Woldia University

School of computing

Department of software engineering:

Course name: SOFTWARE ENGINEERING TOOLS AND PRACTICE

Content: individual assignment

Course code:SEng3051

Name: fuad nuru

Id: WLD1301239

Email: fudinuru55@outlook.com

Submitted to: Mr. Esmail

Content:

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

Historically, software security has primarily been addressed at the **project level**, focusing on activities like code scanning, penetration testing, and reactive incident response. However, this approach has limitations. To align security practices with business objectives and enhance overall security, organizations have shifted their focus to the **program level**. This shift aims to ensure that software development teams act in alignment with business goals, organizational risk, and solution architectures, recognizing that security is integral to business success.

DevSecOps emerged as a response to these challenges. It builds upon **DevOps principles** but places additional emphasis on security throughout all phases of the **software development lifecycle (SDLC)**. Here are **five common challenges** that led to the initiation of DevSecOps:

1. **Lack of Security Assurance at the Business and Project Levels**:
    - Historically, security assurance was often an afterthought, leading to vulnerabilities and incidents.
    - DevSecOps aims to integrate security practices from the outset, ensuring that security is considered at every stage of development.
2. **Organizational Barriers Related to Collaboration, Tooling, and Culture**:
    - Silos between development (Dev), security (Sec), and operations (Ops) hinder effective collaboration.
    - DevSecOps promotes cross-functional collaboration, breaking down these silos and fostering a culture of shared responsibility.

3. **Iterative and Incremental Development Challenges**:
    - DevSecOps encourages iterative and incremental development, breaking down software development into smaller, manageable steps.
    - Implementing continuous integration/continuous deployment (CI/CD) practices ensures automation, quality, and security throughout the SDLC.
4. **Continuous Feedback and Measurement**:
    - Collecting feedback throughout the entire lifecycle is crucial for expert validation and observability.
    - Metrics and measurement help evaluate performance in terms of productivity and quality.
5. **Balancing Speed of Delivery and Security**:
    - Achieving a suitable balance between rapid delivery and robust security practices is essential.

In summary, DevSecOps addresses these challenges by integrating security practices, fostering collaboration, emphasizing automation, and maintaining a balance between speed and security. It's a holistic approach that aligns software development with business goals and risk management.

2. What is DevSecOps?

DevSecOps is the practice of integrating security testing at every stage of the software development process. It includes tools and processes that encourage collaboration between developers, security specialists, and operation teams to build software that is both efficient and secure. DevSecOps brings cultural transformation that makes security a shared responsibility for everyone who is building the software.

DevSecOps stands for development, security and operation. It is an extension of the DevOps practice.

DevSecOps (short for development, security, and operations) is a development practice that integrates security initiatives at every stage of the software development lifecycle to deliver robust and secure applications.

DevSecOps infuses security into the continuous integration and continuous delivery (CI/CD) pipeline, allowing development teams to address some of today's most pressing security challenges at DevOps speed.

Historically, security considerations and practices were often introduced late in the development lifecycle. However, with the rise of more sophisticated cybersecurity attacks, and development teams shifting to shorter, more frequent iterations on applications, DevSecOps is now becoming a go-to practice for ensuring applications are secure in this modern development ecosystem.

DevSecOps stands for development, security, and operations. It is an extension of the DevOps practice. Each term defines different roles and responsibilities of software teams when they are building software applications.

Development

Development is the process of planning, coding, building, and testing the application.

Security

Security means introducing security earlier in the software development cycle. For example, programmers ensure that the code is free of security vulnerabilities, and security practitioners test the software further before the company releases it.

Operations

The operations team releases, monitors, and fixes any issues that arise from the software.

3.Briefly explain DevSecOps lifecycle?

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

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

1. Plan

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

2. Develop

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

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

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

Optional: Test

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

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

3. Deploy and Monitor

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

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

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

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

Benefits of Following The DevSecOps Lifecycle

Following a DevSecOps approach has many benefits. By integrating security directly into the software development lifecycle, organizations can proactively identify and mitigate security risks early in the development process, leading to:

- Early detection of security vulnerabilities: Integrating security from the beginning of the SDLC helps detect security vulnerabilities at an early stage. This enables developers to address these vulnerabilities before they turn into major security threats.
- Reduced time and cost: Integrating security into the SDLC reduces the costs associated with fixing security vulnerabilities at a later stage. Addressing security issues early in the SDLC helps avoid costly rework, saves time, and ensures that the software is secure from the outset.
- Improved software quality: Integrating security into the SDLC improves the overall quality of the software. By identifying and addressing security issues early on, developers can ensure that the software is more reliable and less prone to errors.
- Compliance with regulations: Many industries have regulations and standards that require software to meet specific security requirements. Integrating security into the SDLC ensures that the software meets these requirements, reducing the risk of non-compliance.
- Increased customer trust: By helping teams find - and fix - application vulnerabilities before release - DevSecOps helps organizations deliver more secure, reliable software to customers to build trust

  In addition:

There's no prerequisite for adopting DevSecOps principles - you can shift security left no matter what methodologies your development team follows. That said, teams adopting agile principles in how they write and ship code often see the most benefit from DevSecOps practices, due to a similarity in the principles of the two.

Comparing Agile, DevOps, and DevSecOps

Agile, DevOps, and DevSecOps are extremely similar approaches. The Agile methodology is an iterative approach that prioritizes customer satisfaction, collaboration, and working software as the primary measures of progress. It involves breaking down the development process into smaller sprints that deliver a working software increment, with an emphasis on continuous feedback and flexibility.

The DevOps methodology is an extension of Agile that focuses on the collaboration between development and operations teams. It aims to deliver software quickly and reliably by automating human operations tasks such as building and shipping code, as well as emphasizing continuous integration, continuous testing, and continuous delivery. The DevOps methodology also includes a focus on monitoring and feedback, with the goal of identifying and resolving issues as quickly as possible.

DevSecOps is an extension of DevOps in the same way that DevOps is an extension of Agile. It follows the DevOps methodology, focusing on security as an integral part of the development process. The DevSecOps process involves continuous integration, continuous testing, and continuous deployment, with a focus on security at every stage.

Since DevOps already emphasizes the importance of collaboration between development and operations teams, it's easy to integrate security testing into the DevOps process. By breaking down the silos between these teams, organizations can ensure that security is not an afterthought, but an integral part of the entire software development process.

Both Agile and DevOps also emphasize continuous integration and delivery (CI/CD), which allows for rapid feedback and iteration. This is essential for identifying and fixing security issues early in the development process. By adopting a DevSecOps approach, organizations can build products that are secure, reliable, and of high quality.

4. How does DevSecOps works?

DevSecOps will integrate automation across your software release pipeline to reduce downtime, security attacks, and fix code issues faster.

Businesses who want to employ security into their DevOps framework need to utilize optimal DevSecOps processes and tools.

Here's how your DevOps and DevSecOps framework will look like:

- A developer writes code in a version control management system.
- Any changes made are applied throughout the version control management system.
- Other developer retrieves this code from the version control system to analyze the static code quality and look for any potential security bugs.
- Utilizing infrastructure-as-code DevSecOps tools, developers create an environment such as Chef. In this infrastructure, you can easily deploy applications and perform environment-specific configurations.
- Perform test automation suite on the newly deployed application consisting of security tests, API, UI, integrations, and back-end.
- If the application runs seamlessly in testing, it is sent forward to the production environment.
- The production environment is kept under continuous monitoring to detect any active security risks, bugs, or attacks.

DevSecOps creates a test-driven development infrastructure that executes continuous integration and automated testing to create quality code, enhanced security, and compliance.

DevSecOps offers organizations a stronger approach to address modern security challenges in software development. DevSecOps helps teams create more secure software essentially by "shifting security left," or by incorporating the first security checks early and continuing them all throughout the development lifecycle. With DevSecOps, security optimally is evaluated during the planning stage and then again in every subsequent phase, including coding, deployment, and post-release operations (continuous monitoring and updating). This merging of security checks into existing Dev and Ops workflows is achieved through a combination of automation and more fundamental cultural changes.

Automation in DevSecOps

Automation is an important tool that helps teams meet the goals of DevSecOps, with continuous integration/continuous delivery (CI/CD) playing a particularly key role. Through CI/CD, teams can configure various jobs to run automatically in predefined pipelines (sequences) when code is submitted to an application repository such as Github, GitLab, or Bitbucket. The DevSecOps approach normally includes automated security tests in these CI/CD pipelines, which ensures that each code update undergoes some degree of security screening. These automated security tests each perform different types of scans, and they

can be created manually by the DevSecOps team or obtained through third-party sources.

The following are some examples of the types of automated security tests that teams can add to their CI/CD pipelines:

1. **Static application security testing (SAST) tools** scan source code or binaries before the application runs. These types of tests attempt to identify code-level vulnerabilities to various security threats, such as buffer overflows, SQL injection, and cross-site scripting (XSS).
2. **Software composition analysis (SCA) tools** scan for instances of open-source code and components that were used in development. SCA tools then review the open-source code for potential licensing issues or known vulnerabilities.
3. **Interactive application security testing (IAST) tools** are useful for analyzing the behavior of web applications. IAST relies on code instrumentation to monitor an application in its running state and to detect vulnerabilities in real time.
4. **Dynamic application security testing (DAST) tools** also test applications as they are running, but from the outside perspective of an attacker. DAST essentially simulates known attack methods such as cross-site scripting (XSS) and SQL injection to determine whether a running application is susceptible to them.

Note that these types of security tests aren't necessarily automated, and teams can perform them outside of CI/CD pipelines as well as inside them. However, teams that embrace a DevSecOps methodology to develop and maintain their software can use CI/CD pipelines to automate these tests and approach their larger security goals. Through automation, teams can improve application security by implementing mandatory checks throughout the software lifecycle—including early in that lifecycle, when problems are often easier and cheaper to fix.

5. Exline well known DevSecOps tools.

DevSecOps tools can be categorized into several groups based on their functionality. These categories include:

1. *Static Application Security Testing (SAST) Tools*

2. Dynamic Application Security Testing (DAST) Tools

3. Software Composition Analysis (SCA) Tools

4. Container Security Tools

5. Infrastructure as Code (IaC) Security Tools

6. Continuous Integration/Continuous Deployment (CI/CD) Security Tools

7. Compliance and Governance Tools

8. Security Dashboard and Analytics Tools

So let's see different tools in these categories:

Top OpenSource tools DevSecOps for 2023

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**: FindBugs 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.

6.What are the benefits of DevSecOps?

Generate business value

Including security at every stage of the software delivery lifecycle will allow to catch vulnerabilities earlier. The cost associated with detecting and fixing security issues will therefore be mitigated. This also means that, as the data and the products, the business as a whole is safer. In the long run, this generates value by strengthening your business image and brand.

Improve deliverability without compromises

With DevSecOps, security bottlenecks can be drastically reduced. Customers' security requirements are easier to meet on a schedule. One thing is true no matter what kind of customers you are serving: they use your product because they can trust it. As a consequence, you cannot make any compromise on security. Because threats are evolving rapidly, and at the same time the market

always wants features shipped more quickly, pressure can accumulate on the shoulders of software engineers. DevSecOps comes with the promise of improved security. As more compliance requirements will roll out, organizations will also be better prepared. DevSecOps enables companies to deliver features fast and often.

Better agility

Thanks to automation, the security teams have more time to perform global audits and to update their knowledge on what matters: assets, defining priorities, evolve strategies. They are also able to answer quicker to threats and overall to accelerate the SDLC. Speed of recovery is enhanced in the case of a security incident by using templates and pet/cattle methodology. But they are not the only ones benefiting in terms of agility. Indeed, as with any type of testing, when the pipeline is properly covered you are not afraid to make experiments anymore. That means that you can iterate faster and be less shy about implementing small or radical changes on any brick supporting your product. This is what is called a "business enabler".

Break the silos

The DevSecOps accelerates the development process. It automates everything related to security or policy, and more importantly, it's a repeatable process. The artifact is reusable for future projects and can be well integrated with your CI/CD pipelines.

1. Development, deployment and recovery acceleration

DevSecOps is a management lifecycle approach that combines application planning, delivery and monitoring approaches under a single framework. Part of the allure of DevSecOps is it can speed up many steps in the software development lifecycle (SDLC) and ensure continuous code integrations and updates are handled at the ever-increasing speed of business.

An automation framework can be constructed and executed during the deployment phase of the SDLC process. Applications can be placed into a framework where security functions are added, tested and automatically pushed into production. DevSecOps tools can also automatically monitor newly launched applications and can trigger a rollback to a previous version if an application-breaking bug is detected.

2. Elimination of remedial tasks

As with most technology automation practices, low-level, remedial tasks can be automated and eliminated throughout the SDLC. This includes the implementation and monitoring of security features within applications, as well as the monitoring of apps from a cybersecurity perspective.

3. Accurate auto-verification checks

When speed is critical to software development, it often comes at the cost of code accuracy. It's important to implement automated code verification checks into DevSecOps frameworks. These checks can identify errors and potentially point to remediation steps that won't slow down software updates and deployment schedules.

4. Security uniformity

A detailed DevSecOps framework should include processes that automatically integrate security functions across all software builds in a uniform manner. This highly structured approach creates a consistent security foundation where security is built in the same way every time an application moves through the continuous integration/continuous delivery lifecycle process.

5. Self-service functions

Mature DevSecOps automation involves providing developers with self-service security tools that remediate identified vulnerabilities without the need to directly interact with IT security staff. Self-service tools can be ingrained into the DevSecOps process during the following:

- secure application platform provisioning
- configuration management and control
- vulnerability and bug tracking
- reporting and auditing

Self-service tools within DevSecOps not only empower developers to take control of security without human bottlenecks, but also encourage cross-team skill development.

6. AI-backed threat analysis

Advanced DevSecOps frameworks take advantage of AI and machine learning techniques to streamline, simplify and speed up complex DevSecOps tasks. Two examples are the following:

1. Collecting and analyzing software and OS logging information identify which aspects of software bad actors are attempting to target. Based on this information, AI can suggest code alterations, adds or architectural changes to proactively identify code vulnerabilities.

2. From a testing perspective, code adds or changes can be run through finely tuned machine learning tools to identify how a particular change might affect other aspects of the application.

7. Ease of scalability

Once DevSecOps tools and processes are developed and tuned, it makes little sense to manually replicate them when more compute resources are required or when entire frameworks need to be replicated and placed in other physical locations. Scaling these systems and processes upward or downward at a moment's notice can be fully automated and kicked off with just a few clicks thanks to automated DevSecOps. A recent case study from Comcast showed 85% fewer security incidents with DevSecOps in place.

8. Streamlined compliance reporting

Adhering to business and industry policies and government compliance mandates is important for most business verticals. Auditing and reporting functions must,

therefore, identify relevant information, ensure accuracy and display data in an understandable and consistent manner.

For many security teams, auditing and reporting can be arduous tasks. They can be rife with complications due to lack of visibility, constantly changing data collection sources, and manually configured and operated tools that deliver varying results.

Automated auditing and compliance tools take a holistic approach to this process using a DevSecOps framework. Tools use AI and machine learning to intelligently learn a software's underlying infrastructure architecture and perform auditing scans on VMs or containers to verify if they have the proper security controls in place. The same tool set can also move up the stack to identify software-specific security controls, such as authentication, authorization and accounting, that may or may not meet acceptable compliance levels.

9. Bonus: Cost savings potential

One additional benefit that can result from proper DevSecOps automation is inherent cost savings. Gains can be found in several areas, including the speed at which software can be delivered, the lower likelihood of a catastrophic cybersecurity incident and the reduction in the number of operations staff required to thoroughly execute a secure SDLC process.

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

Certainly! Let's explore the exciting world of DevSecOps—a field that combines development, security, and operations. Whether you're interested in local or international opportunities, DevSecOps offers a dynamic career path. Here's what you need to know:

Why DevSecOps?

The corporate landscape faces increasing cybersecurity threats, making DevSecOps crucial.

ISO27001, the international standard for information security, now reflects this need for heightened cybersecurity awareness.

The DevSecOps industry was estimated at $2.79 billion in 2020, with a projected growth rate of 24.1% from 2021 to 20281.

DevSecOps Career Opportunities:

As a DevSecOps professional, you'll:

Automate security scans.

Verify code.

Develop security protocols.

Collaborate with operations staff and developers to ensure secure software design and continuous monitoring.

Starting Your DevSecOps Career:

Experience matters: Employers value practical experience.

Consider these routes:

Software Development: Gain coding and application development experience.

Operations or Security Roles: Learn about business tools, systems, and processes used to manage and secure software applications.

College Degree: Choose a major aligned with your career goals.

Certifications and Learning Paths:

Explore DevSecOps certifications to boost your resume.

Continuously learn about security best practices and emerging technologies.

Remember, DevSecOps offers a blend of technical skills, collaboration, and a commitment to safeguarding digital assets. Whether you're working locally or globally, this field promises exciting challenges and meaningful impact.

References

- **COPILOT**
- **HTTPS://TECHVIFY-SOFTWARE.COM**
- **HTTPS://BROWSERSTACK.COM**
- **HTTPS://GEEKSFORGEEKS.COM**