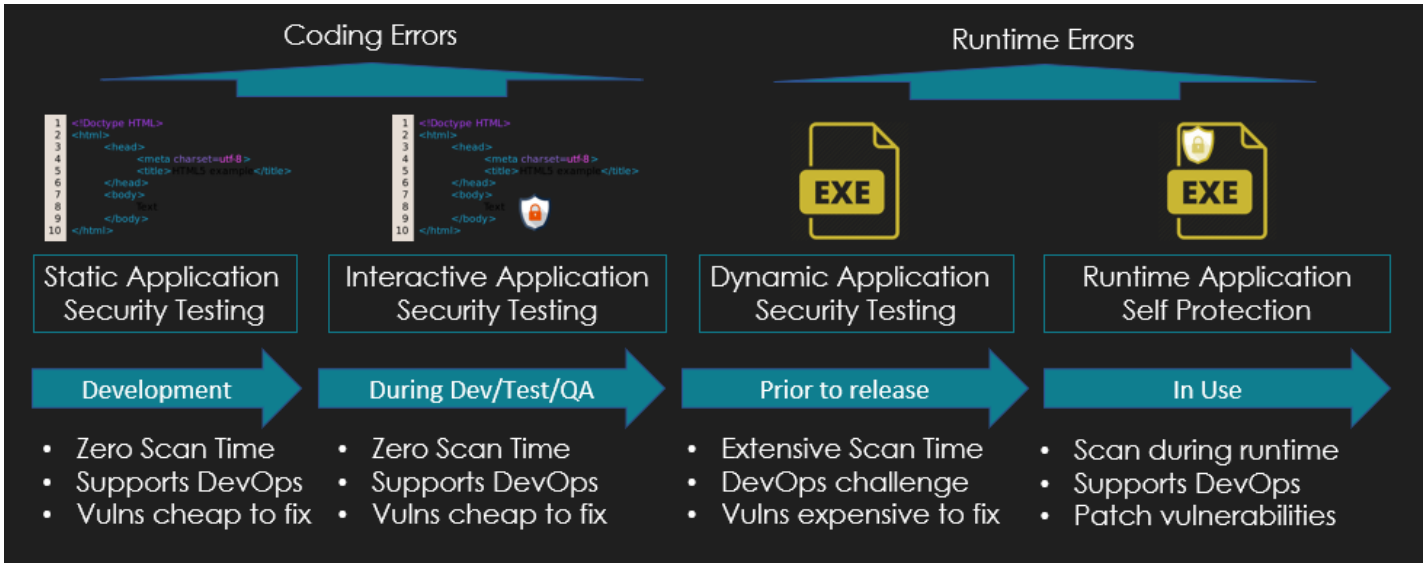
Secure Application Development

Software vulnerabilities are now recognised as a fact of life, and users and IT teams are accepting that even the largest most security conscious organisations will develop applications with flaws which require patching. Even the operating systems we all run our critical services on have regular patch schedules – each patch a recognition that something 'slipped through the net' and needs to be repaired to keep your organisation safe. If the likes of Microsoft and Oracle regularly make mistakes – what chance does a smaller organisation have who is writing their own code? Luckily, toolsets have become available over recent years to help you to detect and fix these errors in the code you create for use internally and with your clients.



Static Application Security Testing (SAST)

SAST was the first generation of tools to help coders do their job more securely. The aim of SAST is to detect errors early in the dev cycle where they are easiest and cheapest to fix. These tools operate by interrogating the source code before it is turned into executable binaries. Tools can be used to integrate into the code check-in process, and more popularly, integrated right into the development environment to check code as it is typed – giving instant feedback to the coder allowing them to immediately fix mistakes. With the integration into the IDE, links to training can be made available if the developer needs more information.

Dynamic Application Security Testing (DAST)

DAST was designed to solve some of the blind spots of SAST – as there will be errors which can only be detected in the executable binary during runtime. DAST usually works in tandem with SAST so both coding errors and runtime errors can be detected. One of the uses for DAST is to test edge cases in input validation using a process known as 'fuzzing' where thousands of inputs are fired into an application to see how it deals with them. DAST scanning can only be done on executable code, and due to the many permutations of testing, it can take a substantial amount of time to complete. This causes some issues where the DevOps methodology is in use, and rapid deployment cycles are required.

Interactive Application Security Testing (IAST)

IAST is an attempt to solve the problems of SAST and DAST. IAST performs security testing as the application code is exercised, which fits in with modern CD/CD cycles. As various branches of the code and APIs are tested during the QA process, the same branches are assessed in real time for coding errors. This fits much more neatly with modern DevOps methodology with a zero scan time.

IAST analyses from within applications and has access to application code, runtime control and dataflow information, memory and stack trace information, HTTP requests and responses, and libraries, frameworks, and other components. This analysis allows developers to pinpoint the source of an identified vulnerability and fix it quickly.

Runtime Application Self-Protection (RASP)

RASP is similar to IAST but protects the code while it is in production use, after deployment. Web Applications Firewalls (WAFs) are traditionally deployed to protect internal applications from exploitation by users – but what if the code is going somewhere else, outside of your network? Think of RASP like a WAF-like wrapper around your application which can detect common issues like the OWASP Top 10. It can be configured to alert you to a problem or even automatically block issues. RASP comes into its own where issues are only detected after the rollout of an application where taking an application offline for remediation is going to be extremely costly. RASP therefore provides the ability to patch applications after deployment as a last line of defence in those rare cases where SAST, DAST, and IAST have failed.

Discovery and Scoping Questions for Secure Application Development

There are multiple different vendors competing in this space – and the best tool for the job depends very much on your requirements. Some of the questions we will ask when helping you to determine the correct vendor are:

- What does your current development process look like?
- Is a DevOps methodology in place, or part of your future strategy?
- How many developers do you have?
- How many projects are ongoing at one time?
- Which key languages and frameworks are in use?
- What is the likely outcome of doing nothing?
- What have you done before?
- Do you have a preference between services offered from the cloud, or on-premises?
    - Some organisations are not able to send code to cloud services for security purposes
    - Some vendors offer additional functionality when coupled with cloud services

How Armadillo can help

At Armadillo, we listen to your requirements and act as an honest broker to advise you on the best combination of products and services to meet your needs. We strive to understand the entirety of your IT estate so we can offer recommendations which will integrate with your existing platforms and operating model and increase your security maturity.