INSTITUTE OF TECHNOLOGY

SCHOOL OF COMPUTING

DEPARTMENT OF SOFTWARE ENGINEERING

Course title: Software Tools and Practices

Course code:SEng3051

INDIVIDUAL ASSIGNMENT

Summitted To: Mr.Esmael

Summited Date: 20/07/2016E.C

table of content

Content page

DevSecOps

Introduction

DevSecOps is a compound of Development, Security, and Operations, is a methodology that integrates security practices into the DevOps process. This synergistic approach aims to create a culture of shared responsibility for security, where teams collaborate seamlessly to deliver secure and reliable software at a faster pace.

In this document, we delve into the meaning and significance of DevSecOps, exploring how it transforms traditional approaches to software development by emphasizing security from the outset. We will discuss the key benefits of implementing DevSecOps, such as enhanced security posture, faster time-tomarket, and improved collaboration among teams.

Furthermore, we will examine the lifecycle of DevSecOps, outlining the stages involved from planning and development to testing, deployment, and monitoring. Understanding the lifecycle is crucial for organizations looking to adopt DevSecOps practices effectively.

Lastly, we will explore the tools and technologies that support DevSecOps implementation. These tools play a pivotal role in automating security processes, enabling continuous integration and deployment, and facilitating the seamless integration of security into every phase of the software development lifecycle.

Join us on this journey as we unravel the essence, advantages, lifecycle, and tools of DevSecOps, and discover how it can revolutionize your approach to software development and security.

Software Engineering Problems Causes For The Initiation Of DevSecOps

In traditional software development, projects are split into distinct phases for planning, design, development, integration, and testing, which happen sequentially over several months or even years. Although this approach is very methodical, many organizations have found it to be too slow, making it difficult to meet customers' expectations for continuous product improvements. Additionally, security typically gets bolted on at the very end, which puts companies at risk of a breach.To remain competitive, many companies have adopted a DevOps model that prioritizes delivery of smaller packets of high-quality code rather than feature-rich projects that take longer. In this framework, software development and operations teams collaborate to incorporate testing and integration throughout the process. Automation, standardized processes, and collaboration help teams move quickly without sacrificing quality.

DevSecOps is an enhancement to DevOps that builds security into all aspects of the process. The goal is to address security issues from the very start of the project. In this framework, not only does the entire team take responsibility for quality assurance and code integration but also security. In practice, this means teams discuss security implications during planning and begin testing for security issues in development environments, rather than waiting until the end. Another name for this approach is shift left security.The initiation of DevSecOps was driven by several software engineering problems that highlighted the need for integrating security practices into the software development lifecycle. Some of the key issues that led to the emergence of DevSecOps include:

Silos Between Development , Operations, and Security Teams

Traditional software development often involved separate silos for development, operations, and security teams. Lack of communication and collaboration between these groups led to security being an afterthought rather than a priority.

Rapid Development and Deployment

The increasing pace of software development and deployment cycles with practices like Continuous Integration/Continuous Deployment (CI/CD) meant that security measures were often bypassed or overlooked in favor of speed.

Security Breaches and Data Leaks

High-profile security breaches and data leaks due to vulnerabilities in applications highlighted the importance of integrating security practices early in the development process to prevent such incidents.

Manual Security Practices

Reliance on manual security testing and compliance checks resulted in slower processes, increased human error, and inconsistency in security measures, making it challenging to maintain a secure development environment.

Compliance Challenges

Meeting regulatory requirements and compliance standards became more complex with evolving regulations. Without automated security checks and controls, ensuring compliance was a time consuming and error-prone task.

Lack of Security Awareness

Many developers lacked in-depth knowledge of security best practices, leading to the introduction of vulnerabilities in the code unintentionally. Integrating security into the development process aimed to increase security awareness among all team members Inadequate Security Testing

Traditional testing methodologies often focused more on functionality rather than security. This resulted in security vulnerabilities being missed until later stages of development or even after deployment, leading to costly fixes.

By addressing these software engineering challenges through the adoption of DevSecOps practices, organizations aim to overcome these issues and build a more secure and resilient software development process. DevSecOps strives to embed security into every phase of development, promoting a culture of security awareness, collaboration, and continuous improvement.

What is DevSecOps?

DevSecOps, which stands for development, security, and operations, is a framework that integrates security into all phases of the software development life cycle. It is a methodology that integrates security practices within the DevOps (Development and Operations) process. It focuses on incorporating security early in the software development lifecycle, rather than treating it as an afterthought. Organizations adopt this approach to reduce the risk of releasing code with security vulnerabilities. Through collaboration, automation, and clear processes, teams share responsibility for security, rather than leaving it to the end when issues can be much more difficult and costly to address. DevSecOps is a critical component of a multi cloud security strategy.

Briefly Explain DevSecOps Lifecycle

The DevSecOps lifecycle involves the following stages:-

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 stages, 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. Security requirements are defined at this stage. This includes identifying security objectives, risk assessments, and creating security policies.

2. Develop or build

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.

3. Test

Since testing happens during development, a separate testing phase is optional 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. 4. Deploy

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.

DevSecOps ensures that security configurations are applied, vulnerabilities are addressed, and access controls are properly set.

5. Monitoring

After deployment, teams then monitor the application for security threats and respond to any incidents that occur. Continuous monitoring is essential to detect and respond to security threats. Security incidents are identified, investigated, and mitigated in real-time to maintain the security of the application.

6. Feedback

Feedback is gathered from security incidents, testing results, and monitoring data. This feedback loop helps in identifying areas for improvement and enhancing security practices in future iterations.

How DevSecOps Works?

DevSecOps is a practice that integrates security measures at every stage of the software development lifecycle.it involves collaboration between development, security and operations teams to ensure that security is considered from the beginning of a project rather than being added as an afterthought. So it does like the following:

1. Integration of Security

In DevSecOps, security is integrated throughout the software development pipeline. Security considerations are made at every stage, from planning and coding to testing and deployment.

2. Automation of Security Processes

Security practices are automated wherever possible to ensure consistency and efficiency. This includes automated security testing, vulnerability scanning, and compliance checks.

3. Collaboration and Communication

DevSecOps promotes collaboration between development, operations, and security teams. By breaking down silos and fostering communication, teams can better identify and address security vulnerabilities.

4. Continuous Monitoring

Continuous monitoring is a key aspect of DevSecOps. Teams use monitoring tools to detect and respond to security threats in real-time, thus enhancing the overall security posture of the software.

5. Shift Left Approach

DevSecOps follows a shift left approach, which means addressing security concerns early in the development process. By identifying and mitigating security issues at the outset, the likelihood of vulnerabilities slipping through to production is reduced.

6. Security as Code

Developers write security configurations as code, just like they do with the application code. This practice ensures that security policies are version controlled, automated, and auditable.

7. Shared Responsibility

DevSecOps emphasizes that security is everyone's responsibility, not just the security team's. Developers, operations, and security professionals all play a role in ensuring the security of the software.

To sum up , DevSecOps aims to create a culture of security awareness and accountability within the organization. By embedding security practices into every stage of the software development lifecycle, DevSecOps helps in building secure, resilient, and high-quality software products.

Explain well known DevSecOps tools

The DevSecOps approach requires the use of various tools and strategies to identify and address security risks.

Software Composition Analysis (SCA) Tools:

1. ***OWASP Dependency-Check:*** OWASP Dependency-Check is a software composition analysis tool that identifies known vulnerabilities in project dependencies.
2. ***Retire.js:*** Retire.js is a scanner that detects vulnerable JavaScript libraries in your web application.
3. ***WhiteSource Bolt:*** WhiteSource Bolt is an open-source SCA tool that scans your project dependencies for known vulnerabilities and provides actionable remediation steps.
4. ***Dependency-Track:*** Dependency-Track is an open-source platform that tracks and monitors your project's dependencies, providing insights into their known vulnerabilities.
5. **OSSIndex:** OSSIndex is an open-source vulnerability database and analysis platform that integrates with various development tools to provide real-time security intelligence on project dependencies

Static Application Security Testing (SAST) Tools:

1. ***SonarQube:*** SonarQube is an open-source platform for continuous code quality inspection that includes static code analysis for identifying security vulnerabilities.
2. ***Bandit:*** Bandit is a Python-focused SAST tool that analyzes Python code for common security issues and vulnerabilities.
3. ***SpotBugs:*** Find Bugs is an open-source static analysis tool for Java applications that detects common coding errors, potential vulnerabilities, and performance issues.
4. ***RIPS:*** RIPS is an open-source PHP security analysis tool that helps identify security vulnerabilities and coding flaws in PHP applications.

5. ***PMD:*** PMD is an open-source source code analyzer for various programming languages, including Java, JavaScript, and XML, which identifies potential bugs, dead code, and security vulnerabilities.

Dynamic Application Security Testing (DAST) Tools:

1. ***OWASP ZAP:*** OWASP ZAP (Zed Attack Proxy) is an open-source web application security scanner that helps you identify vulnerabilities in web applications.
2. ***Nikto:*** Nikto is an open-source web server scanner that performs comprehensive tests against web servers to identify potential vulnerabilities.
3. ***Wapiti:*** Wapiti is an open-source web application vulnerability scanner that audits the security of web applications by performing black-box testing.
4. ***Arachni:*** Arachni is an open-source, modular web application security scanner that checks for a wide range of vulnerabilities and provides comprehensive reports.
5. ***Grabber:*** Grabber is an open-source web application scanner that detects security vulnerabilities by crawling and scanning web pages.

Container Security Tools:

1. ***Clair:*** Clair is an open-source container vulnerability scanner that analyzes container images and provides reports on known vulnerabilities.
2. ***Trivy:*** Trivy is an open-source vulnerability scanner for containers and other artifacts, such as operating system packages and application dependencies. It scans container images and provides detailed reports on any vulnerabilities detected, including their severity and remediation steps.

3. ***Anchore Engine:*** Anchore Engine is an open-source tool for analyzing container images for vulnerabilities, policy violations, and best practices.
4. ***Sysdig Falco:*** Sysdig Falco is an open-source behavioral activity monitoring tool designed specifically for containers and Kubernetes. It detects and alerts on anomalous behavior and potential security threats in real-time. Falco uses rules and policies to define expected container behavior and raises alerts when deviations occur.

Infrastructure Security Tools:

1. ***OpenSCAP:*** OpenSCAP is an open-source framework for compliance checking and vulnerability management, which includes capabilities for assessing and securing infrastructure systems.
2. ***Lynis:*** Lynis is an open-source security auditing tool that assesses the security configuration of Linux and Unix-based systems.
3. ***Dagda:*** Dagda is an open-source container security analysis tool that performs static analysis of container images to detect security issues and vulnerabilities.
4. ***ScoutSuite:*** ScoutSuite is an open-source multi-cloud security auditing tool that assesses the security posture of containerized infrastructure in public cloud environments.

Compliance Tools:

1. ***OpenSCAP:*** OpenSCAP is a Security Content Automation Protocol (SCAP) framework for compliance checking, vulnerability management, and measurement.
2. ***OpenVAS:*** OpenVAS (Open Vulnerability Assessment System) is a full-featured vulnerability scanner that can detect security vulnerabilities in systems and networks.

3. **Wazuh:** Wazuh is an open-source host-based intrusion detection system (HIDS) that helps with compliance monitoring, file integrity monitoring, and log analysis.

Dashboard Tools:

1. ***Grafana:*** Grafana is an open-source analytics and monitoring platform that allows you to create customizable dashboards for visualizing various metrics and data sources.
2. ***Kibana:*** Kibana is an open-source data visualization dashboard for Elasticsearch, used for exploring, analyzing, and visualizing data stored in Elasticsearch indices.
3. ***Metabase:*** Metabase is an easy-to-use open-source business intelligence and analytics tool that allows you to create dashboards and visualize data from various sources.

Vulnerability Tracking Tools:

1. ***OWASP DefectDojo:*** DefectDojo is an open-source vulnerability management tool that helps you track and manage vulnerabilities in your applications and infrastructure.
2. ***TheHive:*** TheHive is an open-source incident response and case management platform that includes features for tracking and managing vulnerabilities.

In conclusion, open-source tools play a crucial role in the field of cybersecurity, offering a wide range of solutions for different categories such as Software Composition Analysis (SCA), Static Application Security Testing (SAST), Dynamic Application Security Testing (DAST), Container Security, and Infrastructure Security. These tools provide valuable support in identifying vulnerabilities, assessing security risks, and ensuring compliance.

What Are The Benefits Of DevSecOps?

The benefits of implementing DevSecOps in software development include:

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.

Generally, DevSecOps enables organizations to build and deliver secure software at a faster pace, with reduced security risks and improved collaboration across teams, ultimately leading to more resilient and secure applications.

DevOps career paths

There are several DevOps career paths you can pursue in this exciting and indemand field. Here are some examples of the top DevOps career paths and what each entails, plus what you can expect in terms of your DevOps salary.

DevOps Software Tester

DevOps Software Testers test software applications to make sure they meet stakeholder expectations. This DevOps career involves responsibilities such as:

- Test planning.
- Designing and implementing automated testing frameworks.
- Implement continuous testing processes and workflows.
- Quality assurance.

To be a DevOps Software Tester, be familiar with DevOps, software development, and testing principles. Also know our way around testing frameworks, continuous testing tools, and quality assurance frameworks. we can learn more about the various DevOps tools and software by reading our product highlight.

Junior DevOps Engineer

One of the most common entry-level positions in this field is the Junior DevOps Engineer. A Junior DevOps Engineer works under Senior DevOps Engineers and has several responsibilities, such as:

- Troubleshooting issues.
- Writing scripts.
- Completing standard system administration tasks.

Junior Engineers may also be tasked with enhancing and maintaining DevOps processes.

To become a Junior DevOps Engineer, you should have a solid understanding of operating systems, cloud infrastructure, and programming languages. You should also be well-versed in DevOps principles and practices, including automation, continuous integration and deployment, monitoring, and source code management. DevOps Engineer

A DevOps Engineer builds, maintains, and enhances DevOps processes and infrastructure. They often work alongside development, testing, and operations teams, ensuring the software delivery pipeline is smooth and

efficient. Managerial in nature, the DevOps Engineer position absorbs several roles and responsibilities.

Perform the following tasks repeatedly

- Writing scripts that deploy.
- Debug, and test software.
- Building reusable code for your organization.
- Collaborate with developers, getting feedback to determine software condition.

You will also need to keep projects on track by troubleshooting issues as they pop up while also keeping team members motivated to meet goals. And you may also need to adapt to changes on the fly using Agile principles, make sure that computer systems and networks are running as they should, and, most importantly, promote a culture that leads to the timely development of highquality software.

DevOps engineers should have extensive technical knowledge in scripting and languages like Python, Ruby, or JavaScript. They should also be comfortable working with configuration management tools, automation frameworks, and Linux environments or shells. Many employers require at least a bachelor’s degree in software development, software engineering, computer programming, or a similar field. Beyond those technical requirements, soft skills like collaboration, time

management, and leadership can be helpful during your DevOps career as an engineer.

DevOps Architect

A DevOps Architect is in charge of designing and implementing DevOps processes and infrastructure to meet an organization's specific needs. Responsibilities of this DevOps career path begin with collaborating with developers, IT operations, executives, and other stakeholders to discover the company's requirements and devise a DevOps strategy that fulfills them.

DevOps Architects work with development teams to ensure infrastructure matches software application needs while being scalable. Additionally, they are responsible for:

- Designing and implementing systems for testing.
- Deployment and monitoring to enhance software delivery processes.
- Evaluating and selecting new technologies and tools to optimize DevOps pipelines.

A DevOps Architect should have a broad knowledge of system administration, infrastructure design, and software development. They should be well-versed in how cloud infrastructure, containerization, and orchestration work, while also having familiarity with automation tools and frameworks that can help enhance software delivery. To be able to recommend proper technology to stakeholders, DevOps Architects must stay up-to-date on the latest DevOps trends. And since they must foster collaboration between teams and stakeholders, DevOps Architects should also have strong communication skills.

DevOps Release Manager

A DevOps Release Manager manages the release of software to ensure it is delivered on time, up to par, and within budget. Choose this DevOps career, and you will plan and coordinate software releases by working with development, testing, and operations teams.

To ensure that software releases remain reliable, predictable, and repeatable, DevOps Release Managers must:

- Design and implement automated release processes.
- Manage change requests, ensuring that any changes are made in a manner that is both auditable and controlled. • Identify and mitigate potential risks that could negatively impact release, plus create contingency plans to fix

them. • Communicate with stakeholders via status updates and release schedules to keep everyone on the same page.

DevOps Release Managers should have extensive knowledge in release management, software development, system administration, automation tools and frameworks, and change management while having strong communication skills.

DevOps Automation Engineer

A DevOps Automation Engineer optimizes the software development lifecycle by automating the software delivery pipeline. They design and implement automation frameworks to boost the efficiency of developers and operations teams, plus CI/CD pipelines to automate software delivery.

To make sure software applications are thoroughly tested before being deployed, DevOps Automation Engineers also:

- Design and implement automated testing frameworks.
- Automate the provisioning and configuration of infrastructure using infrastructure-as-code (IaC) tools.

Choose this DevOps career path, and you will also find yourself working with development, testing, and operations teams to keep the software delivery pipeline as smooth and efficient as possible.

DevOps Automation Engineers should understand DevOps and software development principles, automation tools and frameworks, CI/CD tools, testing frameworks, and IaC tools while also having strong collaboration skills.

DevOps Security Engineer

DevOps or DevSecOps Security Engineers must ensure that software applications and their supporting infrastructure are secure. DevSecOps stands for Development, Security, and Operations. Such engineers design and implement secure architectures for software and infrastructure, manage vulnerabilities, protect against known security threats, and create automated scanning and testing processes.

Other responsibilities of DevOps Security Engineers include:

- Threat modeling to spot potential threats.
- Design strategies to minimize potential threats.

- Communicating security-related information to developers, IT teams, and executives.

Anyone interested in becoming a DevOps Security engineer should be familiar with DevSecOps principles, security frameworks and regulations, vulnerability scanning tools, automation tools and frameworks, and security threat modeling frameworks. Since they must relay security-related news to stakeholders, DevOps Security Engineers should also be strong communicators

Conclusion

DevSecOps represents a paradigm shift in the world of software development, where security is not an afterthought but an integral part of the entire development process. By weaving security practices into the fabric of DevOps, organizations can reap a myriad of benefits, ranging from improved security posture to accelerated delivery cycles and enhanced collaboration among disparate teams.

Throughout this document, we have explored the meaning and essence of DevSecOps, underlining its pivotal role in fostering a culture of shared responsibility and proactive security measures. We have also delved into the array of benefits that come with embracing DevSecOps, including heightened resilience against cyber threats, enhanced customer trust, and increased efficiency in software delivery.

Understanding the lifecycle of DevSecOps is paramount for organizations embarking on this transformative journey. From the initial planning stages through development, testing, deployment, and monitoring, each phase of the lifecycle presents unique opportunities to embed security into the software delivery pipeline.

Lastly, the tools and technologies that support DevSecOps implementation serve as enablers for seamless integration of security into every step of the development lifecycle. Automation, continuous monitoring, and actionable insights provided by these tools empower organizations to proactively address security vulnerabilities and deliver robust and secure software solutions.

As we conclude our exploration of DevSecOps, it is evident that this methodology presents a holistic approach to software development that prioritizes security without compromising speed and agility. Embracing DevSecOps is not just a choice; it is a strategic imperative for organizations looking to stay ahead in today's rapidly evolving digital landscape. By adopting DevSecOps principles, organizations can build a strong foundation for secure, reliable, and resilient software solutions, safeguarding their assets, reputation, and stakeholders in an increasingly interconnected world.

References

https://www.mavhem.securitv/blog/the-devsecops-lifecycle-how-to-automate-securitv-in-softwaredevelopment. "Lifecycle of DevSecOps"

https://ranianiitian.medium.com/top-devsecops-tools-for-2023-open-source-solutions-for-enterprises-7c146f80b325. "what are well known tools for DevSecOps?"