Institute of Technology

School of Computing

Department of Software Engineering

Software Engineering Tools and Practices

COURSE TITLE: SOFTWARE ENGINEERING TOOLS AND PRACTICE

COURSE CODE: SEng3051

INDIVIDUAL ASSIGNMENT

STUDENT NAME ID

1. DAGIM SEWNET ---------- 145771

SUBMITTED DATE: MAY /30/ 2024

SUBMMITED TO: Esmael M.

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

The initiation of DevSecOps was driven by several software engineering problems that highlighted the need for integrating security practices into the software development process. Here are some key problems that led to the emergence of DevSecOps:

1. **Security as an Afterthought** Traditional software development processes often treated security as an afterthought, with security measures being added late in the development lifecycle or even after the software was deployed. This approach led to security vulnerabilities being identified only during or after the deployment phase, resulting in costly and time-consuming security fixes.
2. **Lack of Collaboration** Development, security, and operations teams traditionally operated in silos with limited collaboration and communication. The lack of interaction between these teams often resulted in misaligned priorities, delays in addressing security concerns, and a lack of shared responsibility for security.
3. **Slow Feedback Loops** Traditional software development processes had slow feedback loops for security issues. Security assessments and testing were typically performed at the end of the development cycle, leading to delayed identification and resolution of vulnerabilities. This slow feedback loop hindered the ability to address security issues promptly and increased the risk of security breaches.
4. **Compliance Challenge** Organizations faced challenges in meeting regulatory and compliance requirements due to the absence of dedicated security practices integrated into the software development process. Compliance with standards such as Payment

Card Industry Data Security Standard (PCI DSS), Health Insurance Portability and Accountability Act (HIPAA), and General Data Protection Regulation (GDPR) became more complex and time-consuming.

5. **Increasing Security Threats** the evolving threat landscape and the rise in sophisticated cyber-attack highlighted the need for a proactive approach to security. Traditional software development practices often failed to address emerging security threats effectively, leaving software systems vulnerable to attacks.

6. **Slow Time to Market** in some cases, security concerns led to delays in software releases and hindered the organization's ability to respond quickly to market demands. The lack of integrated security practices and the manual nature of security assessments and testing slowed down the software delivery process.

7. **Lack of Security Assurance**

Implement security testing tools and processes early in the development lifecycle to identify and address security vulnerabilities. Conduct regular security assessments and penetration testing to ensure the security of applications. Provide security training and awareness programs for developers and stakeholders to increase security assurance.

8. Organizational Barriers

Foster a culture of collaboration between development, operations, and security teams by promoting cross-functional teams and communication. Invest in DevSecOps tooling that enables seamless integration of security practices into the CI/CD pipeline. Educate stakeholders on the benefits of DevSecOps and the importance of collaboration for achieving security goals**.**

9. **Lack of Security Skills** Provide training and resources for developers to enhance their security skills, such as secure coding practices

and vulnerability detection. Collaborate with security experts or hire external consultants to bridge the gap in security skills within the organization. Encourage continuous learning and professional development in security for both developers and business stakeholders.

10. **Lack of Security Resources**

Allocate budget and resources for implementing security measures, such as security tools, training, and hiring security professionals. Establish security standards and guidelines within the organization to ensure consistent security practices are followed. Leverage open-source security tools and resources to supplement existing security capabilities and overcome resource constraints.

2. What is DevSecOps?

DevSecOps is an approach to software development that integrates security practices into the entire software development lifecycle (SDLC). It combines the principles of DevOps, which focuses on collaboration and automation between development and operations teams, with an added emphasis on incorporating security measures from the very beginning of the development process.

DevSecOps :which is
shortfor *development*, *security* and *operations*, is an application development practice that automates the integration of security and security practices at every phase of the software development lifecycle, from initial design through integration, testing, delivery and deployment.

What does DevSecOps stand for?

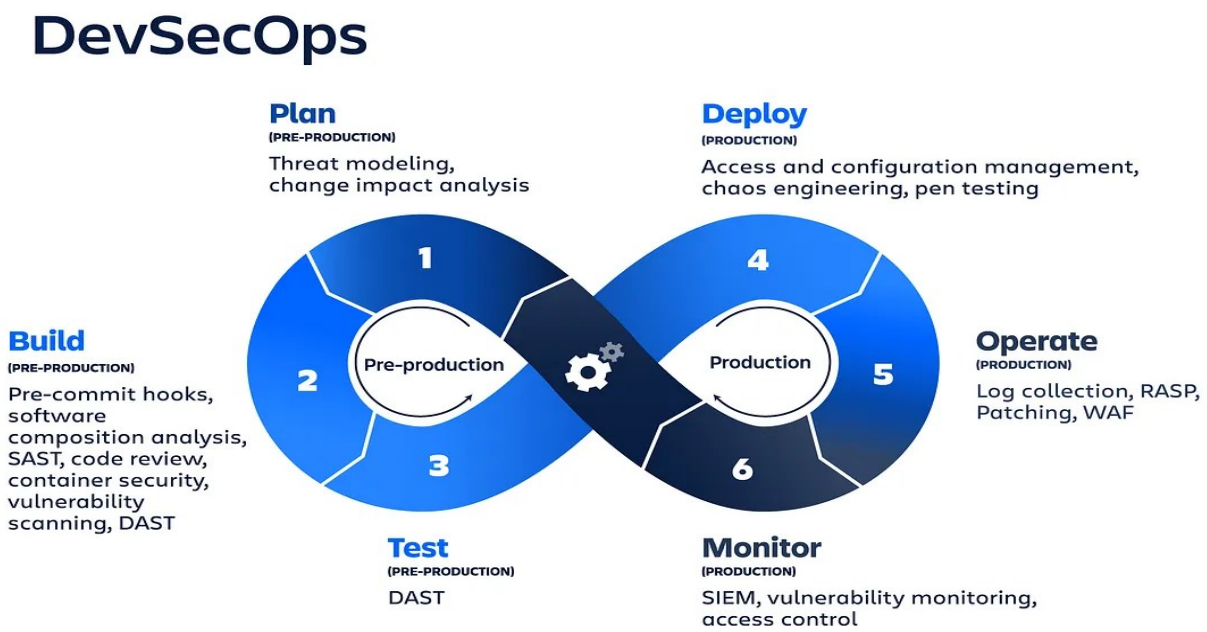
DevSecOps stands for development, security, and operations. It is an extension of the DevOps practice. Each term defines different roles and responsibilities of software teams when they are building software applications.

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.

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

3. Briefly explain DevSecOps lifecycle?



1. **Plan**

   In the planning phase, the development team collaborates with security and operations teams to define the security requirements and goals for the software project. This involves identifying potential security risks, establishing security controls, and defining compliance requirements.

2. **Develop**

   During the development phase, developers write code while incorporating security best practices. Secure coding guidelines and standards are followed to minimize vulnerabilities. Static code analysis tools can be used to identify potential security issues early on.

3. **Build**

   In the build phase, the code is compiled, packaged, and built into executable software. Continuous integration tools automate the build process and run automated security tests, such as static application security testing (SAST), to identify security weaknesses in the code.

4. **Test**

   the testing phase focuses on comprehensive security testing. It includes various types of security assessments, such as dynamic application security testing (DAST), penetration testing, vulnerability scanning, and security-focused unit testing. These tests help identify and address security vulnerabilities and weaknesses.

5. **Deploy**

   In the deployment phase, the software is deployed to the target environment. Automation tools like continuous deployment (CD) pipelines and infrastructure-as-code (IaC) facilitate consistent and secure deployments. Security checks, such as environment

hardening and secure configuration, are performed during this phase.

6. **Operate**

   the operations phase involves monitoring the deployed software for security incidents, performance issues, and compliance. Security monitoring tools and techniques, like log analysis, intrusion detection systems, and security information and event management (SIEM) systems, are employed to detect and respond to security threats.

7. **Maintain**

   the maintenance phase focuses on ongoing security and maintenance tasks. It includes patch management, vulnerability management, and regular security updates. Secure coding practices should continue to be followed for any new features or changes introduced.

8. **Respond**

   the response phase involves promptly addressing security incidents and vulnerabilities that are discovered in production. Incident response plans and procedures are in place to handle security breaches effectively. Lessons learned from security incidents are used to improve security practices for future development cycles.

These phases are iterative and continuous, with feedback loops and automation integrated throughout the lifecycle. By following this DevSecOps lifecycle, organizations can ensure that security is integrated at every stage of the software development process, leading to more secure and reliable software deployments.

Real-World scenario: DevSecOps lifecycle in action

Imagine a scenario where a tech-savvy software development team embraces the DevSecOps lifecycle for a new application launch. By weaving security controls into the planning phase, rigorously testing for vulnerabilities during development, and orchestrating robust monitoring post-deployment, the team successfully fortifies the application against potential threats, ensuring a robust security posture throughout the lifecycle.

Conclusion: Embracing the essence of DevSecOps lifecycle

In embracing the DevSecOps lifecycle, organizations open the gateway to enhanced security resilience and optimized software development practices. By championing collaboration, automation, and persistent improvement imbibed in the DevSecOps lifecycle, organizations cultivate a security-first mindset that shields their digital assets against evolving threats. Witness the transformative power of DevSecOps as it reshapes security paradigms and propels organizations toward a secure digital future.

4. How dose DevSecOps works?

The approach to DevSecOps is designed to equip development teams with a complete security framework. This is achieved through continuous collaboration among development, release management (operations), and the security team, emphasizing teamwork throughout each CI/CD Pipeline stage.

The CI/CD Pipeline comprises six phases: Code, Build, Store, Prep, Deploy, and Run.

Each phase is outlined below to demonstrate the benefits of incorporating security early in the process:

1. Code

The first step in a DevSecOps-aligned development approach is to code in secure and trustworthy segments. Tools are provided that regularly update these fast building blocks, enhancing the protection of data and applications from the beginning.

2. Build

Transforming code into comprehensive container images, which include a core OS, application dependencies, and runtime services, requires a secure process. This process is managed securely, with runtime dependency scans to improve security, allowing DevSecOps teams to develop with both security and agility.

3. Store

Every pre-packaged technology stack is a potential risk in the current cybersecurity context. Developers can securely obtain specific dependencies and conduct vulnerability scans on container images to mitigate these risks.

4. Prep

Before deployment, it's essential to ensure applications comply with security policies. This involves validating configurations against the organization's security policies before moving to the following stages of the development cycle. These configurations, which determine how the workload should run, not only identify potential vulnerabilities but also set the stage for successful deployment in subsequent CI/CD pipeline phases.

5. Deploy

Scans performed in earlier stages give a comprehensive view of the application's security status. At this stage, any identified vulnerabilities or misconfigurations in the development process are presented, allowing organizations to address issues and establish more robust security standards, thus enhancing their security posture.

6. Run

As deployments are executed, teams can utilize active deployment analytics, monitoring, and automation to ensure continuous compliance and address vulnerabilities that arise after deployment.

5 .Explain well known DevSecOps tools.

What is DevSecOps Tools?

DevSecOps tools are a set of software and applications that facilitate the integration of security practices into the software development and operations lifecycle. These tools play a pivotal role in ensuring that security measures are seamlessly woven into every step of the development process – from code creation to deployment and beyond.

1. Software Composition Analysis (SCA)

Software composition analysis (SCA) tools scan applications to detect and address issues (security vulnerabilities, problematic OSS licenses, and quality issues) in open source code. SCA solutions also offer reporting functionality, including the ability to generate a software bill of materials.

If and when SCA does identify a vulnerability, it provides a host of information (including severity score, inclusion path, and remediation guidance) to help users properly address the issue.

For the open source license compliance use case, SCA inventories the different licenses involved in your code, flagging any components with licenses that violate an organization's compliance policies.

Finally, modern SCA tools also help teams implement the key DevSecOps principle of delivering quality software. SCA offers code quality and provenance checks, helping users identify and upgrade outdated and/or poorly maintained software components.

2. Static Application Security Testing (SAST)

SAST refers to a set of tools that scan codes (source code, binary code, byte code) in a non-running (read: static) state. SAST flags weaknesses in the code it scans, effectively surfacing common issues like CWE-79 (cross-site scripting), buffer overflow errors, SQL Injection, and more.

Much like SCA, SAST flags vulnerabilities and offers remediation guidance. Both tools analyze source code/binaries as opposed to running applications. And, both SCA and SAST are frequently used during the "build" stage of the software development lifecycle, in line with the "shift-left" principle of conducting security testing as early as possible in the SDLC.

3. Dynamic Application Security Testing (DAST)

In contrast to SAST and SCA, DAST (Dynamic Application Security Testing) tests for vulnerabilities in a running application. As such, it's used later in the software development lifecycle.

DAST does not require access to source code. Instead, DAST tools detect vulnerabilities in a running application by (safely) injecting malicious inputs to identify potential security vulnerabilities within the application. A DAST tool will make HTTP requests and uncover issues like SQL injections, OS injections, and cross-site scripting errors. It also finds bugs that are important to application security contexts, like security headers, cookie safety, content security policies, and X-Frame-Options.

4. Automated Testing Tools

The days of large, dedicated QA teams are a thing of the past for organizations with successful DevSecOps implementations. As the U.S. government's DevSecOps Fundamentals Guidebook puts it: "Testing is about automation, and testers will need to become coders of that automation."

Although some manual testing work will still be required — it's not possible to automate every part of every test — the majority can be automated. For example:

Unit tests: Unit tests analyze individual units of code to make sure they perform as expected. Unit testing tools tend to be language-specific.

Integration tests: Integration tests are performed after unit tests and deal with the interaction between units of code. Again, many of these tests are language-specific.

System tests: System tests are performed after integration tests and analyze the entire application. System testing tools analyze areas like usability, reliability, scalability, and more.

Performance testing, regression testing, and acceptance testing are also among the areas that can be automated.

Veracode is an amazing cloud-based security tool created to simplify developer security testing. It provides comprehensive visibility into your application's security posture and offers remediation tips for any vulnerabilities it detects.

.

Check Marx provides AI-powered software security solutions that help identify and remediate code vulnerabilities. It integrates easily into your development pipeline and provides actionable insights into your security posture.

OWASP ZAP is a free and open-source web application security scanner. It is highly customizable and can identify vulnerabilities in your application and works by intercepting and modifying HTTP and HTTPS traffic between the web application and client. ZAP has the capability to scan for a range of security issues and includes automated and manual scanning modes.

Burp Suite is a leading platform for web application security testing. It offers a variety of tools to help you identify and remediate vulnerabilities and integrates seamlessly into your DevSecOps pipeline.

SonarQube is a popular code quality tool that offers security-focused plugins to help identify code vulnerabilities during development, provides continuous feedback on your code, and enables you to maintain high code quality.

Fortify is an industry-leading application security tool that offers comprehensive testing capabilities, including static, dynamic, and interactive application security testing. It also offers integrations with leading tools for seamless DevSecOps.

Snyk is a popular developer-first application security tool that integrates directly into your development tools and workflows. It supports multiple languages and offers actionable insight into your app's security posture.

Coverity is a static analysis tool that detects and helps you remediate critical software defects that could impact the security of your application. It also offers integrations with all the leading DevSecOps tools, making it a popular choice for large organizations.

AppScan is a popular application security tool produced by HCL Technologies, a leader in the cybersecurity field. Its AI-powered solution is easy-to-use and supports both static and dynamic applications.

6. What are the benefits of DevSecOps?

The two main benefits of DevSecOps are speed and security. Therefore, development teams deliver better, more-secure code faster and cheaper.

The purpose and intent of DevSecOps is to build on the mindset that everyone is responsible for security with the goal of safely distributing security decisions at speed and scale to those who hold the highest level of context without sacrificing the safety required

1. <u>Rapid, cost-effective software delivery</u>

When software is developed in a non-DevSecOps environment, security problems can lead to huge time delays. Fixing the code and

security issues can be time-consuming and expensive. The rapid, secure delivery of DevSecOps saves time and reduces costs by minimizing the need to repeat a process to address security issues after the fact.

This process becomes more efficient and cost-effective since integrated security cuts out duplicative reviews and unnecessary rebuilds, resulting in more secure code.

2 .Improved, proactive security

DevSecOps introduces cybersecurity processes from the beginning of the development cycle. Throughout the development cycle, the code is reviewed, audited, scanned and tested for security issues. These issues are addressed as soon as they are identified. Security problems are fixed before additional dependencies are introduced. Security issues become less expensive to fix when protective technology is identified and implemented early in the cycle.

Additionally, better collaboration between development, security and operations teams improves an organization's response to incidences and problems when they occur. DevSecOps practices reduce the time to patch vulnerabilities and free up security teams to focus on higher value work. These practices also ensure and simplify compliance, saving application development projects from having to be retrofitted for security.

3. Accelerated security vulnerability patching

A key benefit of DevSecOps is how quickly it manages newly identified security vulnerabilities. As DevSecOps integrates vulnerability scanning and patching into the release cycle, the ability to identify and patch common vulnerabilities and exposures (CVE) is diminished. This capability limits the window that a threat

actor has to take advantage of vulnerabilities in public-facing production systems.

4. Automation compatible with modern development

Cybersecurity testing can be integrated into an automated test suite for operations teams if an organization uses a continuous integration/continuous delivery pipeline to ship their software.

Automation of security checks depends strongly on the project and organizational goals. Automated testing can ensure that incorporated software dependencies are at appropriate patch levels, and confirm that software passes security unit testing. Plus, it can test and secure code with static and dynamic analysis before the final update is promoted to production.

5. A repeatable and adaptive process

As organizations mature, their security postures mature. DevSecOps lends itself to repeatable and adaptive processes. DevSecOps ensures that security is applied consistently across the environment, as the environment changes and adapts to new requirements. A mature implementation of DevSecOps will have a solid automation, configuration management, orchestration, containers, immutable infrastructure and even server less compute environments

6. Quick Resolution of Security Flaws

A key benefit of DevSecOps is its quick response to security weaknesses. Dealing with common vulnerabilities during the development phase reduces the risks linked to flaws in development frameworks.

7. Automated Security Monitoring and Testing

DevSecOps enhances security monitoring and testing through automation. This method uses automated testing to check and compare actual results with expected ones, either through

automated test scripts or testing tools. It also ensures thorough code testing and validation with static and dynamic assessments before integration into the development cycle.

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

DevSecOps is a rapidly growing field within the technology industry, and professionals with expertise in this area are in high demand both locally and internationally.

Local DevSecOps Career Opportunities:

1. Application Security Specialist: Application security specialists focus on securing applications by implementing secure coding practices, conducting security assessments, and addressing vulnerabilities.

2. Security Analyst: Security analysts monitor and analyze security threats, conduct risk assessments, and provide recommendations for improving security practices within an organization.

3. DevSecOps Engineer: DevSecOps engineers specialize in integrating security practices into the DevOps pipeline, automating security testing, and ensuring compliance with security standards.

4. Security Engineer: Security engineers focus on implementing security measures in software development processes, including code reviews, vulnerability assessments, and security testing.

International DevSecOps Career Opportunities:

1. Security Operations Center (SOC) Analyst: SOC analysts monitor and investigate security incidents, analyze security logs, and respond to cyber security threats in real-time.

2. Chief Information Security Officer (CISO): CISOs are responsible for overseeing an organization's information security program, managing security initiatives, and ensuring compliance with security regulations.

3. Cybersecurity Consultant: Cybersecurity consultants offer expertise in evaluating and enhancing an organization's cybersecurity posture, conducting security assessments, and developing security strategies.

4. Security Architect: Security architects design and implement secure systems and applications, develop security policies and procedures, and provide guidance on security best practices.

DevSecOps Career Path:

- API Security Engineer
- Cloud DevSecOps Engineer
- Development Security Operations Engineer
- DevSecOps Analyst
- DevSecOps And Site Reliability Engineer
- DevSecOps Architect
- DevSecOps Automation Engineer
- DevSecOps CI/CD Engineer
- DevSecOps Container Engineer
- DevSecOps Engineer
- DevSecOps Platform Engineer
- DevSecOps Site Reliability Engineer
- DevSecOps Testing Engineer
- IT Security DevSecOps Inter

Conclusion

A new approach called DevSecOps integrates security into the early phases of software development. It ensures complete operation, lessens cyber dangers, and quick software product launches. Software solutions can be produced fast by implementing security at every level of the SDLC. Those who work in the automobile, healthcare, financial, or retail sectors can use these security solutions.

It is a management strategy incorporating a continuous delivery cycle with security, operations, application development, and IaaS. DevSecOps Services aim to integrate security into all phases of the SDLC. Continuous integration, cost-effective compliance, and speedy software delivery are all made possible using security at every level of the SDLC. Making everyone responsible for security is its fundamental goal.

REFERENCE

1.https://cybersn.com/role/devsecops/#:~:text=Role%20overview,stage%20of%20the%20SDLC%20lifecycle.

2. https://career.luxoft.com/job/devsecops/371578/

3.https://www.ibm.com/topics/devsecops#:~:text=The%20two%20main%20benefits%20of,secure%20code%20faster%20and%20cheaper.

4. https://fossa.com/blog/must-have-devsecops-tools/

5. https://aws.amazon.com/what-is/devsecops/#:~:text=DevSecOps%20introduces%20security%20to%20the,before%20they%20write%20any%20code.

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

7. https://www.synopsys.com/glossary/what-is-devsecops.html#D