Institute of Technology
School of Computing

Department of Software Engineering

Software Engineering Tools and Practices

Assignment 1 DevSecOps

NAME == BIRUK GIRMA , ID == 1300655;

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

Software engineering problems which caused the initiation of DevSecOps include:

***1. Lack of security awareness**: Traditional software development processes often prioritize speed and functionality over security. This lack of focus on security can lead to vulnerabilities in the software that can be exploited by attackers.*

***2. Siloed teams**: In many organizations, security teams operate separately from development and operations teams. This siloed approach can lead to miscommunication, delays in addressing security issues, and a lack of shared responsibility for security.*

***3. Slow response to security incidents:** Traditional software development processes often do not include mechanisms for quickly responding to security incidents. This can result in delays in identifying and addressing vulnerabilities, which can increase the risk of a successful attack.*

***4. Compliance challenges:** Many industries are subject to regulatory requirements that mandate certain security practices. Traditional software development* *processes may not easily accommodate these requirements, leading to compliance challenges.*

5. Increasing complexity: *Modern software systems are becoming increasingly complex, with multiple layers of infrastructure, dependencies, and third-party components. Managing the security of these complex systems can be challenging without a unified approach like DevSecOp*

While DevSecOps can benefit your organization in many ways, we've observed several challenges to its adoption, the most common of which are the following:

lack of security assurance at the business and project levels

organizational barriers related to collaboration, tooling, and culture

impact to quality because security is not a priority while systems are getting more complex

lack of security skills for developers, business stakeholders, and auditors

insufficient security guidance due to lack of resources, standards, and data

The rest of this section examines each of these challenges and provides approaches for overcoming them.

Security Assurance

How do we know that the security practices we've adopted for our development lifecycle and built into our software are adequate and appropriate for the business purpose we're trying to address? Addressing this challenge can be hard, especially when your industry, business, and/or project lacks security assurance.

The security requirements for your industry or domain are different from those of other industries or domains. For instance, the health care industry has different security requirements from the financial sector. If your industry lacks assurance models, you might begin by assessing your own organization's security posture and requirements, then engage with similar organizations in your industry or domain to share information. In addition, we recommend the following:

Don't wait for an industry standard to emerge.

Join or create informal working groups with industry peers.

Attend conferences and network with like organizations.

Share your experiences and lessons learned.

Work with others to extend the body of knowledge and establish best practices.

Your Business Lacks Security Assurance

There is often a disconnect between business needs, mission, and vision when it comes to security. Developers need to understand the business context of security. They must think about the organization's security policies, its business drivers, and how these apply to the software being developed. In so doing, they should address security as early as possible in the lifecycle, preferably during the requirements stage. As you do, keep the following recommendations in mind:

Focus on fundamentals: What are the threats? What are the business drivers? Balance the two.

Align with development with business needs (time to market, cost savings, resilience).

Conduct external audits.

Understand the business context.

Identify, link, and rank business and technical risks.

Identify security requirements in early.

Educate top management and get them onboard.

Engage more senior technical people first to work with security teams.

Make security part of senior technical reviews; organically spread the word.

Your Project Lacks Assurance of Security

Once you've identified security assurance needs within your industry or domain (and perhaps your specific business within that domain), you need to map that data to your project. For instance, perhaps your organization is already following guidance, such as General Data Protection Regulation (GDPR) or the Health Insurance Portability and Accountability Act (HIPAA). You need to account for any security activities stipulated in that guidance in your project planning, and you need to do so early in the lifecycle when there's still time to address it. As you do, keep in mind the following recommendations:

Map reporting data from tools to form a continuous view of value.

Run security tools on all code to measure code quality and standards.

Review code changes for security and document approval prior to release.

Use dedicated testing resources in the case of significant changes.

Track all changes and approvals for incident purposes.

Conduct code reviews.

Expose security team to your metrics and data.

Organizational Barriers

If you're not sharing your DevSecOps journey across the 1. What are Software engineering problems which was cause for initiation of DevSecOps., from concept to product, you should expect problems since you won't have a clear understanding of the business needs your software needs to address. You might not even have a clear vision of the customer's needs and the environment in which the customer operates. Communication is key to breaking down organizational *barriers, but often different units inside an organization use different communications tools, structures, and goals.*

To begin to break down these barriers, briefly document your journey from idea to product. Look at points of interaction among the various components of your organization. Educate executives who likely don't know the details of the DevSecOps process. Build connections and a culture that reinforce the sharing of the same goals and vision. Often, poor stakeholder collaboration, difficulty integrating pipeline security, and an unwillingness to make security a priority stand in the way of successful

The product you're creating touches many other stakeholders in your organization, including marketing, IT, and legal teams, but communication among them might be lacking. For example, you may have different tools, may not share the same infrastructures, and may not even share the same vision and goals. To address these issues, you need to come together and document your journey from concept to product. A simple cheat sheet will suffice, one that reminds all stakeholders of the vision, the mission, and the roles they will play in the lifecycle. Our recommendations for improving stakeholder collaboration include the following:

Document your current state and identify silos (e.g., development, infrastructure, and security).

Start building collaboration between the Security, Dev, and Ops teams.

Be prepared: people generally don't want to change their culture and/or workflow.

Make sure everyone gets on the same page regarding the importance of security (from executives to DevSecOps teams).

Instill a continuous security mindset.

Focus on partnership, not unhealthy conflict. Destroy the blame culture.

Get stakeholders to agree on a shared a vision for the project.

Balance workload among teams involved.

Put security people into development teams.

Integrating Pipeline Security

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 *approach that combines application development, security, operations and infrastructure as code (IaC) in an automated continuous integration/continuous delivery (CI/CD) pipeline.*

The main objective of DevSecOps is to automate, monitor and apply security at all phases of the software lifecycle: plan, develop, build, test, release, deliver, deploy, operate and monitor. Applying security at every stage of the software development process supports CI/CD, reduces the cost of compliance and enables faster software delivery.

DevSecOps means that every employee and team is responsible for security from the outset, and they must make decisions efficiently and put them into action without forfeiting security.

DevSecOps is a methodology that integrates security practices into the DevOps process, aiming to shift security left in the software development lifecycle. By incorporating security early in the development process, DevSecOps seeks to build secure software from the start and minimize vulnerabilities that could be exploited by attackers.

3. Briefly explain DevSecOps lifecycle?

Plan

The plan phase is the least automated phase of DevSecOps, involving collaboration, discussion, d perform a security analysis and create= a plan that outlines where, how, and when security testing will be done. A popular planning tool for DevSecOps is IriusRisk, a collaborative design tool for threat modeling. Additional tools include issue tracking and management tools like Jira and communication and chat tools like Slack.

Build

The build phase begins once developers commit code to the source repository. DevSecOps build tools focus on automated security analysis against the build *output artifact. Important security practices include software component analysis, static application software testing (SAST), and unit tests. Tools can be plugged into an existing CI/CD pipeline to automate these tests.*

Developers regularly install and build upon third-party code dependencies, which may be from an unknown or untrusted source. External code dependencies may accidentally or maliciously include vulnerabilities and exploits. During the build phase, it is critical to review and scan these dependencies for any security vulnerabilities.

Some well-known tools to execute build phase analysis include: OWASP Dependency-Check, SonarQube, SourceClear, Retire.js, Checkmarx, and Snyk.

DevSecOps tools for the code phase help developers write more secure code. Important code-phase security practices include static code analysis, code reviews, and pre-commit hooks.

When security tools plug directly into developers' existing Git workflow, every commit and merge automatically triggers a security test or review. These tools support different programming languages and integrated development environments. Some of the more popular security code tools include Gerrit, Phabricator, SpotBugs, PMD, CheckStyle, and Find Security Bugs.

Test

The test phase is triggered after a build artifact is created and successfully deployed to staging or testing environments. A comprehensive test suite takes a considerable amount of time to execute. This phase should fail fast so that the more expensive test tasks are left for the end.

The test phase uses dynamic application security testing (DAST) tools to detect live application flows like user authentication, authorization, SQL injection, and API-related endpoints. The security-focused DAST analyzes an application against a list of known high-severity issues, such as those listed in the OWASP Top 10.

There are numerous open source and paid testing tools available, which offer a variety of functionality and support for language ecosystems, including BDD Automated Security Tests, JBroFuzz, Boofuzz, OWASP ZAP, Arachi, IBM AppScan, GAUNTLT, and SecApp suite.

Deploy

If the previous phases pass successfully, it's time to deploy the build artifact to production. The security areas of concern to address during the deploy phase are those that only happen against the live production system. For example, any differences in configuration between the production environment and the previous staging and development environments should be thoroughly reviewed. Production TLS and DRM certificates should be validated and reviewed for upcoming renewal.

The deploy phase is a good time for runtime verification tools like Osquery, Falco, and Tripwire, which extract information from a running system in order to determine whether it performs as expected. Organizations can also run chaos engineering principles by experimenting on a system to build confidence in the system's capability to withstand turbulent conditions. Real-world events can be simulated, like servers that crash, hard drive failures, or severed network connections. Netflix is widely known for its Chaos Monkey tool, which exercises chaos engineering principles. Netflix also utilizes a Security Monkey tool that looks for violations or vulnerabilities in improperly configured infrastructure security groups and cuts any vulnerable servers.

By the release phase of the DevSecOps cycle, the application code and executable should already be thoroughly tested. The phase focuses on securing the runtime environment infrastructure by examining environment configuration values such as user access control, network firewall access, and secret data management.

The principle of least privilege (PoLP) is a key concern of the release phase. PoLP means that any user, program, or process, has minimum access to perform its function. This involves auditing API keys and access tokens so that the owners *have limited access. Without this audit, an attacker may find a key that has access to unintended areas of the system.*

Configuration management tools are a key ingredient for security in the release phase, since they provide visibility into the static configuration of a dynamic infrastructure. The system configuration can then be audited and reviewed. The configuration becomes immutable, and can only be updated through commits to a configuration management repository. Some popular configuration management tools include Ansible, Puppet, HashiCorp Terraform, Chef, and Docker.

The security community provides guidelines and recommendations on best practices for hardening your infrastructure, such as the Center for Internet Security (CIS) benchmarks and NIST configuration checklists.

Once an application is deployed and stabilized in a live production environment, additional security measures are required. Companies need to monitor and observe the live application for any attacks or leaks with automated security checks and security monitoring loops.

Runtime application self-protection (RASP) automatically identifies and blocks inbound security threats in real-time. RASP acts as a reverse proxy that observes incoming attacks and enables the application to reconfigure automatically without human intervention in response to explicit conditions.

A specialized internal or external team can perform penetration testing to find exploits or vulnerabilities by deliberately compromising a system. Another security technique is to offer a bug bounty program that pays external individuals who report security exploits and vulnerabilities.

Security monitoring uses analytics to instrument and monitor critical security-related metrics. For example, these tools flag requests to sensitive public endpoints, like user account access forms or database endpoints. Some examples of popular runtime defense tools include Imperva RASP, Alert Logic, and Halo.

As more development teams evolve their processes and embrace new tools, they need to be diligent with security. DevSecOps is a cyclical process, and should be continuously iterated and applied to every new code deployment. Exploits and attackers are constantly evolving and it is important that modern software teams evolve as well

4. How dose DevSecOps works?

DevSecOps, a combination of Development, Security, and Operations, is an approach to software development that integrates security practices into the entire software development lifecycle. The goal of DevSecOps is to build security into the development process from the outset rather than treating it as an afterthought. Here's how DevSecOps works:

***1. Collaboration**: DevSecOps emphasizes collaboration between development, security, and operations teams. By breaking down silos and fostering communication and collaboration among these teams, organizations can ensure that security is integrated into every stage of the development process.*

***2. Automation**: Automation is a key aspect of DevSecOps. By automating security processes, such as code analysis, vulnerability scanning, and compliance checks, organizations can identify and address security issues early in the development cycle. Automation helps to streamline security practices and ensures consistent application of security controls.*

***3. Continuous Integration/Continuous Deployment (CI/CD):** DevSecOps leverages CI/CD pipelines to automate the build, test, and deployment processes. By integrating security checks into the CI/CD pipeline, organizations can automatically scan code for vulnerabilities, assess security risks, and enforce security policies before deploying applications to production.*

***4. Security as Code**: DevSecOps promotes the concept of "security as code," where security controls and policies are codified and treated as infrastructure code. By defining security requirements in code, organizations can automate security configurations, monitor security posture, and enforce security policies consistently across environments.*

5. Shift-Left: *DevSecOps advocates for a "shift-left" approach to security, meaning that security considerations are brought earlier in the development process. By addressing security requirements at the beginning of the development lifecycle, organizations can proactively identify and remediate security issues before they escalate into larger problems.*

6. Continuous Monitoring: *DevSecOps emphasizes continuous monitoring of applications and infrastructure to detect and respond to security incidents in real-time. By monitoring systems for anomalies, unauthorized access, and other security events, organizations can quickly identify and mitigate security threats.*

7. Culture of Security: *DevSecOps promotes a culture of security awareness and responsibility among all team members. By educating developers, operations staff, and other stakeholders about security best practices and fostering a shared commitment to security, organizations can create a strong security posture.*

5. Exline well known DevSecOps tools.

1. Aqua Security *:is a security tool designed for cloud-based applications. It offers comprehensive vulnerability scanning and seamless integration with the CI/CD pipeline. The solution is suitable for organizations of all sizes, thanks to the various available versions, including a free version for basic needs.*

2. Checkmarx *:is a comprehensive static application security testing (SAST) solution that analyzes the source code of software applications and identifies security vulnerabilities and weaknesses in real-time. It can scan and analyze various programming languages, including Java, C/C++, C#, .NET, Ruby, Python and many. Checkmarx also offers software composition analysis (SCA) and interactive application security testing (IAST) capabilities. SCA helps identify open source components and dependencies and their associated vulnerabilities, while IAST provides real-time application security testing during runtime*

3. Prisma Cloud :*is a cloud-native security platform designed to protect cloud infrastructure, applications, and data across multi-cloud environments. It offers a range of security features and capabilities, including vulnerability management, compliance management, network security, and cloud workload protection. Prisma Cloud integrates with popular cloud platforms like AWS, Azure, and Google Cloud Platform, as well as DevOps tools like Kubernetes, Jenkins, and Terraform.*

4. Codacy: *is a software tool that automates code reviews and comes with a static code analysis feature. It assists developers in detecting security weaknesses early in the development phase, which can significantly reduce long-term security vulnerabilities*

5. ThreatModeler *:is a threat modeling platform designed to help organizations identify and mitigate security risks in software applications. ThreatModeler integrates with various DevOps tools and workflows, including Jira, Azure DevOps, and GitHub, allowing organizations to incorporate threat modeling into their development process. This helps ensure that security is built into the application from the start and that potential vulnerabilities are identified and addressed early on in the development cycle.*

6. SonarQube *:is a powerful and flexible platform for code quality inspection and code analysis that helps organizations maintain high standards for code quality and security. SonarQube integrates with popular DevOps tools like Jenkins, GitLab, and Azure DevOps, allowing organizations to incorporate code analysis into their continuous integration and delivery workflows.*

7. Acunetix: *is a web vulnerability scanner that helps organizations identify and remediate security vulnerabilities in their web applications. It uses a combination of black-box and white-box testing techniques to provide comprehensive coverage of potential security issues. Acunetix can be integrated with various DevOps tools and workflows, such as Jenkins, Azure DevOps, and TeamCity*

8. CyberRes Fortify: i*s a software security platform that provides automated static and dynamic application security testing. CyberRes Fortify also offers integrations* *with various DevOps tools, such as Jenkins, GitLab, and Azure DevOps, enabling organizations to incorporate security testing into their continuous integration and delivery (CI/CD) pipelines.*

9. Docker Security Scanning*: Docker Security Scanning is a tool that automatically scans Docker container images for security vulnerabilities. It integrates with Docker Hub and Docker Trusted Registry to provide vulnerability reports and recommendations for securing containerized applications.*

10. GitLab Secure*: GitLab Secure is a set of security tools integrated into the GitLab CI/CD pipeline to help identify and remediate security issues in code. It includes features such as static application security testing (SAST), dynamic application security testing (DAST), dependency scanning, and container scanning.*

11. HashiCorp Vault*: HashiCorp Vault is a popular tool for managing secrets and sensitive data in modern infrastructure. It provides secure storage, encryption, and access control for secrets such as API keys, passwords, and certificates, helping organizations maintain strong security practices in their 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," 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.

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 serverless compute environments.

7. About Local and international DevSecOps career opportunities, career pathThe DevSecOps Career Path

The DevSecOps career path starts with a solid foundation in software development. Many DevSecOps engineers start as software developers or system administrators before transitioning to a DevSecOps role. Relevant certifications, such as Certified DevSecOps Professional (CDP) and Certified DevSecOps Expert (CDE), can also help you to advance in this career path. You can also reach for leadership roles in DevSecOps through certifications like Certified DevSecOpsDevSecOps professionals have a wide range of career opportunities both locally and internationally, given the increasing demand for individuals with expertise in integrating security into the software development process. Here are some career paths and opportunities in the DevSecOps field:

1. Security Engineer: Security engineers play a critical role in implementing security measures, conducting security assessments, and ensuring compliance with security standards. They work closely with development and operations teams to integrate security into the software development lifecycle.

2. DevSecOps Engineer: DevSecOps engineers are responsible for automating security processes, implementing security tools and practices, and managing security incidents within the DevOps pipeline. They bridge the gap between development, security, and operations teams to ensure that security is a priority at every stage of the development process.

3. Security Analyst: Security analysts assess and monitor security risks, analyze security incidents, and recommend security solutions to protect an organization's systems and data. They play a crucial role in identifying vulnerabilities and implementing security controls to mitigate risks.

4. Security Architect: Security architects design and implement secure systems, networks, and applications by developing security strategies, policies, and architectures. They work closely with development teams to ensure that security requirements are integrated into the design and development of software applications.

5. Security Consultant: Security consultants provide expertise and guidance on implementing security best practices, conducting security assessments, and developing security policies and procedures. They work with organizations to assess their security posture and recommend solutions to improve their overall security posture.

In terms of career paths, individuals looking to pursue a career in DevSecOps can start as entry-level security analysts or engineers and then progress to more senior roles such as security architect or consultant. Continuous learning and professional development through certifications such as Certified DevSecOps Professional (CDP) or Certified Information Systems Security Professional (CISSP) can help advance one's career in the DevSecOps field.

Internationally, there is a growing demand for DevSecOps professionals across various industries, including technology, finance, healthcare, and government sectors. Organizations around the world are recognizing the importance of integrating security into their development processes to protect against cyber threats and ensure the integrity of their systems and data