



# Using IAST to Unlock the Benefits of DevSecOps

Jeff Williams, Founder and CTO  
Contrast Security  
@planetlevel  
November 2022

**YOW! LONDON**



# We all blindly trust software with the most important things in life



**Government**



**Social**



**Elections**



**Business**



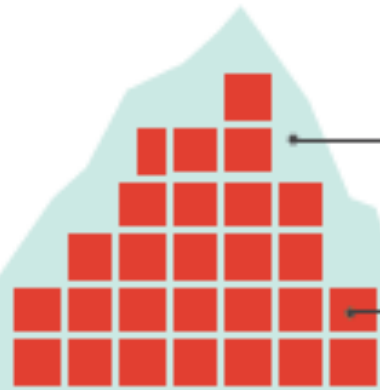
**Power Grid**



**Money**

# The Average Application Has Serious Security Issues

**5,000+** ATTACKS PER MONTH



**21%** CUSTOM CODE

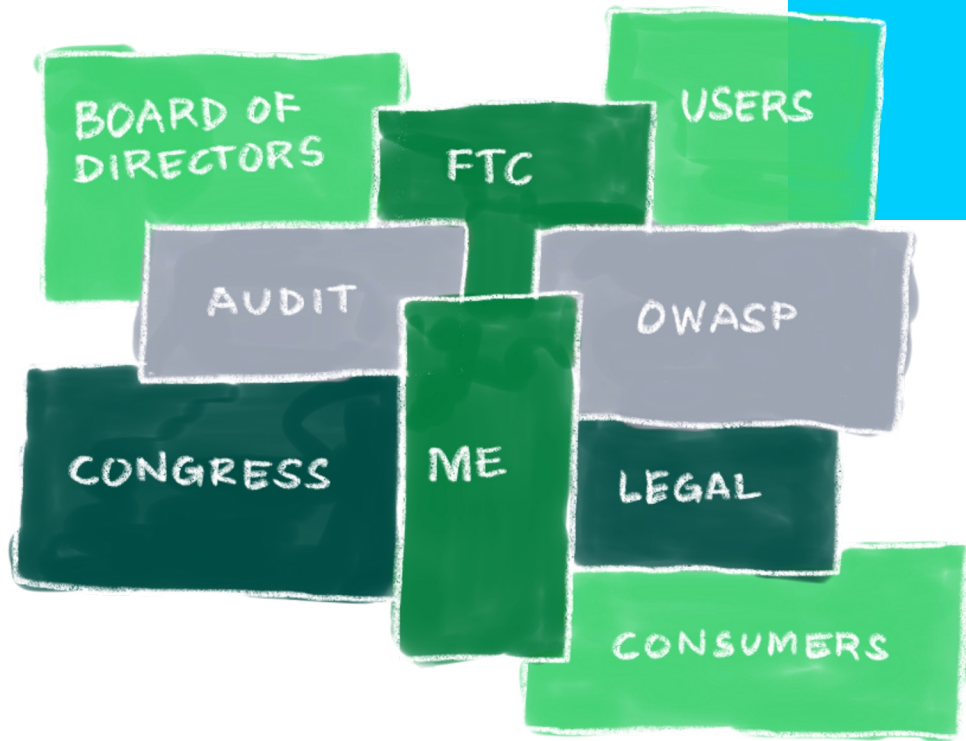
**26.7** SERIOUS VULNERABILITIES

**8.5%** ACTUALLY INVOKED LIBRARY CODE ACROSS 27 LIBRARIES

**70.5%** UNUSED LIBRARY CODE ACROSS 30 LIBRARIES

**2.0** VULNERABILITIES (CVE)

# Public expectations don't match reality







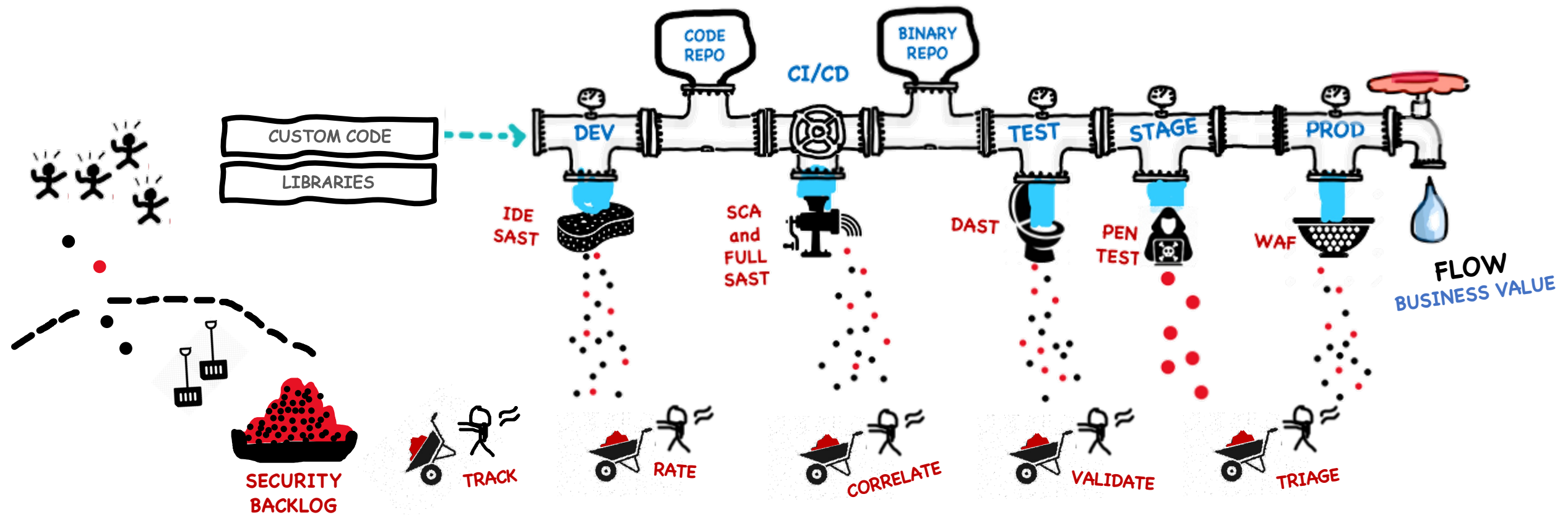
SECURITY  
TRANSPARENCY  
IS COMING

# DevSecOps Will Fix Everything! /s

- **Problem:** **software** was poor quality, late, slow, and doesn't provide business value.
- **Approach:** DevOps
- **Results:**
  - 200x more deploys
  - 24x faster service recovery
  - 3x less change failures
  - 22% less unplanned work

- **Problem:** **security** is poor quality, late, slow, and doesn't provide business value.
- **Approach:** DevSecOps
- **Results?**
  - 10x faster security results?
  - 80% less vulnerabilities in prod?
  - 0x increase in time to market?
  - No log4j-style firedrills?

# So Why Does DevSecOps Feel Awful?



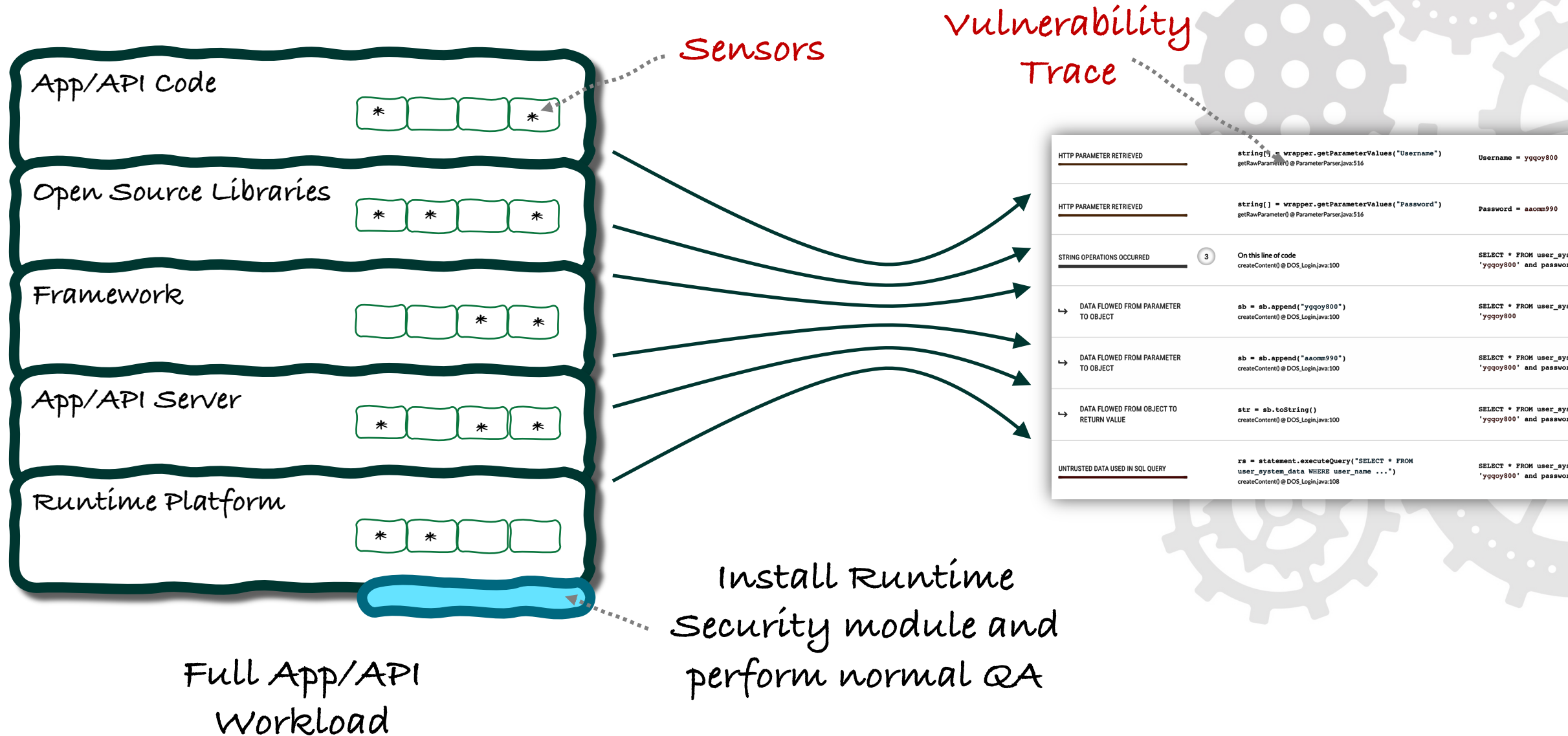




**INSTRUMENTATION CHANGES EVERYTHING**



# Security Instrumentation!



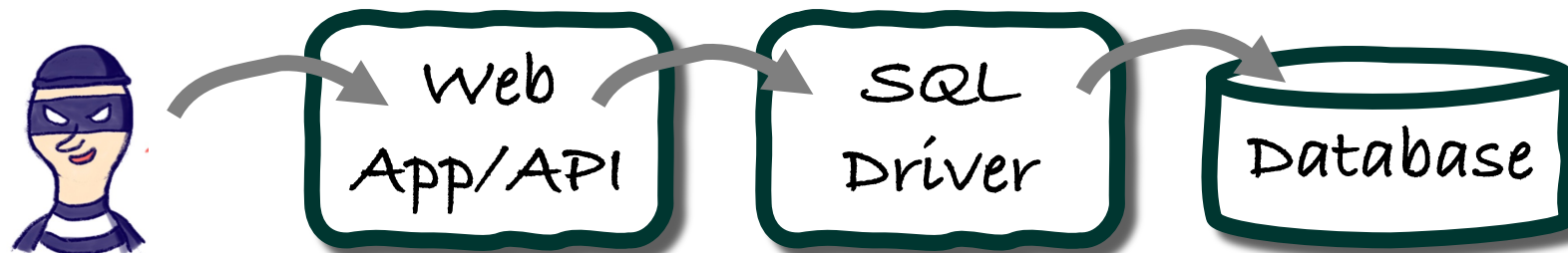
# Example: Detecting SQL Injection

```
String name=request.getParameter("Username");  
String pass=request.getParameter("Password");  
String query="select * from user_system_data \\  
             where user='" + name + "' and user='" + pass + "'";  
Statement statement = connection.createStatement(...);  
ResultSet results = statement.executeQuery( query );
```

No escaping or  
parameterization!

?name=ygqoy800 &  
pass=aaommm990

select \* from users where  
user='ygqoy800' and pass='aaommm990!'



HTTP PARAMETER RETRIEVED

```
string[] = wrapper.getParameterValues("Username")
getRawParameter() @ ParameterParser.java:516
```

Username = ygqoy800

HTTP PARAMETER RETRIEVED

```
string[] = wrapper.getParameterValues("Password")
getRawParameter() @ ParameterParser.java:516
```

Password = aaomm990

*Security trace -  
data flow events*

STRING OPERATIONS OCCURRED

3

On this line of code  
createContent() @ DOS\_Login.java:100

```
SELECT * FROM user_system_data WHERE user_name =
'ygqoy800' and password = 'aaomm990'
```

→ DATA FLOWED FROM PARAMETER  
TO OBJECT

```
sb = sb.append("ygqoy800")
createContent() @ DOS_Login.java:100
```

```
SELECT * FROM user_system_data WHERE user_name =
'ygqoy800'
```

→ DATA FLOWED FROM PARAMETER  
TO OBJECT

```
sb = sb.append("aaomm990")
createContent() @ DOS_Login.java:100
```

```
SELECT * FROM user_system_data WHERE user_name =
'ygqoy800' and password = 'aaomm990'
```

→ DATA FLOWED FROM OBJECT TO  
RETURN VALUE

```
str = sb.toString()
createContent() @ DOS_Login.java:100
```

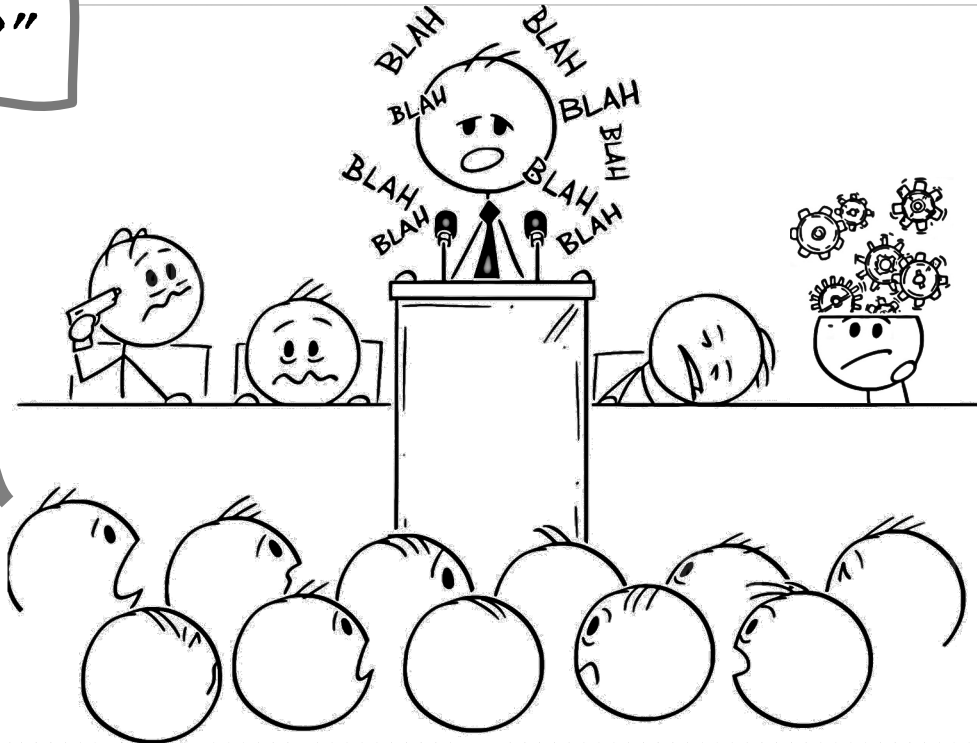
```
SELECT * FROM user_system_data WHERE user_name =
'ygqoy800' and password = 'aaomm990'
```

UNTRUSTED DATA USED IN SQL QUERY

```
rs = statement.executeQuery("SELECT * FROM
user_system_data WHERE user_name ...")
createContent() @ DOS_Login.java:108
```

```
SELECT * FROM user_system_data WHERE user_name =
'ygqoy800' and password = 'aaomm990'
```

"Big deal –  
another  
security tool?"



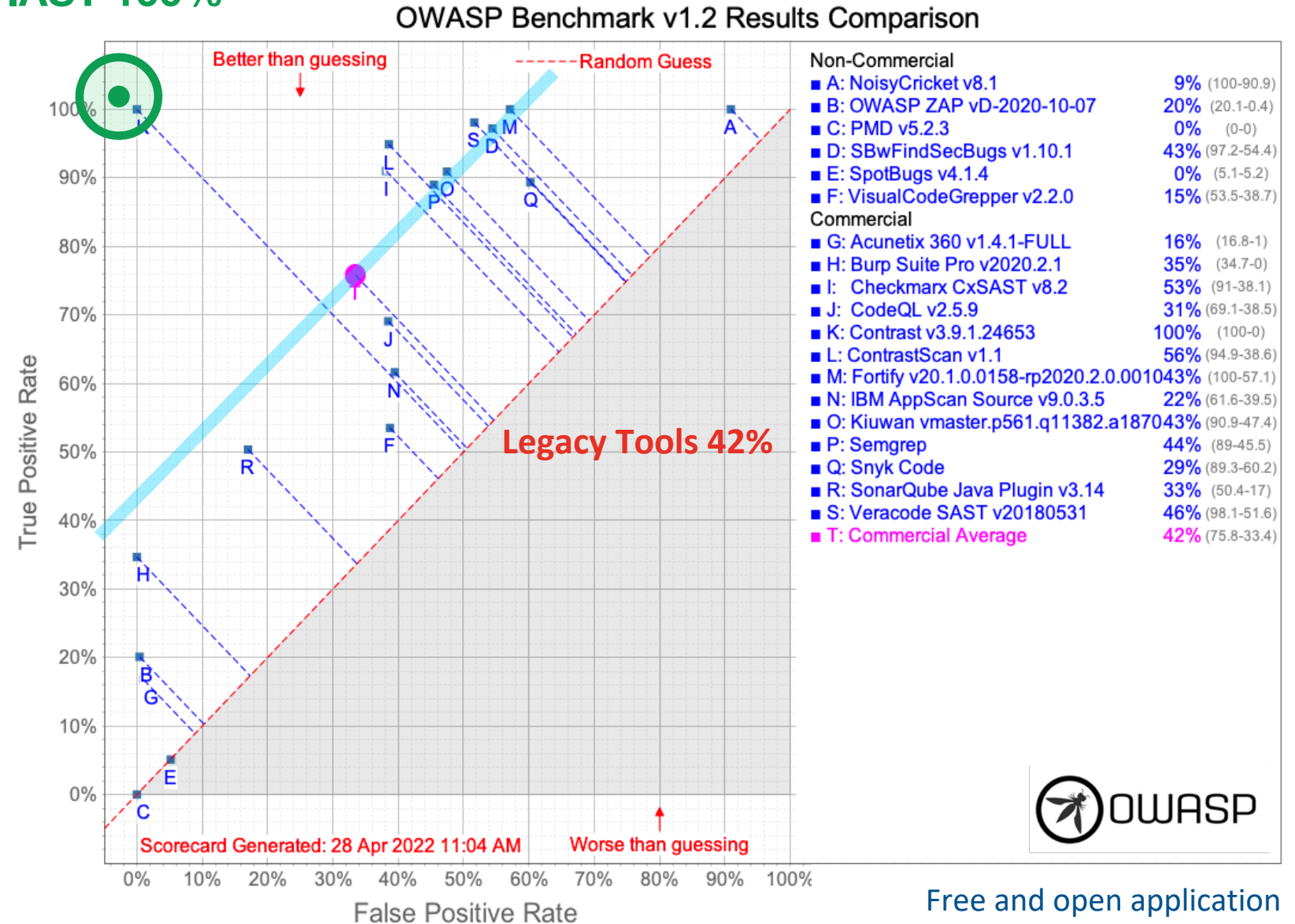
With IAST...

ANYONE can get  
instant, detailed, and  
accurate security  
feedback on their  
projects without ANY  
extra work.



Accuracy is  
the most  
developer-  
friendly  
feature in a  
security  
tool

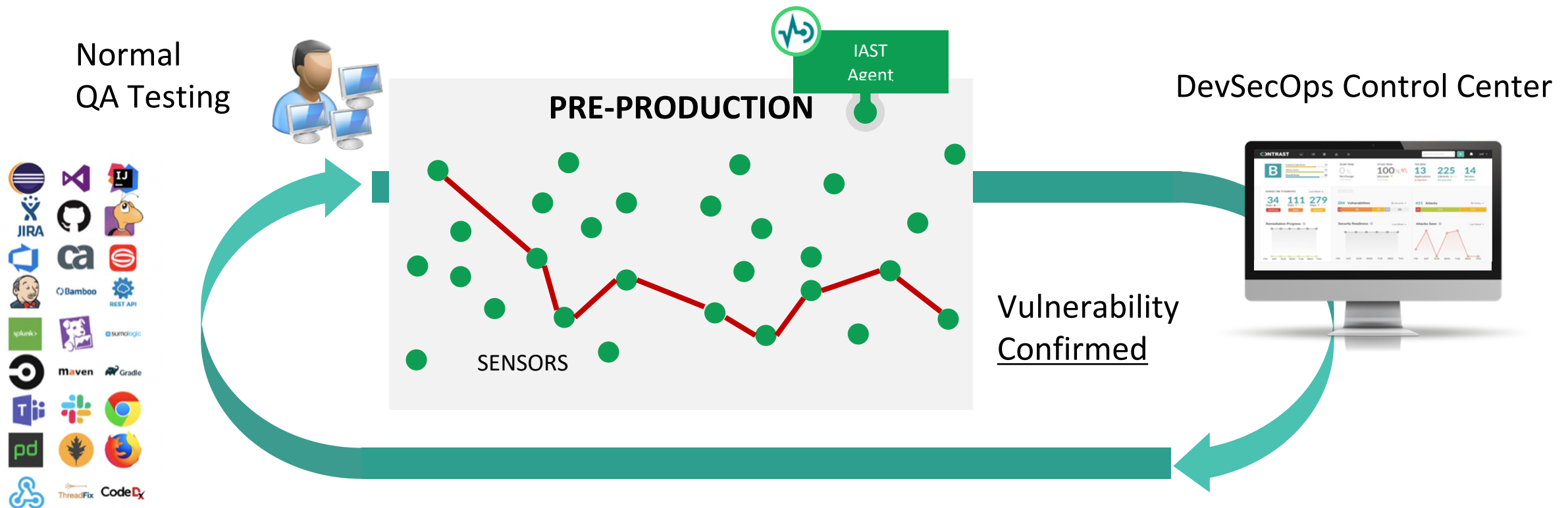
IAST 100%



Free and open application  
benchmark with thousands  
of security test cases

# Interactive Application Security Testing (IAST)

-- using instrumentation to detect vulnerabilities during normal QA



# Runtime vulnerability snapshots

## What happened?

We tracked the following data from "lastName" Parameter:

```
GET /owners?lastName=Yow%21+LONDON
```

```
lastName=Yow! LONDON
```

...which was accessed within the following code:

```
org.hibernate.jpa.spi.AbstractEntityManagerImpl#createQuery(), line 305
```

...and ended up in this database query:

```
SELECT DISTINCT owner FROM Owner owner left join fetch owner.pets WHERE owner.la  
'Yow! LONDON%'
```

HTTP view, Code view,  
and Data view – FULL CONTEXT

Replay Request

```
GET /owners?lastName=Yow%21+LONDON HTTP/1.0
Sec-Fetch-Mode: navigate
Cookie: jenkins-timestamper-offset=18000000; jenkins-timestamper=system; jen
ASP.NET_SessionId=gztggqnanuc4ausaaocgr3ci; JSESSIONID=2E651D61C504B1
Referer: http://localhost:8000/owners/find
Sec-Fetch-Site: same-origin
Accept-Language: en-US,en;q=0.9
Cookie: jenkins-timestamper-offset=18000000; jenkins-timestamper=system; jen
ASP.NET_SessionId=gztggqnanuc4ausaaocgr3ci; JSESSIONID=2E651D61C504B1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (Kl
Sec-Fetch-User: ?1
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/
Sec-Ch-Ua: "Google Chrom
Sec-Ch-Ua-Mobile: ?0
Sec-Ch-Ua-Platform: "Wir
Host: localhost:8000
Upgrade-Insecure-Request: 1
Connection: keep-alive
Accept-Encoding: gzip, deflate
Sec-Fetch-Dest: document

HTTP PARAMETER
RETRIEVED

string[] =
facade.getParameterVal
ues("lastName")
lastName = Yow! LONDON
getParametersStartingWith() @
WebUtils.java:672
```

### HTTP Parameter Retrieved

Class.Method	org.apache.catalina.connector.RequestFacade.getParameterValues(java.lang.String)
Object	org.apache.catalina.connector.RequestFacade@252c198d
Return	[Yow! LONDON] *
Parameters	Parameter 0: lastName*
Stack Trace	All Code

Copy

```
org.apache.catalina.connector.RequestFacade.getParameterValues(RequestFacade.java:427)
```

```
org.springframework.web.util.WebUtils.getParametersStartingWith(WebUtils.java:672)
```

```
org.springframework.web.bind.ServletRequestParameterPropertyValues.<init>(ServletRequestParameterProp
ertyValues.java:77)
```

```
org.springframework.web.bind.ServletRequestParameterPropertyValues.<init>(ServletRequestParameterProp
ertyValues.java:52)
```

```
org.springframework.web.bind.ServletRequestDataBinder.bind(ServletRequestDataBinder.java:100)
```

# Runtime library analysis

spring-core-4.3.6.release.jar (v4.3.6.RELEASE)

Released: Jan 25, 2017 | SHA1: 690da099c3c2d2536210f0fd06ff3f336de43ad9 | License: Apache-2.0

Overview Usage



2/100

6

Vulnerabilities

0

Policy  
Violations

10

Apps  
Using

498/790

Classes Used

## What happened?

This library has known CVEs

## What's the risk?

**CRITICAL** 9.8 CVE-2018-1275

**CRITICAL** 9.8 CVE-2018-1270

**HIGH** 7.5 CVE-2018-1272

**MEDIUM** 6.5 CVE-2018-1257

**MEDIUM** 4.3 CVE-2021-22060

**MEDIUM** 4.3 CVE-2021-22096

## Applications

PetClinic-AF

PetClinic-PD

## Servers

PetClinic-bw-Dev

PetClinic-bw-QA

PetClinic-bw-Prod

PetClinic-KK-QA

PetClinic-KK-Prod

PetClinic-kk-Dev

PetClinic-kk-QA

AST and SCA are  
one thing

... and should always  
be performed by  
one tool

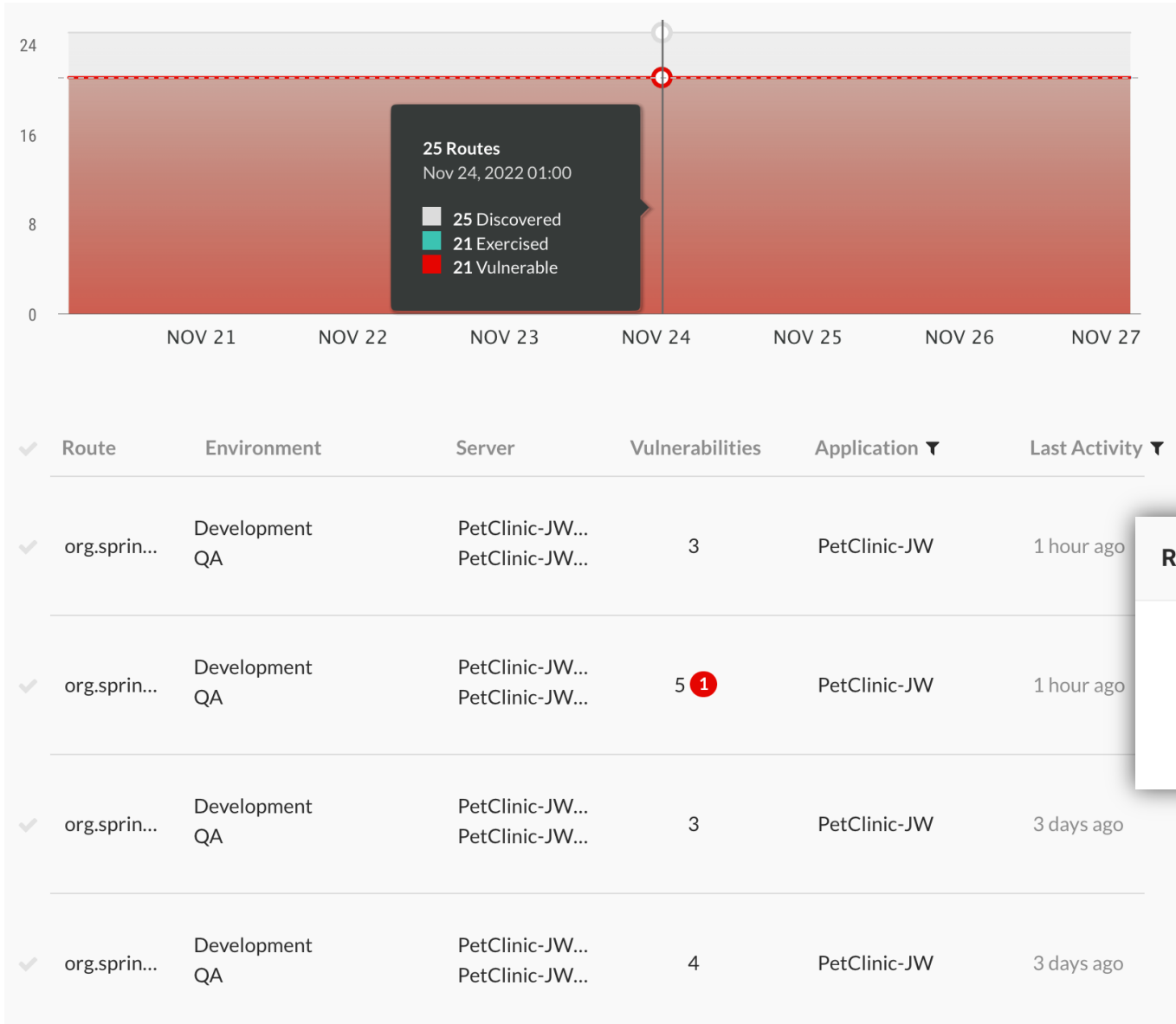
Classes Loaded (260/451)	First Seen	Last Seen
org.springframework.boot.logging.logback.LogbackLoggingSystem	9 months ago	2 months ago
org.springframework.boot.CommandLineRunner	9 months ago	2 months ago
org.springframework.boot.SpringApplicationRunListeners	9 months ago	2 months ago
org.springframework.boot.logging.logback.configuration.ConfigurationComparator	9 months ago	2 months ago
org.springframework.boot...		
org.springframework.boot...		
org.springframework.boot...		
org.springframework.boot...		
org.springframework.boot.bind.RelaxedDataBinder\$BeanPath\$MapIndexNode	9 months ago	2 months ago

Measure exactly what classes  
are used by your code

Find both known and unknown  
vulnerabilities in libraries



# Runtime route coverage



Ensure all exposed endpoints are security tested

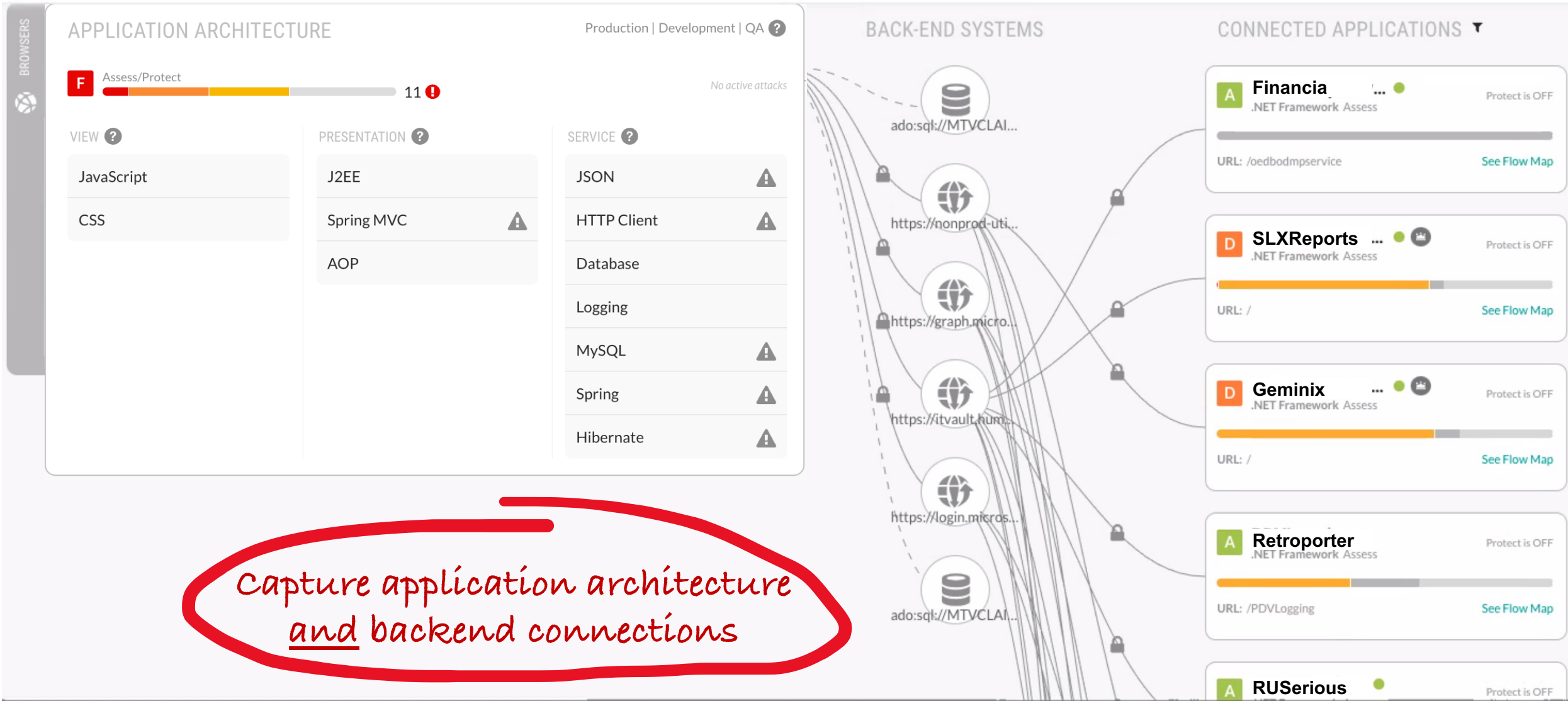
**Route URLs** ×

- POST /owners/new

Close

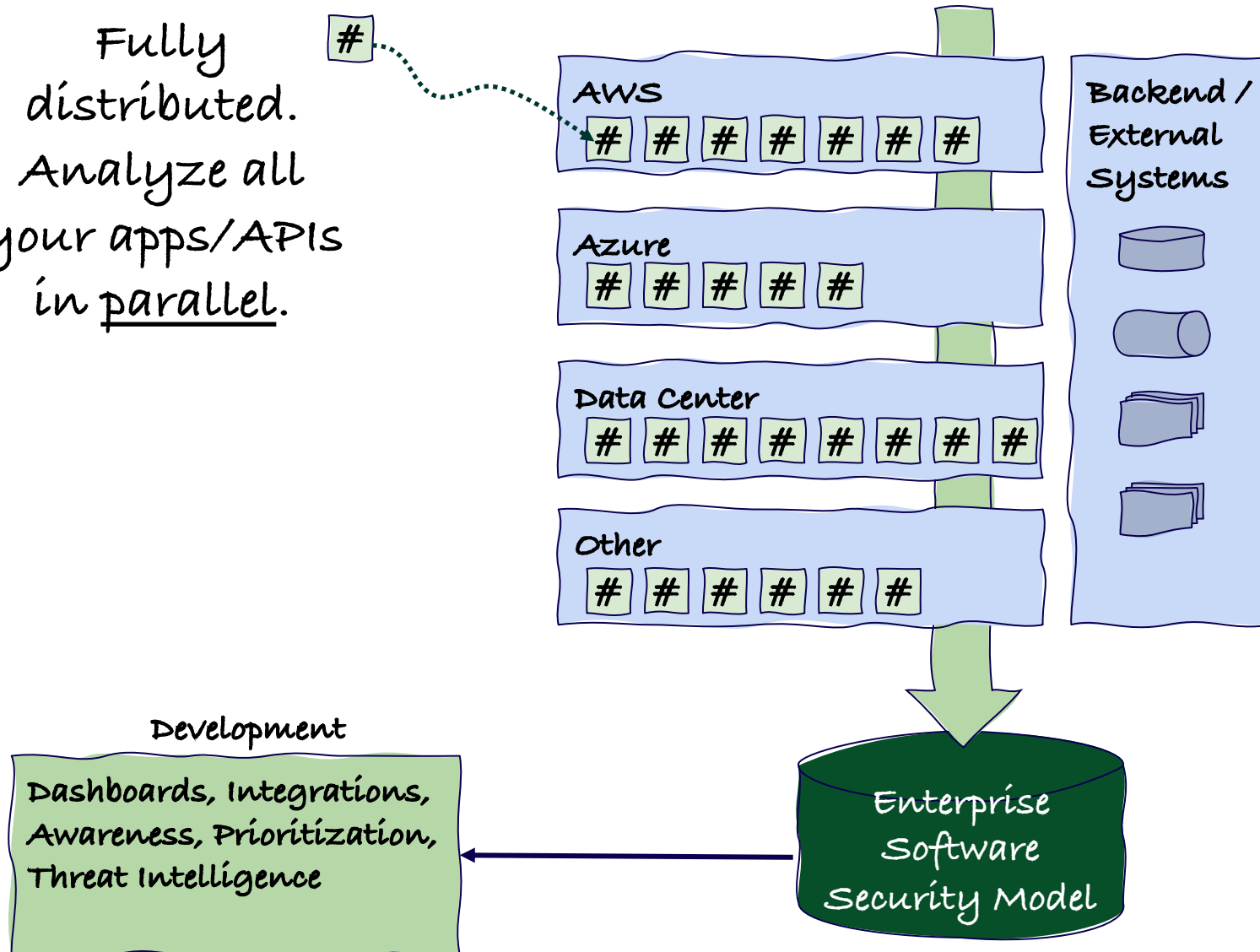
Discover unknown routes from frameworks, plugins, old code, etc...

# Runtime architecture diagrams



# Deploying IAST at Scale

Fully distributed.  
Analyze all  
your apps/APIs  
in parallel.



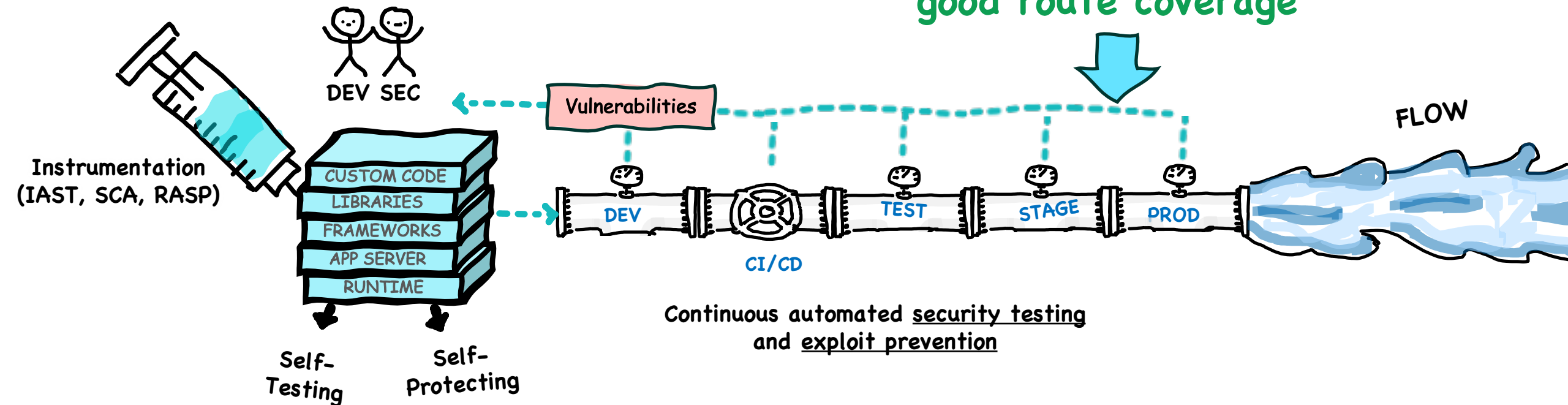
Deploy IAST  
automatically to all your  
servers via:

- Kubernetes operator
- Platform engineering
- Gold Server
- Container build
- Etc...

# DevSecOps - Getting Secure Code Moving

Automatically cleared  
for production if...

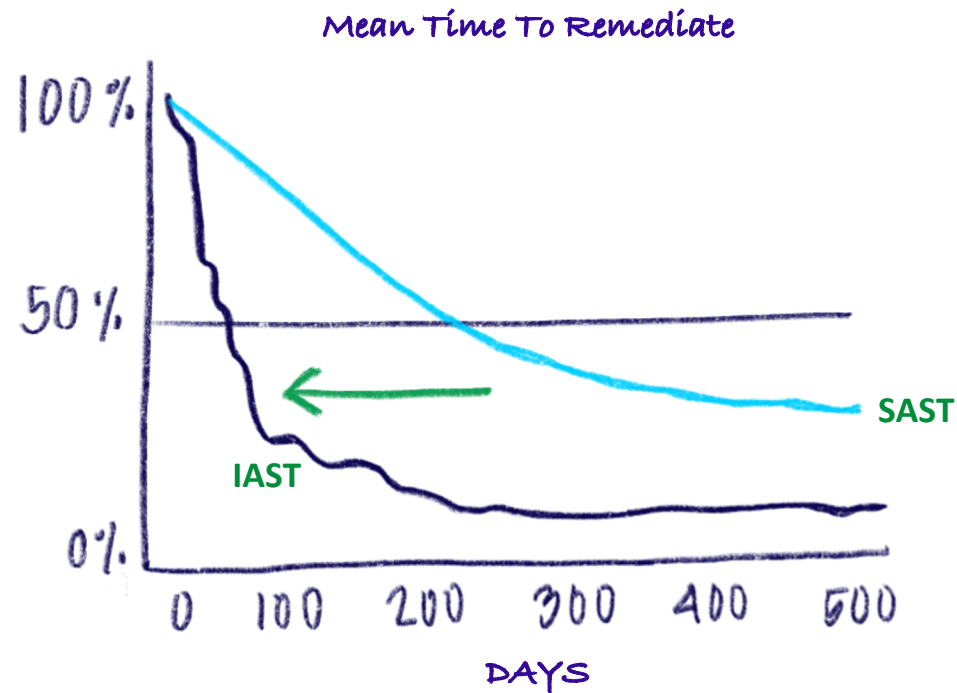
clean code +  
clean libraries +  
good route coverage



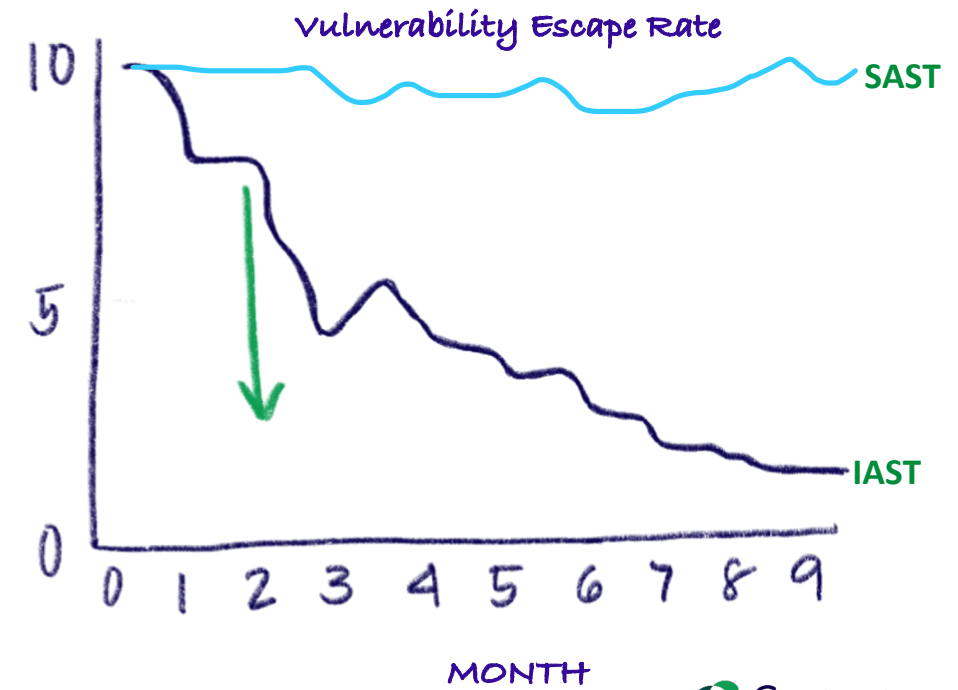


# Metrics that matter

Reduce  
MTTR to < 1  
week



Reduce  
VER to < 1  
per month





# Ask me ANYTHING!

**Jeff Williams @planetlevel**  
Cofounder and CTO  
Contrast Security



Follow us **@goto\_con**  
Share your experience using **#YOWlondon**



Follow us **@GOTOcon**  
Share your experience using **#YOWlondon**



Follow us **GOTO Conferences**  
Join **322k+ subscribers & 36m+ views**

=====EXTRA=====



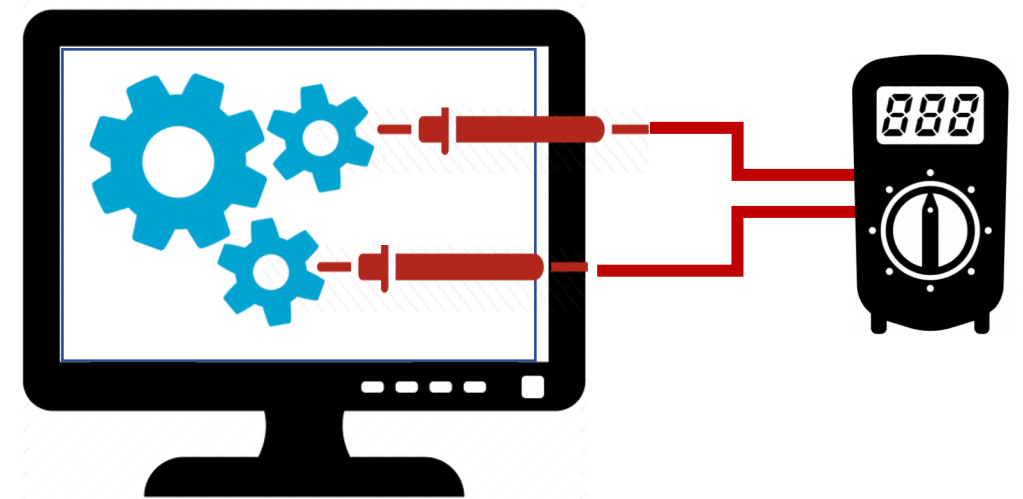
# Try the Java Observability Toolkit (JOT)

Free and Open Source

Make your own runtime protection with a few lines of yaml

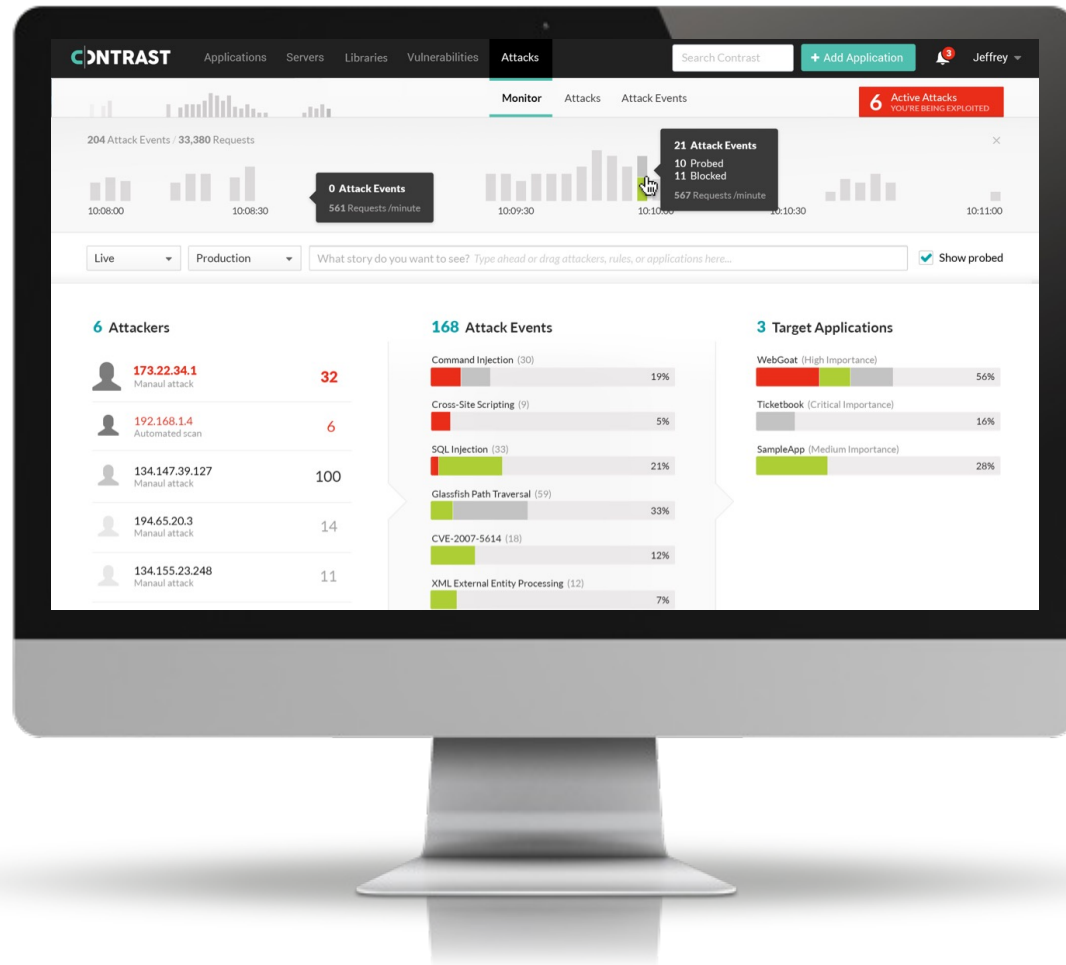
```
- name: "sandbox-expressions"
  description: "Prevents harmful methods from being used"
  methods:
    - "java.lang.ProcessBuilder.<init>"
    - "java.io.Socket.<init>"
  scopes:
    - "javax.el.ValueExpression.getValue"
  captures:
    - "#p0"
  exception: "Attempt to escape expression language sandbox"
```

**$\${JOT}$**



<https://github.com/planetlevel/jot>

# Try it for free...



### SQL Injection Event from 18.59.254.198

**BLOCKED** When: 01/25/2017 03:35 PM URL: /WebGoat/attack

**Overview** Details Request Discussion

We observed the following suspicious value enter the application through the HTTP Request Parameter "Password":

```
POST /WebGoat/attack?Screen=1314&menu=1200 HTTP/1.0
Password=%27+or+1%2B2%3D3+--+&SUBMIT=mzjgk305&Username=cprft051
```

This value was again observed altering the meaning of the SQL query executed within org.hsqldb.jdbc.Statement.executeQuery(Unknown Source):

```
SELECT * FROM user_system_data WHERE user_name = 'cprft051' and password = '' or 1+2=3 --'
```

We blocked this attack.

**Code Location**

File: Unknown File for class org.hsqldb.jdbc.Statement.java  
Method: executeQuery()  
Stack:

```
org.hsqldb.jdbc.Statement.executeQuery(Unknown Source)
org.owasp.webgoat.plugin.DOS_Login.createContent(DOS_Login.java:108)
org.owasp.webgoat.Lessons.AbstractLesson.handleRequest(AbstractLesson.java:868)
org.owasp.webgoat.HammerHead.makeScreen(HammerHead.java:300)
org.owasp.webgoat.HammerHead.doPost(HammerHead.java:148)
Show more...
```

<https://contrastsecurity.com/ce>