INSTITUTION OF TECHNOLOGY

SCHOOL OF COMPUTING

DEPARTMENT OF SOFTWARE ENGINEERING

Software Engineering Tools and Practice (SEng3051)

Individual Assignment

DevSecOps

Name: SULEYMAN ABDU MOHAMMED ID: 147312

Submitted to: Instructor Esmail M.

MAY 29, 2024

Woldia, Ethiopia

Table of Contents

Introduction

The integration of security into the software development lifecycle (SDLC) has become a paramount concern in today's fast-paced, technology-driven world. Traditional methods of addressing security as an afterthought have proven insufficient in mitigating the sophisticated cyber threats that continue to evolve. DevSecOps, a practice that amalgamates development, security, and operations, emerged as a robust response to these challenges. By embedding security practices within the DevOps framework, DevSecOps ensures that security is a shared responsibility and an integral part of the software development process. This report delves into the genesis of DevSecOps, its lifecycle, operational mechanisms, key tools, benefits, and career opportunities in both local and international contexts.

1. What are Software engineering problems which was cause for initiation of DevSecOps.

DevSecOps emerged as a response to several software engineering problems that highlighted the need for integrating security into the DevOps practices. Here are some of the key issues that led to the initiation of DevSecOps:

1. **Security as an Afterthought:**
   - Traditional software development methodologies often treated security as a final step, addressed only during the testing phase or just before deployment. This approach led to discovering vulnerabilities too late in the development cycle, making them costly and time-consuming to fix.
2. **Siloed Teams:**
   - Development, operations, and security teams traditionally worked in silos with limited collaboration. This lack of integration resulted in communication gaps, inefficiencies, and a slower response to security threats.
3. **Increased Complexity of Modern Applications:**
   - Modern applications are more complex, involving microservices, containers, and continuous integration/continuous deployment (CI/CD) pipelines. Managing security across these dynamic and distributed environments is challenging.
4. **Accelerated Development Cycles:**
   - Agile and DevOps practices have significantly accelerated the development cycles, leading to multiple deployments per day. This rapid pace made it difficult to perform thorough security checks using traditional methods, often leading to insecure code being deployed.
5. **Growing Threat Landscape:**
   - The sophistication and frequency of cyber threats have increased dramatically. This growing threat landscape requires continuous and proactive security measures that traditional security approaches couldn't adequately address.
6. **Manual Security Processes:**

- Many security processes were manual and not integrated into the automated CI/CD pipelines, leading to delays and human errors. Manual reviews and testing couldn't keep up with the speed of modern development practices.

7. **Compliance and Regulatory Requirements:**
    - Increasing regulatory requirements and industry standards necessitate continuous monitoring and documentation of security practices. Traditional methods made it difficult to maintain compliance in a fast-paced development environment.
8. **Lack of Security Awareness:**
    - Developers often lacked the necessary security knowledge and training, leading to the introduction of vulnerabilities in the code. There was a need for integrating security education and practices directly into the development process.
9. **Resource Constraints:**
    - Security teams are often understaffed and overburdened, unable to handle the volume of security issues that arise in modern development environments. Integrating security into the DevOps pipeline helps distribute the responsibility of security across all teams.

2. What is DevSecOps?

DevSecOps is the practice of integrating security into a continuous integration, continuous delivery, and continuous deployment pipeline. By incorporating DevOps values into software security, security verification becomes an active, integrated part of the development process.

DevSecOps stands for development, security, and operations. It is an extension of the DevOps practice. Each term defines different roles and responsibilities of software teams when they are building software applications.

Development: Development is the process of planning, coding, building, and testing the application.

Security: Security means introducing security earlier in the software development cycle. For example, programmers ensure that the code is free of security vulnerabilities, and security practitioners test the software further before the company releases it.

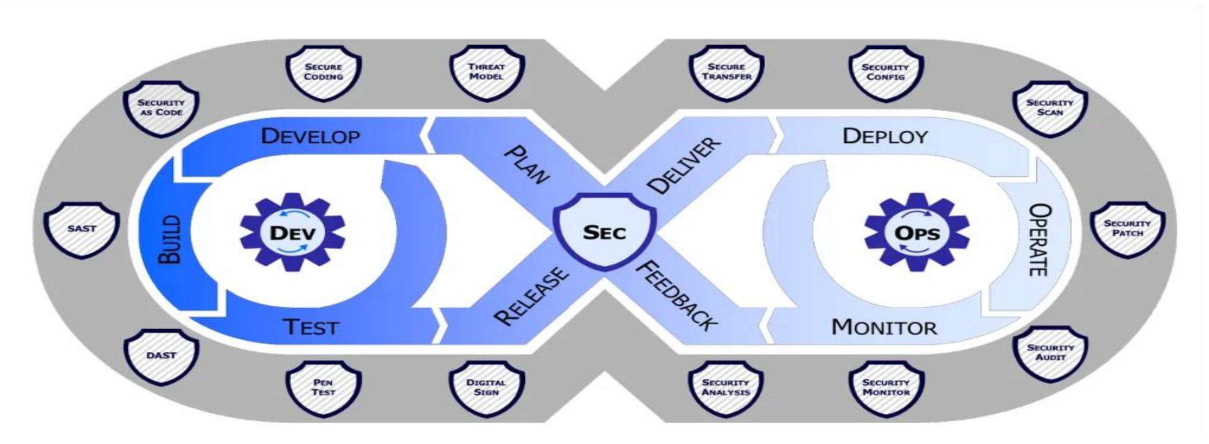
Operations: The operations team releases, monitors, and fixes any issues that arise from the software.

DevSecOps is a framework that integrates security into all phases of the software development lifecycle. Organizations adopt this approach to reduce the risk of releasing code with security vulnerabilities. Through collaboration, automation, and clear processes, teams share responsibility for security, rather than leaving it to the end when issues can be much more difficult and costly to address. DevSecOps is a critical component of a multicloud security strategy.

DevSecOps is an approach to software development that integrates security practices into every phase of the software development lifecycle (SDLC).

DevSecOps is the practice of integrating security testing at every stage of the software development process. It includes tools and processes that encourage collaboration between developers, security specialists, and operation teams to build software that is both efficient and secure. DevSecOps brings cultural transformation that makes security a shared responsibility for everyone who is building the software.

3. Briefly explain DevSecOps lifecycle?



The DevSecOps lifecycle integrates security practices into every phase of the software development lifecycle (SDLC), ensuring that security is a continuous, collaborative, and automated process. The key phases of the DevSecOps lifecycle include:

3.1. Planning and Design

The planning phases of DevSecOps integration are the least automated, involving collaboration, discussion, review, and a strategy for security analysis. Teams must conduct a security analysis and develop a schedule for security testing that specifies where, when, and how it will be carried out. Tools like IriusRisk for collaborative threat modeling, Slack for collaboration and conversation, and Jira Software for managing and tracking issues are popular in this phase.

3.2. Coding

Developers can produce better secure code using DevSecOps technologies during the code phase. Code reviews, static code analysis, and pre-commit hooks are essential code-phase security procedures. Every commit and merge automatically starts a security test or review when security technologies are directly integrated into developers' existing Git workflow. Popular security tools include PMD, Gerrit, SpotBugs, CheckStyle, Phabricator, and Find Security Bugs.

3.3. Building

The 'build' step begins once developers develop code for the source repository. The primary objective of DevSecOps build tools is automated security analysis of the build output artifact. Static application software testing (SAST), unit testing, and software component analysis are crucial security procedures. Tools can be implemented into an existing CI/CD pipeline to automate these tests. The most popular tools to create build phase analysis include Checkmarx, SourceClear, Retire.js, SonarQube, OWASP Dependency-Check, and Snyk.

3.4. Testing

The test phase is initiated once a build artifact has been successfully built and delivered to staging or testing environments. Dynamic application security testing (DAST) tools are used throughout the testing process to detect application flows such as authorization, user authentication, endpoints connected to APIs, and SQL injection. Multiple open-source and paid testing tools are available in the current market, including BDD Automated Security Tests, Boofuzz, JBro Fuzz, OWASP ZAP, SecApp suite, GAUNTLET, IBM AppScan, and Arachi.

3.5. Releasing

The application code should have undergone extensive testing when the DevSecOps cycle is released. This stage focuses on protecting the runtime environment architecture by reviewing environment configuration values, including user access control, network firewall access, and personal data management. Configuration management solutions are a crucial security component. Reviewing and auditing the system configuration is possible in this stage. Popular configuration management tools include HashiCorp Terraform, Docker, Ansible, Chef, and Puppet.

3.6. Deployment

If the earlier process goes well, it's the proper time to deploy the build artifact to the production phase. Security issues affecting the live production system should be addressed during deployment. This includes examining any configuration variations between the current production environment and the initial staging and development settings. Runtime verification tools such as Osquery, Falco, and Tripwire can gather data from an active system to assess if it functions as intended. Chaos engineering principles can be applied to test system resilience.

3.7. Operations

Another critical phase is operation, where operations personnel frequently do periodic maintenance. Zero-day vulnerabilities are particularly concerning and should be monitored regularly. IaC tools can protect the organization's infrastructure while swiftly and effectively preventing human error.

3.8. Monitoring

A breach can be avoided if security is constantly being monitored for abnormalities. It's crucial to put in place a robust continuous monitoring tool that operates in real-time to maintain track of system performance and spot any exploits at an early stage.

By integrating these phases into a unified DevSecOps workflow, organizations can ensure that security is an ongoing priority, enhancing the overall speed, security, and reliability of software delivery.

4. How dose DevSecOps works?

DevSecOps integrates security practices into every stage of the software development lifecycle (SDLC). It aims to create a culture where security is a shared responsibility among development, security, and operations teams. By using automation, collaboration, and continuous monitoring, DevSecOps ensures that security is built into the software from the start and maintained throughout its lifecycle.

4.1 Planning and Development

Introducing security early into development sprints not only helps reduce vulnerabilities later down the line but also saves time because it's easier to address issues before code has been built and integrated. During planning and development:

- Threat Modeling: Use threat modeling to identify and mitigate potential threats to the application. This helps build security into the application right from the start.
- Automated Checks: Implement automated checks, such as integrated development environment security plugins, which give developers immediate feedback if there's a potential security risk in the code they've written.
- Code Review: Have someone with security expertise provide recommendations during code reviews to ensure improvements.
- **Code Commit: One of the keys to a successful DevSecOps process is continuous integration. Developers typically commit their code to a central repository several times a day to ensure integration issues are caught early. During this phase:**
- Automated Security Checks: Add automated security checks, including scanning third-party libraries and dependencies, unit testing, and static application security testing.
- Role-Based Access Controls: Deploy role-based access controls to protect your continuous integration and continuous delivery infrastructure from attackers seeking to run malicious code or steal credentials.

4.2 Building and Testing

Running automated security scripts on the test environment helps uncover potential issues that weren't previously detected. Security tests during this phase can include:

- Dynamic Application Security Testing (DAST)
- Infrastructure Scanning
- Container Scanning
- Cloud Configuration Validation
- Security Acceptance Testing

Production

Once the application is deployed to production, some organizations engage in penetration testing to try to find weaknesses in the live environment. In penetration testing, people adopt the mindset of an attacker and search for ways to breach the application.

Operation

Even the best DevSecOps process won’t catch everything, so it’s critical to continuously monitor applications for vulnerabilities and threats. Analytics data can help you evaluate if your security posture is improving and highlight areas for optimization. Continuous monitoring includes:

- Real-Time Monitoring: Tools track system performance and detect anomalies in real time, allowing for prompt responses to potential security incidents.
- Log Management: Tools like Splunk and ELK Stack collect and analyze logs to identify and investigate security events.

4.3 Key Practices in DevSecOps

1. Collaborative Culture

- Shared Responsibility: Security is a collective responsibility, with all team members—developers, security professionals, and operations staff—working together to ensure the software is secure.
- Communication: Open and continuous communication among teams helps identify and address security issues early in the development process.

2. Automation

- Automated Testing: Security tests are automated and integrated into the CI/CD pipeline, ensuring that code is continuously checked for vulnerabilities.

- Infrastructure as Code (IaC): Infrastructure is managed through code, allowing for automated, consistent, and repeatable deployments.

3. Continuous Integration and Continuous Delivery (CI/CD)

- CI/CD Pipelines: Automated pipelines for integrating and delivering code ensure that security checks are part of every build and deployment.
- Early Detection: By integrating security into the CI/CD process, vulnerabilities can be detected and addressed early, reducing the risk of security breaches in production.

4. Security Tools Integration

- Static Application Security Testing (SAST): Tools like Checkmarx and SonarQube analyze source code for vulnerabilities during development.
- Dynamic Application Security Testing (DAST): Tools like OWASP ZAP and IBM AppScan test running applications for security issues.
- Dependency Scanning: Tools like Snyk and OWASP Dependency-Check scan third-party libraries and dependencies for known vulnerabilities.

5. Proactive Security Practices

- Threat Modeling: Teams conduct threat modeling sessions to anticipate potential security threats and design countermeasures.
- Security Training: Regular training and education ensure that all team members are aware of security best practices and emerging threats.

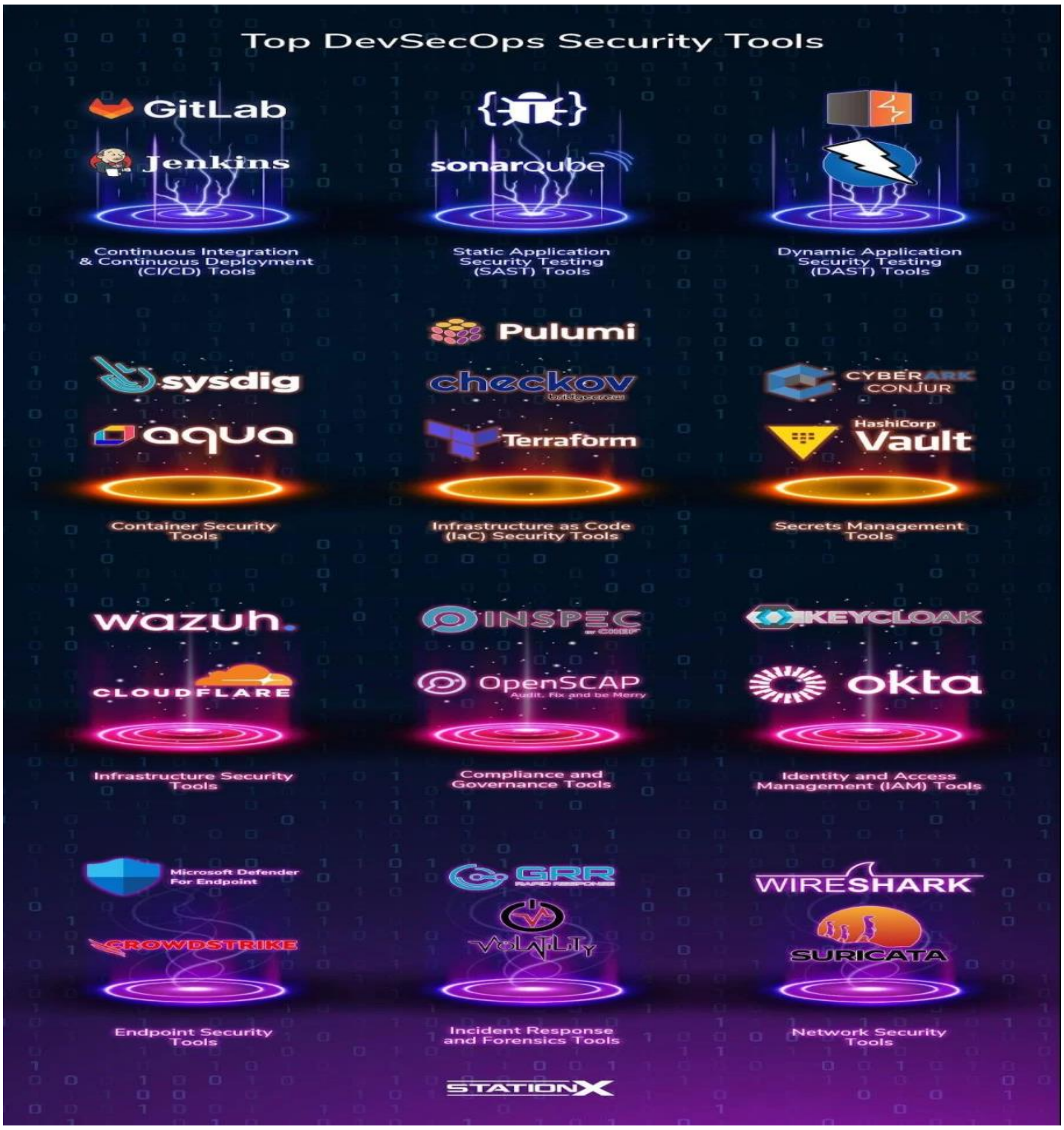
6. Incident Response

- Preparedness: Incident response plans are developed and regularly tested to ensure teams can quickly and effectively respond to security incidents.
- Post-Incident Analysis: After an incident, teams analyze the response and outcomes to identify areas for improvement and prevent future occurrences.

7. Continuous Improvement

- Feedback Loops: Continuous feedback from monitoring, testing, and incident response is used to refine and improve security practices.
- Adaptation: DevSecOps teams continuously adapt to new security challenges, incorporating the latest tools, techniques, and knowledge into their workflows.

5. Explain well known DevSecOps tools.


Top DevSecOps Security Tools
GitLab
Jenkins
Continuous Integration & Continuous Deployment (CI/CD) Tools
sonarqube
Static Application Security Testing (SAST) Tools
Dynamic Application Security Testing (DAST) Tools
sysdig
aqua
Container Security Tools
Pulumi
checkov
Terraform
Infrastructure as Code (IaC) Security Tools
CYBERARK CONJUR
HashiCorp Vault
Secrets Management Tools
wazuh.
CLOUDFLARE
Infrastructure Security Tools
INSPEC
OpenSCAP
Compliance and Governance Tools
KEYCLOAK
okta
Identity and Access Management (IAM) Tools
Microsoft Defender For Endpoint
CROWDSTRIKE
Endpoint Security Tools
GRR
Volatility
Incident Response and Forensics Tools
WIRESHARK
SURICATA
Network Security Tools
STATIONX

some well-known DevSecOps tools that are widely used in the industry:

1.Jenkins

- Purpose: An open-source automation server that helps automate parts of software development related to building, testing, and deploying, facilitating continuous integration and continuous delivery.

- Features: Extensible with a vast array of plugins, integrates with various CI/CD tools, and supports complex build pipelines.

2. GitLab

- Purpose: An end-to-end platform that integrates source code repositories, CI/CD pipelines, and security scanning tools.
- Features: Comprehensive DevSecOps features including source control, CI/CD, code review, and built-in security scanning.

3.SonarQube

- Purpose: Continuous inspection of code quality to perform automatic reviews with static analysis of code to detect bugs, code smells, and security vulnerabilities.
- Features: Supports multiple programming languages, integrates with CI/CD pipelines, and provides detailed reports and metrics.

4. HashiCorp Terraform

- Purpose: A tool for building, changing, and versioning infrastructure safely and efficiently.
- Features: Infrastructure as code, supports multi-cloud environments, and ensures infrastructure compliance through code.

5. Ansible

- Purpose: An open-source software provisioning, configuration management, and application-deployment tool.
- Features: Agentless architecture, easy-to-read YAML configuration files, and strong support for automation and orchestration.

6. Puppet

- Purpose: A software configuration management tool that allows you to manage infrastructure as code.
- Features: Automates the provisioning, configuration, and management of infrastructure, supports complex environments, and integrates with various DevOps tools.

7. Chef

- Purpose: A powerful automation platform that transforms infrastructure into code, enabling you to automate how you build, deploy, and manage your infrastructure.
- Features: Uses a domain-specific language (DSL) for writing system configuration, supports multi-cloud environments, and provides comprehensive automation capabilities.

8. Docker

- Purpose: A set of platform-as-a-service products that use OS-level virtualization to deliver software in packages called containers.
- Features: Simplifies application deployment, consistent environments, and strong support for microservices architecture.

9. Veracode

- Purpose: A cloud-based service that provides automated static and dynamic security testing to find and fix security vulnerabilities.
- Features: Static, dynamic, and software composition analysis, integrates with development tools, and provides detailed security analytics and reporting.

10.Snyk

- Purpose: A developer-first cloud-native application security solution.
- Features: Real-time vulnerability scanning, automated fixes, integration with various development tools and environments, and detailed reporting.

11.Checkmarx

- Purpose: Static Application Security Testing (SAST) to identify vulnerabilities in source code.
- Features: Supports multiple languages, integrates with IDEs, CI/CD pipelines, and issue trackers, and provides detailed reports on vulnerabilities.

12.Aqua Security

- Purpose: Security for containerized environments and cloud-native applications.
- Features: Container image scanning, runtime protection, compliance checks, and integration with CI/CD pipelines.

13. Netsparker

- Purpose: Web application security testing through automated scans.

- Features: Scans for various vulnerabilities, provides detailed reports, integrates with CI/CD tools, and offers both on-premises and SaaS options.

14. OWASP ZAP (Zed Attack Proxy)

- Purpose: Open-source web application security scanner.
- Features: Intercepting proxy, automated scanner, passive scanner, and extensive scripting options for custom tests.

15. HashiCorp Vault

- Purpose: Managing secrets and protecting sensitive data.
- Features: Secure storage of secrets, dynamic secrets, data encryption, and identity-based access.

16.Fortify

- Purpose: Static and dynamic application security testing.
- Features: SAST and DAST capabilities, integrates with IDEs and CI/CD tools, and provides detailed vulnerability analysis and remediation guidance.

17. Black Duck

- Purpose: Open-source security and license management.
- Features: Identifies open-source components, detects vulnerabilities, and manages license compliance.

These tools help integrate security into the CI/CD pipeline, ensuring that security is a part of the software development process from the start. They are designed to automate and streamline various aspects of the DevSecOps workflow, making it easier for teams to incorporate security measures into their development practices.

6. What are the benefits of DevSecOps?

DevSecOps integrates security practices into the DevOps process, promoting a culture where security is a shared responsibility across development, operations, and security teams. Here are the key benefits:

6.1 Enhanced Security

- Continuous Security: Security checks are integrated into every stage of the CI/CD pipeline, ensuring vulnerabilities are identified and addressed early in the development process.
- Automated Security Testing: Automated tools perform regular and consistent security testing, reducing the chances of human error and ensuring thorough coverage.
- Proactive Security: Vulnerabilities in code can be detected early if you implement a DevSecOps approach. The DevSecOps model involves analyzing code and performing regular threat assessments, enabling teams to take control of an application's risk profile instead of merely reacting to issues as they arise.

6.2 Faster Time-to-Market

- Streamlined Processes: Integrating security into the development pipeline reduces the need for lengthy security reviews at the end of the development cycle.
- Early Detection of Issues: Identifying and fixing vulnerabilities early in the development process minimizes delays caused by security issues discovered late in the cycle.

- Automation: Automation in DevSecOps allows software teams to conduct security tests automatically. This reduces human errors and speeds up the development cycle, enabling faster delivery of secure software.

6.3.Cost Savings

- Reduced Remediation Costs: Addressing security issues early in the development process is significantly less expensive than fixing them after deployment.
- Efficient Resource Utilization: Automated security tools free up security experts to focus on more complex and high-value tasks.
- Conserving Person-Hours: With development security operations as an inherent part of the process, vulnerabilities are addressed at each design phase, allowing the development team to release a more secure iteration of the program faster. This also leads to better ROI for security infrastructure as issues are fixed upfront, precluding many future problems.

6.4 Improved Compliance

- Automated Compliance Checks: Continuous monitoring and automated compliance checks ensure that applications and infrastructure adhere to regulatory requirements.
- Audit Trails: Detailed logs and reports from automated tools provide clear audit trails, simplifying compliance audits.
- Integrated Compliance: DevSecOps practices help organizations meet regulatory requirements by incorporating security measures from the beginning, preventing last-minute adjustments.

6.5 Increased Collaboration and Shared Responsibility

- Unified Teams: DevSecOps promotes a culture where development, operations, and security teams work collaboratively towards common goals.
- Shared Responsibility: Security becomes a shared responsibility, reducing the chances of security being an afterthought or overlooked.
- Build a Security-Aware Culture: DevSecOps encourages collaboration between development, security, and operations teams, fostering a culture where security is a shared priority.

6.6 Higher Quality Software

- Comprehensive Testing: Continuous and automated security testing improves the overall quality of the software by ensuring it is secure and free of vulnerabilities.
- Improved Code Quality: Regular code reviews and security scans help maintain high code quality, reducing technical debt and enhancing maintainability.

6.7 Enhanced Trust and Reputation

- Secure Products: Delivering secure products builds customer trust and enhances the organization's reputation.
- Proactive Security Posture: Adopting a proactive security approach demonstrates a commitment to security, which can be a competitive differentiator.

6.8 Adaptability to Change

- Agile Security Practices: DevSecOps practices are agile and adaptable, enabling organizations to quickly respond to new security threats and vulnerabilities.
- Continuous Improvement: Feedback loops and continuous learning enable teams to improve security practices and tools continuously.

6.9 Scalability

- Automated Security Scaling: Automated security tools can scale with the application, ensuring consistent security checks regardless of the application's size or complexity.
- Efficient Resource Management: Scalable security processes allow organizations to manage resources efficiently as they grow.

6.10 Continuous Feedback

- Feedback Loop: DevSecOps creates a continuous feedback loop that interweaves security solutions during the software development process. This enables constant communication and improvement between development and security teams.

- Improved Automation: Continuous feedback helps in developing better-automated security functions and designing automation protocols that benefit specific processes.

6.11 Improved Collaboration Between Teams

- Enhanced Communication: Throughout the entire development lifecycle, communication is enhanced because team members must understand how each facet of an application interfaces with the necessary security measures. This leads to increased collaboration and a more cohesive organization and product.

7. About Local and international DevSecOps career opportunities, career path

Overview

DevSecOps is a growing field that combines development, security, and operations into a cohesive practice. As organizations increasingly prioritize security throughout the software development lifecycle, the demand for DevSecOps professionals has surged both locally and internationally. Career opportunities are available across various sectors, including technology, finance, healthcare, and government.

7.1 Local Opportunities

1. Startups and SMEs: Smaller companies and startups often look for versatile DevSecOps professionals who can wear multiple hats and handle a variety of tasks from coding to security monitoring.
2. Large Enterprises: Corporations with substantial IT infrastructure invest heavily in DevSecOps to ensure their systems are secure. Opportunities in these organizations often come with specialized roles.
3. Government and Public Sector: Governments and public institutions are adopting DevSecOps to secure sensitive data and critical infrastructure. These roles may require specific certifications or clearances.
4. Consulting Firms: Local consulting firms hire DevSecOps experts to help other companies implement and optimize their DevSecOps practices.

7.2 International Opportunities

1. Global Tech Giants: Companies like Google, Amazon, Microsoft, and Facebook have extensive DevSecOps teams and often recruit internationally.
2. Remote Work: The rise of remote work has opened international career opportunities, allowing professionals to work for companies abroad without relocating.
3. International Consultancies: Firms like Accenture, Deloitte, and PwC offer DevSecOps consulting services globally and often look for talent from different parts of the world.
4. Multinational Corporations: Many multinational corporations have DevSecOps roles across various geographical locations, offering opportunities for relocation or remote work.

7.3 DevSecOps Career Path

Entry-Level Positions

1. Junior DevSecOps Engineer: Entry-level role focusing on learning and applying basic DevSecOps principles, assisting in automation, and basic security tasks.
2. Security Analyst: Focuses on monitoring and analyzing security systems, responding to incidents, and conducting vulnerability assessments.

Mid-Level Positions

1. DevSecOps Engineer: Works on implementing and maintaining CI/CD pipelines, integrating security tools, and automating security checks.
2. Security Engineer: Focuses on designing and implementing security measures within the DevOps process, performing advanced security testing and risk assessments.
3. Cloud Security Engineer: Specializes in securing cloud environments, managing cloud security tools, and ensuring compliance with cloud security standards.

Senior-Level Positions

1. Senior DevSecOps Engineer: Leads DevSecOps initiatives, mentors junior staff, and works on complex security challenges and advanced automation.

2. Security Architect: Designs and oversees the implementation of secure systems architecture, ensuring all components adhere to security best practices.
3. DevSecOps Manager: Manages DevSecOps teams, coordinates between development, operations, and security teams, and ensures alignment with business objectives.

Leadership and Specialist Roles

1. Director of DevSecOps: Oversees the entire DevSecOps function within an organization, sets strategic direction, and ensures the integration of security practices across all development processes.
2. Chief Information Security Officer (CISO): Senior executive responsible for an organization's overall security posture, including DevSecOps practices.
3. DevSecOps Consultant: Provides expert advice to organizations on implementing and optimizing DevSecOps practices, often working with multiple clients.

7.4 Skills and Qualifications

Technical Skills

1. Programming and Scripting: Proficiency in languages like Python, Java, and shell scripting.
2. Security Tools: Experience with tools like SonarQube, Snyk, Checkmarx, and OWASP ZAP.
3. Cloud Platforms: Knowledge of AWS, Azure, GCP, and their security features.
4. CI/CD Tools: Experience with Jenkins, GitLab CI/CD, CircleCI, etc.
5. Infrastructure as Code (IaC): Familiarity with Terraform, Ansible, Puppet, and Chef.

Soft Skills

1. Collaboration and Communication: Ability to work effectively across development, operations, and security teams.
2. Problem-Solving: Strong analytical skills to identify and mitigate security risks.
3. Adaptability: Willingness to continuously learn and adapt to new technologies and threats.

DevSecOps is a dynamic and evolving field with abundant opportunities both locally and internationally. A career in DevSecOps can be highly rewarding, offering the chance to work on cutting-edge technologies and play a critical role in securing modern software applications. With the right combination of technical skills, certifications, and experience, professionals can advance through a well-defined career path from entry-level roles to senior leadership positions.

Conclusion

DevSecOps represents a transformative approach in software engineering, prioritizing security throughout the SDLC and fostering a culture of collaboration and shared responsibility among development, security, and operations teams. The integration of automated security checks and proactive security measures significantly enhances the overall quality, speed, and reliability of software delivery. Additionally, the adoption of DevSecOps not only mitigates security risks but also improves compliance, reduces costs, and builds trust with stakeholders. As the demand for secure software development practices continues to rise, the career opportunities in DevSecOps are expanding, offering promising pathways for professionals in the tech industry. Embracing DevSecOps is not merely a trend but a necessity for organizations aiming to thrive in the digital era.

8.Reference

https://cloudsecurityalliance.org/articles/devsecops-tools

https://github.com/paulveillard/cybersecurity-devsecops

https://dodcio.defense.gov/Portals/0/Documents/Library/DevSecOpsReferenceDesign.pdf

https://www.microsoft.com/en-us/security/business/security-101/what-is-devsecops

https://www.practical-devsecops.com/devsecops-life-cycle/

https://www.ibm.com/topics/devsecops

https://et.linkedin.com/jobs/view/leuwint-technologies-devops-engineer-openshift-at-leuwint-technologies-3852308148