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. AWOL TILAHUN --	145506

SUBMITTED DATE:
JUNE/1/ 2024
SUBMMITED
TO: Ismael M.

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

There are several software engineering problems that can lead to the initiation of DevSecOps practices, including:

1. **Lack of security awareness:** Developers may not have a strong understanding of security best practices, leading to vulnerabilities in the code.
2. **Siloed development and operations teams:** When development and operations teams work in isolation, security concerns may not be adequately addressed, leading to vulnerabilities in the deployment process.
3. **Slow response to security incidents:** Traditional software development processes may not have mechanisms in place to quickly respond to security incidents, leaving systems vulnerable for longer periods of time.
4. **Inadequate testing:** Testing processes may not adequately cover security vulnerabilities, leading to potential issues being missed until they are exploited in production.

5. lack of automation: Manual processes for security checks and deployments can lead to errors and delays, making it difficult to keep up with the fast-paced nature of modern software development.

By implementing DevSecOps practices, organizations can address these software engineering problems by integrating security into every stage of the development process, automating security checks, and fostering collaboration between development, operations, and security teams.

2. What is DevSecOps?

DevSecOps, which is short for *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.

DevSecOps represents a natural and necessary evolution in the way development organizations approach security. In the past, security was 'tacked on' to software at the end of the development cycle, almost as an afterthought. A separate security team applied these security measures and then a separate quality assurance (QA)
team tested these measures.

This ability to handle security issues was manageable when software updates were released just once or twice a year. But as software developers adopted Agile
and DevOps practices, aiming to reduce software development cycles to weeks or even days, the traditional 'tacked-on' approach to security created an unacceptable bottleneck.

DevSecOps integrates application and infrastructure security seamlessly into Agile and DevOps processes and tools. It addresses security issues as they emerge, when they're easier, faster, and less expensive to fix, and before deployment into production.

Additionally, DevSecOps makes application and infrastructure security a shared responsibility of development, security and IT operations teams, rather than the sole responsibility of a security silo. It enables “software, safer, sooner”—the DevSecOps motto–by automating the delivery of secure software without slowing the software development cycle.

3. Briefly explain DevSecOps lifecycle?

1. **planning stage**:Identify the threat model and its risks, security requirements and tools deployment strategies.
   a. ***Security requirements:*** technical specifications needed to implement a project/application:
   b. ***Threat model:*** aims to understand the possible threats to the organization/project, the goal being to create a model to identify and mitigate vulnerabilities before the cybercriminal exploits them.

2. **coding stage**: require secure and reliable development
   a. ***Code analysis:*** fixing bugs, known vulnerabilities, licensing management .
   b. ***Static Application Security Testing (SAST):*** static technique used before deployment to statically analyze the code, looking for standardization, quality and security (known vulnerabilities):
   c. ***Dynamic Application Security Testing (DAST):*** a dynamic technique used after deploy stage, it analyzes the application’s behavior (response) based on known real attacks. It is considered a black-box test because it simulates an external attack without knowing the architecture and source code:
   d. ***Interactive Application Security Testing (IAST):*** an interactive technique that analyzes and identifies vulnerabilities in real-time (runtime). Unlike SAST and DAST, which are performed manually or scheduled before and after deployment, IAST performs automatically through agents and sensors in real-time. It fills a gap that both [SAST and DAST] are unable to fill:
   e. ***Runtime Application Self-Protection (RASP):*** application self-protection technique.

3. **Build stage**: where the code is compiled, integrated and assembled into binaries and other artifacts. The main objective at this stage is to converge the language into binary, as well as resolving dependencies. When it comes to secure development, it becomes a stage where tools must be deployed to check for vulnerabilities and countermeasures:
   a. Static Application Security Testing (SAST)
   b. SCA
   c. Detection of sensitive data in code (e.g. passwords)
   d. Dynamic Application Security Testing (analysis of vulnerabilities in APIs and container image)

4. **Testing stage:** various tests are carried out in order to guarantee the performance, quality, functionality, credibility and security of the application — before deploying it into production:
   a. ***Unit testing:*** isolated testing of each component, checking its behavior and results (when necessary);
   b. ***Integration testing:*** testing in conjunction with other integrations to check their behavior and results;
   c. ***Functional testing:***validates the application.
   d. ***Regression testing:*** ensures that new changes and functionalities do not break the application (like Content Security Policy [CSP]);

e. Performance testing: analyzes the application's responsiveness and scalability, including stress testing,load balancing, network bottleneck,etc;
f. Security testing: identifies vulnerabilities and potential security risks;
g. User Acceptance Testing (UAT): end users assess whether the application meets the security standards and requirements agreed at the start of the project. This part is extremely important for validating the security features in the pipeline:

5. Release stage: the aim is to automate the release process for deployment, as well as to optimize and guarantee efficiency for the next stage, which is delivery:

a. Security automation: control and verification of configurations, patches, updates (latest version available) in all environments;
b. Versioning control (Azure DevOps):
c. CI: Continuous Integration provides functional consistency of code across multiple contributors for automated delivery to the next layer;
d. Infrastructure as Code (IaC): manages and provisions cloud resources (infrastructure):

6. Deliver stage: the crucial process of copying [signing] the code and infrastructure configurations and securely transferring to the environment.

a. ***Secure integration:*** if there are several cloud providers (CSP) involved in the deployment, sensitive information and passwords must be isolated and encrypted;
b. ***Monitoring:*** any failure in delivery must be notified to the system administrator.

7. Deployment stage: which the application becomes available to end users:

a. ***Validation:*** check that the application complies with the requirements and conformities;
b. ***Rollback:*** well-defined procedure for returning to the previous version — if necessary;
c. ***Continuous monitoring:*** record the entire deployment process (logs);
d. ***Dynamic Application Security Testing (DAST).***

8. Operate and Monitor stages: manages application environments, the aim is to guarantee availability, integrity:

a. ***SIEM:*** correlate and filter logs in different environments on a single platform;
b. ***SOAR:*** orchestrate and respond automatically to threats and failures;
c. ***Patch management:*** manage updates;
d. ***Scalability and Elasticity:*** provide dynamically resources (like vCPU, RAM, NVMe, LB, VMs, etc.);
e. ***Resilience:*** define the acceptable level of impact and implement mechanisms to help at this stage (e.g. failover, disaster recovery, etc.);
f. ***SOC:*** a specialized team monitoring the application (in real-time);
g. ***Application Performance Monitoring (APM):*** important for the user experience, it collects response metrics, resources used, packets trafficked, storage volume, overload notification, etc.

9. Feedback stage: proposes continuous communication with everyone involved in the project:

a. KPIs;
b. Agile;
c. Review process.

4. How dose DevSecOps works?

DevSecOps works by integrating security practices into the DevOps workflow, ensuring that security is prioritized throughout the software development lifecycle. Here are some key aspects of how DevSecOps works:

1.shift kift: DevSecOps emphasizes shifting security practices to the left in the development process, meaning that security considerations are introduced as early as possible in the software development lifecycle. By addressing security from the planning and design phases, teams can proactively identify and mitigate security risks before they become more costly to fix later in the process.

2.Automation: Automation is a fundamental aspect of DevSecOps. Security testing tools, such as static code analysis, vulnerability scanning, and compliance checks, are integrated into the CI/CD pipeline to automate security testing and validation. This automation enables teams to detect security issues quickly and consistently throughout the development process.

3.collaboration: DevSecOps encourages collaboration between development, operations, and security teams. By breaking down silos and fostering communication between these teams, organizations can ensure that security requirements are understood and implemented effectively across all stages of the software development lifecycle.

4. Continuous Monitoring: DevSecOps promotes continuous monitoring of applications and infrastructure to detect security incidents in real-time. Security monitoring tools are used to track system behavior, analyze logs and metrics, and respond to security events promptly. Continuous monitoring helps organizations identify and remediate security issues before they escalate into larger problems.

5. Security as Code: DevSecOps advocates for treating security as code, where security policies, configurations, and controls are defined and managed using code- based approaches. Infrastructure as code (IaC) tools, configuration management tools, and security automation scripts enable teams to define security controls programmatically and apply them consistently across environments.

6. Feedback Loop: DevSecOps emphasizes the importance of feedback loops to continuously improve security practices. By collecting feedback from security testing, incidents, and monitoring activities, organizations can identify areas for improvement, address vulnerabilities, and enhance their overall security posture over time.

Overall, DevSecOps works by embedding security into the DevOps workflow, promoting collaboration, automation, continuous monitoring, and feedback to build secure and resilient software applications. By adopting DevSecOps practices, organizations can strengthen their security posture and deliver secure software faster and more reliably.

7. Feedback and Continuous Improvement: Feedback from security testing and incidents is used to improve security practices and processes continuously. Lessons learned are incorporated into future development cycles to enhance the overall security posture of the application.

By following the DevSecOps lifecycle, organizations can build secure applications, detect security issues early, respond to incidents effectively, and continuously improve their security practices.

5. Explain well known DevSecOps tools.

There are several well-known DevSecOps tools that organizations can use to integrate security practices into their DevOps workflows. Here are some popular DevSecOps tools:

1. Static Application Security Testing (SAST) Tools:
- **Checkmarx:** Checkmarx is a leading SAST tool that helps developers identify and fix security vulnerabilities in their code.
- **Veracode:** Veracode offers a cloud-based SAST solution for secure software development.

2. Dynamic Application Security Testing (DAST) Tools:
- **OWASP ZAP (Zed Attack Proxy):** ZAP is an open-source DAST tool that helps developers find security vulnerabilities in web applications.
- **Burp Suite:** Burp Suite is a popular DAST tool for web application security testing.

3. Container Security Tools:
- ***Docker Bench for Security:*** Docker Bench for Security is a tool that checks for common best practices in Docker deployments.
- ***Clair:*** Clair is an open-source container vulnerability scanning tool that helps identify security issues in container images.

4. Infrastructure as Code (IaC) Security Tools:
- ***Terraform:*** Terraform is a popular IaC tool that allows infrastructure to be defined as code. Security best practices can be integrated into Terraform configurations.
- ***AWS Config:*** AWS Config provides continuous monitoring and assessment of AWS resource configurations for security compliance.

5. Security Orchestration, Automation, and Response (SOAR) Platforms:
- ***Demisto:*** Demisto is a SOAR platform that helps automate incident response processes and integrates with various security tools for orchestration.

6. Security Information and Event Management (SIEM) Tools:
- ***Splunk:*** Splunk is a widely-used SIEM tool that collects and analyzes security event data to provide insights into security incidents.

7. Open Source Security Tools:
- ***OpenSCAP:*** OpenSCAP is an open-source Security Content Automation Protocol (SCAP) toolkit for compliance checking and vulnerability scanning.
- ***OSSEC:*** OSSEC is an open-source host-based intrusion detection system that provides log analysis, file integrity checking, and rootkit detection.

These are just a few examples of the many DevSecOps tools available to help organizations enhance their security practices within the DevOps workflow. It's important to evaluate the specific needs and requirements of your organization to choose the most suitable tools for your DevSecOps initiatives.

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,” describes Shannon Lietz, co-author of the “DevSecOps Manifesto.”

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.

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.

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.

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.

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

DevSecOps is a rapidly growing field within the broader realm of cybersecurity and software development. As organizations increasingly prioritize security in their software development processes, the demand for professionals with DevSecOps skills is on the rise both locally and internationally. Here are some insights into local and international DevSecOps career opportunities and career paths:

Local DevSecOps Career Opportunities:

1. ***Security Engineer/Analyst:*** Entry-level positions in security engineering or analysis often provide a good starting point for individuals looking to specialize in DevSecOps. These roles involve implementing security measures and monitoring systems for vulnerabilities.

2. ***DevSecOps Engineer:*** DevSecOps engineers are responsible for integrating security practices into the software development lifecycle. They work closely with development and operations teams to ensure that security is embedded throughout the process.

3. ***Security Consultant:*** Security consultants may work for consulting firms or as independent contractors, helping organizations assess their security posture and implement DevSecOps best practices.

4. ***Security Architect:*** Security architects design and implement security solutions for organizations, including developing secure architectures and guiding the implementation of security controls.

International DevSecOps Career Opportunities:

1. ***DevSecOps Manager/Director:*** In larger organizations or multinational companies, there may be opportunities for experienced DevSecOps professionals to take on managerial or leadership roles overseeing security initiatives across different teams or regions.

2. ***Cloud Security Specialist:*** With the increasing adoption of cloud technologies, there is a growing demand for professionals who specialize in securing cloud environments and integrating security into cloud-native applications.

3.

3. ***Compliance Specialist/Auditor:*** Professionals with expertise in regulatory compliance, such as GDPR, HIPAA, or PCI DSS, can find opportunities to work internationally helping organizations navigate complex compliance requirements while implementing DevSecOps practices.

DevSecOps Career Path:

1. ***Entry-Level:*** Start with foundational knowledge in cybersecurity, software development, and operations. Gain experience in security practices and tools.

2. ***Mid-Level:*** Specialize in DevSecOps practices, automation, and integration of security tools into the development pipeline. Obtain relevant certifications like Certified DevSecOps Professional (CDP) or Certified Information Systems Security Professional (CISSP).

3. ***Senior-Level:*** Lead DevSecOps initiatives, mentor junior team members, and contribute to strategic security planning. Consider pursuing advanced certifications or degrees in cybersecurity or related fields.

4. ***Management/Leadership:*** Transition into managerial or leadership roles overseeing DevSecOps teams or security programs within organizations.

Overall, DevSecOps offers a promising career path for individuals passionate about security and software development. By continuously learning and adapting to new technologies and best practices, professionals can advance their careers both locally and internationally in this dynamic field.

CONCLUSION

In conclusion, DevSecOps represents a critical evolution in the way organizations approach cybersecurity within the software development lifecycle. By integrating security practices early and consistently throughout the development process, DevSecOps aims to proactively address security vulnerabilities and mitigate risks effectively. The demand for DevSecOps professionals is growing rapidly, both locally and internationally, as organizations recognize the importance of building secure and resilient software systems. Pursuing a career in DevSecOps offers exciting opportunities for individuals to contribute to enhancing cybersecurity posture, driving innovation, and shaping the future of secure software development practices. Embracing continuous learning, staying updated on industry trends, and obtaining relevant certifications can help professionals thrive in this dynamic and rewarding field.

Reference

http://www.ibm.com
https://www.devsecops.org/
https://owasp.org/www-project-devsecops-studio/