The bridge to possible

White paper
Public

DevSecOps – Addressing Security Challenges in a Fast Evolving Landscape

© 2021 Cisco and/or its affiliates. All rights reserved.

Contents

© 2021 Cisco and/or its affiliates. All rights reserved.

DevSecOps introduces a cultural change into the fabric of how we manage and integrate security into application development and operation; a concept that facilitates the practical fusion of development, security and operations, and the continuous development of new concepts and tools that support the DevSecOps practice.

The capability of being able to automatically provision environments using infrastructure components such as containers as part of deploying applications is generally referred to as Infrastructure-as-Code. The use of containers and container orchestration tools (ex. Kubernetes) is considered one of the mainstream approaches to realize practical DevOps, while noting that serverless is catching up in some application development areas, with new technology innovations advancing how applications can be quickly delivered.

In such an evolving technology landscape, how should you approach securing your DevOps lifecycle and how can you practically realize the concept of “Security as Code”, in which you are expected to approach security controls in the same way as other code requirements?

In this whitepaper, we explore a framework that takes continuous security deeper in the DevSecOps lifecycle, addressing challenges in culture, people, processes and technology.

DevSecOps - The Attack Surface Keeps Growing – We Need Continuous Security

Continuous security in DevOps: security is not a point of time activity in the DevSecOps lifecycle. Security should be realized early and continuously through automation delivered by the Continuous Integration (CI) and Continuous Delivery (CD) pipeline.

This whitepaper focuses on how to *build* and *streamline* robust automated security controls for the DevSecOps lifecycle, which covers topics related to how applications are developed and delivered through infrastructure components such as orchestrated containers or through serverless deployments.

Taking a discrete approach to addressing security challenges in the DevOps lifecycle, described in the “DevSecOps – What” section, can lead to the mis-representation of the actual security posture profile and/or compliance status, inaccurate or conflicting findings, non-actionable recommendations, and/or ending up with non-standard set of security controls across the phases of DevOps lifecycle. The reader may refer to the “DevSecOps – The Threat Landscape” section, in which we list a number of security questions that many of today’s DevSecOps deployments would have challenges addressing.

DevSecOps - Deploy Code Faster and Safer

DevSecOps is an organizational software engineering culture/fabric that aims at unifying and automating the practices of development (Dev), security (Sec) and operations (Ops), with an ultimate objective of achieving *agile* and *safe* deployment of code. DevSecOps can be also be thought of as integrating security controls in the fabric of the established DevOps lifecycle, i.e. baking in security, compared to the legacy model in which security gates are introduced after each of the DevOps lifecycle. While DevOps itself is a fast-evolving landscape with new tools being frequently introduced, existing rapidly enhanced and new practices and processes shared in a thriving community of developers and operators, the reader can imagine the challenges and opportunities that security practices and tools can introduce to this landscape.

According to a recent DevSecOps report published by Contrast Security[1], a staggering 99% of the survey responders report that an average application in production, contains at least four (4) vulnerabilities, while 79% reporting more than twenty (20) as the average number of vulnerabilities in an application that is in the development stage. With vulnerability scanning tools being one of the mostly deployment, 91% of the respondents report that it takes an average of more than 3 hours to complete vulnerability scans, with half of the survey respondent reporting that the average remediation of a vulnerability in production requires at least 10 hours of work.

DevSecOps – What

"Shift-Left" introduces *continuous security* by integrating security in every stage of the application and infrastructure life cycles. Figure 1 shows the phases of the DevOps lifecycle. Other DevOps/DevSecOps references might have a more or less detailed version of the lifecycle or might use different naming for the phases, but the fundamental phases stay the same. The reader may refer to the DoD Enterprise DevSecOps Reference Design" [2] or the "OWASP DevSecOps Guideline"[3] for more information about the lifecycle.

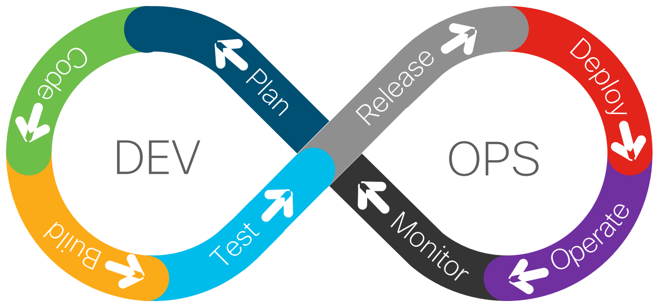


Figure 1.
DevOps Lifecycle

The following provides a short description of the DevOps phases shown in Figure 1:

- **Plan**: Define business values and requirements
- **Code**: Design and the create software code
- **Build**: Manage software builds and versions, and use automation tools to help compile and package code
- **Test**: Continuous testing to ensure optimal code quality
- **Release**: Manage release approvals and schedule
- **Deploy**: Release build into production

[1] https://www.contrastsecurity.com/hubfs/DocumentsPDF/The-State-of-DevSecOps_Report_Final.pdf

[2] https://dodcio.defense.gov/Portals/0/Documents/DoD%20Enterprise%20DevSecOps%20Reference%20Design%20v1.0_Public%20Release.pdf

[3] https://github.com/OWASP/DevSecOpsGuideline

- **Operate**: Manage software during production
- **Monitor**: Collect and analyze data from production.

Shifting left gives you the ability to automatically achieve visibility and security decisions earlier in the lifecycle; this is achieved by integrating security policies in the difference phases of a CI/CD pipeline so that automated actions are triggered when deviations are detected. For example, performing code scanning for known vulnerabilities when developers commit code, allowing you to make decisions early in the lifecycle.

DevSecOps does not only entail shifting security controls to the left of a CI/CD pipeline. Important aspects of continuous security in DevSecOps include connecting the outputs of different security controls deployed to form a coherent security posture view, in real-time, of the whole pipeline that can be presented and acted upon.

Continuous observability in DevSecOps, a common term that is used to refer to being able to gain knowledge of the state of assets such as code, images, etc. across the various phases of the pipeline, is critical to achieving real-time contextual awareness of the security risks. Figure 2 shows how real-time security and continuous monitoring and analytics are enclosed in a continuous observability model.

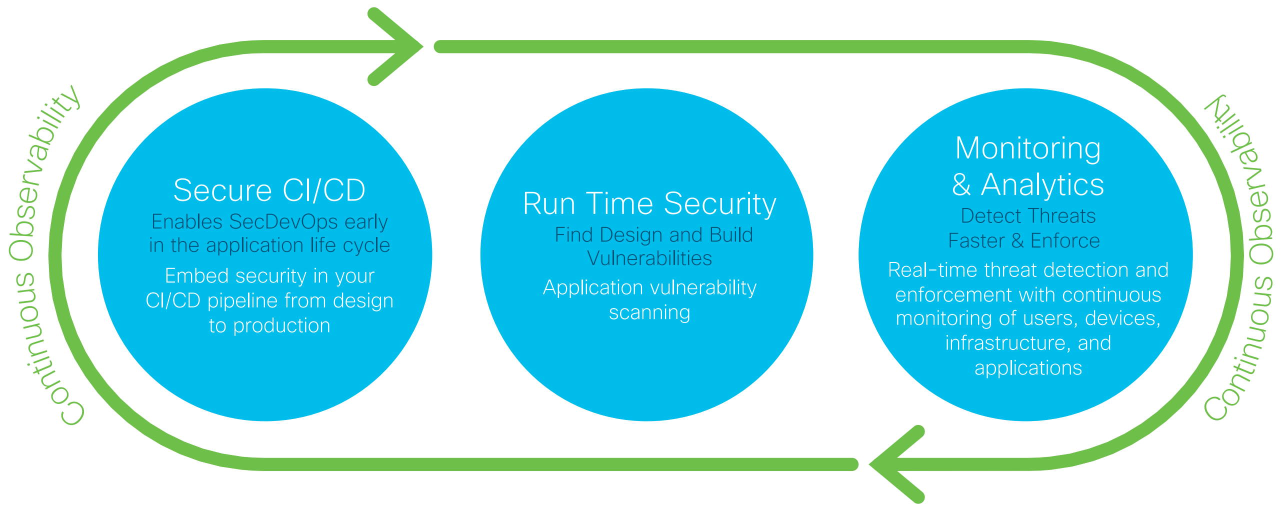

Continuous Observability
Secure CI/CD
Enables SecDevOps early in the application life cycle
Embed security in your CI/CD pipeline from design to production
Run Time Security
Find Design and Build Vulnerabilities
Application vulnerability scanning
Monitoring & Analytics
Detect Threats Faster & Enforce
Real-time threat detection and enforcement with continuous monitoring of users, devices, infrastructure, and applications
Continuous Observability

Figure 2.
Continuous Observability in the DevSecOps Lifecycle

DevSecOps – The Threat Landscape

The threat landscape that DevSecOps addresses covers a wide spectrum of security challenges that relate to both, applications and infrastructure components hosting these applications. The number of workloads, deployed for example as Kubernetes pods, that deliver applications in a microservice-based architecture would typically be large. These workloads require to communicate between themselves and with other internal and/or external systems and users, with possible Internet exposure. With that in mind, security-related questions asked by many security professionals include:

Q. Does a workload deployed in production contain security vulnerabilities? And if so, which of these workloads have been exposed to the Internet or to external parties?

Q. Can I continuously track security vulnerabilities of a workload from testing to deployment, and then in production?

Q. Are there users with excessive or inappropriate permissions in any of my cloud deployments?

Q. Can I know when the behavior of a workload starts deviating from what it has been designed for during the development phase?

Q. Can I build my security connection policy based on workloads and not just IP addresses?

Q. How easy and how fast can I deploy my intra- and inter-cluster security connection policy?

Q. Can I continuously monitor the security status of all my images posted to a registry?

Q. How can I achieve intra- and inter-cluster encryption without making changes to my applications?

Q. Can I know at what point of time and in which phase of the CI/CD pipeline a vulnerability was introduced to my code?

Q. Can I continuously monitor for privileged workloads?

Q. Does my Kubernetes cluster deployment meet security benchmark standards such as ones from the Center for Internet Security (CIS)[4][5]?

Q. Can I map my DevSecOps security status to a framework such as the MITRE ATT&CK framework?

Q. Can I continuously ensure that pods deployed have been validated in the CI pipeline first?

Q. Can I continuously identify and contain workloads deployed from unknown registries?

Q. Can I continuously ensure that only workloads with valid identifies are allowed?

Q. How can I ensure that my zero-trust strategy is enforced on workloads delivered by the CI/CD pipeline?

Q. I understand how security controls work in a container-based environment, but how can I deploy similar measures, if any, in a serverless deployment, and would the same controls apply?

Q. Can I have a dashboard (i.e. a pane of glass) that provides me answers for the above questions, even if I have a multi-cloud deployment (public and private)?

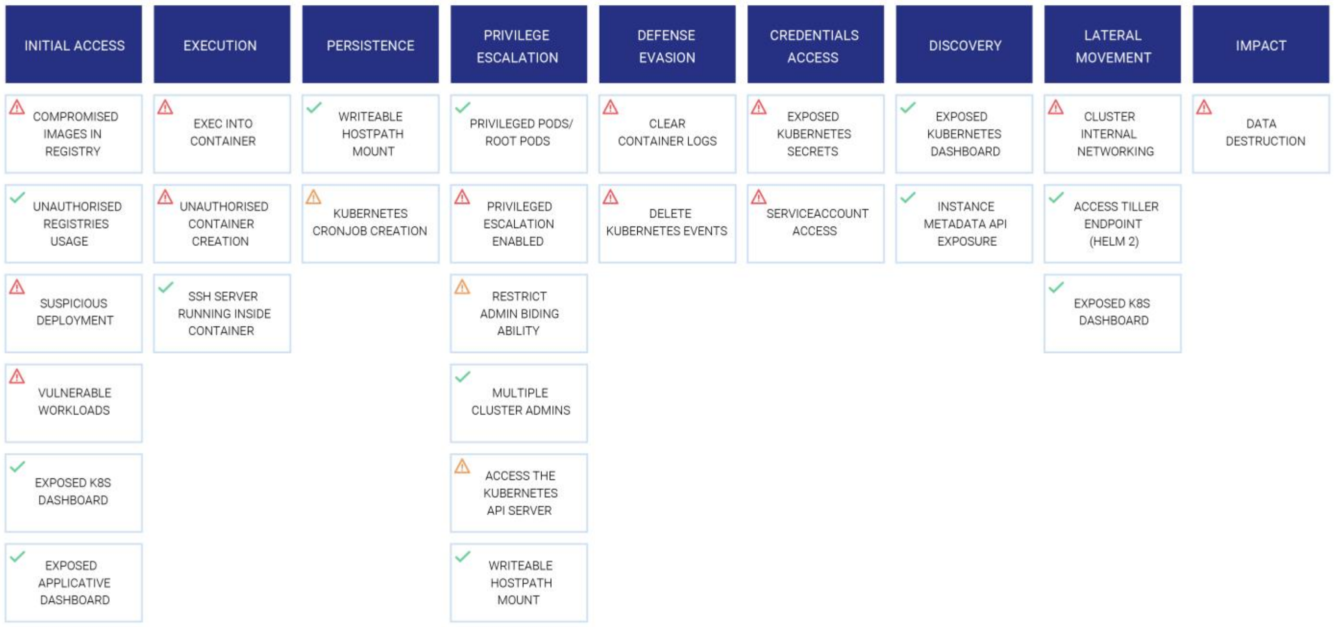
The above questions relate to different security aspects such as user access, vulnerability management, compliance, zero-trust, etc. A mature security team seeks to get answers to the above questions, and

[4] https://www.cisecurity.org/benchmark/docker

[5] https://www.cisecurity.org/benchmark/kubernetes

more, in real-time, accompanied with actionable information that allows them to activate incident response actions that might include automated investigation and perform automated containment and remediation activities, whenever possible.

Figure 3 shows a number of typical threats categorized under the MITRE© ATT&CK tactics that should be continuously monitored, covering initial access, execution, persistence, privilege escalation, defense evasion, credential access, discovery, lateral movement and impact, with the ability to customize the view per deployment and across all deployments.



INITIAL ACCESS	EXECUTION	PERSISTENCE	PRIVILEGE ESCALATION	DEFENSE EVASION	CREDENTIALS ACCESS	DISCOVERY	LATERAL MOVEMENT	IMPACT
⚠ COMPROMISED IMAGES IN REGISTRY	⚠ EXEC INTO CONTAINER	✓ WRITEABLE HOSTPATH MOUNT	✓ PRIVILEGED PODS/ ROOT PODS	⚠ CLEAR CONTAINER LOGS	⚠ EXPOSED KUBERNETES SECRETS	✓ EXPOSED KUBERNETES DASHBOARD	⚠ CLUSTER INTERNAL NETWORKING	⚠ DATA DESTRUCTION
✓ UNAUTHORISED REGISTRIES USAGE	⚠ UNAUTHORISED CONTAINER CREATION	⚠ KUBERNETES CRONJOB CREATION	⚠ PRIVILEGED ESCALATION ENABLED	⚠ DELETE KUBERNETES EVENTS	⚠ SERVICEACCOUNT ACCESS	✓ INSTANCE METADATA API EXPOSURE	✓ ACCESS TILLER ENDPOINT (HELM 2)	
⚠ SUSPICIOUS DEPLOYMENT	✓ SSH SERVER RUNNING INSIDE CONTAINER		⚠ RESTRICT ADMIN BIDING ABILITY				✓ EXPOSED K8S DASHBOARD	
⚠ VULNERABLE WORKLOADS			✓ MULTIPLE CLUSTER ADMINS					
✓ EXPOSED K8S DASHBOARD			⚠ ACCESS THE KUBERNETES API SERVER					
✓ EXPOSED APPLICATIVE DASHBOARD			✓ WRITEABLE HOSTPATH MOUNT					

Figure 3.
Monitoring and Reporting Threats in DevSecOps - Mapped to the MITRE© ATT&CK Matrix Tactics

DevSecOps – Security Controls

The yearly cloud landscape map published by the Cloud Native Computing Foundation (CNCF)[6] shows the year-over-year increase in the number of open source and commercial cloud tools. The landscape map is too large to be embedded in this whitepaper, demonstrating the increased demand and innovation happening in this area. It is worth highlighting the fact that these tools, some of which are in the early stages of deployment, would introduce their own set of vulnerabilities, which should be managed under DevSecOps.

[6] https://landscape.cncf.io

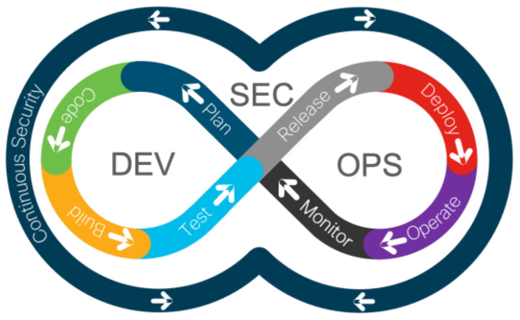
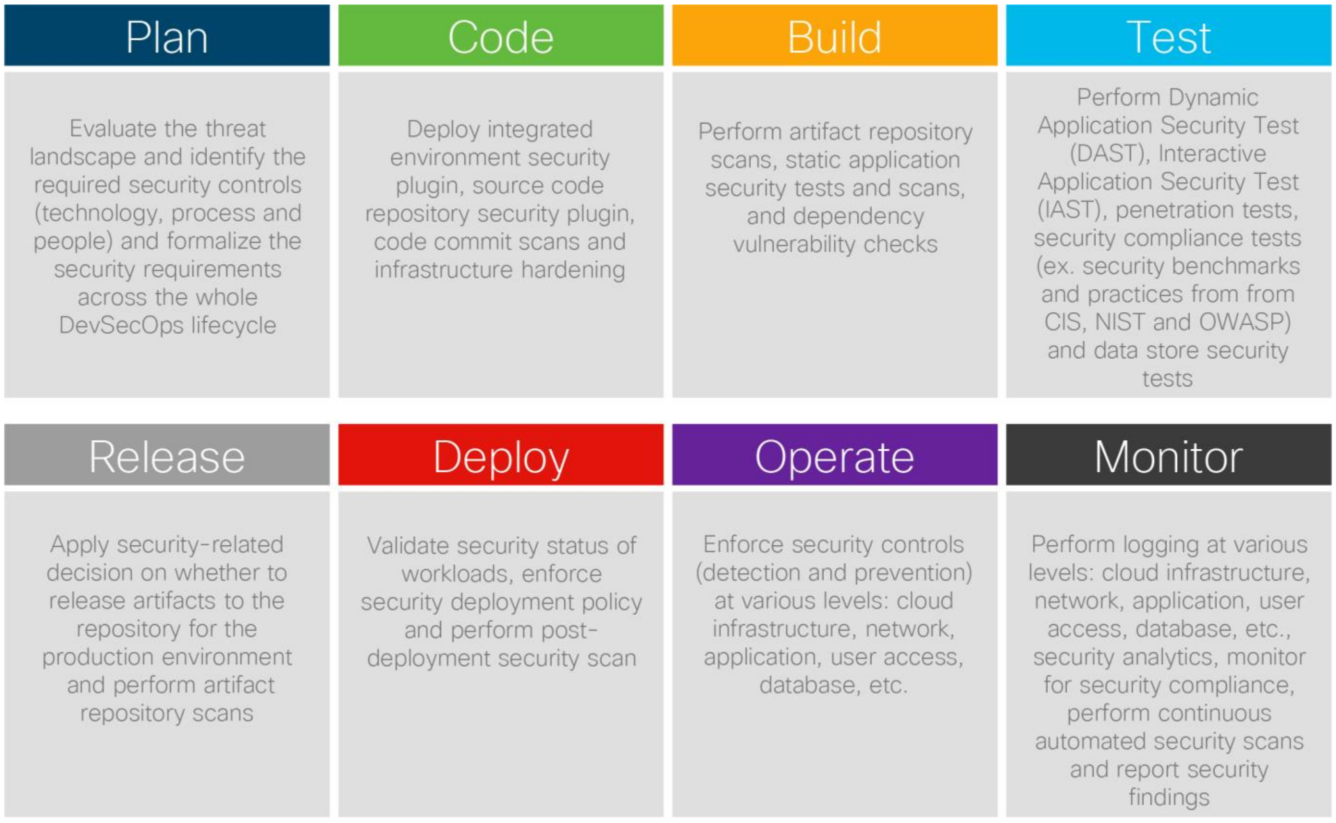


Figure 4.
Continuous Security in the DevOps Lifecycle

Securing DevOps requires a number of security controls in each phase and across the phases. Figure 5 shows the expected security controls in each phase.



Plan	Code	Build	Test
Evaluate the threat landscape and identify the required security controls (technology, process and people) and formalize the security requirements across the whole DevSecOps lifecycle	Deploy integrated environment security plugin, source code repository security plugin, code commit scans and infrastructure hardening	Perform artifact repository scans, static application security tests and scans, and dependency vulnerability checks	Perform Dynamic Application Security Test (DAST), Interactive Application Security Test (IAST), penetration tests, security compliance tests (ex. security benchmarks and practices from from CIS, NIST and OWASP) and data store security tests

Release	Deploy	Operate	Monitor
Apply security-related decision on whether to release artifacts to the repository for the production environment and perform artifact repository scans	Validate security status of workloads, enforce security deployment policy and perform post-deployment security scan	Enforce security controls (detection and prevention) at various levels: cloud infrastructure, network, application, user access, database, etc.	Perform logging at various levels: cloud infrastructure, network, application, user access, database, etc., security analytics, monitor for security compliance, perform continuous automated security scans and report security findings

Figure 5.
Security Controls in the DevOps Lifecycle

The security controls shown in Figure 6 must be automated within the CI/CD pipeline. For example, publishing core, creating a new image or publishing a helm chart should trigger, based on a pre-defined security policy, relevant security controls such as compliance checks against the CIS benchmark hardening standards, vulnerability scans, etc.

DevSecOps – A Unified Approach to Security

The security controls shown in Figure 5 should be context-aware and should not be delivered as silos. For example, you should be able to track an image from the moment it is published to a registry until it makes it to deployment. This way for example you have a full audit trace of all the security vulnerabilities and fixes introduced on the way. Another example is being able to monitor images and workloads for compliance (ex. to CIS benchmarks) starting from the moment they are in CI, all the way to runtime, so that changes impacting compliance are immediately captured and fed into your overall risk management process in general and to the CI/CD pipeline risk score in specific.

Figure 5 takes a technology approach by introducing technologies to the phases of DevOps. We propose a more holistic framework which encompasses and expands the security controls show in Figure 5, in which we consider the following:

- The critical DevSecOps dimensions around culture and people
- Continuous security traceability in which artefacts and actions are tracked throughout the CI/CD pipeline, regardless of when and how they are introduced.
- Real-time CI/CD pipeline security posture risk scoring
- High-value metrics, also referred to as Key Performance Indicator (KPIs), to monitor

The proposed framework, shown in Figure 6, takes a product-agnostic approach to achieving the above. We refer to it as Continuous Security for DevSecOps Framework (CSDF).

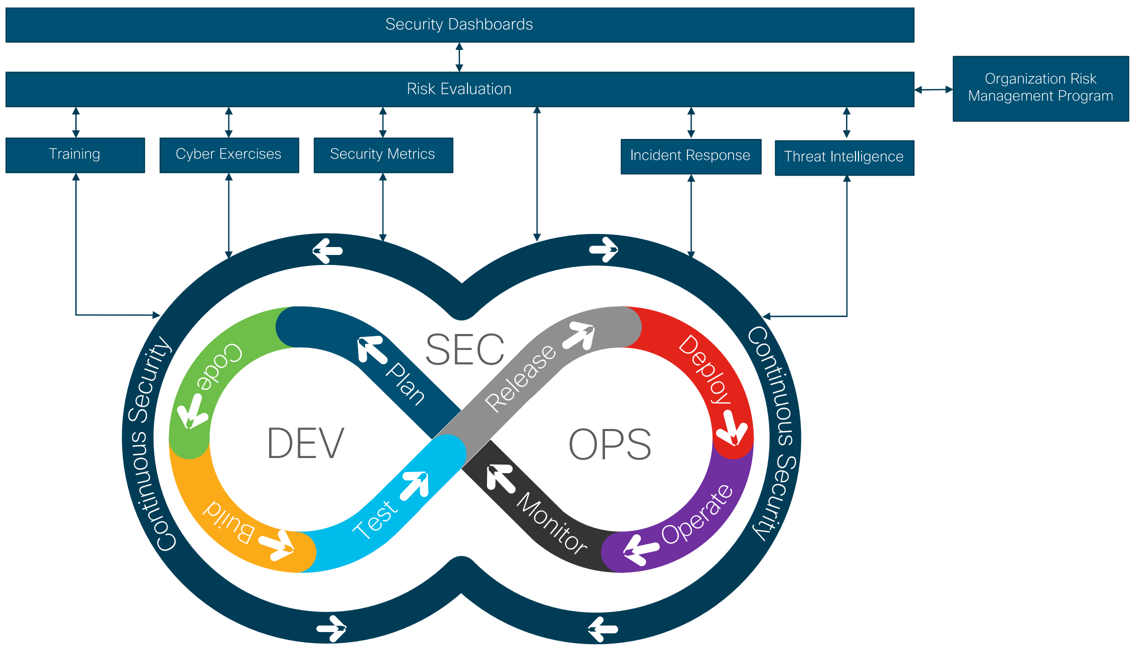


Figure 6.
Continuous Security for DevSecOps Framework (CSDF)

DevSecOps – Security High-Value Metrics

Metrics are important in measuring the success of a DevSecOps security program. They provide a mechanism that can present the outcome of investing in the DevSecOps adoption. In addition, monitoring and responding to metrics provide good insights on how you are performing against your DevSecOps strategy. Examples of relevant security high-value metrics include the following:

- Mean (time) to detect security incidents
- Mean (time) to investigate security incidents
- Mean (time) to contain security incidents
- Security compliance level (in %)
- (Number) of security incidents related to misconfigured DevSecOps security controls
- (Number) of security trainings attended by members from the DevSecOps team
- Integration of DevSecOps systems with the organization's SOC (in %) for event collection, security analytics and incident investigation and containment
- (Number) of security monitoring use cases deployed in relation to security threats that apply to the CI/CD pipeline
- (Number) of vulnerabilities reported in the development, release and runtime phases
- (Number) of re-occurring vulnerabilities during development, release or runtime phases
- (Number) of pods deployments bypassing the CI pipeline

DevSecOps – How about Serverless?

Serverless, also known as Function as a Service (FaaS), forces us to change the way we gain access to development and runtime artifacts. Although some standard security checks such a security code analysis and review still apply, the opportunity of inserting security in the code itself is critical to gain visibility due to the remote execution nature that serverless introduce. In addition, with serverless, integrating observability and security controls into the code provides the DevSecOps team with a much more granular capability to quarantine parts of the code, without the need to contain the whole workload in the case of container-based deployments, allowing for only safe execution paths to be executed in runtime, until the code is fixed and rolled out through the DevSecOps lifecycle.

DevSecOps – Opportunities

The evolving DevSecOps landscape provides challenges that can be converted to opportunities through the integration of innovations of products and services. Examples of opportunities that organizations might want to explore include the following:

- Cisco CX Cloud is an innovative customer experience platform powered by collaborative intelligence. The platform can extend its services to providing customers with a single source of truth about their security status/posture for DevSecOps, continuously integrating, ingesting and analyzing data from various sources such as Cisco Portshift, Cisco AppDynamics, Cisco Secure Workload (formerly Tetration), Cisco Secure Network Analytics (formerly Stealthwatch), Cisco Secure Cloud Analytics (Formerly Stealthwatch Cloud), Cisco SecureX and other open source and commercial tools.

- Having a multi-cloud security dashboard for DevSecOps is crucial to understanding and reporting the overall security posture of the whole DevSecOps lifecycle across public and private clouds.
- A DevSecOps security assessment service provides organizations with a detailed overview of their capabilities using the Continuous Security for DevSecOps Framework (CSDF), providing actionable information on how to address security issues tactically and strategically.
- DevSecOps strategy and roadmap development allows you to establish a solid DevSecOps practice, with a practical execution roadmap, covering various areas under the Continuous Security for DevSecOps Framework (CSDF).
- Extending managed security monitoring to the area of DevSecOps allows organizations to fully or partially outsource security monitoring capabilities to a managed detection and response providers, allowing organizations to focus on their core business, while gaining access to security subject matter experts in the area of security detection and response for DevSecOps.
- In this evolving landscape, organizations should maintain a DevSecOps radar that can prepare them for new innovations in DevSecOps practices and technologies.
- Evaluate your infrastructure as code deployments against technical best practices such the ones from the Center for Internet Security (CIS).
- A serverless security design and assessment service provides organizations with an overview of their serverless deployments using the Continuous Security for DevSecOps Framework (CSDF).
- Automated security threat exposure assessment service of a CI/CD pipeline is an active engagement in which an security assessment report is generated showing the various security issues and vulnerabilities discovered in the environment.
- Extending zero-trust assessment and design to cover workloads and serverless deployments across multi-cloud, public and private.
- Continuous observability design and implementation work allows you to deploy the right controls across the CI/CD pipeline to answer questions highlighted earlier in the whitepaper.
- DevSecOps SOC use cases design and deployment, allowing the effective integration of DevSecOps with SOC for rea-time security monitoring and cyber threat hunting.
- A DevSecOps cyber range platform allows your team to gain invaluable insights on the threat landscape, practice security investigations and apply continuous security through automation.
- Organizations should have access to threat intelligence information that covers latest tactics, techniques and procedures (TTPs) used to exploit applications and infrastructure components.

Conclusion

DevSecOps is not just a collection of security controls thrown over a CI/CD pipeline to tactically manage security threats; DevSecOps should be thought of as a framework that covers culture, methodology, practices and technologies, all integrated to strategically address a continuously changing and growing threat landscape.

Cisco can help you in DevSecOps journey with our Continuous Security technologies and services.

Authors

Nadhem AlFardan, Principal Architect, CX TTG, Cisco Systems Inc

Chocks Ramiah, Software Architect, CX TTG, Cisco Systems Inc

Contributors

Vijay Raghavendran, Distinguished Engineer, CX CTO, Cisco Systems Inc

Abdallah Atie, Software Architect, CX TTG, Cisco Systems Inc

Ariel Shuper, Engineering Product Manager, Cisco Systems Inc

© 2021 Cisco and/or its affiliates. All rights reserved.