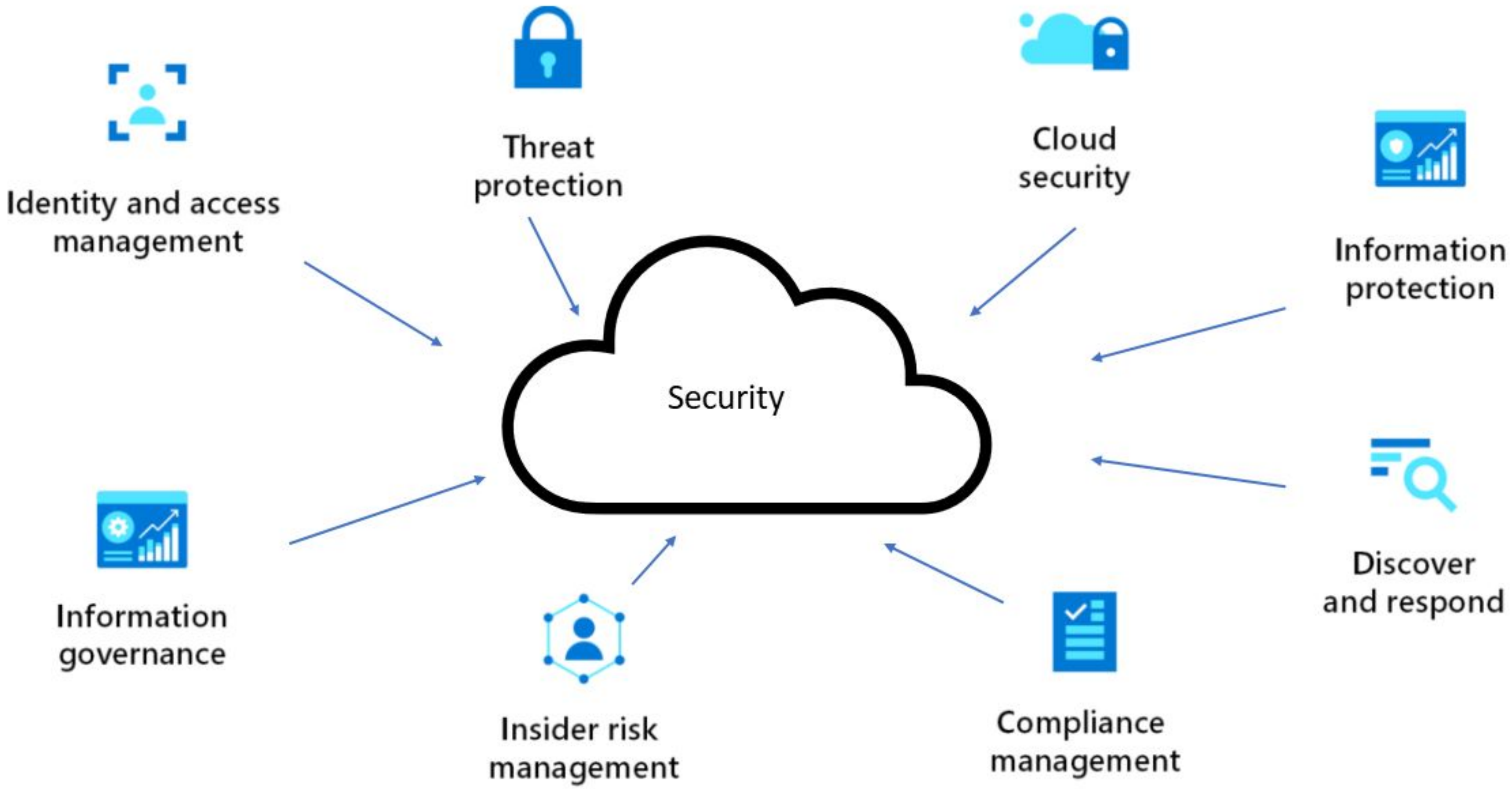
Continuous security testing and monitoring are crucial to identify and address vulnerabilities after deployment, ensuring the ongoing protection of the application.

“Security is not just a technical problem, it's a people problem.”

Secure Software Development Life Cycle (SSDLC)

The Secure Software Development Life Cycle (SSDLC) is a structured approach to software development that integrates security practices into each stage of the Software Development Life Cycle (SDLC) to minimize vulnerabilities, threats, and risks from the outset. The SSDLC encompasses various phases, including planning and requirements gathering, design, development, testing, deployment, maintenance, and retirement, to ensure the development of secure applications.





Securing Cloud Applications: Architecting for Resilience and DevSecOps

Cloud Impacts on Architecture-Level Security

Shared Responsibility Model

Cloud providers responsible for infrastructure security, customers manage application and data security

Data Sovereignty

Data may be stored in multiple regions, legal requirements for data protection can vary

Multi-tenancy

Ensuring isolation between tenants in a shared environment is a challenge

Careful consideration of cloud-specific security factors is crucial when designing the architecture of secure cloud applications.

Architectural Resilience

Redundancy

Deploy multiple instances of critical systems to ensure service continuity during failures

Scalability

Design systems that can scale up or down based on demand while maintaining performance and security

Automated Monitoring and Response

Implement automated security monitoring to detect anomalies and respond quickly to incidents

Building resilient cloud architectures is crucial for ensuring the security and availability of applications.

Identity & Access Management and Application Security

- **Secure Secrets Management**

  Encrypt sensitive information like API keys, passwords, and encryption keys. Use secure vaults and environment variables to avoid hardcoding secrets.

- **Identity and Access Management (IAM)**

  Ensure only authorized users can access sensitive resources, reducing the risk of data breaches.

- **Shift Left Security**

  Integrate security tests and checks into the early stages of the CI/CD pipeline, catching vulnerabilities early in the development process.

- **Web Application Firewalls (WAF)**

  Monitor and filter HTTP traffic to protect against attacks like SQL injection, cross-site scripting, and DDoS.

- **API Gateways**

  Manage rate-limiting, authentication, logging, and other security controls for APIs.

DevOps & DevSecOps

- **DevOps: Collaboration between Dev and Ops teams**

  Focuses on streamlining the software delivery process through automation and communication

- **DevSecOps: Integrating security practices into the DevOps pipeline**

  Ensures security is addressed early in the development cycle, reducing risks during deployment

- **CI/CD Pipelines and Shift Left**

  Automatically build, test, and deploy applications, with security tests integrated into the early stages

- **Web Application Firewalls (WAF)**

  Monitor and filter HTTP traffic to protect against attacks like SQL injection and DDoS

- **API Gateways**

  Control and monitor traffic between clients and backend services, managing security controls for APIs

Secure Cloud Application Architecture

- **Secure Cloud Application Architecture**

  Designing cloud applications with the assumption of breaches to ensure security is woven into the fabric of the system.

- **Shared Responsibility Model**

  Cloud providers are responsible for the security of the cloud infrastructure, while customers must manage the security of applications, data, and workloads.

- **Architectural Resilience**

  Building systems that can withstand attacks or failures without compromising security or availability, including redundancy, scalability, and automated monitoring.

- **Identity & Access Management**

  Ensuring only authorized users can access sensitive resources to reduce the risk of data breaches.

- **Secrets Management**

  Securely storing, accessing, and managing sensitive information like API keys, passwords, and encryption keys.

- **DevSecOps**

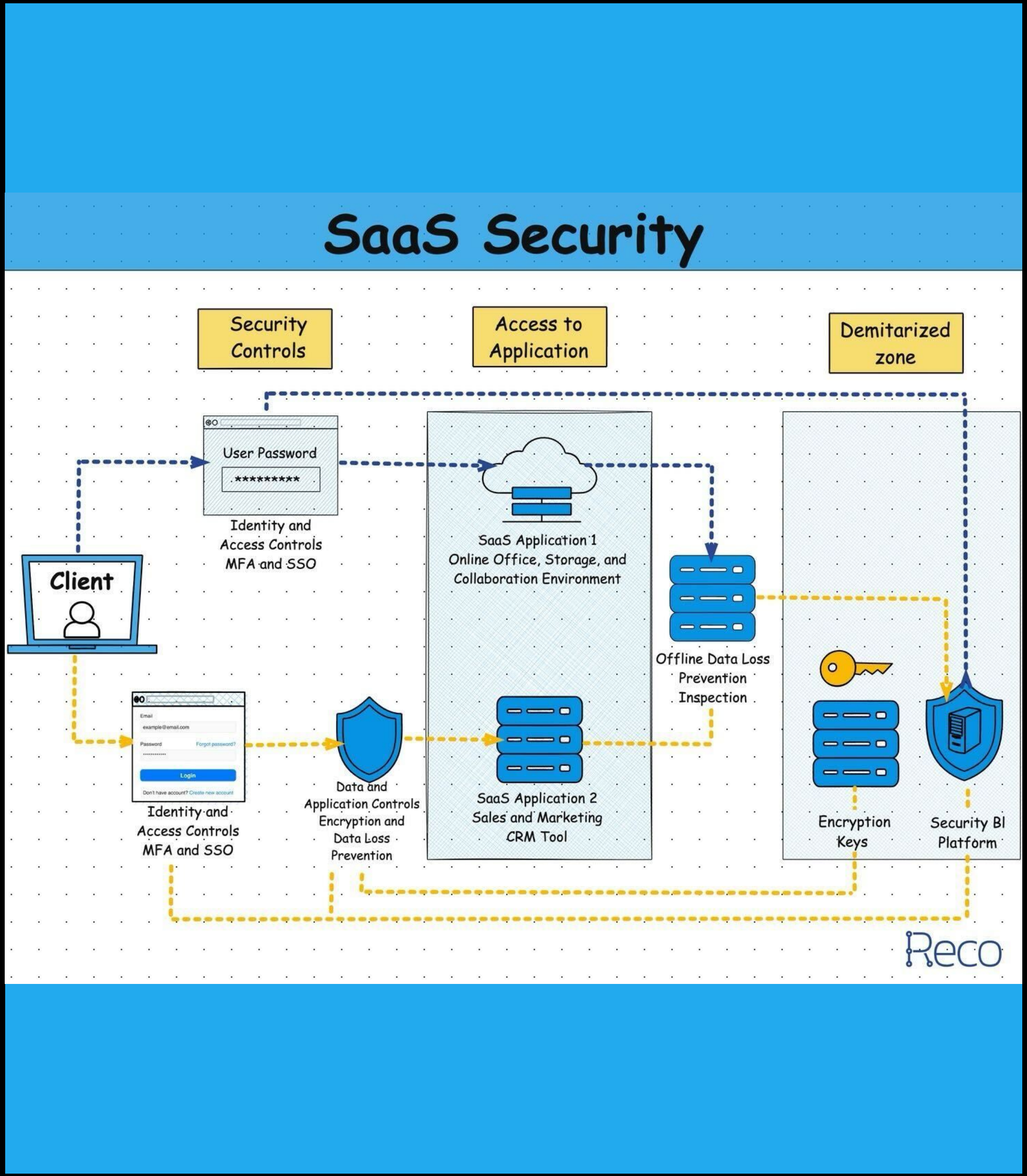
  Integrating security practices into the DevOps pipeline to address security early in the development cycle.

- **Web Application Firewalls (WAF)**

  Monitoring and filtering HTTP traffic to protect web applications against attacks like SQL injection, XSS, and DDoS.

- **API Gateways**

  Controlling and monitoring traffic between clients and backend services, managing rate-limiting, authentication, logging, and other security controls for APIs.


SaaS Security
Security Controls
Access to Application
Demitarized zone
User Password
****
Identity and Access Controls MFA and SSO
Client
SaaS Application 1 Online Office, Storage, and Collaboration Environment
Offline Data Loss Prevention Inspection
Identity and Access Controls MFA and SSO
Data and Application Controls Encryption and Data Loss Prevention
SaaS Application 2 Sales and Marketing CRM Tool
Encryption Keys
Security BI Platform
Reco

Case Study: Security Implementation in a Cloud Application

This case study explores the security measures implemented in a cloud-based healthcare application that handles sensitive patient data. The application was developed using a microservices architecture, and the focus was on ensuring security at every stage of the software development lifecycle (SLDLC).

Key Security Measures for Secure Cloud Applications

Shared Responsibility Model

Cloud providers are responsible for securing the infrastructure, while customers must manage the security of their applications, data, and workloads.

Data Sovereignty

Data may be stored in multiple regions or countries, and legal requirements for data protection can vary depending on the location.

Multi-tenancy

Ensuring isolation between tenants in a shared cloud environment is a key security challenge.

Architectural Resilience

Designing systems with redundancy, scalability, and automated monitoring and response to withstand attacks or failures.

Secrets Management

Securely storing, accessing, and managing sensitive information like API keys, passwords, and encryption keys.

Shift Left with CI/CD

Integrating security tests and checks into the early stages of the CI/CD pipeline to catch vulnerabilities early in the development process.

Architectural Resilience in Action

Architecture's Role in Secure Cloud Applications

Cloud architectures need to be designed with the assumption that breaches will happen, so security needs to be woven into the system from the outset.

Cloud Impacts on Architecture-Level Security

The transition to cloud environments introduces new security considerations that can affect the architecture design, including the shared responsibility model, data sovereignty, and multi-tenancy.

Architectural Resilience

Building resilient architectures includes implementing redundancy, scalability, and automated monitoring and response to ensure service continuity and security during attacks or failures.

Identity & Access Management and Application Security

Effective Identity and Access Management (IAM) ensures that only authorized users can access sensitive resources, reducing the risk of data breaches. Secure secrets management is critical for maintaining the confidentiality and integrity of application data.

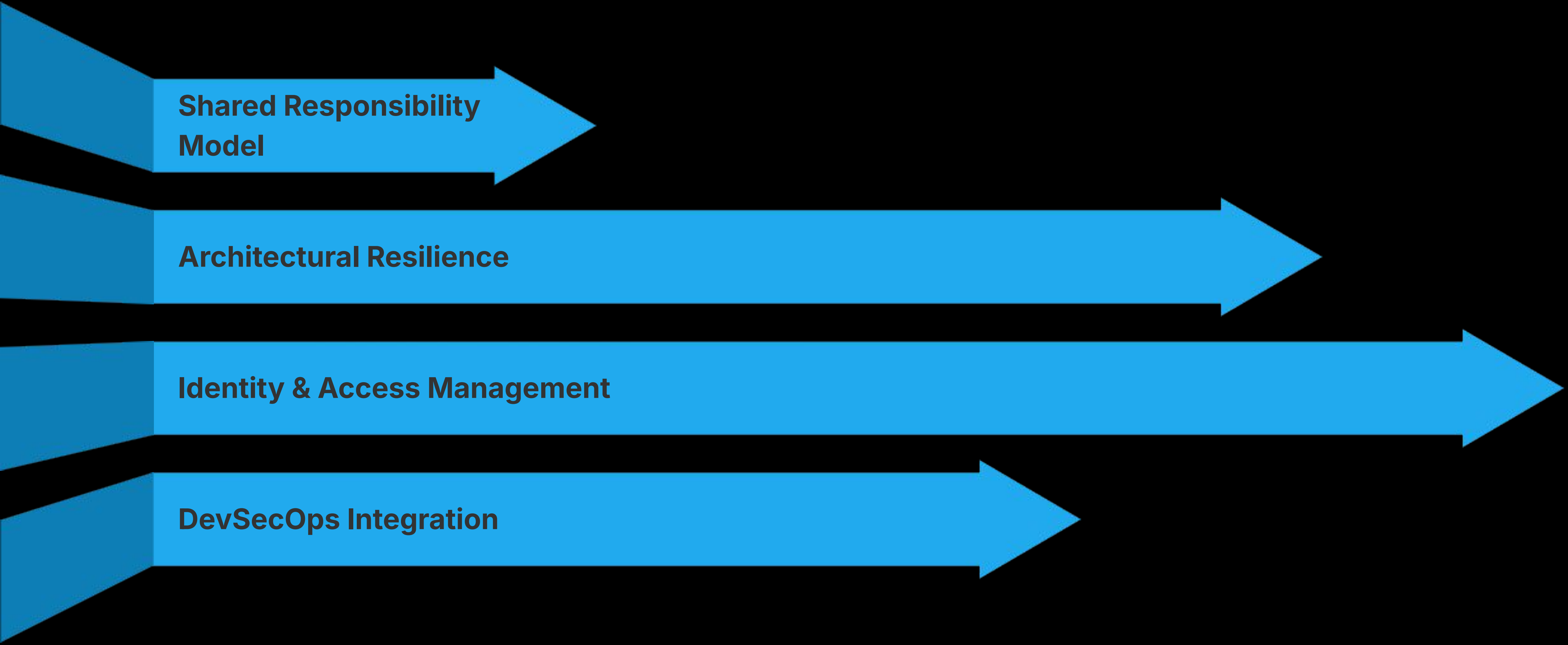
DevOps & DevSecOps

DevSecOps integrates security practices into the DevOps pipeline, ensuring that security is addressed early in the development cycle and vulnerabilities are caught before deployment.

Web Application Firewalls & API Gateways

Web Application Firewalls (WAFs) monitor and filter HTTP traffic to protect against attacks, while API Gateways control and monitor traffic between clients and backend services, managing security controls.

Securing the Cloud Application Architecture



Shared Responsibility Model

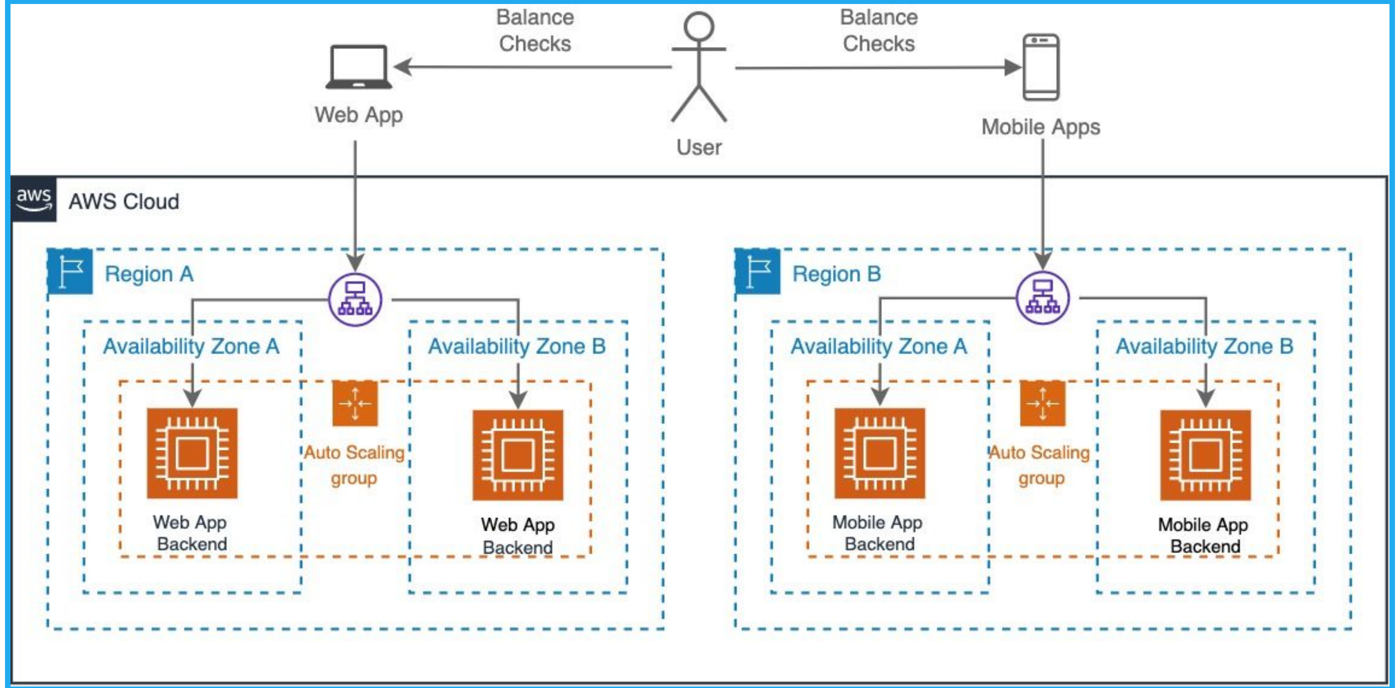
Architectural Resilience

Identity & Access Management

DevSecOps Integration

Securing Cloud Applications: Architecting for Resilience and DevSecOps

This slide focuses on the role of architecture in securing cloud applications. Cloud architectures need to be designed with the assumption that breaches will happen, so security needs to be woven into the system from the outset. Key architectural considerations include the shared responsibility model, data sovereignty, and multi-tenancy challenges.


Balance Checks
Balance Checks
Web App
User
Mobile Apps
AWS Cloud
Region A
Availability Zone A
Availability Zone B
Auto Scaling group
Web App Backend
Web App Backend
Region B
Availability Zone A
Availability Zone B
Auto Scaling group
Mobile App Backend
Mobile App Backend