WOLDIA UNIVERSITY

COLLEGE OF TECHNOLOGY

SCHOOL OF COMPUTING

DEPARTMENT OF SOFTWARE ENGNEERING

® COURSE TITLE: SOFTWARE ENGINEERING TOOLS AND PRACTICES.

® COURSE CODE : SEng3051

® INDIVIDUAL ASSIGNMENT

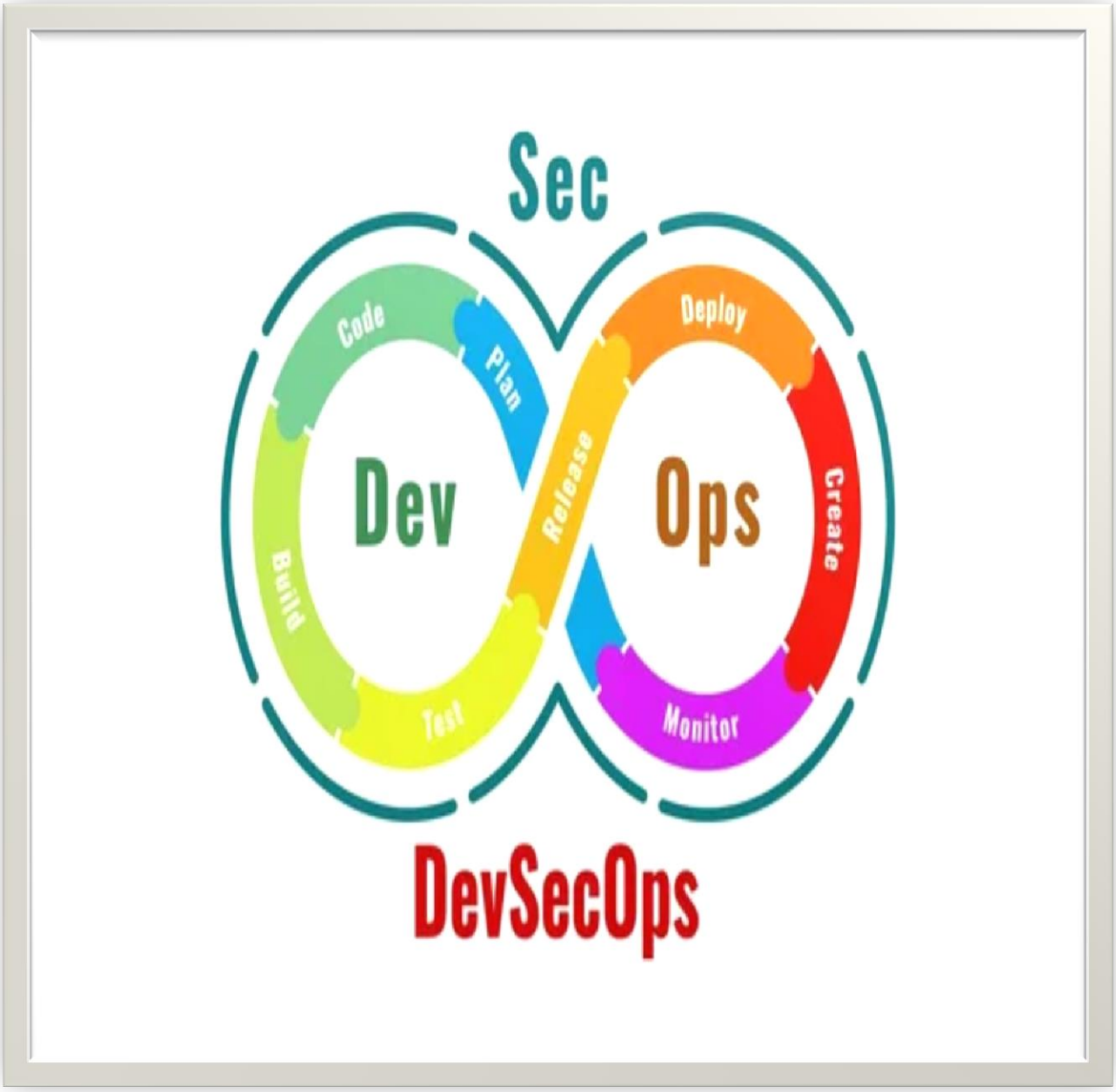
® ACADAMIC YEAR 2016E.C

THIRD YAER FIRST SEMESTER

INSTRUCTOR:- ESMAIL .M

THIS ASSIGNMENT IS DONE By :-

MOHAMMED ALI =ID 1302225



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

Software Engineering Problems Leading to Initiation of DevSecOps **are**

1. Lack of Security Integration in Software Development: One of the primary software engineering problems that led to the initiation of DevSecOps was the lack of security integration in the software development lifecycle. Traditionally, security was often treated as an afterthought in the development process, with security measures being implemented only at the end stages or even post-deployment. This approach left software vulnerable to cyber threats and attacks, as security considerations were not woven into the fabric of the development process from the beginning.

2. Siloed Teams and Lack of Collaboration: Another significant issue in software engineering was the presence of siloed teams and a lack of collaboration between developers, operations, and security teams. This lack of communication and collaboration often resulted in security vulnerabilities being overlooked or not adequately addressed during development. DevSecOps emerged as a response to this problem by promoting a culture of collaboration and shared responsibility among all stakeholders involved in software delivery.

3. Slow Response to Security Threats: Traditional software development practices often led to slow responses to emerging security threats. Security patches and updates were typically applied reactively, after vulnerabilities had been exploited by threat actors. DevSecOps aims to address this issue by integrating security practices throughout the development pipeline, enabling faster identification and remediation of security issues.

4. Compliance Challenges: Compliance with regulatory requirements and industry standards is a critical aspect of software development, especially in sectors such as finance, healthcare, and government. However, ensuring compliance can be challenging without incorporating security practices early in the development process. DevSecOps helps organizations meet compliance requirements by automating security checks and audits, thereby reducing the burden on development teams.

5. Increasing Complexity of Software Ecosystems: The modern software landscape is characterized by complex ecosystems comprising diverse technologies, platforms, and third-party dependencies. Managing the security risks associated with this complexity requires a proactive approach that integrates security into every stage of the software development lifecycle. DevSecOps provides a framework for addressing these challenges by emphasizing continuous monitoring, testing, and automation to enhance overall security posture.

6. Need for Continuous Delivery and Deployment: With the growing demand for rapid delivery of software products and updates, traditional approaches to security become inadequate. DevSecOps recognizes the importance of continuous delivery and deployment practices but also emphasizes the need for continuous security testing and validation throughout the process. By automating security checks and integrating them into CI/CD pipelines, organizations can achieve both speed and security in their software delivery.

7. Evolving Threat Landscape: The evolving nature of cybersecurity threats poses a constant challenge for software engineering teams. From ransomware attacks to data breaches, organizations face a wide range of threats that require proactive measures to mitigate risks effectively. DevSecOps enables organizations to stay ahead of these threats by embedding security controls into their development processes and fostering a culture of vigilance towards emerging risks.

In conclusion, the initiation of DevSecOps was driven by various software engineering problems such as the lack of security integration, siloed teams, slow response to threats, compliance challenges, increasing complexity in software ecosystems, the need for continuous delivery, and deployment practices, as well as the evolving threat landscape.

2. What is DevSecOps?

DevSecOps is a software development approach that integrates security practices within the DevOps process. It aims to shift security left in the software development lifecycle, meaning that security is incorporated from the beginning rather than being added as an afterthought. By integrating security into the DevOps pipeline, DevSecOps seeks to automate security testing and processes, enabling faster and more secure software delivery.

Key Components of DevSecOps:

1. **Automation:** DevSecOps heavily relies on automation tools to integrate security testing and processes seamlessly into the development pipeline.
2. **Collaboration:** It emphasizes collaboration between development, operations, and security teams to ensure that security is a shared responsibility across all stages of software development.
3. **Continuous Monitoring:** DevSecOps promotes continuous monitoring of applications and infrastructure to identify and address security vulnerabilities in real-time.

Benefits of DevSecOps:

1. **Improved Security Posture:** By integrating security early in the development process, organizations can proactively identify and mitigate security risks.
2. **Faster Time to Market:** Automation in DevSecOps streamlines the development process, allowing for quicker delivery of secure software.
3. **Enhanced Compliance:** DevSecOps helps organizations meet regulatory requirements by embedding compliance checks into the development pipeline.

Challenges of Implementing DevSecOps:

1. **Cultural Shift:** Adopting DevSecOps requires a cultural shift towards collaboration and shared responsibility among different teams.
2. **Skill Gaps:** Organizations may face challenges in finding professionals with expertise in both security and development.
3. **Tool Integration:** Integrating various security tools into the DevOps pipeline can be complex and require careful planning.

In conclusion, DevSecOps is a methodology that emphasizes the importance of incorporating security practices throughout the software development lifecycle. By automating security processes, fostering collaboration, and continuously monitoring for vulnerabilities, organizations can enhance their overall security posture while accelerating software delivery.

3.Briefly explain Dev SecOps lifecycle ?

Dev SecOps Life Cycle:

DevOps follows a traditional development cycle that involves phases like Plan, Code, Build, Test, Release, Deploy, Operate, and Monitor. Whereas, in Dev SecOps, some distinct security steps are integrated into each of the DevOps development phases for thorough security checks, which help organizations build and deliver increasingly secure products at an accelerated rate.

- ✓ **Threat Modeling:**

The first phase of the Dev SecOps lifecycle, thread modeling, helps the team assess an application and its surrounding environment to find as many vulnerabilities as possible before attackers do.

By implementing threat modeling within the traditional development process, teams are able to gather a summary of possible attack scenarios, outline the sensitive data workflow, identify vulnerabilities and potential mitigation options.

Like the majority of the processes in Dev SecOps, this is also implemented with the help of tools like OWASP Threat Dragon, Irius Risk, Threat Modeler, etc.

- ✓ **Scan & Analyze:**

After the threat modeling phase, the code is analyzed in the scanning phase to ensure it is secure from security vulnerabilities. This phase involves both manual and automated code review, which helps developers to identify security vulnerabilities and bugs earlier in the software development life cycle.

This phase involves the use of tools like Static Application Software Testing (SAST) and Dynamic Application Security Testing (DAST).

- ✓ **Identity:**

After code analysis, the team reviews all the data and metrics collected from the previous phases to identify security risks. These risks are then compiled based on their severity and priority.

Tools like Klocwork can be used to identify security vulnerabilities within the data and metrics collected.

- ✓ **Remediate:**

Once all the security vulnerabilities are identified and organized in the previous phases, the team moves on to the remediation phase, where steps are taken to rectify issues. This involves the use of various SAST tools that suggest solutions for the identified vulnerabilities, errors, and bugs.

This makes it easier for the team to address and rectify the security issues as they arise.

- ✓ **Monitor:**

Though last, this is another critical phase of the Dev SecOps lifecycle, where the team is responsible for tracking all the identified vulnerabilities, the steps taken to mitigate or eliminate those vulnerabilities, and the overall status of the application's security. This allows them to make informed data-driven decisions during the software development lifecycle, which further helps them deliver quality and secure products/features to the users.

Apart from tracking the aforementioned aspects, the team can also track and manage the differences between the actual and target metric values, which will allow the organization to experience advancement in operational efficiency across various departments.

Though there is no concrete process for implementing Dev SecOps, these steps are usually present. Depending on the complexity and size of your project, your development lifecycle might include some other sequential steps.

Best Practices for Dev SecOps Implementation:

To ensure maximum benefits with Dev SecOps, the team must follow certain best practices like:

- Make sure to build security into the application.
- Follow necessary secure code guidelines.
- Scan and secure all the open-source and third-party components within the application.

4. How does Dev SecOps works ?

Dev SecOps, which stands for **development, security, and operations**, is a framework that integrates security into all phases of the software development

lifecycle. Organizations adopt this approach to **reduce the risk of releasing code with security vulnerabilities**.

Here's how Dev SecOps works:

1. **Continuous Integration**:
    - ✓ Developers commit their code to a central repository multiple times a day.
    - ✓ Code is automatically integrated and tested.
    - ✓ This approach catches integration issues and bugs early in the process, rather than waiting until the end when there could be several issues that need to be resolved.
2. **Collaboration and Responsibility**:
    - ✓ Dev SecOps encourages **collaboration** between development, security, and operations teams.
    - ✓ **Shared responsibility** for security: The entire team takes responsibility for quality assurance, code integration, and security.
    - ✓ Teams discuss security implications during planning and begin testing for security issues in development environments, rather than waiting until the end.
3. **Shift Left Security**:
    - ✓ Dev SecOps aims to address security issues from the very start of the project.
    - ✓ By integrating security early (also known as **shift left security**), teams can identify and fix vulnerabilities before they become critical.
4. **Automation and Standardization**:
    - ✓ Automation tools are used to scan code for security vulnerabilities.
    - ✓ Standardized processes ensure consistent security practices throughout the development lifecycle.

5. **Benefits**:

   - ✓ **Dynamic Security**: Security is not an afterthought; it's part of the development process.
   - ✓ **Reduced Risk**: Deploying software with fewer misconfigurations and vulnerabilities.
   - ✓ **Faster Response**: Addressing security issues early prevents costly breaches.

In summary, DevSecOps combines development, security, and operations to create a secure and efficient software development process.

   - ✓ By integrating security early (also known as **shift left security**), teams can identify and fix vulnerabilities before they become critical.

6. **Automation and Standardization**:

   Automation tools are used to scan code for security vulnerabilities.

5.Exline well known DevSecOps tools ?

The growth of Dev SecOps tools is an encouraging sign that software and application service providers are increasingly integrating security into the software development lifecycle (SDLC).

The top Dev SecOps vendors offer a comprehensive suite of testing tools, including static application security testing (SAST), dynamic and interactive analysis testing (DAST and IAST), and software composition analysis (SCA). Though still a maturing market, several Dev SecOps vendors stand out, offering tools for containers, continuous integration and continuous delivery (CI/CD) pipelines, and API management.

Dev SecOps is a crucial approach that integrates security practices into the software development and deployment process. It prioritizes security throughout the entire lifecycle, from design to deployment, rather than treating it as an

afterthought. By utilizing a variety of tools, Dev SecOps aims to detect and resolve vulnerabilities early in the development pipeline.

Here are some essential Dev SecOps tools:

1. **Jenkins**: An open-source automation server that streamlines continuous integration and continuous delivery (CI/CD) processes. It seamlessly integrates security checks throughout the development lifecycle
2. **GitLab**: A complete DevOps platform that includes CI/CD pipelines, container scanning, and security testing.
3. **SonarQube**: A code quality and security analysis tool that identifies code vulnerabilities and provides actionable insights.
4. **OWASP ZAP (Zed Attack Proxy)**: A security testing tool for finding vulnerabilities in web applications.
5. **Truffle Hog**: A tool that scans Git repositories for secrets and sensitive information.

6. What are the benefits of DevSecOps ?

The benefits of DevSecOps include:

1. **Reduced Vulnerability Patch Time**: Integrating security into the release cycle allows for quicker identification and patching of vulnerabilities.
2. **Enhanced Security**: Continuous security testing helps create high-quality, secure applications.
3. **Simplified Compliance**: Security is built into the application, avoiding the need for retrofitting.
4. **Increased Efficiency**: Automation in DevSecOps scales development and uniformly adopts security features.
5. **Shared Responsibility**: Security becomes a collective responsibility across all IT teams.

By adopting DevSecOps, organizations can ensure that security is a fundamental part of the development process, leading to more secure and reliable software.

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

DevSecOps career opportunities are growing globally due to the increasing focus on integrating security into the development process. Here's a brief overview of the career path:

- **Starting Point**: Many start as software developers or system administrators before transitioning to DevSecOps roles.
- **Certifications**: Certifications like Certified DevSecOps Professional (CDP) and Certified DevSecOps Expert (CDE) can enhance career prospects.
- **Responsibilities**: As you advance, responsibilities may include leading teams, strategizing security practices, and integrating security at every stage of software development.
- **Job Market**: There is a strong demand for DevSecOps professionals due to the critical role of security in today's software culture.

For local opportunities, it's best to check job boards or professional networks specific to your region. Internationally, job platforms like LinkedIn and Indeed can provide a wide range of listings in various countries.