TECHNICAL BRIEF

Contrast Security architecture

Increase application and API resilience by identifying and eliminating zero day attacks and application vulnerabilities

Architectural overview

Contrast Security's runtime security platform helps organizations block attacks and identify vulnerabilities from QA to production in real-time with innovative, in-app technology. It easily scales to protect your entire software portfolio, including applications, Application Programming Interfaces (APIs), libraries and even third-party applications. Contrast secures the whole application stack at once, instead of scanning pieces separately and overlooking major components. The result is an effective operating model that delivers a high level of security while also accelerating development productivity and innovation.

The Contrast runtime security platform leverages the power of instrumentation to embed sensors within the application's runtime. This means Contrast can see what is happening in real time, and get answers directly from production. This, in turn, limits false positives and for the first time, developers, application security, and the SOC are all working out of the same dataset to reduce and defend the attack surface.

These sensors are central to Contrast's unique approach. Contrast's threat sensors provide visibility and control inside the application layer, operating entirely in the application running in user space, not in kernel space. Unlike other methods such as eBPF programs that run within the OS kernel and are constrained by a variety of restrictions designed to protect the kernel, the Contrast approach provides context for vulnerabilities and accuracy needed to prevent attacks in real-time. The Contrast runtime security platform provides the safest architecture for always-on security for your applications and APIs. Contrast leverages open telemetry protocols and other lightweight and highly scalable approaches to ensure virtually undetectable impact on performance. So it can run everywhere, even in production.

Contrast concepts

Contrast Graph: A digital twin of the entire application and API security landscape. This real-time, unified security model is created from a wide variety of telemetry directly measured from running code by Contrast's threat sensors.

Application: One deployable artifact of code.

Server: A process running one or more applications in an environment. It varies by language, but this process could be Tomcat, IIS, WebSphere, UWSGI and others.

Environments: The stage of the deployment lifecycle in which a specific server is running, such as:development, QA or production. Rules and configurations are specific to each environment.

Routes: A combination of an HTTP verb, resource path and method signature that handles requests to the application. The exact format can vary depending on the language of the Contrast agent in use. A route can have multiple URLs associated with it.

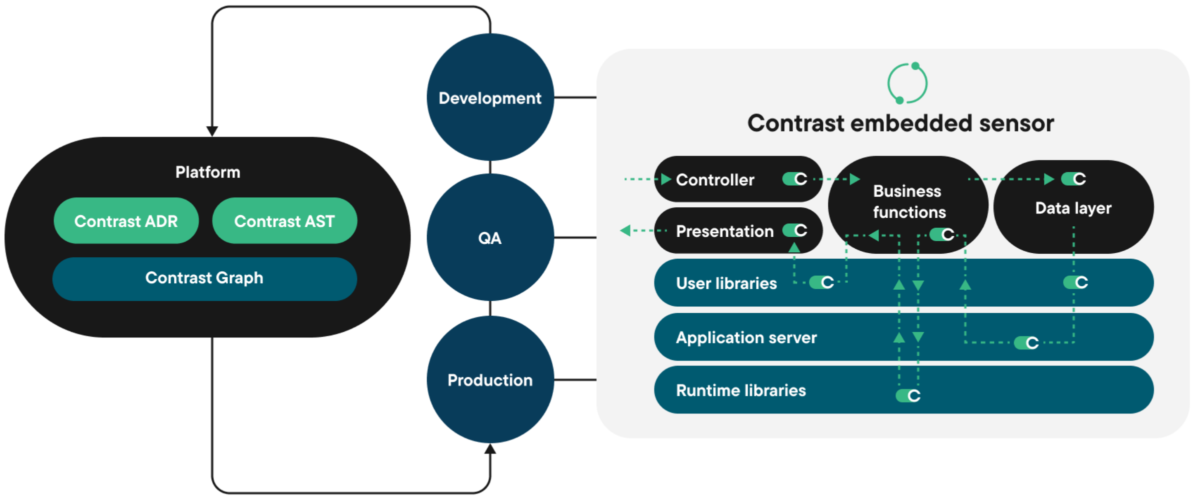
Issues: Vulnerabilities in applications

Incidents: Attacks on vulnerabilities

Instrumentation: Small snippets of code that the Contrast agent places into an application to analyze behavior, data flow and operational context.

Runtime: Contrast actively monitors and analyzes application behavior in real-time, identifying and potentially blocking threats as they occur.

Contrast architecture



The Contrast difference

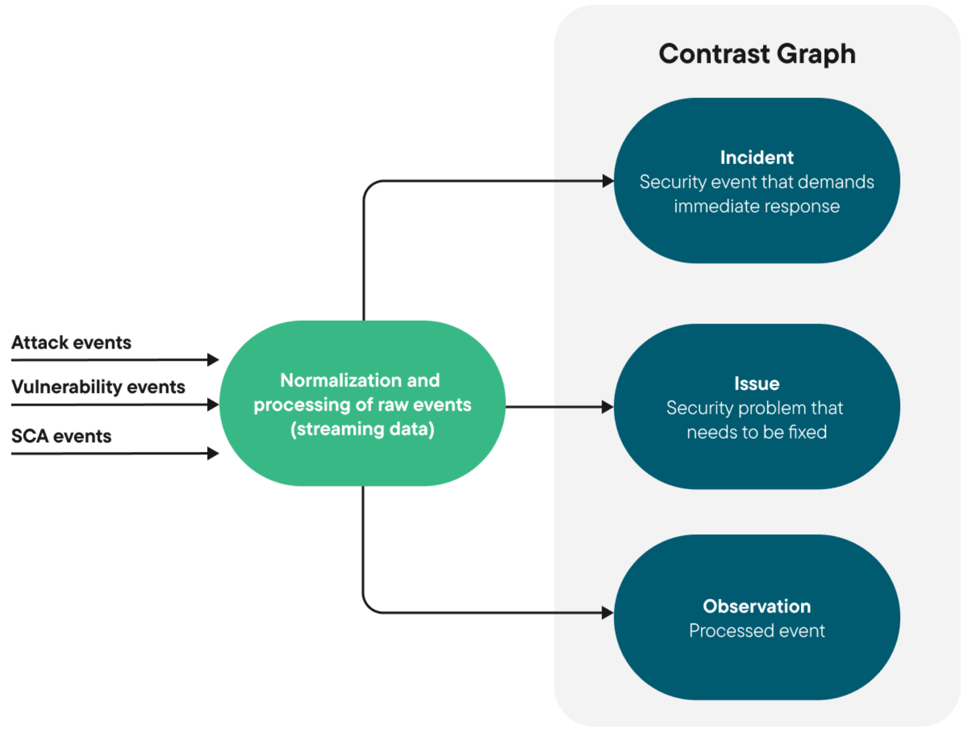
Contrast's threat sensors analyze data flow and identify vulnerabilities in running applications. Contrast inserts Contrast code in the application's existing methods across custom code and libraries. Sensors observe the locations where data enters (sources) and reaches dangerous functions (sinks). This action creates real-time visibility into any data that flows through the application and allows Contrast to detect security flaws or vulnerabilities in this code path and report them to Contrast. This also allows Contrast to take actions within an application to prevent exploitation or enable a security defense against attacks until vulnerabilities can be remediated.

Contrast gathers relevant information using a variety of security instrumentation techniques, including code scanning, library scanning, instrumenting an application, configuration file scanning, and other techniques. Any security instrumentation technique that gathers information is a sensor. Sensors generate events that snapshot information directly from within an application. For example, a sensor might capture an incoming HTTP parameter, or the details of a SQL query being made to the database. Some sensors may also take action if necessary to help strengthen defenses or block malicious activity, typically by throwing a security exception that prevents a vulnerability from being exploited.

Events generated by sensors are all reported to the tracking and analysis part of the agent. Over time, the analysis engine receives events from all over the code of the application and builds them into traces. The analysis engine watches these traces for patterns of behavior that represent a violation of the Contrast rules.

Critical capabilities

Contrast data streaming architecture


Contrast Graph
Attack events
Vulnerability events
SCA events
Normalization and processing of raw events (streaming data)
Incident
Security event that demands immediate response
Issue
Security problem that needs to be fixed
Observation
Processed event

In addition to Contrast's ability to detect real-time attacks, including zero days, and vulnerabilities with accuracy and precision — left through the development pipeline and right into production, Contrast also provides the following key features:

Contrast Graph

Contrast's modern streaming data architecture run by Kafka and graph DB creates a comprehensive digital twin of an organization's entire application and API security environment. It correlates real-time data across services, maps attack paths, and reveals how vulnerabilities, threats and assets are interconnected so you can make decisions fast.

Contrast AI

Contrast uses AI to both suggest fixes that developers can then use but also, for critical and high vulnerabilities, automatically create pull requests that capture a complete fix for issues and a test case to verify correct remediation. This uses Contrast's detailed understanding of all the relevant context, including the full HTTP request, control and data flow traces, route behavior and connections, and successful programming patterns from the same application — reducing remediation time from weeks to minutes. Customers can also plug in their preferred AI via the new Contrast MCP Server, giving them the flexibility to operate within their chosen ecosystem.

Contrast Score

Contrast uses context from the Contrast Graph including a wide range of risk factors from production, such as asset criticality, exploitability, threat intelligence and even active attacks, to update risk scores dynamically to ensure focus on what matters.

Product specifications

The Contrast runtime security platform has broad language support for web and API applications and offers 30+ partner integrations:

Language support

Contrast's Flex Agent deploys and auto-updates the agent that matches the language of the application you want to instrument for Java, .NET, Python, and Node.js. Provided agents are as follows:

- The Java agent supports Java, Kotlin, and Scala web applications and web APIs.
- The .NET Framework agent supports .NET web applications and APIs running on IIS.
- The .NET Core agent supports applications and APIs running in the .NET Core runtime.
- The Node.js agent supports web applications and APIs with support for code written in native Node.js or TypeScript.
- The PHP agent supports analysis of PHP web applications at runtime for library usage and vulnerability detection.
- The Python agent supports a variety of web and API frameworks including Django, Flask and Pyramid.
- The Go agent supports Go web applications and APIs for library support and vulnerability reporting.

Contrast supports additional languages via our Scan offering.

Technical integrations

Contrast leverages several different integrations to provide accurate security feedback with tools you are already using. This approach accelerates the software deployment process by encouraging security and development to work together effectively.

Learn more about integrations