SAST





When testing the security of your applications prior to pushing into production, you want to identify vulnerabilities and mitigate your risks as early as possible. You need to make sure you also have the best coverage and that your scans are integrated across your development pipeline and providing accurate results...the list goes on. But what tool do you need?

Two of the main types of application security testing are Static Analysis (SAST) and Dynamic Application Security Testing (DAST).

Below are some of the key differences between SAST and NeuraLegion's modern DAST:

White-box Security Test

Tested from the inside out, has access to the underlying framework, design and implementation.

Requires source code

Analyzing the source code, SAST does not require a deployed application

Language Specific

Each programming language needs to be supported, often with separate tools at a great cost

Maintaining / updating rules is a substantial burden on resource

Many False Positives

Large volume of false positives.

Time wasted on manual validation and doublechecking valid code.

Alert fatigue causing developers to ignore issues, increasing risk

Quick fixes to remove false alarms leads to bad code quality

Finds vulnerabilities earlier in the SDLC

The scan can be executed as soon as code is deemed feature-complete

Cannot discover runtime

Unable to detect run-time and configuration vulnerabilities, major causes of breaches

Lack of third-party component analysis

Not enough to guarantee application security.

Cannot Detect Business Logic Flaws

Unable to understand context, workflow or logic of application

Need to rely on manual testing to carry this out

Microservices Limitations

Only scratches the surface by scanning each microservice *individuallu*.

Cannot test how each microservice interacts and communicates with another

High TCO

Substantial CapEx investment

Expensive ongoing vendor support and updates

Ongoing manual validation of false positives is a drain on security team, adding delays



Black-box Security Test

Tested from the outside in, requiring no prior knowledge of technologies or frameworks

Requires a runtime application

Does not require source code or binaries, analyzes by executing the application

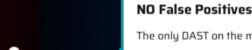


Language Agnostic

Technology and language independent.

Supports any programming language for off-theshelf and custom-built applications

No maintenance required



The only DAST on the market with this capability

Automatic validation of every vulnerability detected with a proof on concept

Real time actionable results for developers to remediate early, with no noise

Immediate snapshot of cyber risk

Finds vulnerabilities early SDLC

Every build / commit / pull request can be tested, using unit tests / mock files to scan even earlier

Enables developers, QA and security to enhance DevSecOps

Can discover runtime issues

Dynamic analysis to detect runtime vulnerabilities.

Only feasible approach to ensuring security of applications that use third party libraries



Can Detect Business Logic Flaws

Only DAST with expanding automated Business Logic Security Testing capabilities

Natural language Processing (NLP) enables understanding of context, responses and logic



Built for Microservices

Full end-to-end tests examine your entire system and verify the entire microservice architecture.

Ability to examine and test each API (SOAP / REST / GraphQL)



Low TCO

SaaS, no CapEx required

Shallow learning curve for developers / QA / Security to learn

No manual validation frees up security for critical

Superior coverage and detection reduces reliance on and cost of manual testing

