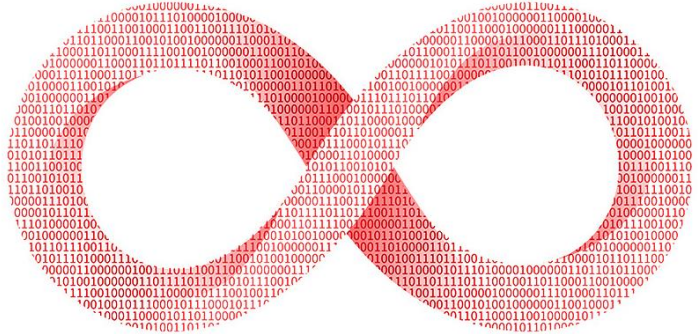


Speed up delivery of secure products

Leverage DevSecOps for proactive prevention of vulnerabilities

Credits | Lakshmiprasad A Kavya Rengaraj Priyankaa

Prodapt

Current state of software production in the connectedness industry

- **83%** of the businesses are implementing **DevOps** to accelerate product releases
- Complex, distributed applications that employ containers, cloud resources and microservices are increasing. This ideally shows that the **services are no more within perimeter security**
- Given the shift to hybrid work environment post the COVID-19 pandemic, more than **60% of businesses experience security breaches**

Source: Forrester

Traditional security approach lacks the following

- Mechanisms to handle agile developments leading to a fragile and vulnerable code
- Sophisticated security measures for new services that are beyond perimeter
- Techniques to secure the evolving ecosystems such as Kubernetes and observability tech stack which are highly prone to attacks

Impact of the traditional security approach

High OpEx due to detection of security issues in production	Delays in new application releases	Reputational damage due to security flaws in software

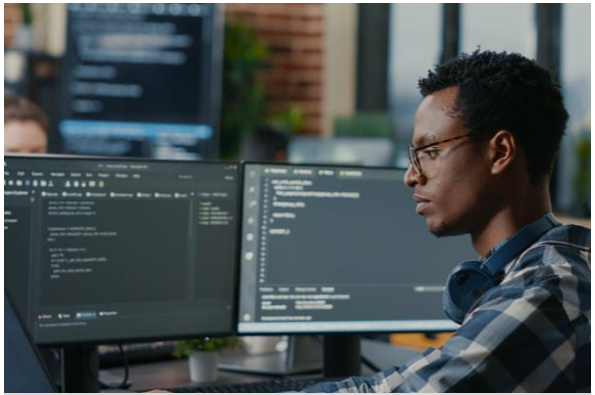
Continuous security: The need of the hour

Increasing data breaches

According to Cybersecurity Ventures, "More than 60% enterprises experience breaches and increase in cyberattacks. Cybercrime damages costs **$6 trillion globally**, and the cost is expected to increase by 15 percent per year over the next five years.

Emerging technologies and evolving threats

The ever-evolving technology landscape exponentially increases the rate of cyberattacks, paving the need for security in development and operations.



Rising vulnerabilities in open-source software

According to 2022 OSSRA report, 97% of the codebases contain open-source components and **81%** of them are **vulnerable**. Hence, it is vital for developers to secure code as they develop.

The need to release feature-rich applications faster makes security an afterthought. But service providers must look for ways to enable and prioritize **continuous security** by **infusing security at every stage of software development lifecycle.** This establishes trust in application usage and accelerates quality releases.

Leveraging DevSecOps to unite the power of agility and security

While service providers strive to shorten the release cycles by adopting **DevOps**, they often **deprioritize security**. Implementing **DevSecOps** helps to break the silos by automating, monitoring, and applying security throughout the software lifecycle.

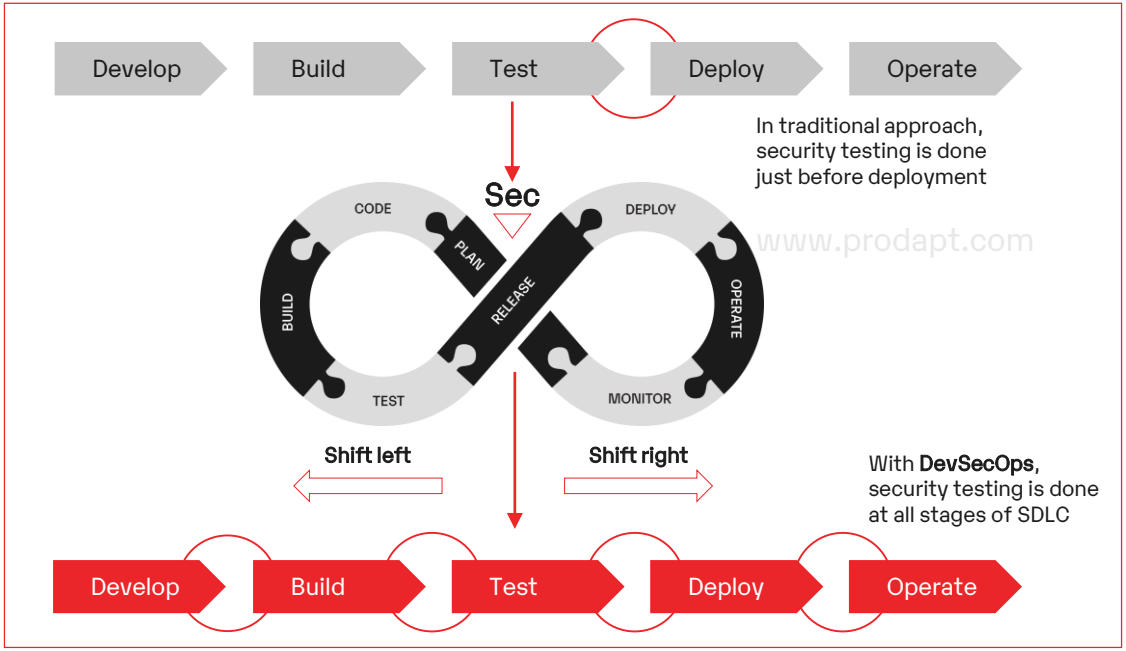
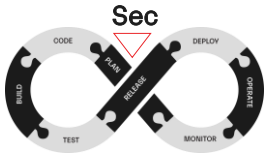


Fig. Leveraging DevSecOps to infuse security in all stages of SDLC

Implementing security at every stage of the **SDLC** enables continuous integration, faster delivery of secure products and reduction of compliance costs.

Prodapt

The four-step approach to implement DevSecOps and accelerate secure product releases



Shift left

Step 1

Shift left: Implement continuous security in agile software development

to secure code and application as they are built

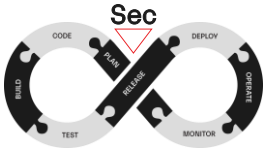
Develop → Build → Test → Deploy → Operate

Step 2

Harden the Kubernetes cluster

to examine potential threats and improve operational efficiency

Develop → Build → Test → Deploy → Operate



Shift right

Step 3

Enable continuous security monitoring

to combat vulnerabilities that continue to arise in the new agile developments

Develop → Build → Test → Deploy → Operate

Step 4

Implement vulnerability management

to track and manage all vulnerabilities in one place

Develop → Build → Test → Deploy → Operate

The following slides elaborates on a four-step approach for successful implementation of DevSecOps. It further helps to identify and **reduce 80% vulnerabilities** in the Software Development Life Cycle(SDLC).

Shift left: Implement continuous security in agile software development

Enable developers to secure code and application as they are built

Application security – 4 key components to secure

Open-source / Third-party libraries

Software Composition Analysis (SCA) 1

- Scan open-source packages using SCA tools such as **Dependency-check** and **npm** to mitigate risk and secure application health
- Integrate SCA tools into the CI/CD pipeline to enable automatic scanning of vulnerabilities

Application code

Static Application Security Testing (SAST) 2

- Implement tools like **SonarQube** and **Bandit** to inspect the code quality continuously and highlight the newly introduced issues
- Integrate SonarQube with Jenkins to execute unit test cases with every build, thus saving time and effort

Endpoints

Dynamic Application Security Testing (DAST) 3

- Leverage DAST tools such as **OWASP ZAP** and **Burp Suite** to automatically identify vulnerabilities in web application
- Perform active and passive scanning to detect specific set of vulnerabilities using known attacks

Containerized applications

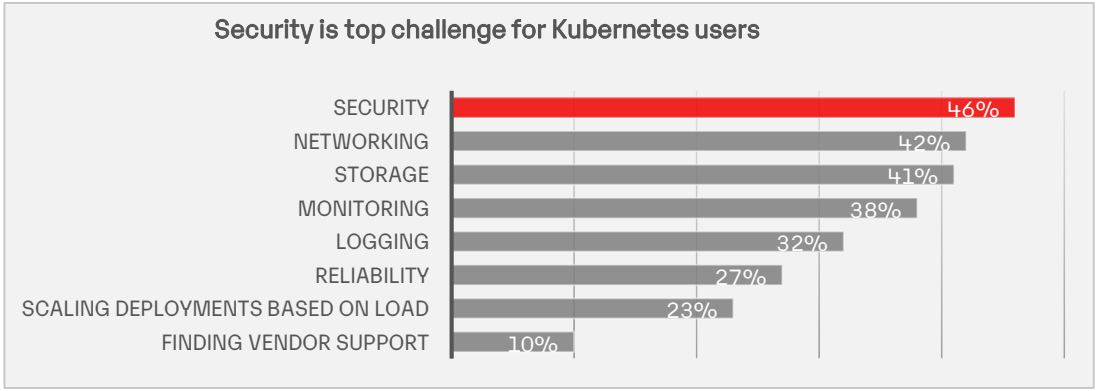
Image Scanning 4

- Implement vulnerability scanning tools such as **Trivy** and **Clair** to scan docker images, IaC scripts, vulnerabilities in OS packages and sensitive information. Drive smarter and reliable application architecture decisions
- Scan the Kubernetes manifest files and helm charts using Trivy to strengthen the security in running containers

The shift left approach in security facilitates identification and resolution of defects early in the software development lifecycle, improving code quality and reducing costs.

Harden the Kubernetes cluster to examine potential threats and improve operational efficiency

Rapid adoption of Kubernetes empowers service providers to embrace cloud-native solutions and achieve carrier-grade network and performance. However, it is critical to solve the issues in containerization, including concerns over system complexity and security.



Source: CNCF Survey

Common security issues in Kubernetes

- Detected misconfigurations
- Run-time security incidents
- Privilege escalation
- Exposed endpoints

According to Gartner

"The container ecosystem is immature and lacks operational best practices, but adoption of containers and Kubernetes is increasing for legacy modernization and cloud-native applications."

Harden the Kubernetes cluster to examine potential threats and improve operational efficiency

Recommendations

1. **Ensure Kubernetes infrastructure security:**
   - Implement **Kube-bench** to adhere to **CIS benchmarks** and deploy a Kubernetes cluster securely
   - Leverage **Kubescape** to harden the Kubernetes cluster by evaluating it against **MITRE ATT&CK matrix**

2. **Validate Kubernetes deployment configurations:** Implement tools like **Kube-score** and **KubeLinter** to identify insecure configurations in Kubernetes YAML files and Helm charts before deployment into a Kubernetes cluster

3. **Identify runtime security issues:** Leverage tools such as **Falco** and **Docker bench** to continuously monitor and identify security issues in the container and the environment in which it runs

4. **Automate the threat discovery process:** Leverage open-source tools, such as **Anchore**, to scan and analyze the container images for security vulnerabilities and automate the threat discovery process

Enable **continuous security monitoring** to combat vulnerabilities that continue to arise in the new agile developments

With the rising threats in agile development environments, uninterrupted monitoring of critical assets has become vital to detect and mitigate potential threats in real-time. Continuous security monitoring helps service providers to identify and detect security issues in container environments and orchestration platforms.

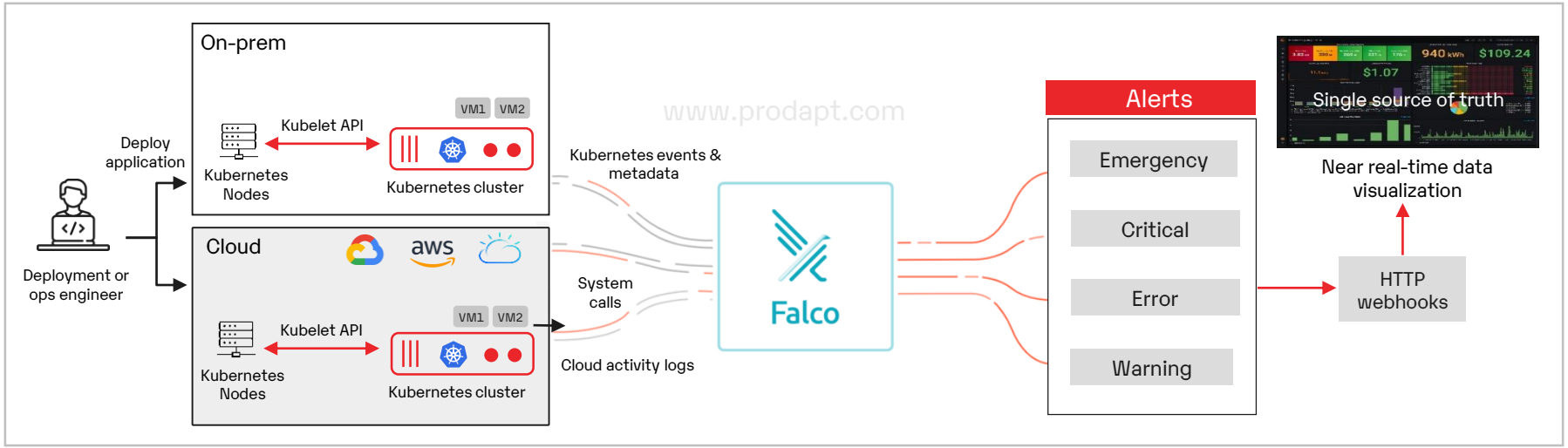
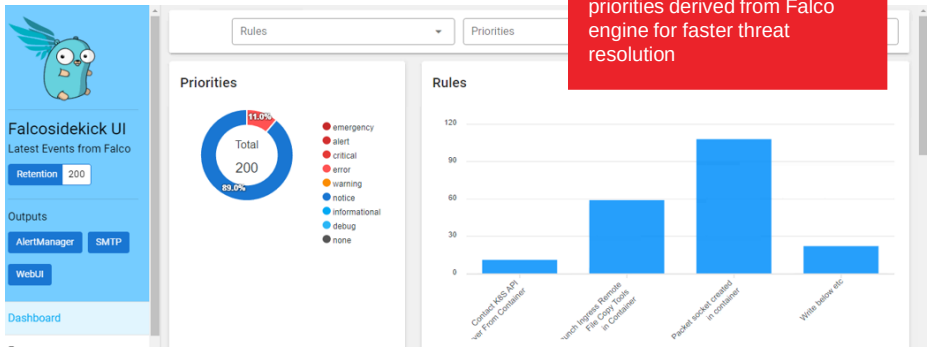


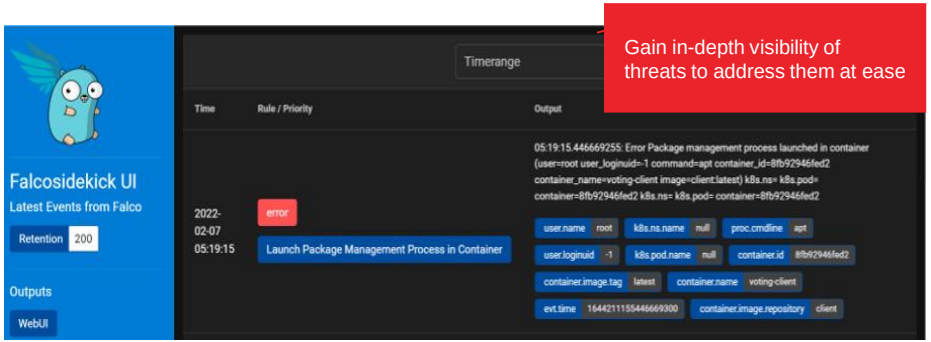
Fig. Sample flow of continuous security monitoring with Falco

Enable **continuous security monitoring** to combat vulnerabilities that continue to arise in the new agile developments

Recommendations

- Implement **Falco** to detect unexpected behavior, configuration changes, and data theft in real-time across containers, cloud and Kubernetes audit logs
- Leverage Falco to notify whenever a user tries to open a shell or delete the shell history. This provides high security to the clusters
- Gain real-time view of vulnerabilities by integrating **Falco with incident response workflow** systems through Alert Manager and Webhooks
- Ensure continuous monitoring of Docker daemons and host configurations to achieve end-to-end observability
- Perform **periodic scanning** of open-source packages to safeguard the application from recently exploited cyberattacks





Prodapt

Implement **vulnerability management** to track and manage all vulnerabilities in one place

As service providers struggle to combat vulnerabilities in their IT environments, they need real-time view of performance and security issues. Leveraging a centralized vulnerability management dashboard saves the time and effort of service providers in figuring out how a vulnerability affects the production environment and which systems are affected.

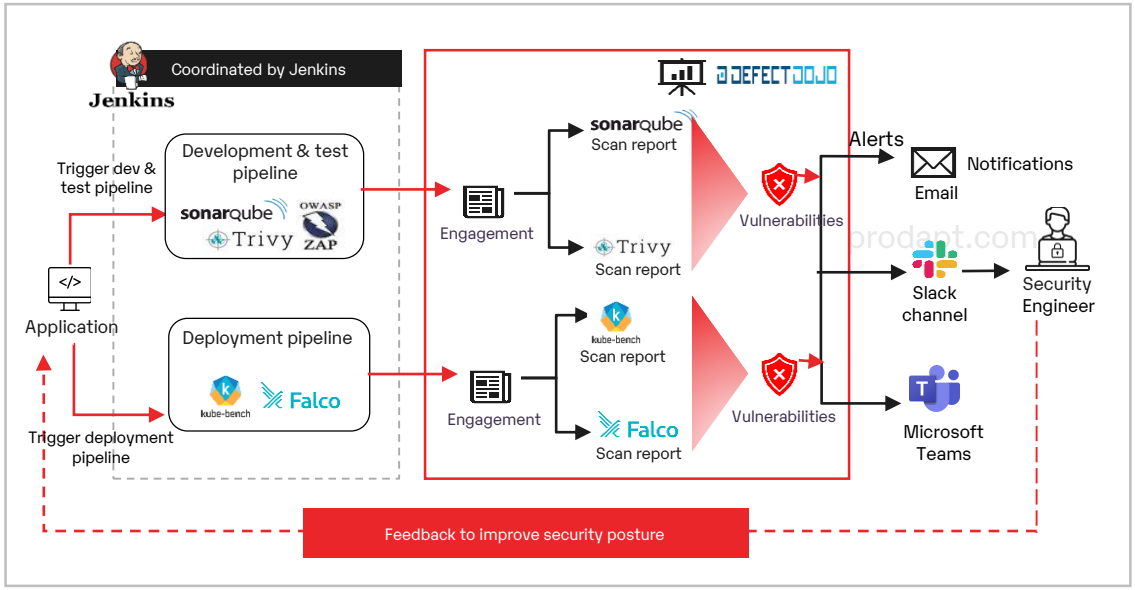
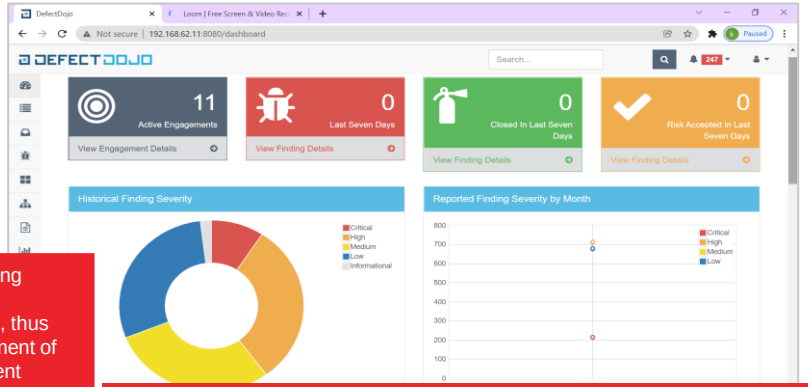


Fig. Sample flow of vulnerability management with DefectDojo

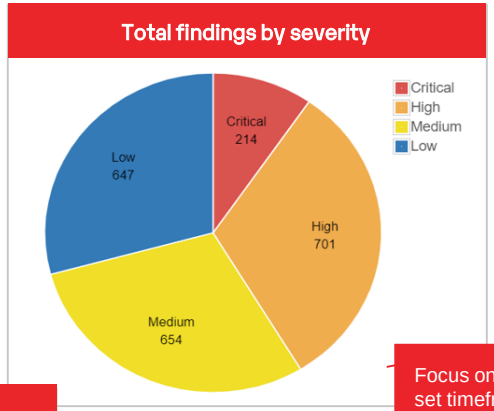
Recommendations

- Implement vulnerability management tools like **DefectDojo** for integration of DevOps and continuous security. It further helps in managing and tracking vulnerabilities raised
- Develop unique reports to understand exactly when new vulnerabilities are introduced in a build or remediated
- Configure **remediation timeframes** based on the criticality of findings which helps with reminders for remediation
- Set thresholds for determining the grade of product to gain a simple view of product health

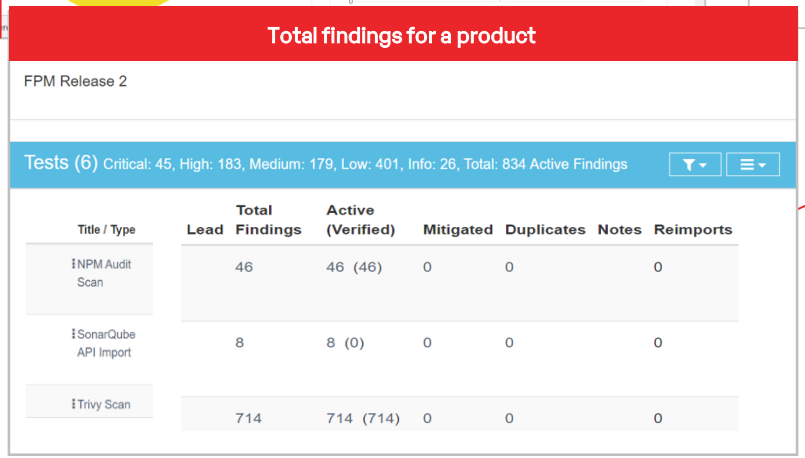
Implement **vulnerability management** to track and manage all vulnerabilities in one place



A unified dashboard providing various reports for tests, engagements and products, thus enabling effective management of vulnerabilities across different products.

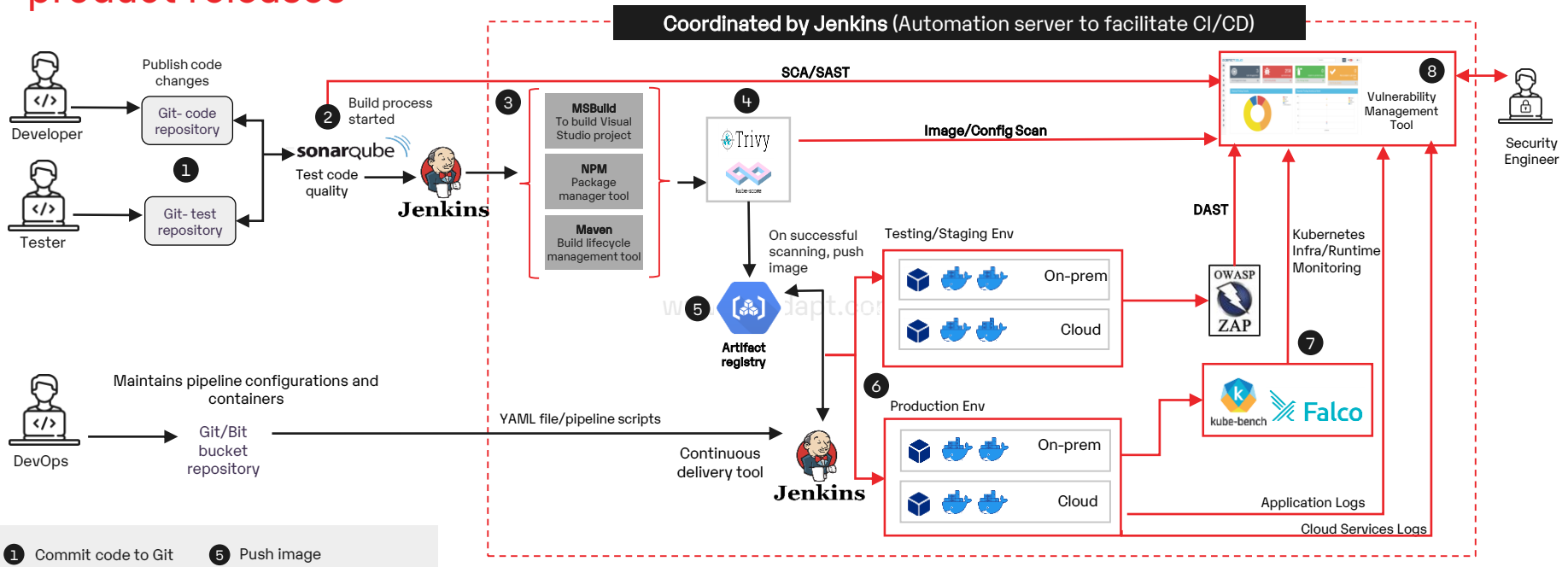


Focus on the critical findings and set timeframes for remediation.



Various scans and vulnerabilities of an engagement by severity to gain in-depth view of the development, test and deployment pipelines.

Sample DevSecOps pipeline implemented by leading service providers in Europe and Americas to fortify development and accelerate secure product releases



1. Commit code to Git
2. Evaluate code quality
3. Build code
4. Scan image
5. Push image
6. K8 deployment
7. K8 hardening and monitoring
8. Vulnerability management

The open-source platform agnostic DevSecOps pipeline ensured up to date security for all types of workloads - J2EE/ Python as well as .NET workloads and resulted in cost optimization. It further reduced the deployment time of security fixes/updates and accelerated secure product releases.

Prodapt

Business benefits achieved by a leading service provider in Europe after implementing DevSecOps approach

Implementing the four-step approach as discussed in this insight, resulted in the following benefits.

80% reduction in vulnerabilities

70% reduction in security incidents, optimizing OpEx

2x faster and secure product releases

Reduction in remediation time with early feedback on application architecture

Thank you

insights@prodapt.com