# Enterprise: The importance of Shifting Application Security Left

a Bright Security White Paper for myredfort

May 2022

# INTRODUCTION

In today's agile world, developers are shipping software and features continuously and at an unprecedented velocity and speed. This results in security vulnerabilities being introduced in the same increased velocity and volume, resulting in organisations being faced with unprecedented levels of exposure and risk across their applications and APIs.

Traditionally, application security testing, especially Dynamic Application Security Testing (DAST), is implemented during the latter phases of the development pipeline (Testing, or Release/Deployment). Legacy DAST solutions were built for security professionals, more commonly used by internal security teams, or via third parties on a periodic basis to complement their manual testing.

These legacy practices, coupled with delays in the developer feedback loops and the many resulting bottlenecks, have not kept up with organisations' demands in a world of agile development, CI/CD and DevOps methodologies. They prevent the guiding themes of automation and ever-increasing development velocity.

> "The ability to scale security testing and truly reduce your risk exposure, is directly linked to your ability to eliminate these manual processes and empower developers to identify and resolve the real vulnerabilities Applications and APIs will have in production"

It is therefore imperative to develop securely from the start, by enabling developers to identify and fix security defects much earlier in the SDLC.

**This is known as Shift-left security.**

## CONTENTS

# WHAT IS SHIFT-LEFT TESTING?

Shifting security left means introducing security as early as possible in the Software Development Life Cycle (SDLC). This means automating testing at the earliest possible phase during development.

> "300:1 – the average ratio between developers and cyber security professionals"

With the ratio of developers to security professionals up to 500:1 in some enterprises, the responsibility for security must be shared across the organisation.

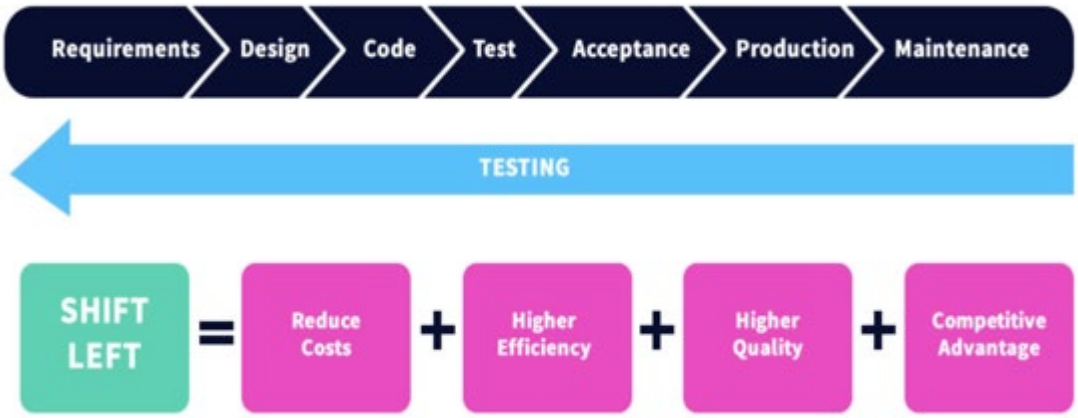
The AppSec team can provide governance across what should be tested. When and how the actual work is done needs to be spearheaded by development teams. This means empowering them to test and secure what they build while maintaining their rapid release cycles.

Developer-focused security tooling provides developer autonomy to ensure products are intrinsically secure by design, while being managed and governed by the security team. Automated testing on every build means security issues are detected and fixed earlier, more often, and faster, so code can be pushed to production faster by removing manual review and delivering consistent testing policies.

## How to effectively and successfully Shift DAST Left:

- Security tooling needs to be built with developers in mind – DAST scanners were traditionally built for security experts. This means you have to be uniquely trained to use them as they are complex, difficult to use and configure.
- Developers must be able to initiate scans automatically and quickly, without leaving their existing toolset, controlling scans via Command Line Interface (CLI).
- Tooling needs to be integrated throughout the SDLC pipeline to automatically test every build, PR or merge, in the CI/CD.
- Integrating DAST security scans during the Unit Testing delivers faster scans much earlier in the SDLC and enables developers to detect and remediate vulnerabilities 60 X faster.

- A feedback loop directly to developers of any security bugs detected is required, so that immediate remediation can be performed while the developer still has the relevant context and understanding of the code they have written.
- Accuracy is paramount to remove the noise and alert fatigue of false positives, which in turn will:
    - Provide developers with clear and actionable results, delivered in real time.
    - Remove the need for costly and time-consuming manual validation of findings by the AppSec team which is already stretched too thin.
    - By implication, reduce both security and technical debt.
- Doing all this while providing the AppSec team complete visibility of the process, vulnerabilities detected and remedied.




With regular testing on every build/commit, or at least daily, everyone can be focused on making better security decisions as part of a unified DevSecOps strategy to deliver software with speed, efficiency and security.

## Accurate and seamlessly integrated

Organisations relying on manual testing cannot keep up with accelerated development timelines, resulting in vulnerable applications and APIs in production. The success of this strategy relies on development teams having easy to use, accurate, and seamlessly integrated automated application and API testing technology.

Traditional legacy Dynamic Application Security Testing (DAST) tools are not built for this regular cadence of security testing that demands speed, accuracy and even earlier deployment in the SDLC. Modern DAST solutions, like Bright's, are leading the charge.

## THE ADVANTAGES OF SHIFTING-LEFT

### Fast, Efficient, Secure Software Delivery

One of the main drivers to shift security left is maximising the delivery of secure software, at scale.

DevOps has been pivotal in transforming the way we embed both unit and integration testing into our release cycles. Functional bugs are found earlier, remediated more often and faster, to release code of superior quality.

Mirroring this with security testing enables developers to detect and fix security bugs earlier and faster. Security, always known to be a blocker, can now be scaled to support large scale development while ensuring your applications are secure.

> "Security vulnerabilities should be treated in the same way any other bug and not dealt with separately as a responsibility of the security team"

### Empowering Developers

Traditionally, security prevents developers from actually focussing on their job of delivering product, with security gating the continuous development process instead of merging into it.

By empowering developers to own security testing means organisations can distribute ownership of delivering secure applications across a much larger team and not rely on a very small AppSec team. This also allows developers who ultimately will remediate the issues, to spend less time dealing with security bugs.

With developer-centric security testing automated across and integrated into your SDLC, the debilitating domino effect of side-lining security is eliminated.

## Prevent interrupted sprints

There's no need to put other work on hold to prioritise fixing what has now become a legacy security issue, or risk found in production. Security bugs are fixed on-the-go by the same developer who wrote the code, instead of the average 280 days this takes with legacy practices.

> "Finding and remediating vulnerabilities early in the SDLC results in a 60X time cost saving vs. remediation of vulnerabilities that were released into production"

## Reduce the cost and time to fix

The earlier security bugs are detected, the easier and less expensive they are to fix. With automated security testing, feedback to and remediation by the developers is carried out continuously and in line with other DevOps processes, allowing organisations to scale their efforts and boost productivity.

As the software development stages progress, the cost of addressing any uncovered bugs also rises, often exponentially.

Research from the Ponemon Institute found that if security vulnerabilities are detected early whilst in production they may cost on average, around **£63.**

The same vulnerability may cost around **£5,939** to remediate post production.

This doesn't include the associated costs of a security incident or data breach as a result of the bug being found by a malicious user.


Source: Ponemon Institute

## Maximise developer productivity

By implementing security testing in the SDLC, developers can treat security bugs the same way they do functional bugs.

With security testing that runs in parallel with both build and integration testing in the CI/CD, every new finding detected is raised with the developers immediately.

Remediation can be carried out faster, as the developer is familiar with the code, has the context of the feature they are creating, and you are able to ensure the same developer is tasked with the fix.

This dramatically reduces the mean-time-to-fix for security bugs, not only maximising productivity, but lowering costs and making organisations more secure.

## Reduce Security and Technical Debt

All of the above efficiencies mean the continual compounding effect of security debt and the resulting technical debt can now be addressed and mitigated by putting security testing into the hands of developers.

# BENEFITS OF SHIFTING-LEFT FOR SECURITY TEAMS

While shifting security left allows developers to be more efficient, it also comes with many advantages for the security teams.

Regardless of your organisation's size, developers can outnumber security by 300:1 and with almost 4 million open cyber security jobs globally, the problem is only set to get worse.

Automation is the only way to bridge the cyber security jobs shortage. By shifting-left, security is scalable by adopting consistent and efficient security testing across your pipelines, managed and governed by the security team.

By shifting-left, the security team can now:

- Create and govern security policies that mandate scanning on every build/commit. This saves time time in finding, validating and assigning security vulnerabilities in production
- Be a consultative organisation. By reducing the need to find vulnerabilities cyber security teams can focus on being more strategic and see the bigger picture. Prioritising remediation efforts early minimises risk and helps developers fix vulnerabilities before they're introduced into production.
- Focus on more complex security vulnerabilities that require unique security expertise, confident that the low-to-medium hanging fruits are being detected via automated tools in the pipeline.
- Rely less on manual testing, whether in-house or via a third-party, in turn reducing costs for the business

## More Secure Applications and APIs

While efficiency is important, shifting-left also means applications, microservices and APIs are going to be intrinsically more secure.

It combats development drag and blocked releases as a result of delayed testing, which impacts engineering teams' velocity and ultimately, the delivery of product.

From a security perspective, testing in production can result in far greater risk.

Aside from the fact that testing for and detecting vulnerabilities in production is too late, as it means that malicious users could find these before you do, there are several other limitations impacting security that can be overcome.

## Reducing the Reliance on Periodic testing

Unless security testing is baked across your pipelines, it takes time once the product is in production. Typically it's carried out monthly, or worse still, annually, purely as a compliance measure.

## Maximise Attack Surface Coverage

Achieving completeness in coverage can be a formidable task and requires security experts with the requisite expert knowledge of manually configuring tools built for security professionals.

Testing in production results in coverage limitations. Depth of testing is harder to achieve because the tester is not the developer and in many cases, is instead a third party penetration testing company, with a limitation of knowledge of the application.

Additionally, there are several security measures that need to be overcome to enable testing, such as adding the scanner to approved lists in a WAF, OTP requirements that affect scanning, CAPTCHA and other proxies that need to be overcome.

Having security testing earlier in the process negates this, as these security measures are not implemented. Security scans are therefore easier to configure and coverage is maximised, delivering efficiencies for all parties as well as being secure by design.

## Faster Scan Times

Testing in production requires the scanning of a large number of entry points, with test times taking 7+ days. This is not conducive to agile development and not only cripples rapid release cycles, but also results in far more work for the security team.

Shorter, scope-defined tests, on every build or as part of your unit testing, means faster scans and immediate feedback on security issues in the pipeline.

## Enhanced Prioritisation of Fix

With potentially hundreds of security findings detected in production, prioritisation and remediation of findings is a constant challenge and results in friction between the engineering and security teams, leading to compounding security debt.

Detecting security bugs early and often removes this, as both teams work together in real time to continually remediate security findings. Developers can treat and fix security issues like their functional bugs, with no requirement for security intervention and protracted meetings that would be essential if remediating a bug in production.

## Evolving role of Application Security Experts

With security vulnerabilities being detected in production, and the constant battle between security and engineering to fix these issues, security is seen as a blocker, causing friction between security and development. Shifting-left means that security becomes an enabler, reinventing themselves as security consultants and collaborators with the dev teams.

> "Creating harmony between security and development disciplines"

Similar to "escalating" infrastructure problems to DevOps or user-facing functionality to Product, security becomes the escalation point for more complex security issues found by R&D in the pipeline.

## ARE YOU READY TO SHIFT-LEFT? YES, YOU ARE!

The matter of shifting-left is not a case of 'if', but 'when'.

Hands-on responsibility for application security design and testing tasks is shifting to development and engineering teams. Gartner research reveals that more than half of software engineering leaders are directly responsible for application security, and another third share responsibility.

> "50% of software engineering leaders are also responsible for security"

It is inevitable that implementing security testing early and often in the CI/CD has to be achieved to deal with the large number of builds and iterations in software we see on a daily basis. Security as we know it cannot keep up - this has to change.

We have already discussed that periodic manual penetration testing is no longer viable, operationally, financially, or from a security perspective.

With modern DAST tools like Bright's, the ability to seamlessly integrate security testing automation into your CI/CD pipelines is achievable quickly, however in some cases, this requires a maturity level and alignment of processes across the DevOps teams to succeed.

Shifting-left does not have to be an all or nothing response.  It is important to start NOW, to ensure that progress is continually made and the organisation works to optimise the Shift-left methodology as processes and teams mature.

Hands-on responsibility for application security design and testing tasks is shifting to development and engineering teams.

Gartner research reveals that more than half of software engineering leaders are directly responsible for application security, and another third share responsibility.

## Start Now and Mature Later

Irrespective of your maturity levels, the process of shifting-left can be started today, by initially enabling smaller teams to carry out security testing. Any issues can then be worked out before scaling it up by rolling it out to other teams.

The goal is to have automated scanning on every build or sooner, as part of your unit testing, but shifting-left can grow as your teams and processes do too to achieve this ultimate goal.

## Leveraging QA

QA can be a vital bridge between development and security and can be used to spearhead your security transformation as you Shift-left.

QA have an intrinsic understanding of the applications and APIs and their existing functional scripts can be leveraged to carry out security testing too.

They can supplement the security testing they are already performing, mainly authentication and authorisation tests and load testing, with more comprehensive security testing. QA regression tests can serve as security regression testing, verifying that no security bugs are reappearing.

Enabling your QA to carry out security testing will refine your processes, introduce enhanced security testing earlier, and build security champions to define a culture of security best practices.

**BENEFITS:**

- Daily or weekly security scans
- Earlier developer and engineering manager feedback
- Alert on regression tests and rollback mistakes
- Ease of testing with no requirement to bypass OTP, WAF, CAPTCHA and other aforementioned scan limitations when testing in production environments

While this is certainly a step in the right direction, the notion of shifting-left is to do this earlier, owned by developers and governed by security. As this is matured, you can start to implement security testing automation into the CI/CD, perhaps starting with a specific team or squad.

## Security testing as part of the CI/CD Pipeline

The ability to run automated tests in the CI/CD enables security testing to keep up with the rapid release cycles and immediately deliver organisation-wide efficiencies, cost savings and improved security posture.

Deploying a developer-first DAST tool in the pipeline has a number of benefits for both the engineering and security teams, namely:

- Faster, scope defined tests that do not delay the CI pipeline
- Seamless integration with CI tools
- Enables specific scan policies per environment, target or development team for full control and visibility
- Ease of testing with no requirement to bypass OTP, WAF, CAPTCHA and other aforementioned scan limitations
- Immediate developer and engineering manager feedback of detected issues integrated with ticketing management allows for immediate triaging, prioritisation and remediation
- Snapshot understanding of your security posture, with reports continually
- Having regular snapshots of all of your applications allows analysis of patterns and trends over time, allowing your security team to measure process and effectiveness of tools, training and strategies
- Dramatically reduces the reliance on and cost of manual testing

## The Ultimate Goal . . . Unit Security Testing

As discussed, shifting-left is a process, but how far left can comprehensive security testing be implemented?

Software Composition Analysis (SCA) and Static Application Security Testing (SAST) have been at the forefront of early security testing, but they have their limitations, mainly around context.

SAST specifically cannot differentiate between code that is or is not run during execution and regular use of the application.

With modern technologies and microservices, this one dimensional view lacks context and how the wider application interacts with microservices and the corresponding APIs.

SCA alerts on insecure open source libraries are often oblivious to the fact the code doesn't use the vulnerable library method, resulting in false positives, developer alert fatigue, and no trust toward security tools at large.

A DAST scan will always be needed, whether carried out on production by a (third party) security team or later in the build phases.

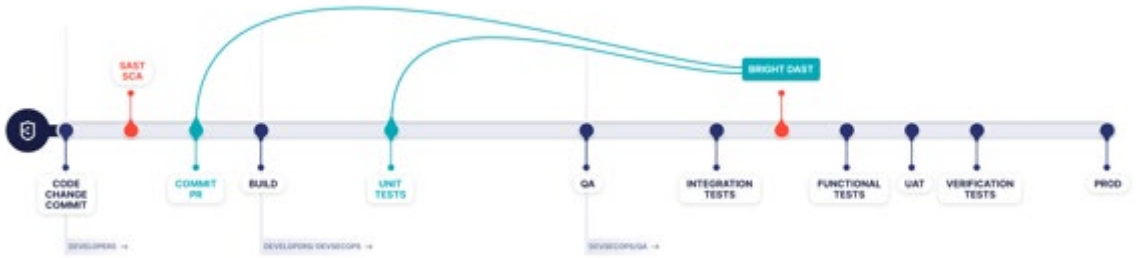
It is the ability of deploying DAST scans earlier and enabling developers to run scans in their unit testing that last for seconds to minutes and not hours to days, on every component / function as they create new features, that will propel your security programme.

**BENEFITS:**

- Earlier testing in the SDLC
- Optimised, scope-defined tests  on every newly created component / feature only achievable with unit security testing
- Lightening quick scans against each function
- Immediate feedback to developers while they code
- No ticket opening is required as the developer gets feedback and guidelines in the CLI
- AppSec intervention is completely eliminated (apart from remediation assistance or advice)
- Developer context switching is eradicated

# WHY DAST? WHY BRIGHT?

DAST means Dynamic Application Security Testing (DAST): it's a process of analysing a web application or API to find weaknesses through simulated attacks. DAST tools – sometimes referred to as "web app scanners" – attempt to attack an application from the "outside in", as a malicious attacker would. Once a DAST scan is complete, it reports any vulnerabilities it found so they can be addressed. DAST is a critical piece in developing, running, and maintaining secure applications and APIs.




Bright's developer-first DAST is the first of its kind to integrate into unit testing, revolutionising the ability to shift security testing even further left. Organisations can seamlessly bake security testing across development and CI/CD pipelines to minimise security and technical debt, by scanning early and often, spearheaded by engineering teams.

With NO false positives, there is no need for manual validation of security findings, removing costly and time consuming human bottlenecks that cripple rapid releases and drain security team's limited resources.

Bright is easy to use, fast, and integrates into pipelines to test applications and APIs (SOAP, REST, GraphQL), built for modern technologies and architectures. With automated Business Logic Security Testing, organisations can detect more complex vulnerabilities to minimise reliance on periodic manual testing to be secure by design, with full visibility of cyber posture to understand risk and compliment compliance.

- No false-positives
- No manual validations
- Easy to use
- Fast
- Integrates into pipelines and APIs
- Built for modern technologies and environments
- Full visibility of the cyber-posture whilst in development

# SUMMARY

Shifting-left is imperative to achieve a best-in-class methodology and process for building secure software at scale. In order to keep up with DevOps practices and implement DevSecOps, organisations cannot continue to perform periodic DAST scans and manual testing.

The need to shift security testing left is real and immediate; legacy DAST solutions simply do not enable organisations to effectively execute on their DevSecOps strategy.

Bright Security offers a modern DAST solution that is purpose-built for DevSecOps and empowers organisations to truly Shift-left, further than before.

Bright

## Bright Security - Background

"Traditional Application Security Testing isn't keeping up and focuses on detecting known vulnerabilities. Legacy tools rely on a heuristics-based approach and lengthy and costly manual testing for finding new issues. This doesn't scale and results in substantial delays to remediation, putting your business at risk."

Bar Hofesh and Art Linkov decided to do something about it. They combined their experience in cybersecurity and biologically-inspired machine learning, creating Bright Security's AIAST technology, which automates a human's critical thinking process when detecting vulnerabilities.