WhiteSource

# THE MAIN APPLICATION SECURITY TECHNOLOGIES TO ADOPT BY 2018

# Application Security Continues to Evolve

This September, consumer credit reporting agency Equifax reported a security breach that occurred between mid-May and July. The breach which affected over 145 million users, is possibly the most extensive breach to date. Hackers acquired access to a massive cluster of personal identifying information including names, social security numbers, birth dates, street addresses and, in some instances, driver's license numbers.

With those sets of data, criminals can steal identities to apply for credit cards, mortgages, loans, and more. The vulnerability that hackers exploited to access Equifax's system was in the open source component Apache Struts, which was used in their web-application. Beyond the sheer scale of the data that was compromised, this attack made it clear once again that the application layer has become a focus for hackers in their efforts to gain a beachhead within your environment.

These days, the reality is that 84% of all security attacks target the application layer. As organizations realize this, the focus toward application security and processes has increased and matured. In fact, the global application security market is expected to grow to $10.7 billion by 2025, according to a report by Grand View Research. Furthermore, application security continues to evolve to adapt to new software development methodologies, as seen in DevOps giving birth to DevSecOps.

Application security has always been important, but until a few years ago, network security was given higher priority. Enterprises were more concerned about implementing packet filtering and inspection on their firewall and security network appliances, and therefore infrastructure security took precedence over application security.

As hacker attacks on the application layer evolve, the need for application security that provides continuous coverage and real-time protection and remediation becomes a top priority. The tools and practices that used to provide security to organizations no longer provide a complete solution in today's development ecosystem. Security practices need to change, being implemented and continuously enforced from the earliest stages of development, and throughout the entire lifecycle.

# In this white paper,

we will present relatively new and trending application security technologies which are important to implement in the next year in order to keep your application security posture up to date and resistant to modern threats. We will also discuss the benefits that you can expect from implementing each technology and how it should affect your application security strategy.

## TOP 3 APPLICATION SECURITY TECHNOLOGIES TO ADOPT IN 2018:

### IAST

### SCA

### CONTAINER SECURITY

# Interactive application security testing (IAST)

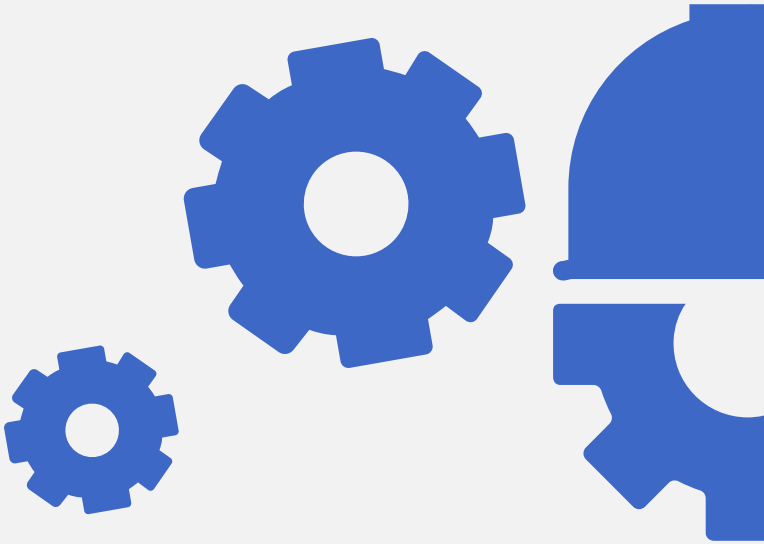
## Understanding the Range of Application Security Testing Methods

**Protecting your applications is a full time job that needs to be tackled from many angles. These are some of the leading methods that are used in the industry that you should know about:**

**Static application security testing (SAST)** is a process where the code is evaluated for security vulnerabilities without executing the code. This method analyzes source code to identify any vulnerabilities and deviations from an organization's secure coding policy. Carrying out this process helps review the early stages of the development cycle, showing the number of defects that might make it through to the production version of the application. Static application security testing is a key phase in Microsoft's published SDLC process as it's one of their key steps to ensuring that secure coding policies are being followed:

| 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|
| Training | Requirements | Design | Implementation | Verification | Release | Response |
| 1. Core Security Training | 2. Establish Security Requirements | 5. Establish Design Requirements | 8. Use Approved Tools | 11. Perform Dynamic Analysis | 14. Create an Incident Response Plan | 17. Execute Incident Reponse Plan |
| | 3. Create Quality Gates/Bug Bars | 6. Perform Attack Surface Analysis/ Reduction | 9. Deprecate Unsafe Functions | 12. Perform Fuzz Testing | 15. Conduct Final Security Review | |
| | 4. Perform Security And Privacy Risk Assessments | 7. Use Threat Modeling | 10. Perform Static Analysis | 13. Conduct Attack Surface Review | 16. Certify Release and Archive | |

**Dynamic application security testing (DAST)** analyzes applications in their dynamic running state and uses known attack methods to identify vulnerabilities. This is achieved by monitoring the system's memory, functional behavior, and the response time of the application by testing for known vulnerabilities within the running application. DAST happens at the verification stage of the development life. DAST will only monitor the code that is being executed within the application, so a comprehensive test plan will be in place to ensure that all areas of the application are analysed.



Did You Know?

## 2013 Target

**110 million records compromised**

In 2013, 40 million Target shoppers' private information was hacked. The hackers escaped with tens of millions of credit and debit card numbers, but the interesting part was discovered only a month later. The air conditioning management system was first hacked via a phishing attack, paving the way for hackers to break into Target's payment card reader system.

Did You Know?

## 2015 Ashley Madison

**36 million users exposed**

In 2015, the "cheating network" Ashley Madison, announced that hackers infiltrated the network and published nearly 10 GB of data on a Tor website, making it accessible to anyone with a Tor browser. The user information of about 36 million users were published, as well as millions of payment card transactions.

**Interactive application security testing (IAST)** is a form of application security testing that combines dynamic application security testing (DAST) and static code security scanning (SAST) technologies. An agent is deployed within the application and analyzes the code dynamically at runtime during its execution within an application server or web site. One of the advantages of IAST is that it can see all the libraries, frameworks, and calls to external applications or services, providing a holistic view on the security of the application. There are two deployment methods for IAST:

**Active IAST**
is a combination of IAST and DAST, utilizing DAST to generate malicious traffic, the IAST agent will monitor the application and understand how the application responded to the malicious traffic. Combining the two components allows the identification of vulnerabilities with a low false positive overhead.

**Passive IAST**
is the IAST agent will monitor the application during runtime without inducing any malicious traffic. This process is reliant on the test plan being comprehensive enough to cover the threats the application may encounter in production.

**DAST and SAST** are complementary. They provide detailed insights into the security of the application. When combined, they tend to discover issues that wouldn't have been apparent if the test had only used one approach.

The growing trend with SAST and DAST vendors is to partner with Software Composition Analysis (SCA) vendors (for example, IBM AppScan and Checkmarx have both partnered with WhiteSource) or develop their own SCA solution (as Veracode, a SAST & DAST vendor has done).

While IAST is a combination of DAST and SAST, it's not a complete replacement, since it misses a few of the SAST functionalities, such as full access to the application code and all data flows. Also, IAST requires additional overhead due to its more complex deployment and maintenance.

Did You Know?

2014
Yahoo

**500 million accounts compromised**

The massive Yahoo breach was revealed in late September 2016 while Yahoo was in the midst of being acquired by Verizon. They announced a breach from 2014 where over 3 billion accounts were compromised, including information like: names, email, birth dates, telephone numbers and more. One of the spooky aspects of this breach is that Yahoo blamed an undisclosed country for the attack.

# The Benefits of Adopting AST Technologies and Tools

1

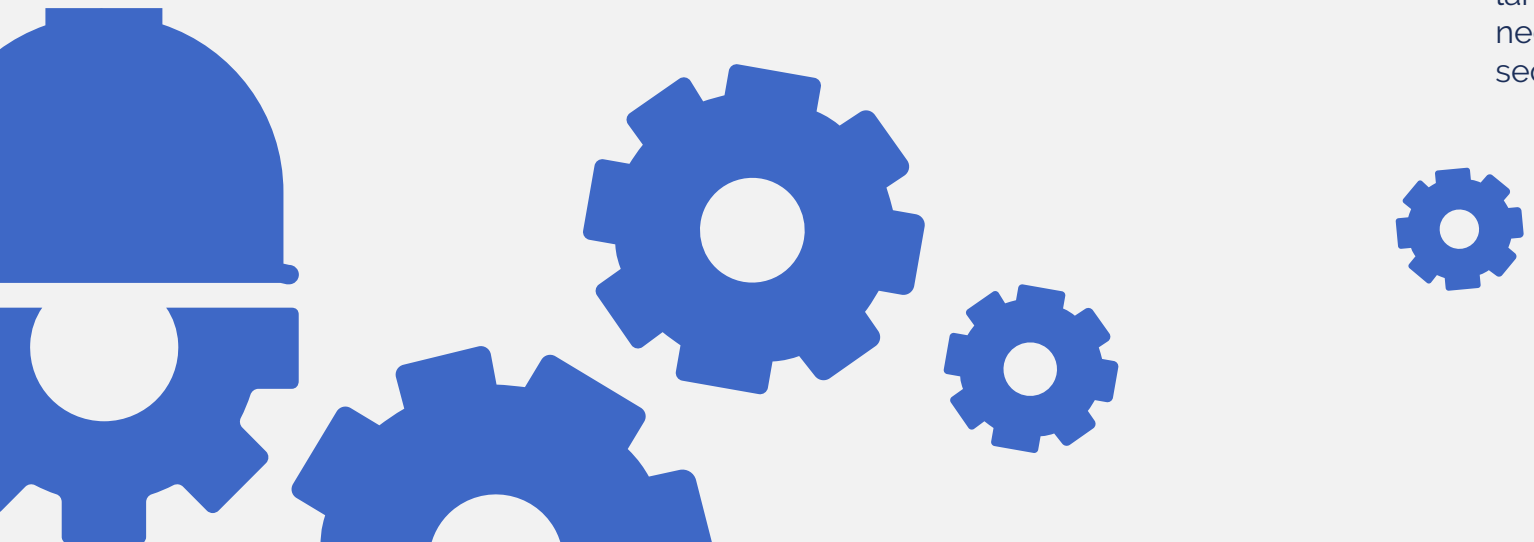
**Easier detection of defects and more secure code** When you integrate Static\Dynamic or Interactive Application Security Testing (AST) tools with existing development systems, defects that are detected will automatically be assigned to the relevant developer. By directly embedding AST tools into SDLC, you are provided with instant feedback to a developer, leading to the delivery of the more secure code and providing an improved delivery time.

2

**Prevents data pollution** Implementation of AST tools, both in development and test environment, will eliminate the possibility of data pollution. This will lower the possibility of bugs entering into production versions of your code. A secure testing environment is necessary for producing accurate test data, since it will be used for optimizing and tailoring the production versions of your applications. When using inaccurate test data, you're risking late delivery of your product or even defects once your application is released.

3

**Planning for the future** When properly deploying and maintaining your code, AST tools will serve your application security needs for years to come. As development practices change over time and new languages and technologies are introduced, the Static\Dynamic AST tools need to be reviewed regularly to ensure that they support the new environments and that the current rules will still be relevant. With proper planning and investment, you will only need to make minor adjustments with AST tools for new languages and technologies, without the need to implement new application security solutions from scratch.



# Software Composition Analysis

Widespread open source usage is simply a fact in organizations of every size. across all verticals and open source components have became the foundation of modern applications. Open source is a force multiplier for developers and it enables companies to accelerate software development cycles. Forrester recently estimated that only 10 - 20% of the average code base is original.

Software Composition Analysis (SCA) tools are enabling software development and security teams to automatically detect all open source components in alerts in real-time on problematic components the minute they are added to the code base, or once an issue is published within the open source community.

The key value of the Software Composition Analysis (SCA) tools is the ability to enforce policies automatically at different phases of the software development lifecycle (SDLC) to block vulnerable or problematic components from entering the software, and therefore avoid license or security issues later on in the lifecycle. Integrating SCA tools into your SDLC offers a unique opportunity to shift left your open source management and significantly reduce the remediation costs when compared to a last minute detection just before release.

Gartner has reported that “SCA is becoming a critical or mandatory feature of application security testing (AST) solutions, as open-source and third-party components are proliferating in applications that enterprises build."

SCA also automates many developers tasks that are usually performed manually in most teams, and therefore increasing efficiency.  For example, automating the entire process of open source components approval, tracking open source inventory, identifying vulnerable components with possible fixes, and more. Another important aspect in regards to open source vulnerabilities is the need to continuously track the history version as in many cases vulnerabilities are discovered in components released years ago, which increases the complexity when tracking is not automated.

The recent Equifax data breach has brought open source software security issues to the forefront, highlighting the fact that the majority of organizations nowadays do not understand the need to implement dedicated tools to find and fix open source vulnerabilities as SAST, DAST and other technologies do not have the capability to do so.

# The Benefits of Adopting Software Composition Analysis Tools

1

You can **Shift Left** detecting problematic open source components before even downloading a certain component, not to mention integrating it with your code. The security, quality and license information on each component is public, but the problem is that the information is spread across thousands of websites and is not indexed for a quick search. SCA tools aggregate publicly available information and index it to enable quick detection. These tools can empower your developers to make better choices to begin with and also alert you in real-time whenever a problematic component is added. Shifting Left is all about uncovering as many issues as possible as early as you can in the software development process, both so that the cost of fixing them is under control and you can avoid the stress when it comes time to release.

2

**Blocking the Unwanted** When the policies within the SCA are configured to be enforced at every stage of the build, creates an opportunity for less frequent issues moving forward. Using the SCA integration will automate your security licenses and quality checks, which will skim out vulnerabile and out of date components that can put your products at risk. These policies need to be continually refined to match the requirements of your business. Once this is achieved, your code will be patched and secured from the vulnerabilities that were presented by the SCA integration, thus utilizing your open source components.

3

**Full visibility of Open Source Code** Implementation of SCA will automate detection of all your open source components that could have vulnerabilities or license issues, and once identified, the issue can be patched and made secure again. At this point, open source software components comprise between 60% to 80% of most organizations' code base. The SCA integration provides companies the opportunity to track the license terms of their open source components, as well as the capability to avoid using code that can introduce risk or potential vulnerabilities into their product.

Did You Know?

## 2012 LinkedIn

**165 million accounts compromised**

In 2012, a whopping 165 million LinkedIn users had been compromised by hackers via by their password-reset notifications. While we all get the monthly warning emails about a mysterious login by an unknown user, this time over 6.5 million of the 165 million LinkedIn users fell for the ploy of the hackers requesting to change their password.

Did You Know?

## 2017 Equifax

**145 million accounts compromised**

On Sept. 7, 2017, consumer credit reporting agency Equifax reported a security breach, impacting 145 million Americans. Hackers acquired the access to a massive cluster of sensitive information from which criminals can steal identities to apply for credit cards, mortgages, loans and more. The hackers exploited an open source component with a known vulnerability. This is considered the biggest breach in U.S. history, and one that could have been easily prevented as the vulnerability was published more than two months before the breach with an easy fix.

# Container Security

Containers are lightweight OS-level visualizations that allow the isolation of an application and its dependencies. When the container is run, it is isolated from the host and then run from a distinct image that provides all files necessary to support the application. This makes containers extremely portable and simplifies the process from development, to testing, and finally production.

Application security was a late bloomer to the industry and is playing catch-up with the DevOps movement. This is especially true with container security. Hyperscale companies such as Google are running and managing billions of containers within their environment. Companies of all shapes and sizes are having success with using containers within their continuous delivery pipeline. As containers move to mainstream use, container security will become an important consideration for all companies which are using them or planning to migrate their application workload to a containerized environment.

# The Benefits of Adopting Container Security Tools

1

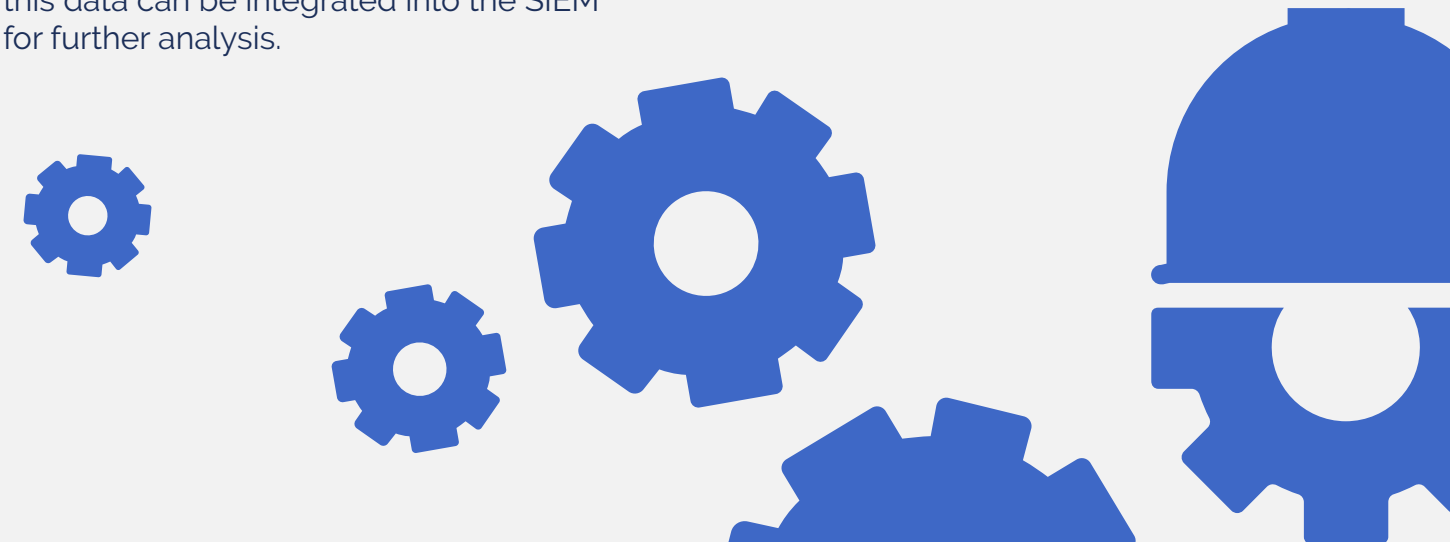
**Reducing the attack surface** By installing and enabling the minimum set of features within the container to support the application and continuously scanning the configuration of your containers, you will significantly lower your footprint and thus lower the chance of being attacked.
In order to achieve this, consider using an OS that is designed for containers such as CoreOS or RancherOS. These operating systems are designed for container usage, but require a proper configuration and constant monitoring of components used inside containers. Consider hardening your containers with a more secure baseline. There are a number of guides available that provide the best information required to build a hardened environment. *Understanding and Hardening Linux Containers* and *CIS Docker Community Edition Benchmark*, for instance. These guides will provide the configuration steps that will allow you to build a secure foundation, which the applications can be developed on and deployed to.

2

**Complete control of your container orchestration platform** If the orchestration platform is compromised, considerable damage could be caused within the environment. Implementing container security tools with the orchestration platform can be controlled to utilize a role based controls which will ensure that only the correct level of access is provided to users of the platform. All access and actions that are carried out should be logged for auditing purposes, and if the organisation has a SIEM then this data can be integrated into the SIEM for further analysis.

3

**Consistent state of runtime configuration** Using automated checking and remediation features from container security tools, you will mitigate the risk that configuration changes inside containers may deviate away from the security baseline and introduce additional risks into the environment. Besides being a security threat, this will also ensure that your own developers can't change the runtime configuration of containers by mistake, ensuring a consistent environment for your applications.



# Summary
# What Does the Future Hold?

It's clear that enterprises should make a priority not to neglect the security of their application layer. Implementation of AST technologies, in combination with SCA tools, is a reliable path towards a more secure future, especially to prevent attacks that abuse vulnerabilities in OSS components or your code base. If you're using containers or considering a move to migrate your applications to a containerized environment, security should be one of your main concerns, be aware that containers are not virtual machines, and you can't protect them the same way you secure your current infrastructures. With the ever evolving changes in security technologies, we need to prioritize  protecting our applications and our environments in general, we also need to adjust the way we think of security.

Shifting away from security as the responsibility solely of the security team is gaining more momentum each year. The SANs state of application security highlights that 30% of respondents assign responsibility for security testing to their development team. This increased 8% from last year. In the near future, we will see a larger scale of adoption of the shift left approach, making "everyone" responsible for security. As we see a shift in responsibility and focusing organizations need to strongly consider:

**Tool integration in SDLC** to simplify the developer's work and increase efficiency. The integration should offer a better guidance for all phases of development from requirements, capture to testing.

**Empower the developer** by providing the right tools to make the right decisions. Security should be frictionless to the developer, enabling immediate feedback on issues within their code.

**Break down the silos:** The developers and security teams must work closely together. A close working relationship makes for better communication and entail the sharing of findings and best practices.

**DevSecOps:** With the introduction of DevOps, agile, and public cloud services, traditional security processes are no longer economical or fit for purpose. Traditionally, security has been engaged at a late stage of the development lifecycle, but with the increasing adoption of DevOps and agile, this is no longer optimal. The iterative approach requires security to be embedded into the lifecycle, as well as tools and processes for analysis and feedback supplied to developers. The security team and developers should work together on defining processes and setting security policies and standards within the tools.

Even though the number of attacks is increasing every day, it's possible to keep your applications secured and updated. By incorporating the technologies we provided in this white paper, along with the shift left approach to your application security, your applications will be developed and managed in a more secure manner. You can also be certain that your SDLC is not harmed in any way, allowing your team to avoid last-minute bug fixes or the release of applications with known security vulnerabilities.


How to Choose a Solution for Managing Your Open Source Components
You know you need to manage your open source for licensing, security and quality issues, but which solution is best for your needs?
DOWNLOAD NOW