QUALIZEAL

Why is Shift Left Security Testing Important for your organization in 2023

Table Of Content

"Shift-left security is about reducing the time to identify and mitigate risks by enabling developers with the right tools and training to make secure coding decisions."

-Paul Farrington, EMEA CTO at Veracode.

Don't leave security at the gates; Shift left and secure the code from day one!

Terminologies:

DevOps: A set of practices that combine software development (Dev) and information technology operations (Ops) to shorten the systems development life cycle and provide continuous delivery with high software quality.

DevSecOps: A software development approach that emphasizes integrating security throughout the entire software development lifecycle (SDLC) to ensure secure and reliable software delivery.

DevOps: A software development approach designed specifically for cloud environments. It involves building and deploying applications as a collection of small, independent services that can run and scale dynamically in the cloud.

Cloud-native software development: A set of practices that combine software development (Dev) and information technology operations (Ops) to shorten the systems development life cycle and provide continuous delivery with high software quality.

Security quality guardrail: Automated controls that enforce security policies and best practices throughout the software development lifecycle.

Git-based development workflow: Software development processes that rely on the Git version control system to manage code changes and collaborate on software development projects.

Buffer overflow: Software vulnerability occurs when a program writes more data to a buffer than it can hold, causing the extra data to spill over into adjacent memory locations and potentially allowing an attacker to execute arbitrary code or crash the program.

SQL injection: A web application vulnerability that allows an attacker to insert malicious SQL code into a web application's input fields. This can enable the attacker to access or modify sensitive data, such as user login credentials or personal information.

Cross-site scripting (XSS): A type of security vulnerability that allows attackers to inject malicious scripts into web pages viewed by other users. This can lead to the theft of sensitive information, such as login credentials or personal data, and even take over of user accounts.

In a Sauce Labs Survey, 61% of respondents reported to have multiplied their investment in shift-left testing practices over the last year.

Software security has become a growing concern in the modern digital landscape as organizations continue to advance their digital transformations and deposit more assets in the cloud. Traditional approaches to application security, where security warnings are only addressed during the final phases of the software development lifecycle, are no longer sufficient.

Conducting security audits later creates a time crunch, as developers work until the last minute and leave the security team with little time to confirm the code is safe and secure. As a result, the emerging approach of Shift Left security is gaining momentum, embedding security into the earliest phases of the software/application development process rather than in the testing phase.

"The sooner you find the bugs, the cheaper they are to fix."

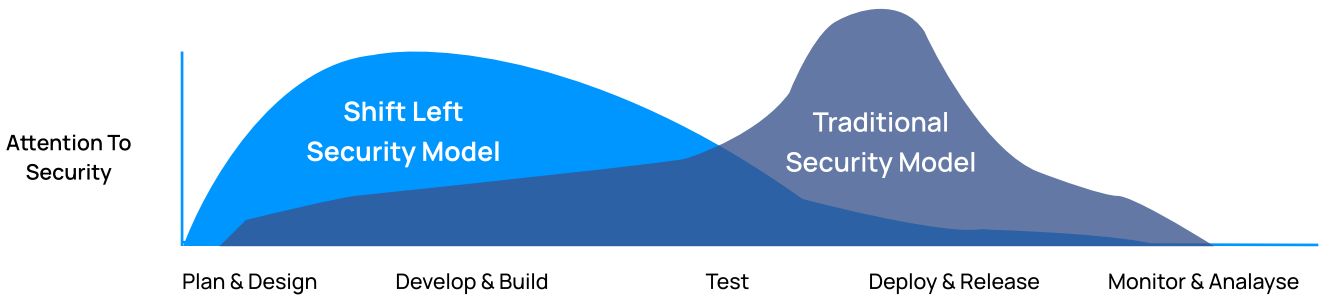
-Steve McConnell.

What is Shift-Left Security Testing?

The shift-left Security testing is an approach that integrates security measures into the programming stage to identify and address the proliferation of security flaws at the outset of the software development lifecycle (SDLC).

Shift-Left approach involves collaboration between development and security teams and the use of automated testing tools and techniques to detect and mitigate potential security incidents and data breaches. It requires continuous monitoring and feedback loops for allowing organizations to improve and refine their security posture over time.

Traditionally, security is addressed only in the final phase before release, which restricts visibility to inspect that code is secure. When security leaks are exposed, the release is delayed, and the development team has to scramble to make corrections. This results in increased expenses and hampers the progress of software releases, and the chances of overlooking security threats are always substantial

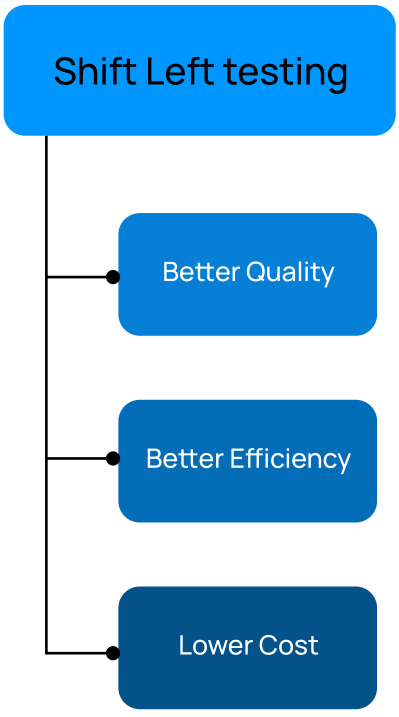


- **A Forrester survey states that organizations adopting shift-left security testing saw a 38% reduction in the number of vulnerabilities that made it to production.**

- **Gartner predicts that by 2024, 60% of enterprises will implement shift-left testing to improve application security and reduce the risk of security breaches.**

Shift-Left security testing is a core element of DevSecOps that guides organizations to consider security a key aspect of the development process and gives developers the ability to deliver secure, reliable solutions without slumping down the deployment process.

Shift-left does not translate to "security testing is not required in post-production." It only implies prioritizing security measures at the inception of SDLC as it drastically reduces the security burdens around cloud-native software development. A Pre-launch review is still irreplaceable for double assessment, but it would be less time-taking if the measures were integrated early.



Benefits of Shift-Left Security Testing:

The importance of shift-left security testing lies in the growing interest around application security and workload protection as organizations advance their digital transformations and place more of their assets in the cloud.

The speed of software releases, the use of cloud-based services, the incorporation of automation into the software building phase, and the rate of innovation in the development toolchain all contribute to eroding application security.

Streamlined automation process:

Automated processes in shift-left security testing result in fewer human errors and production issues. Automation allows for multiple tests to be carried out simultaneously, which increases test coverage. Additionally, testers are freed up to focus on other tasks, ameliorating the overall productivity of the team.

Escalated quality and efficiency:

By conducting unit and integration testing primarily in the development process, you can catch up on potential roadblocks and bottlenecks to prevent them from becoming more extensive and expensive later. This leads to a higher-quality product that meets customer expectations and reduces the risk of critical security vulnerabilities.

Spending more time on testing throughout the development improves code quality, makes your pipeline more stable, and enables quicker, more secure releases. Moreover, it liberates quality engineers and security professionals to focus on more critical activities, such as identifying and debugging problems.

Accelerated Delivery Speed:

Shift-left security testing decreases the time between releases by enabling DevOps and security to work in correspondence. It replaces manual processes with automated security checks causing advancement in the software quality, meaning teams invest more time to identify and resolve issues earlier in the development phase.

Automating your security pipeline can help you deliver a better quality and more trustworthy product with speed and agility to the market. Fewer delays between releases help organizations swiftly deliver the products and simultaneously improve time-to-market and customer satisfaction.

Reduction in Budget:

Shift-left security testing is a proactive approach that emphasizes integrating security testing early rather than waiting until the end. Detecting critical changes in the initial phase saves a lot of time and prevents security breaches and associated expenses such as legal fees, loss of revenue, and damage to reputation. It saves overall development, maintenance, testing, and compliance cost.

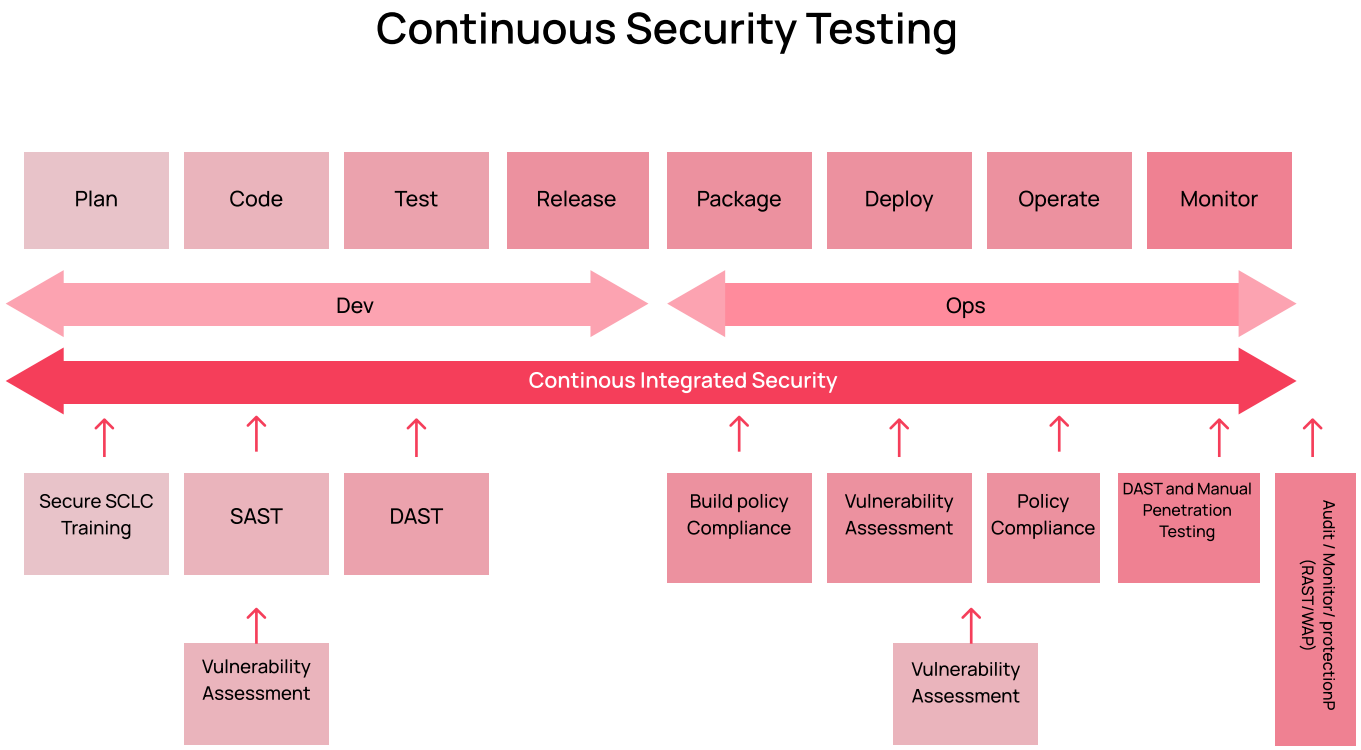
Secured Application Development:

Shift left security testing supports instant application delivery because there is no pause in coding while security performs its reviews. Continuously testing throughout the development process catches security flaws sooner, and fixes are minor in scale and less time-consuming. This saves DevOps and security teams from frustration and late nights and ensures that new user-pleasing features are deployed faster.

Additionally, shift-left security testing helps keep the known vulnerabilities away prior to the product deployment, keeping the cost of remediation at the lowest possible. It is far less costly to fix security issues earlier in the development phase than after going live.

How To Get Started with Shift-Left Security Testing?

The shift-left testing approach may vary depending on various factors, such as the size of the organization, the number of security personnel, current processes, and product risk exposure. However, following some practical steps can help teams make a great start toward introducing shift-left testing into the development process.



1. Define Strategy and Policy for Shift-Left Security:

Defining a shift-left security strategy is the first step in implementing an effective security program. It is highly suggested to document what shift-left security means to your organization and outline the vision, ownership/responsibility, milestones, and metrics. This will provide your teams with a clear roadmap, so they understand what success looks like.

One key aspect of security policies is the establishment of coding standards. These standards help ensure all developers are on the same page and can review code more quickly. The shift-left security strategy should be concisely documented, ideally no more than one page. The documentation will likely evolve over time, so it is important to focus on iteration rather than perfection.

2. Understand Development Processes in the Organization:

The goal of this step is to get a grasp on how and where software is created in your organization. This is important to understand because it allows the security team to know where they can move security closer to development. The process may be straightforward for small and medium-sized organizations, but it can be challenging for larger organizations that have multiple vendors or development teams in different locations.

In order to understand the software flow, it is essential to identify the developer profile, the process used, and the technology involved. This may also include reviewing the CI/CD toolchain. For larger organizations, it is best to start at a macro level and then drill down to the individual business units.

3. Inducting Testing Team Early in the SDLC:

Once security policies are in place, the next step is to implement testing early in the software development lifecycle (SDLC). By understanding your current practices, you can identify small steps to place testing earlier in the process. Adopting an agile SDLC approach that works with small code increments and includes development and testing phases for each sprint can ensure that every small feature gets covered with relevant tests. If a drastic switch is not possible, then the development team can agree to write unit tests for each feature they develop.

One approach to shifting testing to the left is running the four tests outlined below, which represent a minimal commitment while reducing friction within the security and quality assurance phases.

Unit testing:
Unit testing is a type of testing that verifies the performance of a single method, function, or class. Developers run unit tests on the smallest testable unit of software, such as procedures,interfaces, or classes. The primary goal of unit testing is to ensure that the individual units of code are working as expected before integrating them into the larger codebase. If a unit test fails to return the correct value when fed input, it is marked as a failure, and the code is deemed unfit for further use.

Basic functionality:
Basic functionality testing evaluates whether all aspects of the code work correctly rather than focusing on a single output examined in unit tests. For example, functionality tests check whether the application displays correctly, works outside of development environments, allows users to submit data without crashing the application, and supports each feature of the called API. Performing these tests can save downstream time in QA or security.

Code review:
Code review is a manual verification process performed by a peer to ensure code accuracy, consistency, and quality. Code reviews accelerate the coding process and help identify errors in the code. Many

developers overlook errors in their own code, but an unbiased co-worker can often spot inconsistencies. A quick second layer of manual verification can go a long way in cleaning up source code. It's also recommended to check for security issues in the code during a code review.

Static code:
Static code analysis is a type of testing that uses automated tools to scan for errors in the code without executing it. These tests examine code structure and ensure that the code meets standard criteria. Static code analysis tools can detect commonly identified security concerns (SAST), programming errors, standard code violations, syntax errors or anomalies, and undefined values. Static code analysis helps ensure that known security issues do not impede the CI/CD pipeline at the security stage.

4. Implement Security Quality Guardrails:

Quality assurance has always been a part of the software development lifecycle, but it has not traditionally included security. It became relevant to change this by implementing security quality guardrails. Security teams can start small by providing development teams with simple and effective tools that become a part of the daily development routine. These tools can help provide feedback, identify security issues throughout the development phase, and speed up the entire cycle.

An example of automated security is security gating on pull requests, which is often used as the backbone of Git-based development workflows. Pull requests provide an easy collaboration point within applications as developers commit and merge changes into code repositories. Automation tools can test pull requests for security issues and license issues before the code is merged.

5. Train Developers in Secure Coding Practices:

- According to one survey, 19% of developers were unfamiliar with the OWASP Top 10, which highlights the importance of this step.

- A recent survey by GitLab found that only 25% of developers think their organization's security practices are good, indicating the need for ongoing training and improvement.

Developers know how to code but may not know how to do it securely. It is mandatory to verify that developers understand the security context to produce secure code in the first place. This requires assessing the current skill level of your development team and providing continuous training in secure coding practices. Assigning developers with safe coding guidelines and security standards for the project assures your production status remains intact.

Shift-Left Security Testing Tools:

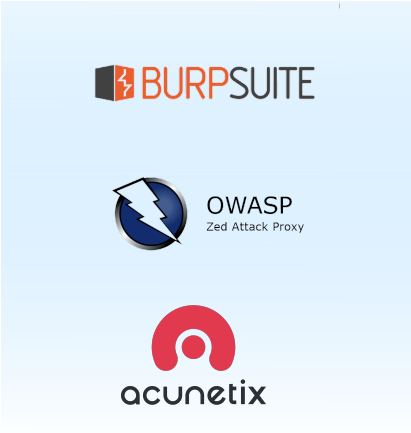
Shift-left testing and tools are vital for organizations to streamline the software development process and minimize the occurrence of bugs and security issues. These tools are designed to detect common vulnerabilities and classify the results, which helps in identifying the trends and patterns of the current system.

One of the significant benefits of using left-shift tools is the rectification of application layer security. Breaches often occur by exploiting the low-hanging fruits that are neglected. With the help of shift-left security tools, developers can identify known threats in the build and release phases, allowing them to fulfill the objective of shift-left testing.

Software Composition Analysis (SCA)

SCA tools are indispensable for identifying vulnerabilities and tracking open-source software components and libraries that an application uses. The SCA tools automatically collect the inventory of all the ingredients used in the application, including version information and licensing terms. This helps developers and security teams identify and fix bugs and errors in third-party libraries before they become a potential risk for the application.

Examples: Black Duck by Synopsys, WhiteSource, Nexus Lifecycle



Dynamic Application Security Testing (DAST)

DAST tools scan the running application to identify potentially endangered practices. These tools analyze requests, responses, scripts, interfaces, injections, authentication, and sessions to identify security breaches. DAST uses fuzzing techniques to send a variety of input values to the application and observe how it responds. It identifies potential weaknesses in the application's logic and can help detect issues with configuration settings.

Examples: Burp Suite, OWASP ZAP, Acunetix

Static Application Security Testing (SAST)

SAST tools examine the application's source code for potential security attacks, such as buffer overflows, SQL injection, and cross-site scripting (XSS). The tools generate a report of identified issues, including the severity and location of the possible attack. Developers can then use this information to fix the issues before the code is released into production.

Examples: Checkmarx, Veracode, SonarQube

Interactive Application Security Testing (IAST)

IAST combines both static and dynamic analysis methods to perform testing on the data flow and application. IAST uses pre-defined test cases and performs dynamic analysis to identify potential security weaknesses while the application is running. The tool recommends additional test cases based on the results and provides real-time feedback to the developer. This approach is highly methodical and provides automatic feedback to developers during software development.

Examples: Contrast Security, Hdiv Security, R2C KICS

Application Security Testing as a Service (ASTaaS)

ASTaaS is an outsourcing approach that provides all security testing services to a third-party company. QualiZeal performs both static and dynamic testing, penetration testing, and evaluates application programming interfaces (APIs). ASTaaS is a popular option for organizations that do not have the expertise or resources to perform the required testing in-house.

Key Challenges and Considerations for Implementation:

Merely 20% of companies that adopted DevOps practices are consistently incorporating security into their development process.

Every team involved in the software development lifecycle should work together to validate the application security from the start. They should be reactive to any changes made to the application. The entire plan of shifting security left can go wrong if there is any mismatch in the communication. Here are some pitfalls that are most likely to interfere with the process:

Time-consuming and expensive:

Organizations that have already invested heavily in tools that detect security issues may find it too time-consuming and expensive to implement the shift-left approach. The sheer volume of security issues can overwhelm security teams, making it laborious for them to manage and prioritize the problems, let alone remediate them. This challenge can be tackled by selecting the right tools and adopting automation to reduce the workload of security teams.

Counterproductive practices:

To meet deadlines and maintain time to market, some organizations may push unresolved vulnerabilities into production despite early screening. This practice is counterproductive and can result in costly bug remediation and damage to reputation, trust, and relationships. The solution is to ensure that security issues are remediated before deployment to production.

Awareness and maturity of teams:

Integrating security into the SDLC is more complex for teams with low to medium maturity levels. They must have a comprehensive understanding of the most common security threats and problem-solving techniques. Developers need to be aware of the latest vulnerabilities and stay updated with the ongoing cybersecurity trends, so they can maintain the best code hygiene practices. This challenge can be mitigated by promoting collaboration, communication, and alignment to foster creativity and security.

Collaboration missing with Infosec Team:

Incorporating InfoSec teams into the software development process is crucial to make sure the security is acknowledged from the beginning.

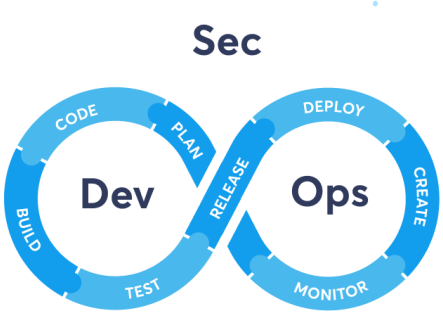
The failure to do so can result in unbearable losses, particularly as threats continue to strike in tandem with technology. Further, it causes increased expenses as making changes to the product that's already released requires an abundance of time and effort.

Potential for Conflict:
The diverse roles of developers and security teams can create inevitable friction. Organizations need to mediate and guide the transition until both teams can understand, adapt to, and accommodate each other's workflows. Everyone must work towards a common goal.

Shortcuts and Compromises:
The temptation to cut corners and take shortcuts may arise when up against deadlines. This practice rarely produces positive results and can increase the risk of security vulnerabilities. The solution is to prioritize security and avoid cutting corners.

Ineffective Tools:
Some organizations may have invested in security monitoring tools that add more alerts and notifications to already overwhelmed security teams. The solution is to select the right tools that can reduce the workload of security teams and provide actionable insights.

Best Practices and Recommendation for Shift-Left Security:



Implementing a shift-left strategy in the SDLC demands a momentous shift in the traditional approach to application security, which can impact organizational culture and the way teams work together. To help organizations successfully adopt the shift-left security approach, we have compiled a list of eminent practices that guarantee a successful outcome.

Assess Development Pipeline for Shifting Security Left:

Assessing how software is created and moves through the development pipeline is critical before implementing shift-left security. This assessment should aim to understand the entire software development lifecycle, including how code moves from development to production, the technologies used throughout the process, and any vulnerabilities in the pipeline.

Implement Security Fixes During Development: To shift security left, security must be integrated into the development process as soon as it occurs, providing timely feedback to developers for implementing fixes before release. This requires administering security like a never-ending process rather than waiting for a security assessment at the end of the development lifecycle. This method saves the organization an extensive amount of time and money.

Automate Security Processes: Automating security processes can ease the continuous work that usually takes more time when performed manually. Automation can help to identify bugs in the code and get rid of security issues at the earliest stage possible. To implement automation, it is advised to test the process and integrate the captured results with defect-tracking tools. Do not forget to assess the existing tools and select the most relevant ones for better implementation.

Maintain Transparency and Visibility: To ensure code is kept secure before and after release, maintaining constant visibility for all teams into the application's performance is unavoidable. Security professionals must be well aware of the development team's standpoint since imposing strict security measures can raise code complexity and prolong the project timeline. Project management teams and stakeholders will benefit from better visibility into the security posture.

Best Practices and Recommendation for Shift-Left Security:

	Shift Left Security	Shift Right Security
Definition	Integrating security early in the software development cycle	Integrating security late in the software development cycle
Goal	Preventing security vulnerabilities	Detecting and responding to security incidents
Focus	Proactive	Reactive
Timeline	During design and development stages	After deployment
Processes	Automated security testing, static code analysis, code reviews	Penetration testing, vulnerability scanning, incident response plans
Benefits	Cost-effective, identifies vulnerabilities early	Better response to real-world threats, addresses unknown threats
Limitations	Can lead to false positives, may not catch all vulnerabilities	More costly, can lead to longer response times

QualiZeal is a leader in advanced cloud-native security solutions.

Only 15% of organizations have the expertise to eradicate security threats or compliance issues in less than a day.

Modern software development is driven by agile methodologies, with rapid and continuous iteration, integration, plus delivery. The increasing complexity of software components and the need to respond quickly to market changes has led many organizations to adopt a shift-Left testing approach.

At QualiZeal, we recognize the importance of shifting left. Therefore, we invest in the right pool of talent, processes, and technology, including training for development and security teams, and adopting secure coding practices. Our team uses an automated testing strategy to seamlessly integrate security into the software development processes.

We believe facilitating communication among team members becomes more effective by discussing development and security in parallel rather than following a rigid linear flow. That being so, our inclusive suite of security testing services would help you identify potential threats, so you can proactively address security issues before they turn into major problems.

QualiZeal offers security training and consulting services to help organizations build a vibrant security culture. We vouch that your security practices remain up to date with the latest industry standards. Adopting a shift-left approach to security and leveraging QualiZeal's solutions must be your preliminary action to minimize the risk of security breaches and protect your sensitive data and assets.

In the eternally evolving world of technology, investing in robust security measures is no longer an option but a necessity. So, make the right choice today and partner with QualiZeal to secure your organization's future.