451 Research

S&P Global
Market Intelligence

DevSecOps: Application security tool use between development and information security nears parity

Analysts - Daniel Kennedy

Publication date: Tuesday, January 12 2021

Introduction

Verizon's 2020 *Data Breach Investigation Report*, which is augmented with public sector incident-response information, suggested that approximately 43% of data breaches could be traced back to the compromise of a web application. The tools to combat this – from identifying vulnerabilities directly in source code to fuzzing web and mobile applications for weak input controls – have long been available, and the final piece of the puzzle, the process of applying application security, is starting to fall into place. Developers and information security personnel are entering a heretofore unseen level of collaborative use of application security testing (AST) tools.

The 451 Take

The 'shift left' concept is not new in application security – for example, plug-ins for integrated development environments (IDEs) for popular AST tools like static AST (SAST) have been around for more than a decade. The reasons for addressing security vulnerabilities within software development lifecycles (SDLCs) are straightforward – fixing a defect while a developer is actively working on a section of code is a lot cheaper than trying to reopen something to fix it later, and a lot less damaging than a vulnerability being exploited by a bad actor in production. However, expectations for production web applications to be largely free of security defects and the pressures of keeping up with release cycles that deploy more frequently than in the past have forced a greater share of day-to-day application security testing to be federated to application developers. Per 451 Research's Voice of the Enterprise: Information Security end-user research, a steady multi-year trend toward greater collaboration has reached near parity in tool usage between the two teams.

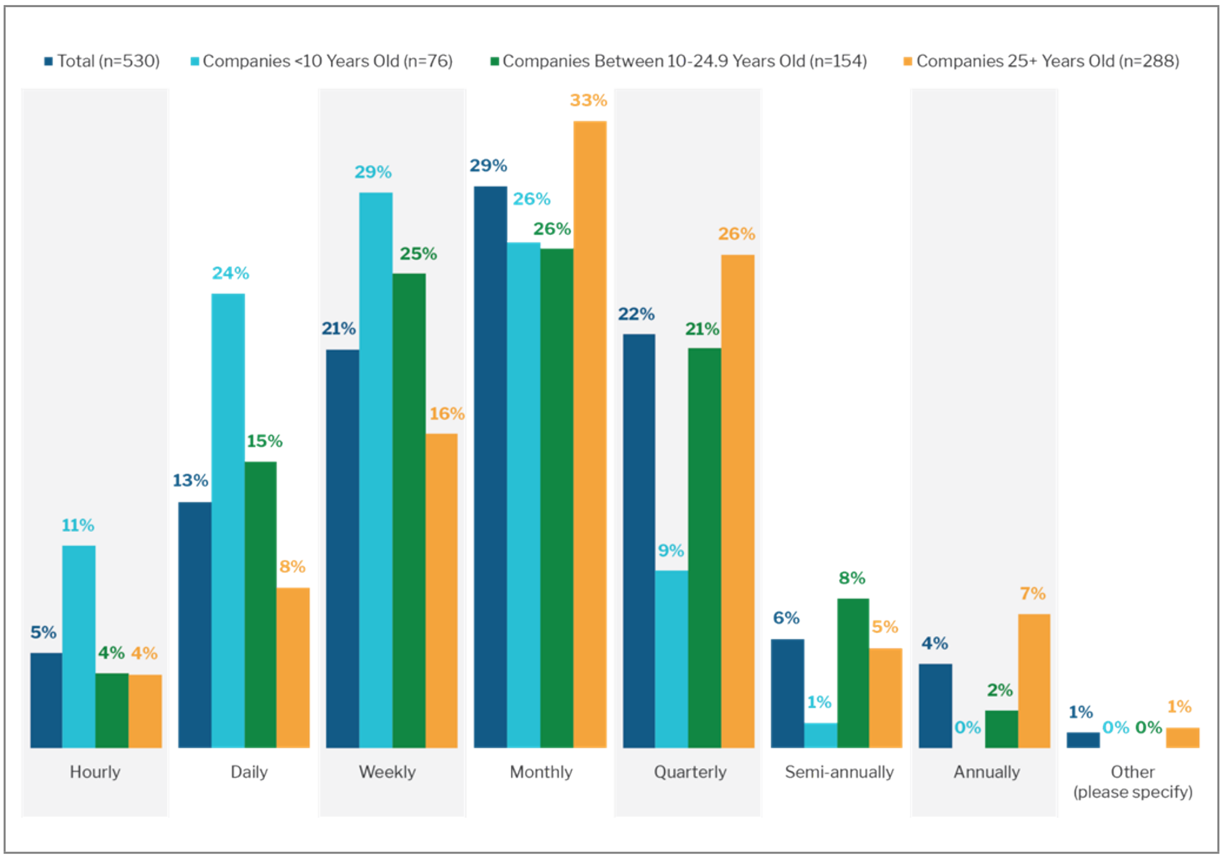
Legacy approaches, and why they're not keeping up

Earlier approaches to application security programs, which persist in some enterprises, are largely characterized by testing applications' security in production only or testing application changes for security vulnerabilities directly before code is released into production. The latter approach involves a potentially significant delay in deployment if serious problems are found or an uncomfortable acceptance of risk due to project timelines that could have been addressed had the problems been found earlier. Some enterprises engage in large-scale 'clean up' projects where potentially thousands of lines of codes are scanned and vulnerabilities corrected under ad hoc projects setup to do so.

Any of these approaches is well ahead of doing nothing to ensure application security, a tactic unfortunately still in use at a rapidly dwindling percentage of enterprises with serious internal application development capabilities in place. Such approaches ignore due care responsibilities and have historically (in other areas of security) been subject to regulatory risk when the risk of an irresponsible approach is passed on externally to customers, suppliers and other third parties, as can happen when application security vulnerabilities are exploited by bad actors.

Running an application security program that tests primarily only in production or in a stage directly before production release is an approach that will have an increasingly difficult time keeping pace with the changes developers are making to applications. The typical enterprise with a serious application development discipline has more developers than information security personnel – certainly more than security personnel who concentrate on application security. The velocity of both code integration and deployment continues to increase, and as shown in the chart below, the majority of newer organizations are doing code production releases at least weekly. Assuming such organizations are a bellwether for where organizations with a legacy footprint are eventually heading, or at least would like to get to, the pressure of release cycles is only going to increase. In addition to increased availability and viability of the AST tools that enable shift-left approaches, this pressure is what's heating the crucible in application security.

Figure 1: Frequency with Which Organizations Deployed Software Apps to Production



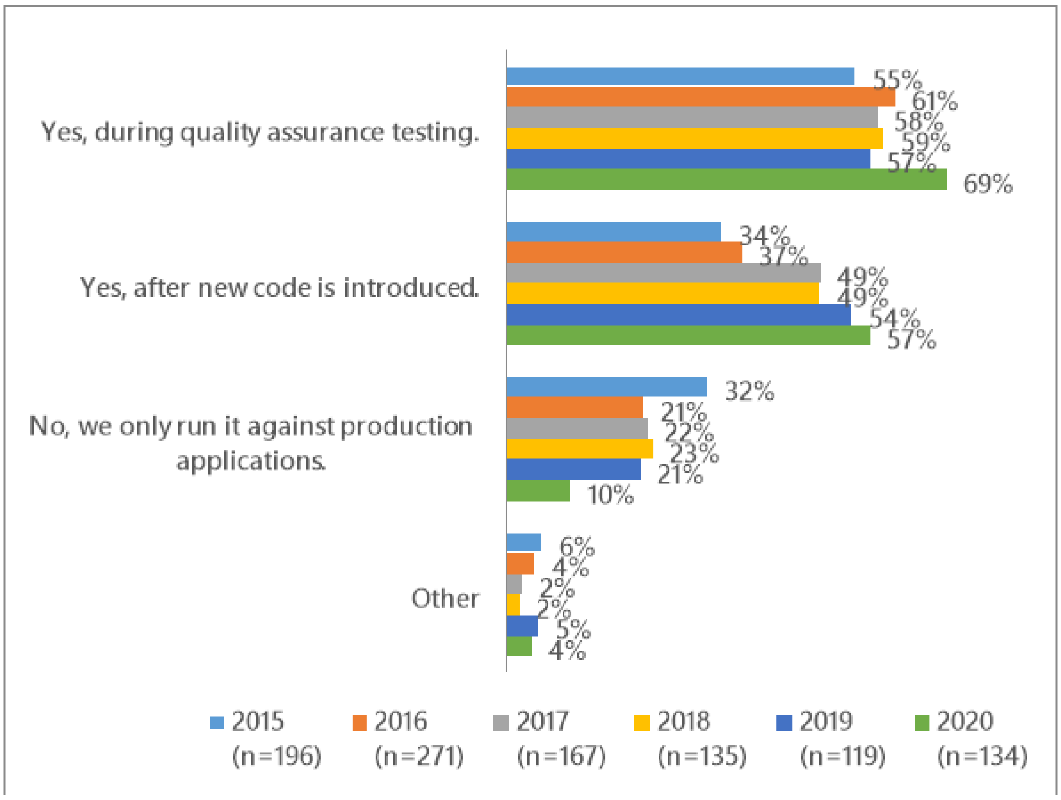
Source: 451 Research's Voice of the Enterprise: DevOps, Workloads & Key Projects - Advisory Report
Q. Over the last 12 months, how often did your organization deploy most software applications to production?
Base: All respondents

Emerging approaches, and why developers are more involved than ever before

As noted above, approaches such as performing SAST during code construction have long been available – for example, kicking off a scan from within an IDE. But other aspects of AST have improved alongside a kind of organic standardization of DevOps toolchains – for example, the ability to kick off processes via Jenkins or checking for open source vulnerabilities (software composition analysis or SCA) at the time of code check-in or pull request. This more macro approach to AST has allowed for improved integration of a greater variety of testing approaches, including but not limited to SCA, SAST, DAST, IAST (integrated AST), MAST (mobile AST), and 'shift right' production level protection offered by web application firewalls, RASP (runtime application self-protection) and bot detection/mitigation.

This improvement in the ability for different types of AST to infiltrate different parts of the development process in a more automated way has led to the steady shift shown in the chart below. The percentage of enterprise respondents stating that they only run AST tools against production applications has dwindled over the past six years, against a marked increase in the percentage running AST tools directly after the introduction of code changes.

Figure 2: System Development Lifecycle Phase Where AST Is Applied

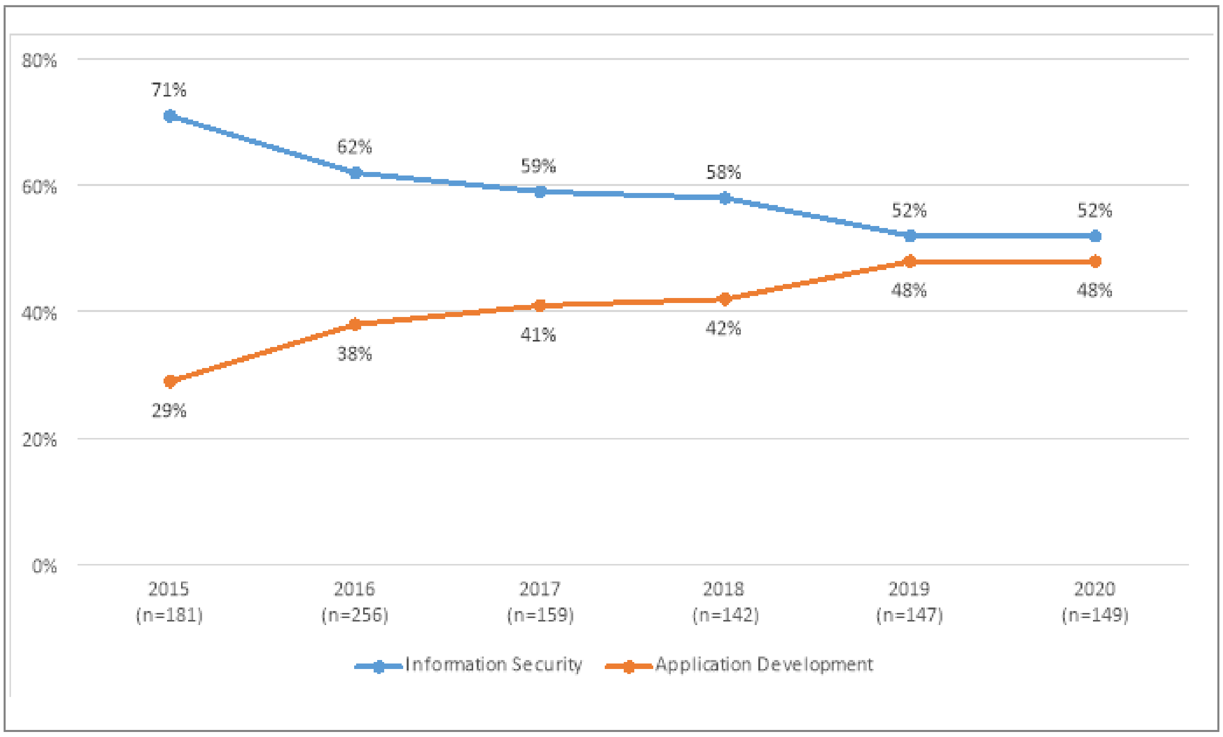


Source: 451 Research's Voice of the Enterprise: Information Security, Vendor Evaluations 2020
Q. Do you run your application security vendor's tools during different phases of the SDLC as part of a secure SDLC? Please select all that apply.

"The teams doing agile are checking a new code many times a day. And every time they check in code, they're most likely going to be triggering a build, and the build will run the application security testing, and they will get the results in real time…. It's not like the waterfall days, where you had worked on an app for 90 days and run the first scan right before the go-live day, and get shocked at how much you're going to get forced to go live with various internet vulnerabilities." – 100,000+ Employees, $10bn+ revenue, Food & Beverage, IT/Engineering Manager/Staff

In Figure 3, below, looking at the same effect from a team collaboration perspective between information security (blue line) and application development (orange line) over the same period, a clear trend line emerges. Information security professionals were, in 2015, the primary users of AST tools. That usage has ceded to developers a little more each year, and has flattened in 2020. For information security, this trend is a positive one. The idea that security personnel in the typical enterprise will have time to review every code change or kick off manual scans to produce vulnerability reports for each pull request isn't realistic. In-depth peer reviews among developers on the same team are a rarity that is often context dependent – a security professional coming in who isn't intimately familiar with a code base or project specifics will have even less of a foundation to work from in what are often abbreviated project timelines. Giving developers the means to efficiently test for and respond to security vulnerabilities during code construction is the most efficient path to keeping up with newly introduced application security issues.

Figure 3: Allocation of Application Security Tool Usage Between Application Development and Information Security



Source: 451 Research's Voice of the Enterprise: Information Security, Vendor Evaluations 2020
Q. How is the usage of application security tools allocated across the following two teams in your organization?

Pendulum swings are often accompanied by those who would see it swing further, and thus an obvious retort to both the conditions outlined above and the trend lines emerging is that enterprises should just make application development entirely responsible for application security. This is an argument along the unfortunate lines of 'security is everyone's responsibility.' Yes, it is, but platitudes rarely make good management practice: When everyone is responsible for something, it can mean no one is actually responsible for driving it. Application developers, like other IT professionals, are driven by a combination of metrics and management expectation, and going back to the days when a single computer could fill a room, those expectations have centered on the delivery of functionality. Thresholds for acceptable defects are a part of that, but organizations that don't take the time to ensure that governance and compliance functions are in place and continually running are typically characterized by atrophied controls following some initial effort.

Security is a team sport, meaning security professionals within an enterprise will be called upon to work in a matrix with a number of different functional teams. This approach is common in endpoint management and network security, and it's increasingly going to be part of the management of cloud infrastructure. Application security is no different – security's reason for existence is the implementation of processes to protect the enterprise from a host of threats, and thus that team typically holds budget, implementation expertise and the motivation (via how it is measured) to insist on continuous process improvement around the protection of information. Enabling other teams by federating certain day-to-day activities can further this mission, but it doesn't absolve information security from, in the context of application security, ensuring tools work for developers, making sure tools are being used and run, and reviewing the status of vulnerabilities. This includes identifying which vulnerabilities are most commonly being found (an opportunity for developer education) or aren't being resolved, which may require direct intervention. That last one is a key input to a macro responsibility of information security – having a solid understanding of the risk posture of the enterprise's applications at any given time.

As this new collaborative approach to application security and 'shifting left' gains traction, AST solutions are going to have to make adjustments. Running in a project-level context within the SDLC for a developer means scans in context – not attempting a scan outside of, for example, the code or functions that have changed, and not reporting everything and the kitchen sink about an application for a project level test. This new approach involves, along the lines of taking advantage of the organic standardization of DevOps tools, not requiring context switching on the part of developers, meaning reporting results in the tools they're already using. The proliferation of AST tools means tools must work better together; having different types of testing enriches results around security vulnerabilities, such that there are fewer false positive reports or missed security issues. Additionally, it means serving both teams, which can mean working within developer tool chains while ensuring that information security is receiving, for example, the dashboard reporting needed to address their oversight responsibilities. For vendors, the sale into an organization is more complex, and they are benefited by ensuring both information security and application development are in the room for their sales pitch and implementation plan.