Out-of-band application security testing (OAST)

What is OAST security testing?

Out-of-band application security testing (OAST) uses external servers to see otherwise invisible vulnerabilities. It was introduced to further improve the DAST (dynamic application security testing) model. PortSwigger was a pioneer in OAST with Burp Collaborator. This added OAST capabilities to Burp Suite - making the method more readily accessible.

What does OAST testing do that other methods can't?

A web application can contain any number of security vulnerabilities. Many of these bugs are widely known, but vulnerabilities are discovered regularly in software both old and new. Compounding this is the fact that web apps - and the languages they are coded in - tend to be under continuous development. Nothing stays the same for very long.

The dynamic nature of this situation makes things tricky. It means that no amount of testing - and no combination of techniques - is ever likely to find every potential vulnerability in an app. Even if it did, the situation wouldn't last long. Security professionals are in constant competition with cybercriminals, and the consequences of failure can be devastating.

Invisible vulnerabilities - DAST

The main selling point of DAST has always been that it can produce results of a very high quality. If you're browsing a report produced using this method, then you can be almost certain you're looking at actual vulnerabilities. This information can go straight to your development team to be fixed.

But when used in isolation, dynamic testing struggles to detect some types of security vulnerability. Blind and asynchronous bugs are easily missed, for instance. As you will see below, augmenting dynamic testing with OAST goes a long way towards fixing this problem.

False positives - SAST

SAST (static application security testing) is another common method of security testing. It takes effectively the opposite approach to dynamic testing. Where DAST considers an app as an attacker might - from the outside in - SAST looks at the code itself. This approach gives it a different set of benefits and drawbacks.

The main problem here is that because SAST doesn't actually execute any code, it can only see what "might" be going on. This means that in general, SAST will produce a larger, noisier set of results than DAST. This noise comes in the form of false positives. Among these will be real vulnerabilities - but it costs time and money to determine which they are.

Want to learn more about web application security testing?

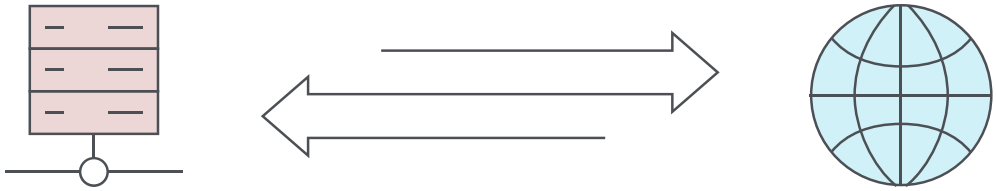
How does OAST work?

OAST improves the results returned by DAST security testing. In many ways, it is itself a dynamic method, albeit one that can see "around corners". This is because "dynamic application security testing" really just denotes a test that can't

see the inner workings of an application. This could also describe OAST.

Attacking from the outside

Conventional dynamic testing is elegant in its simplicity. In essence it sends payloads to a target application and analyses the responses that come back - just like a real attacker might:



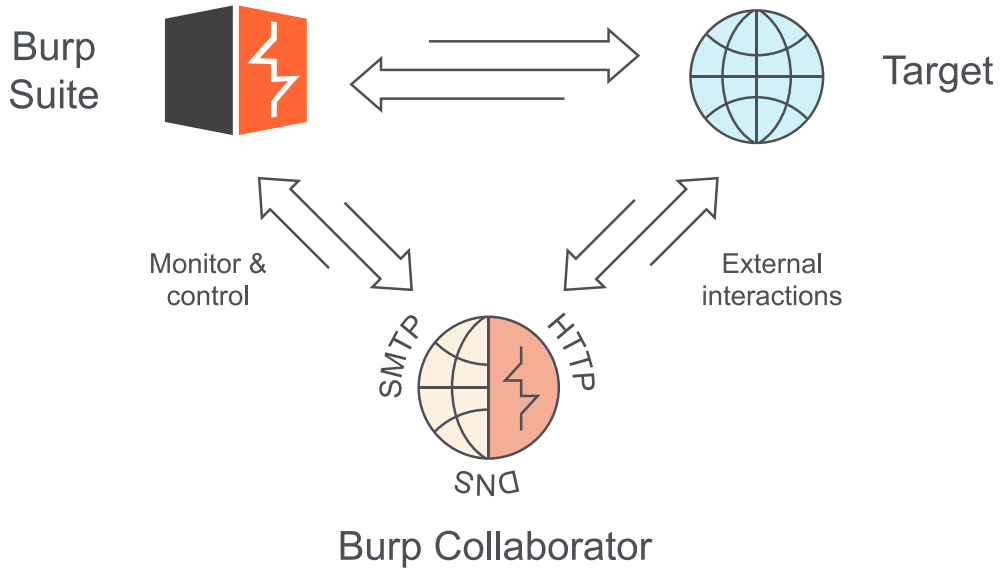
When you send a DAST payload and your target comes back to you with a response suggesting a vulnerability, you can be pretty sure it's real. Dynamic testing has achieved the success it has, because it works well in these situations.

But what if a target app doesn't send back a response to a payload, even though the target is actually vulnerable? This is a particular problem when an app is working asynchronously. Traditional DAST techniques alone just won't see it.

See over the horizon

This is where OAST comes in. When PortSwigger introduced Burp Collaborator, OAST was a revolutionary addition to the field. It allowed Burp Suite to detect a huge new range of bugs, including many blind SQL injection (SQLi), blind cross-site scripting (XSS), and blind OS command injection vulnerabilities.

Burp Collaborator performs OAST by introducing a new channel of communication into the dynamic testing process:



So, what's actually happening here? Well, as we mentioned above, Burp Collaborator can search for a huge range of vulnerabilities that were once invisible to DAST testing.

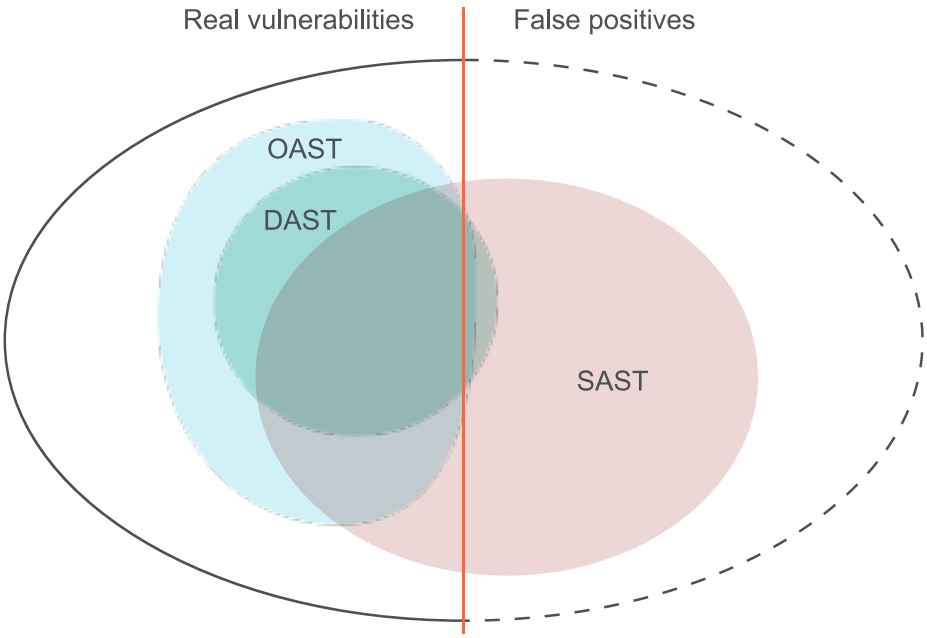
If a vulnerability is blind, then it sends back no useful response to us when we send a test attack - even if that attack is successful. We need a way to bypass this. Out-of-band testing methods are that bypass. It is done by sending an attack payload that causes an interaction with an external system we have control over, that sits outside the target domain.

OAST the easy way

With Burp Collaborator, this is easy to do - even if you don't control an external system to use for this purpose. Burp Suite Enterprise Edition and Burp Suite Professional can both communicate with the Burp Collaborator server for testing purposes. And if you'd rather, then you can always configure a private server to do the same thing.

Burp Collaborator can identify the precise Burp Scanner payloads responsible for each interaction it receives. So if something useful comes back from a target, you'll know exactly what triggered it. This process was designed primarily to be automated - and sits inside Burp Scanner. For advanced users, Burp Suite Professional also includes manual OAST tools.

The advantages of testing out of band



As you can probably see, automated OAST is a powerful technique to add to a security tester's arsenal. The Venn diagram above shows how OAST greatly increases the number of security issues DAST can identify. Some of these might also be picked up by a SAST tool, but in many cases, this would be unlikely.

And OAST comes with all the same advantages as conventional dynamic testing. It rarely produces false positives - meaning that its reports can be trusted.

Like DAST, OAST is agnostic as to the language an app is written in. There's no need for multiple pieces of software, even if you want to scan multiple web apps. This couples well with Burp Suite Enterprise Edition's huge scalability. Your whole web portfolio can now be scanned by a single piece of software.

Does OAST testing have any disadvantages?

We'd be lying if we said that OAST made testing perfect. No method is. There will always be vulnerabilities that DAST and OAST can't see - just as there are others that SAST will miss.

Automated web security scanning is not a panacea for security vulnerabilities. It should always be used in conjunction with regular manual penetration testing. This approach will help to keep your web presence both secure and compliant.

PortSwigger was a pioneer in OAST testing when it introduced Burp Collaborator. This feature comes integrated as part of both Burp Suite Enterprise Edition and Burp Suite Professional. For more information on Burp Suite and how it could

fit into your particular use case, please see the resources below:

Being able to tweak and fine tune the request workflow to be transparent to all tools (Scanner, Repeater, Intruder, Extender) is invaluable for me to properly test multistep operation, adjust CSRF tokens, so that the scan works. With the wide range of different Extenders, almost everything can be done directly from Burp Suite without needing to use Python scripting. I extensively use OAST and IAST, which is the perfect combination for pentesting.

Source: TechValidate survey of PortSwigger customers

See more customer stories →

Andrej Šimko

Security Associate Manager

Learn more

Find out what Burp Suite could do for you, according to your particular use case:

DEVELOPMENT TEAMS ORGANIZATIONS PENETRATION TESTERS