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. NATNAEL ARGAW ----------147000

SUBMITTED DATE: JUNE/1/ 2024

SUBMMITED TO: Ismael M.

Introduction

DevSecOps is a methodology that integrates security practices into the DevOps process, aiming to build security into the software development lifecycle from the beginning. It combines Development (Dev), Operations (Ops), and Security (Sec) to create a culture of shared responsibility for security across the entire software development process.

The main goal of DevSecOps is to shift security left in the development process, meaning that security considerations are addressed early and continuously throughout the software development lifecycle. By incorporating security practices into every stage of development, from planning and coding to testing and deployment, DevSecOps helps organizations build more secure and resilient software applications.

Key principles of DevSecOps include automation of security testing, continuous monitoring and feedback, collaboration between development, operations, and security teams, and the use of security as code practices. By adopting DevSecOps, organizations can improve the security posture of their applications, reduce vulnerabilities, and respond more effectively to security threats in today's fast-paced and dynamic software development environments

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. Lack of security awareness: Traditionally, security has been an afterthought in the software development process, leading to vulnerabilities being introduced during development.

2. Siloed teams: In many organizations, security teams operate separately from development and operations teams, leading to a lack of communication and collaboration between these groups.

3. Slow security testing: Traditional security testing processes are often slow and manual, leading to delays in the release of secure software.

4. Lack of automation: Manual security processes are error-prone and time-consuming, making it difficult to maintain security across a large number of applications.

5. Compliance challenges: Meeting regulatory requirements and industry standards for security can be challenging without a coordinated approach to security across the development lifecycle.

6. Lack of Quality: Security is integral to quality. In our observation, lack of quality is often associated with the security team getting involved too late, a lack of confidence in the release, and system complexity.

7. Lack of Security Skills: Developers, architects, scrum masters, and other key players in

an organization should have the right vocabularies and skills. By vocabularies, we mean some common knowledge or skillset, or a common understanding, such as a knowledge of how to write secure code.

8. Compliance and regulatory requirements : often add an extra layer of complexity to the software development process, making it difficult to ensure security and compliance at the same time.

9. Complexity: The complexity of modern software systems and the increasing use of third-party components and libraries create new attack surfaces for potential security breaches.

2. What is DevSecOps?

DevSecOps is a software development approach that combines Development (Dev), Security (Sec), and Operations (Ops) to integrate security practices throughout the software development lifecycle (SDLC). It builds upon the principles of DevOps, emphasizing collaboration, automation, and continuous improvement, while also prioritizing security considerations from the earliest stages of development.
Traditionally, security has been treated as a separate entity, often added as an afterthought in the software development process. This approach led to vulnerabilities being identified late in the development cycle, resulting in costly and time-consuming fixes. DevSecOps seeks to address this issue by promoting a shift-left mentality, where security is integrated from the beginning and considered at every stage of development.

Key Principles of DevSecOps:

A, **Collaboration**:

DevSecOps encourages collaboration among development, security, and operations teams. By breaking down silos and fostering communication, teams can work together to identify and address security concerns early in the development process

Security professionals work closely with developers and operations teams to integrate security controls, tools, and best practices seamlessly into the development pipeline.

B, **Automation**:

Automation is a core aspect of DevSecOps, enabling teams to implement security checks and measures throughout the SDLC.

Continuous Integration/Continuous Deployment (CI/CD) pipelines are automated to include security testing, such as static code analysis, vulnerability scanning, and compliance checks. Automated tests ensure that security is validated at each stage of the development process, from code commit to deployment.

C, **Shift-Left Approach:**

DevSecOps advocates for a shift-left approach, where security is addressed early in the development cycle.

By integrating security practices into the development process from the outset, teams can identify and remediate vulnerabilities before they become more complex and costly to fix.

D, **Continuous Monitoring:**

DevSecOps emphasizes the importance of continuous monitoring and feedback loops to detect and respond to security threats promptly.

Real-time monitoring of applications and infrastructure allows teams to identify and mitigate security incidents as they occur, reducing the impact of potential breaches.

E, **Culture of Security:**

DevSecOps promotes a culture of security awareness and responsibility across the organization.

Teams are encouraged to prioritize security as a shared responsibility, with everyone understanding their role in maintaining the integrity and resilience of the software.

3. Briefly explain DevSecOps lifecycle?

DevSecOps is a software development methodology that emphasizes security and collaboration between development, security, and operations teams throughout the software development lifecycle. DevSecOps works best with teams that use CI/CD, or continuous integration and delivery process, meaning code changes are integrated and released as part of an automated process.

The DevSecOps lifecycle can be broken down into the following steps, with the development, testing, and deployment stages often happening in a loop as software updates are made and new features are added:

1. Plan
In the planning phase, development teams work with security and operations teams to identify potential security risks and develop a security strategy. This includes identifying security requirements, defining security policies, and selecting the appropriate security testing tools.

2. Develop
During the development phase, development teams both build and test the application. This includes integrating automated security testing into the development process, conducting code reviews, and ensuring that security requirements are met.

Since development and testing happen together in the DevSecOps lifecycle, less secure components, such as third-party code, can be tested as they are put into place.

This is where the continuous integration part of the CI/CD process comes in. Code changes are automatically integrated into a shared repository on a regular basis, allowing developers to identify and address conflicts and issues early in the development process.

Optional: Test
Since testing happens during development, a separate testing phase is not necessary in a DevSecOps approach. When it is included, testing takes much less time than it does in a traditional testing process.

During the testing phase, security teams test the application for security weaknesses, vulnerabilities, and threats using penetration testing, vulnerability scanning, and other security testing techniques.

3. Deploy and Monitor
In a traditional process, the operation team would have deployed the application to production. However, the DevSecOps lifecycle follows the DevOps approach, which shifted the responsibility of deploying the application from operations teams to development teams.

The process of deploying to production includes configuring and securing the infrastructure, implementing access controls, and monitoring the environment for security threats.

Today, many development teams trigger deployments using continuous delivery. This involves the use of tools and processes to automatically build, test, and deploy code changes to production environments.

After deployment, teams then monitor the application for security threats and respond to any incidents that occur.

Benefits of Following The DevSecOps Lifecycle
Following a DevSecOps approach has many benefits. By integrating security directly into the software development lifecycle:

- Early detection of security vulnerabilities: Integrating security from the beginning of the SDLC helps detect security vulnerabilities at an early stage.
- Reduced time and cost: Integrating security into the SDLC reduces the costs associated with fixing security vulnerabilities at a later stage.

- Improved software quality: Integrating security into the SDLC improves the overall quality of the software. By identifying and addressing security issues early on, developers can ensure that the software is more reliable and less prone to errors.

- Compliance with regulations: Many industries have regulations and standards that require software to meet specific security requirements. Integrating security into the SDLC ensures that the software meets these requirements, reducing the risk of non-compliance.

- Increased customer trust: By helping teams find - and fix - application vulnerabilities before release - DevSecOps helps organizations deliver more secure, reliable software to customers to build trust

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. Rapid, cost-effective software delivery

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

There are so many well-known DevSecOps tools that organizations can use to enhance their security practices throughout the software development and operations.These tools help organizations automate security processes, detect vulnerabilities, manage security configurations, and ensure compliance with security standards throughout the software development lifecycle. By integrating these tools into their DevSecOps practices, organizations can strengthen their security posture and build more secure and resilient software systems. Also DevSecOps offers diverse career opportunities locally and internationally.

Reference

- ***https://aws.amazon.com***
- ***https://www.synopsys.com/glossary***
- ***https://www.ibm.com***