Infosys cobalt

WHITE PAPER

DAST AUTOMATION FOR SECURE, SWIFT DEVSECOPS CLOUD RELEASES

Abstract

DevSecOps adoption in the cloud goes well beyond merely managing continuous integration and continuous deployment (CI/CD) cycles. Its primary focus is security automation. This white paper examines the barriers organizations face when they begin their DevSecOps journey, and beyond. It highlights one of the crucial stages of security testing known as Dynamic Application Security Testing (DAST). It explores the challenges and advantages of effectively integrating DAST into the CI/CD pipeline, on-premises and in the cloud. The paper delineates the best practices for DAST tool selection and chain set-up, which assist in shift-left testing and cloud security workflows that offer efficient security validation of deployments with risk-based prompt responses.

Infosys®
Navigate your next

Background

Traditional security practices involve security personnel running tests, reviewing findings, and providing developers with recommendations for modifications. This process, including threat modeling, conducting compliance checks, and carrying out architectural risk analysis and management, is time-consuming and incongruous with the speed of DevOps. Some of these practices are challenging to automate, leading to a security and DevOps imbalance. To overcome these challenges, many organizations have shifted to an agile DevOps delivery model. However, this exerts significant pressure on DevOps to achieve speed with security as part of the CI/CD pipeline. As a result, release timelines and quality have been impacted due to the absence of important security checks or the deployment of vulnerable code under time pressure.

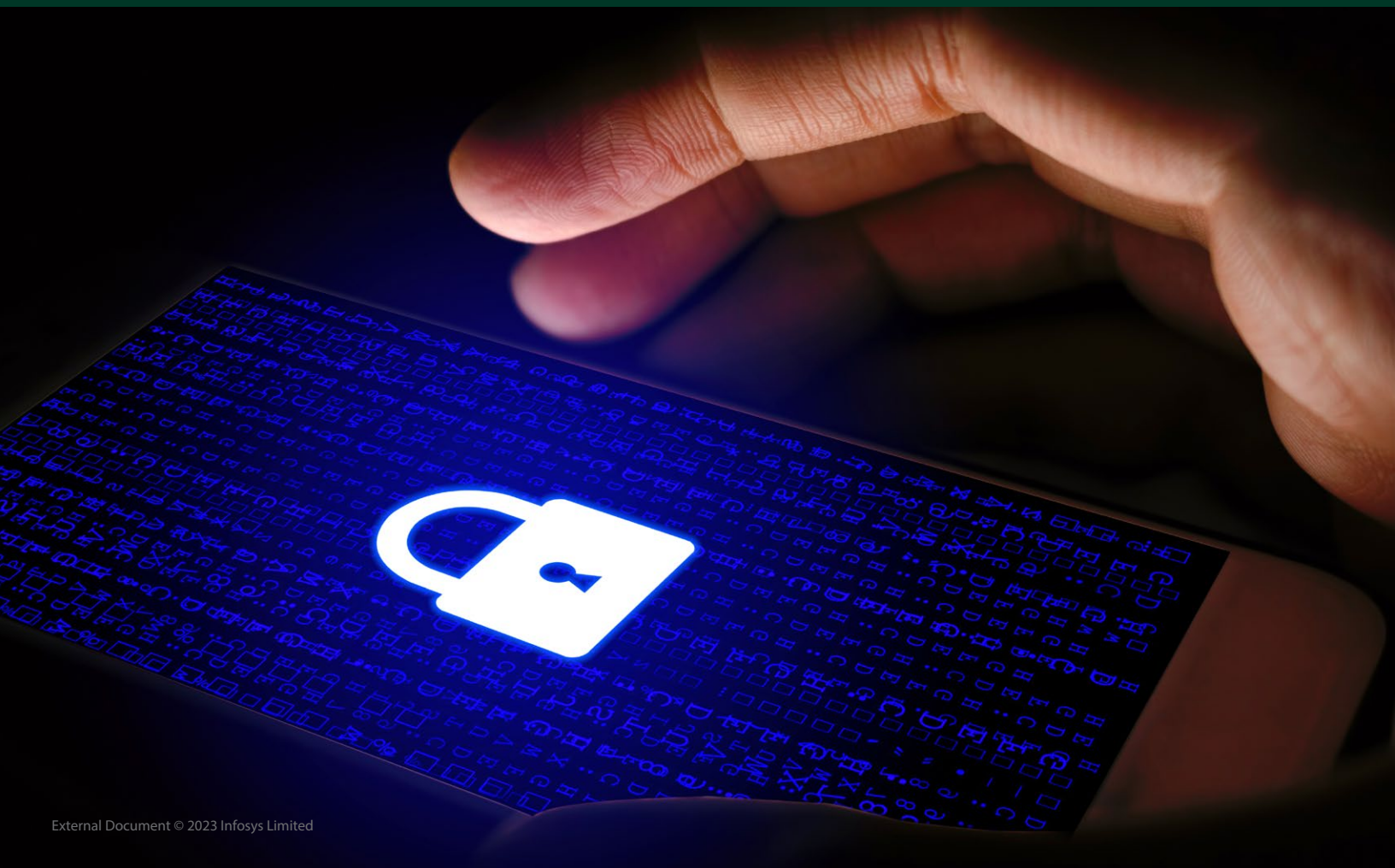
Even as DevOps was evolving, the industry concurrently fast-tracked its cloud transformation roadmap. Most organizations shifted their focus to delivering highly scalable applications built on customized modern architectures with 24/7 digital services. These applications include a wide-ranging stack of advanced tiers, technologies, and microservices, backed by leading cloud platforms such as AWS, GCP, and Azure.

Despite the accelerated digital transformations, a large number of organizations continue to harbor concerns about security. The year-end cybercrime statistics provide good reason to do so:

1. The global average cost of a data breach is an estimated US $4.35 million, as per IBM's 2022 data breach report[1]
2. Cybercrime cost the world US $7 trillion in 2022 and is set to reach US $10.5 trillion by 2025, according to Cybersecurity Ventures[2]

Evidently, security is an important consideration in cloud migration planning. Speed and agility are imperatives while introducing security to DevOps processes. Integrating automated security checks directly into the CI/CD pipeline enables DevOps to evolve into DevSecOps.

DevSecOps is a flexible collaboration between development, security, and IT operations. It integrates security principles and practices into the DevOps life cycle to accelerate application releases securely and confidently. Moreover, it adds value to business by reducing cost, improving the scope for innovation, speeding recovery, and implementing security by design. Studies project DevSecOps to reach a market size of between US $20 billion to US $40 billion by the end of 2030.



DevSecOps implementation challenges

As enterprises race to get on the DevSecOps bandwagon, IT teams continue to experience issues:

- *60% find DevSecOps technically challenging* [3]
- *38% report a lack of education and adequate skills around DevSecOps* [3]
- *94% of security and 93% of development teams report an impact from talent shortage* [1]

Some of the typical challenges that IT teams face when integrating security into DevOps on-premise or in the cloud are:

People/culture challenges:

- Lack of awareness among developers on secure coding practices and processes
- Want of collaboration and cohesive skillful teams with development, operations, and security experts

Process challenges:

- Security and compliance remain postscript
- Inability to fully automate traditional manual security practices to integrate into DevSecOps
- Continuous security assessments without manual intervention

Tools/technology challenges:

- Tool selection, complexity, and integration problems
- Configuration management issues
- Prolonged code scanning and consumption of resources

Solution

Focusing on each phase of the modern software development life cycle (SDLC) can help strategically resolve DevSecOps implementation challenges arising from people, processes, and technology. Integrating different types of security testing for each stage can help overcome the issues more effectively (Figure 1).

PLAN Requirements	**CODE** Code Repository	**BUILD** CI Server	**TEST** Integration Testing	**RELEASE** Artifact Repository	**DEPLOY** **CD** Orchestration	**OPERATE** Monitor
Threat Modelling	**Software Composition Analysis and Secret Management**	**Secure Code Analysis and Docker Linting**	**Dynamic Application Security Testing**	**Network Vulnerability Assessments**	**System/Cloud Hardening**	**Cloud Configuration Reviews**

Figure 1: Modern SDLC with DevSecOps and Types of Security Testing

What is DAST?

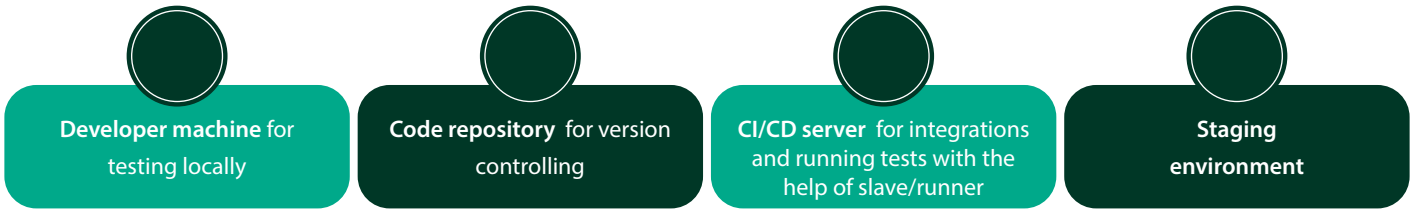
DAST is the technique of identifying the vulnerabilities and touchpoints of an application while it is running. DAST is easy even for beginners to get started on without in-depth coding experience. However, DAST requires a subject matter expert (SME) in the area of security to configure and set up the tool. An SME with good spidering techniques can build rules and configure the correct filters to ensure better coverage, improve the effectiveness of the DAST scan, and reduce false positives.

Best practices to integrate DAST with CI/CD

The last few years have shown that next-generation CX requires heavy doses of perseverance and attitudinal focus. At Infosys, we have extended this to the way we deliver projects by relying on a few key cultural principles:

- Integrate DAST scan in the CI/CD production pipeline after provisioning the essential compute resources, knowing that the scan will take under 15 minutes to complete. If not, create a separate pipeline in a non-production environment
- Create separate jobs for each test in the case of large applications. E.g., SQL injection and XSS, among others
- Consider onboarding an SME with expertise in spidering techniques, as the value created through scans is directly proportional to the skills exhibited
- Roll out security tools in phases based on usage, from elementary to advanced
- Fail builds that report critical or high-severity issues
- Save time building test scripts from scratch by leveraging existing scripts from the functional automation team
- Provide links to knowledge pages in the scan outputs for additional assistance
- Pick tools that provide APIs
- Keep the framework simple and modular
- Control the scope and false positives locally instead of maintaining a central database
- Adopt the everything-as-a-code strategy as it is easy to maintain

Besides adopting best practices, the CI/CD environment needs to be test-ready. A basic test set-up includes:



There can be several alternatives to the set-up based on the toolset selection. The following diagram depicts a sample (see Figure 2).

Figure 2: DevSecOps Lab Set-up

Right tool selection

With its heavy reliance on tools, DevSecOps enables the automation of engineering processes, such as making security testing repeatable, increasing testing speed, and providing early qualitative feedback on application security. Therefore, selecting the appropriate security testing tools for specific types of security testing and applying the correct configuration in the CI/CD pipeline is critical.

Challenges in tool selection and best practices

Common pitfalls

- Lack of standards in tool selection
- Security issues from tool complexity and integration
- Inadequate training, skills, and documentation
- Configuration challenges

Best practices in tool selection

- Expert coverage of tool standards
- Essential documentation and security support
- Potential for optimal tool performance, including language coverage, open source or commercial options, the ability to ignore issues, incident severity categories, failure on issues, and results reporting feature
- Cloud technology support
- Availability of customization and integration capabilities with other tools in the toolchain
- Continuous vulnerability assessment capability

Best practices in tool implementation

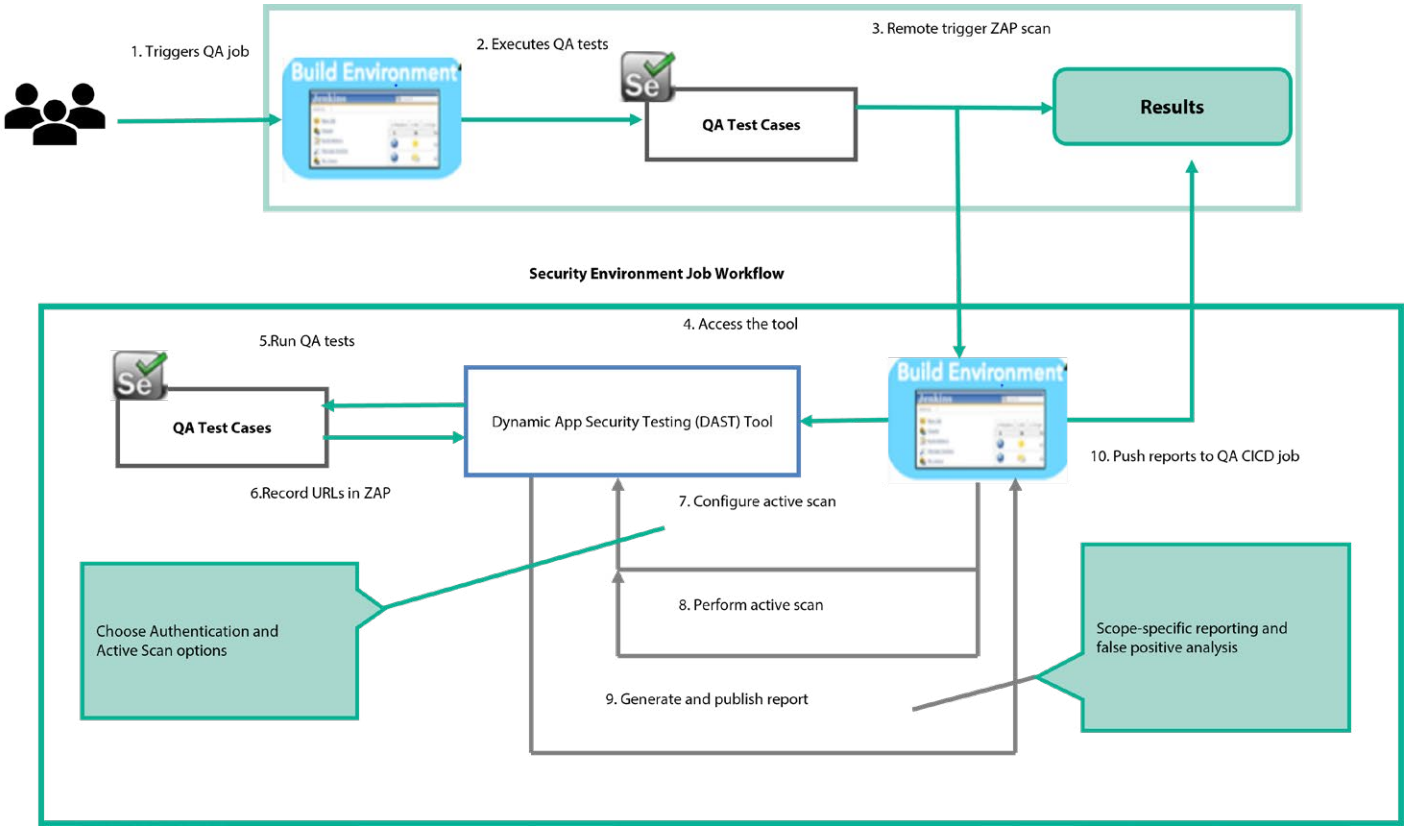
- Create an enhanced set of customized rules for tools to ensure optimum scans, and reliable outcomes
- Plan incremental scans to reduce the overall time taken
- Use artificial intelligence (AI) capabilities to optimize the analysis of vulnerabilities reported by tools
- Aim for zero-touch automation
- Consider built-in quality through automated gating of the build against the desired security standards

After selecting the CI/CD and DAST tools, the next step is to set up a pre-production or staging environment and deploy the web application. The set-up enables DAST to run in the CI/CD pipeline as a part of integration testing. Let us consider an example using the widely available open-source DAST tool, Zed Attack Proxy (ZAP). Some of the key considerations for integrating DAST in the CI/CD pipeline using ZAP (see Figure 3) are listed below:

- Test on the developer machine before moving the code to the CI/CD server and the Gitlab CI/CD
- Set up the CI/CD server and Gitlab. Ensure ZAP container readiness with Selenium on Firefox, along with custom scripts
- Reuse the functional automation scripts, only modifying them for security testing use cases and data requirements
- Push all the custom scripts to the Git server and pull the latest code. Run the pipeline after meeting all prerequisites

Some of the key considerations for integrating DAST in the CI/CD pipeline using ZAP (see Figure 3) are listed below:

- Test on the developer machine before moving the code to the CI/CD server and the Gitlab CI/CD
- Set up the CI/CD server and Gitlab. Ensure ZAP container readiness with Selenium on Firefox, along with custom scripts
- Reuse the functional automation scripts, only modifying them for security testing use cases and data requirements
- Push all the custom scripts to the Git server and pull the latest code. Run the pipeline after meeting all prerequisites



DevSecOps with DAST in the cloud

Integrating DAST with cloud CI/CD requires a different approach.

Approach:

- Identify, leverage, and integrate cloud-native CI/CD services, continuous logging and monitoring services, auditing, and governance services, as well as operation services with regular CI/CD tools – mainly DAST
- Control all CI/CD jobs with server and slave architecture by using containers, such as Docker, to build and deploy applications as cloud orchestration tools.

An effective DAST DevSecOps in cloud architecture appears as shown in Figure 4:

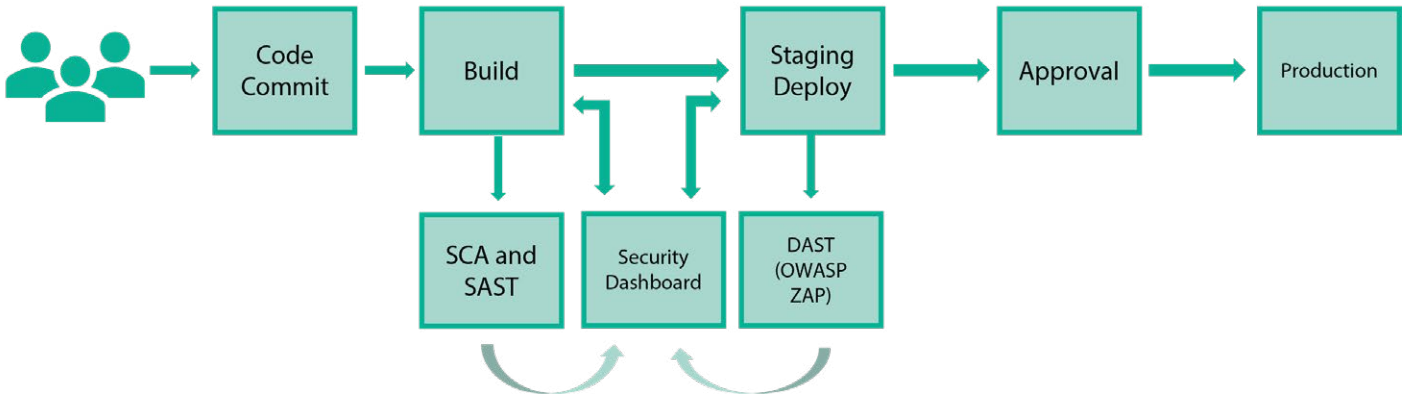


Figure 4: DAST DevSecOps in Cloud Workflow

Key steps

1. The user commits the code to a code repository
2. The tool builds artifacts and uploads them to the artifact library
3. Integrated tools help perform the SCA and SAST tests
4. Reports of critical/high-failure vulnerabilities from the SCA and SAST scans go to the security dashboard for fixing
5. Code deployment to the staging environment takes place if reports indicate "no or ignore vulnerabilities"
6. Successful deployment triggers a DAST tool, such as the OWASP ZAP, for scanning
7. User repeats steps 4 to 6 in the event of a vulnerability detection
8. If no vulnerabilities are reported, the workflow triggers an approval email.
9. Receipt of approval schedules automatic deployment to production

Best practices

- Control access to pipeline resources using identity and access management (IAM) roles and security policies
- Encrypt data at transit and rest always
- Store sensitive information, such as API tokens and passwords, in the Secrets Manager

Conclusion

DevOps is becoming a reality much faster than we anticipate. However, there should be no compromise on security testing to avoid delayed deployments and the risk of releasing software with security vulnerabilities. Successful DevSecOps requires integrating security at every stage of DevOps, enabling DevOps teams on security characteristics, enhancing the partnership between DevOps teams and security SMEs, automating security testing to the extent possible, and shift-left security for early feedback. By leveraging the best practices recommended in this paper, organizations can achieve a more secure and faster release by as much as 15%, both on-premises and in the cloud.

About the authors

Kedar J Mankar

Kedar J Mankar is an Infosys global delivery lead for Cyber Security testing with Infosys. He has extensive experience across different software testing types. He has led large size delivery and transformation programs for global Fortune 500 customers and delivered value through different COEs with innovation at core. He has experience working and handling teams in functional, data, automation, DevOps, performance and security testing across multiple geographies and verticals.

Amlan Sahoo

Amlan Sahoo has an overall 27+ years in IT industry in application development and testing. He is currently the head of Cyber Security testing division. He has a proven track record in managing and leading transformation programs with large teams for Fortune 50 clients, managing deliveries across multiple geographies and verticals. He also has 4 IEEE and 1 IASTED publications to his credit on bringing efficiencies in heterogeneous software architectures.

Vamsi Kishore

Vamsi Kishore Sukla is a Security consultant with over 8 years of professional experience in the security field, specializing in application security testing, cloud security testing, network vulnerability assessments following OWASP standards and CIS benchmarks. With a deep understanding of the latest security trends and tools, he provides comprehensive security solutions to ensure the safety and integrity of organization and clients.

References

1. https://www.cobalt.io/blog/cybersecurity-statistics-2023
2. https://cybersecurityventures.com/boardroom-cybersecurity-report/
3. https://strongdm.com/blog/devsecops-statistics

For more information, contact askus@infosys.com

Infosys®
Navigate your next