Contrast SECURITY

Pipeline-Native Scanning for Modern Application Development

Executive overview

The application security landscape is dominated by tools loosely referred to as code scanners, which includes static application security testing (SAST). Twenty years ago, these tools represented a leap forward in an organization's ability to understand the security of its applications. Today, however, ever-increasing development agility and application complexity reveal fatal weaknesses with regard to the usability, scalability, and effectiveness of these tools. For scanning to become an effective tool for vulnerability remediation, organizations need a tool that will be used by developers and is purpose-built for modern applications and continuous integration/ continuous deployment (CI/CD) pipelines.

As a complement to the Contrast Application Security Platform, Contrast Scan offers a pipeline-native approach for scanning applications for vulnerabilities early in the software development life cycle (SDLC). It combines demand-driven static analysis, risk-based policies, and a product design that enables developers to find and fix issues as they code. As a result, Contrast Scan harmonizes the objectives of security and development teams while improving the quality of code that's ultimately sent to production.

Why scanning for "more results" Outcompeted "accurate results"

Historically, organizations relied on SAST to help analyze the security of their software. Even with considerable changes in the software development landscape, significant academic advances in static code analysis research, and increasing maturity of application security programs, SAST offerings have not fundamentally advanced since the early 2000s

When evaluating different application security testing (AST) tools for purchase in the past, many customers chose tools that produce more results (quantity), rather than more accurate results (quality). There are multiple reasons for this—the main one being the particular software development paradigms of the day, which predated the wide adoption of Agile and DevOps:.

When it comes to accuracy, traditional SAST solutions achieve a mere 26%.[1]

- **Fear of false negatives.** The application security team, which typically led the tool evaluation effort, was concerned that selecting a tool that produced fewer results simply opted for false negatives (i.e., "Isn't it my job to find all the bad stuff?").
- **Optimizing for application security use cases vs. development use cases.** All of the alert noise was not perceived as a negative outcome by application security teams, as this simply gave them more "leads" to chase down. False positives did not cause major workflow delays to development because scans were infrequent.
- **Confidence in the ability to tune.** A solution that provided a wide superset of potential vulnerabilities wasn't considered a downside by security teams because they believed that the tool could be tuned for greater accuracy.

In this case, customer behavior drove vendor design. Security vendors raced to create tools that yielded more and more results, regardless of result validity. Tools went from just being noisy to becoming mathematically impossible to effectively triage. Multiple studies show that while these traditional scanning tools warn users of a huge number of possible threats, the output is so consistently riddled with false positives that results are largely ignored inside organizations. As a result, few of these threats ultimately get addressed in development.[2,3,4] While the tools create a lot of work, they deliver little value.

> Fixing a vulnerability gets more expensive as the development process gets further from where the error was introduced.[5]

As application demand increased over time and developers adopted methodologies to ship code more frequently, application security was forced to play catch-up. Vendors created plugins and integrations to support developers in their repository or in their pipeline, without addressing the problem of cataclysmically imprecise vulnerability testing. This sea change only magnified SAST's existing problems. Instead of having a few days to triage results once a quarter or once a year, developers needed to scan applications every day—sometimes multiple times a day—and often without the involvement of security experts.

All of this tallies up to a huge time expenditure. The vast majority of organizations (73%) report that each security alert they receive consumes an hour or more of application security time.[6] Further, 72% of organizations indicate that true vulnerabilities consume 6+ hours of application security team time; 68% say that true vulnerabilities consume 10+ hours of development team time.[7]

More than half (55%) of developers admit to sometimes skipping security scans to meet deadlines.[8]

Why SAST needs a new approach

Traditional SAST solutions bury true vulnerabilities in a noisy sea of false positives. Only about one-quarter of organizations report being capable of reviewing all alerts in scan reports and returning guidance for remediation to the development team.[9]

One vendor's data showed that their legacy SAST solution finds an average of 1,700 results in a selection of open-source libraries[10]—but they do not provide good feedback on the relative importance or irrelevance of those results. Although public data is not available for all traditional scanning tools, any experienced practitioner most likely will indicate that these kinds of results are quite typical. Even assuming that the tools find all the true positives amongst their total results, their poor accuracy shows a minimum of 99% noise (accurate but irrelevant findings and false positives).

Contrast customers report an average of 21 vulnerabilities per application—down from 26 in 2020.[11] In another survey, the majority (79%) of organizations reported that their average application in development has about 20+ actual vulnerabilities.[12]

Effective static analysis for today's current landscape must provide answers for these unfortunate truths:

1. **Vast quantities of false positives obfuscate the small number of true positives to the point where they cannot be found.** While a proper investigation of each finding could eventually identify the few true positives among the total accumulation of results, most organizations today don't have time to go through thousands of alerts per scan. Because only trained security experts can perform this sort of work, traditional scanning solutions are essentially specialized tools designed for expert analysts. A majority (85%) of organizations report that application security processes slow down development cycles at least sometimes.[13] To achieve scale, organizations need fast and accurate scanning that separates actual vulnerabilities from false positives.

2. **The tuning that organizations must do to make scanning tools reach sustainable output levels effectively disables the tools' ability to and true positives.** In the evaluation phase, organizations often fixate on making sure the tool finds every possible theoretical vulnerability in test applications. But when using traditional SAST in practice, the need for tuning to increase speed and efficiency simultaneously reduces the tool's sensitivity for finding actual vulnerabilities. This suggests the "true-positives-at-all-costs" strategy backfires, allowing many false negatives to slip into production without being remediated.

3. **Full static analysis is prohibitively expensive, and thus all vendors have checkpoints and governors that prevent full analysis.** Scanning every line of code is possible, but scanning all code paths is not.[14] In addition to knowable a priori, Contrast's collective staff has also worked on competitive tools and has faced this reality before. There are many limits to static analysis modeling and all tools have analysis governors that limit scan times. Security vendors will understandably get uncomfortable if challenged on how this affects scan completeness. To date, the most important objectives are:

    - Scans always finish in reasonable time (which is an admirable goal).
    - Scans always find many vulnerabilities, hopefully more than my competition (which is a contemptible goal).

Some of the challenges here involve branching, reflection, inversion of control, dependency injection, dynamic code, and inheritance. All of these pose significant obstacles to delivering complete analysis in times that customers would find acceptable.

How Contrast came to reimagine scanning for modern needs

In 2014, Contrast Security pioneered using code instrumentation as a breakthrough technology to analyze the security of applications. This approach avoids the weaknesses inherent in code scanning tools to provide accurate and actionable information. Since then, hundreds of large enterprises around the globe have adopted this technology through the Contrast Application Security Platform—with outstanding results.

But despite the availability of more accurate tools like Contrast Assess interactive application security testing (IAST), legacy code scanning technology is still widely used across the industry. Often, this is because existing processes are designed around scanning, and these kinds of established norms can be very hard to change.

As the leader in using instrumentation for security, Contrast continues to push the boundaries of application security research and developed a new breakthrough technology for scanning complex and distributed applications within today's rigorous development systems. This new approach ("demand-driven static analysis") is designed to act as a companion to Contrast's existing platform technologies and to provide unprecedented ease of use and speed with superior, accurate, and actionable results.

Contrast Scan uses demand-driven scanning technology to allow development teams to achieve high-quality results while using a familiar "scanning style" methodology. This allows customers to achieve dramatically better security outcomes than when using traditional static analysis without having to change their existing processes.

Measuring real application risks—quickly and accurately

Modern security needs to be reoriented toward the real risk posed by the application via a tool that can identify a small subset of critical vulnerabilities with as little obfuscation by false positives as possible. Another consideration is that there are limits to any type of analysis. If discovery isn't possible, then development deserves a clean "go-to-the-next-step" signal—rather than a pile of results that leave everyone involved with just enough of a case to support their go/no-go decisions to drive organizational conflict.

For example, consider two tests that doctors use to check heart health: a blood pressure test and an exercise stress test. Almost anyone can measure blood pressure; it can be done in 10 seconds and the equipment is ubiquitously available. One can easily discern if a pressure is much too high or much too low and get visibility into broader heart disease risk factors. But physicians do not look at these numbers too closely because the results have limited value with respect to comprehensive cardiac evaluation. It's just an easy (but important) first step in the diagnostic process.

To get a more complete and accurate picture of heart health, doctors put sensors on the patient and place them on a treadmill for an exercise stress test. Doctors use those sensors to get a much deeper view of the heart while it is in operation to identify weaknesses (e.g., arrhythmia, cardiovascular disease, etc.). The first few minutes of the stress test are enormously informative. This means that even though some candidates can only make it halfway through the 15-minute test (and therefore have imperfect test coverage), the amount and quality of the data observed still allows the doctor to perform a much deeper analysis and detect more issues than they can with just the blood pressure numbers. Thus, for example, if someone has great blood pressure numbers and shows no warning signs after a partial stress test due to being unable to complete it because of a work emergency, we can be extremely confident they are not walking around in a dangerously diseased state.

While Contrast Assess is our "stress test" that provides authoritative results, Contrast Scan is built on the vision of the initial blood pressure test. Our approach to modern scanning was designed to bring customers great security results with low noise, speed, and high confidence. To accomplish this outcome, we need the test to be fast and accurate. It combines a number of critical capabilities:

DEMAND-DRIVEN STATIC ANALYSIS

To make static analysis scale to cover large code bases, traditional whole-program algorithms compromise on precision—causing large numbers of false positives as a side effect.[15]

Demand-driven analysis uses lightweight pre-analyses to quickly "zero-in" on the parts of the code that are security-relevant, and then use maximally precise analyses to determine the security state of the code.[16] The high level of precision (which maximizes full context-, flow-, and field-sensitivity) not only reduces false positives to a minimum, but it also helps the analysis itself focus on just the code that really matters.[17]

As a result, code paths that are probably irrelevant to the security state are minimally inspected, which also reduces analysis times by several orders of magnitude. Theoretical and empirical studies show that demand-driven analysis boosts both analysis precision and speed to levels that traditional, whole-program analyses cannot possibly accomplish.[18]

PRODUCT DESIGN THAT INSPIRES CONFIDENT ACTION

How a product is designed has a big influence on how it is used, shaping how users think about the results. At a design level, Contrast leads to a drastically different user experience. An important example of this is how we categorize the findings of the scanner into groups. Contrast results fall in one of these categories:

1. **Vulnerabilities.** These issues rank as higher severity with a high level of confidence. Therefore, they warrant immediate analysis and triage and often lead to a fix by a developer.

2. **Warnings.** These are issues with either lower severity or lower confidence. It is generally safe to ship code with issues of this type, although in rare cases they may lead to exploitable conditions. This process is very much like compiler warnings for comparable software projects.

3. **Leads.** These are discoveries that cannot be confidently cited as issues. Rather, they need to be viewed as potential "jumping-off points" that require analysis by an application security expert or a developer security champion. This is due to the fact that SAST is limited by the nature of its analysis and business context. These issues may need to be triaged by developers with specialized security knowledge or by application security team members with coding backgrounds.

The formation of these three buckets introduces a fundamentally new way to view SAST results, automatically triaging vulnerabilities according to automated next-step actions within the modern development environment. For instance, the discovery of a vulnerability should stop a build. The discovery of a warning should give pause to a developer, but ultimately should not break the build or demand immediate triage. The discovery of a lead is not important to anyone except experts who must chase down the derivation in an application security assessment setting—not an analytic to act on in a pipeline.

Case study examples

Although we believe customers get the best combination (or coverage) and correctness by combining IAST and SAST on the Contrast Application Security Platform, it is still worthwhile to compare the efficacy of Contrast Scan with legacy SAST alternatives.

CASE STUDY #1: FORTUNE 25 BANK APPLICATION

INITIAL RESULTS	CONTRAST SCAN	MAJOR COMPETITOR
CRITICAL/HIGH/MEDIUM SEVERITY ALERTS	3	8
LOW SEVERITY ALERTS	1	252
TOTAL ALERTS	4	260

Application Security Time and Resources Wasted Due to High Signal to Noise

CONTRAST SCAN	TRUE POSITIVE	FALSE POSITIVE	SIGNAL TO NOISE
CRITICAL/HIGH/MEDIUM SEVERITY ALERTS	3	0	100%
LOW SEVERITY ALERTS	1	0	100%
TOTAL ALERTS	4	0	100%

MAJOR COMPETITOR	TRUE POSITIVE	FALSE POSITIVE	SIGNAL TO NOISE
CRITICAL/HIGH/MEDIUM SEVERITY ALERTS	3	5	38%
LOW SEVERITY ALERTS	3	249	1%
TOTAL ALERTS	6	254	2%

CASE STUDY #2: NARCOS (OPEN-SOURCE APPLICATION)

METRIC	CONTRAST SCAN	NEW DEV-FRIENDLY COMPETITOR #1	NEW DEV-FRIENDLY COMPETITOR #1
TRUE POSITIVES	38	6	4
FALSE POSITIVES	2	1	2
FALSE NEGATIVES	2	34	36
ACCURACY SCORE	95%	15%	10%

Strengthening contrast's platformbased approach to application security

Contrast Scan gives developers fast and accurate security feedback at the pace of innovation without the disruptions of legacy solutions. However, when viewed as part of the overall Contrast Application Security Platform, Contrast Scan offers customers even greater benefits. There are two questions that Contrast customers need to ask themselves to determine what deployment is most appropriate for their environments:

1. When to use Contrast Assess? In the pipeline or in a pre-production environment, customers should run IAST (the "exercise stress test" in our analogy). Not only does it offer the most accurate results, but it also offers a superset of vulnerabilities beyond what SAST can offer, given its deeper visibility. And when the application has good pre-production code coverage (i.e., a near 15-minute stress test), we believe organizations do not need anything more than IAST. In the modern application security portfolio, we believe IAST is the centerpiece of vulnerability detection because it offers the most depth in vulnerability discovery, and it offers the most useful context to enable fast triaging and fixing with the least possible noise.

2. When to add on Contrast Scan? Contrast Scan can be added to the Contrast Application Security Platform in cases where test coverage is a concern, where policy requires the use of SAST, and/or when customers want faster vulnerability feedback. Including Contrast Scan early on (focused on coverage, correctness, and speed), in combination with Contrast Assess (focused on depth), offers the most comprehensive, accurate, and frictionless experience for both developers and application security teams. One university study showed that if one were to combine any two solutions to get the best accuracy, the top five combinations all included Contrast Assess.[19] With that in mind, we aim for studies in the future to call for the top combination to include Contrast Scan and Contrast Assess.

In the real world, some small percentage of vulnerabilities will escape detection no matter what one does—either by bad luck, human error, or just because vulnerabilities are complex. Having visibility and protection in production is an additional requirement for modern applications and application programming interfaces (APIs). Here, the Contrast Application Security Platform also includes Contrast Protect to deliver this critical ability—using the same technology as Contrast Assess, but reorienting it toward the performant protection of applications.

The addition of Contrast Scan to the platform of solutions strengthens our mission to secure software across all stages of the software development life cycle (SDLC). This comprehensive suite of capabilities was purpose-built for modern, distributed applications—offering embedded, continuous testing and protection that reduce application security risks.

1. "OWASP Benchmark v1.1, 1.2," OWASP, accessed August 20, 2021.
2. "Foteini Cheirdari and George Karabatis, "Analyzing False Positive Source Code Vulnerabilities Using Static Analysis Tools," IEEE Xplore, January 24, 2019.
3. "Zachary P. Reynolds, et al., "Identifying and Documenting False Positive Patterns Generated by Static Code Analysis Tools," IEEE Xplore, July 3, 2017.
4. Joonyoung Park, et al., "Battles with false positives in static analysis of JavaScript web applications in the wild," Association for Computing Machinery, May 14, 2016.
5. Jeff Williams, "How To Start Decluttering Application Security," Forbes, January 27, 2021.
6. "The State of DevSecOps Report," Contrast Security, November 2020.
7. "2021 State of Application Security in Financial Services Report," Contrast Security, May 2021.
8. "The State of DevSecOps Report," Contrast Security, November 2020.
9. "2021 State of Application Security in Financial Services Report," Contrast Security, May 2021.
10. "Micro Focus Fortify Static Code Analyzer: Performance Guide," Micro Focus, November 2018.
11. "2021 Application Security Observability Report," Contrast Security,
12. "The State of DevSecOps Report," Contrast Security, November 2020.
13. "2021 State of Application Security in Financial Services Report," Contrast Security, May 2021.
14. Paul Anderson, "Human Factors in Evaluating Static Analysis Tools," GrammaTech, March 27, 2017.
15. Brittany Johnson, et al., "Why Don't Software Developers Use Static Analysis Tools to Find Bugs?," North Carolina State University, May 2013.
16. Swati Jaiswal, et al., "Bidirectionality in flow-sensitive demand-driven analysis," Science of Computer Programming, May 1, 2020.
17. Johannes Sp th, et al., "Context-, flow-, and field-sensitive data-flow analysis using synchronized Pushdown systems," Proceedings of the ACM on Programming Languages (Vol. 3), January 2019.
18. Yuanbo Li, et al., "Fast graph simplification for interleaved Dyck-reachability," Proceedings of the 41st ACM SIGPLAN Conference on Programming Language Design and Implementation, June 11, 2020.
19. Francesc Mateo Tudela, et al., "On Combining Static, Dynamic and Interactive Analysis Security Testing Tools to Improve OWASP Top Ten Security Vulnerability Detection in Web Applications," Applied Sciences/MDPI, December 20, 2020.

Contrast SECURITY

Contrast Security provides the industry's most modern and comprehensive Application Security Platform, removing security roadblocks inefficiencies and empowering enterprises to write and release secure application code faster. Embedding code analysis and attack prevention directly into software with instrumentation, the Contrast platform automatically detects vulnerabilities while developers write code, eliminates false positives, and provides context-specific how-to-fix guidance for easy and fast vulnerability remediation. Doing so enables application and development teams to collaborate more effectively and to innovate faster while accelerating digital transformation initiatives. This is why a growing number of the world's largest private and public sector organizations rely on Contrast to secure their applications in development and extend protection in production.

240 3rd Street
2nd Floor
Los Altos, CA 94022
Phone: 888.371.1333

Contrast SECURITY

contrastsecurity.com