

DOCKTER-TOM

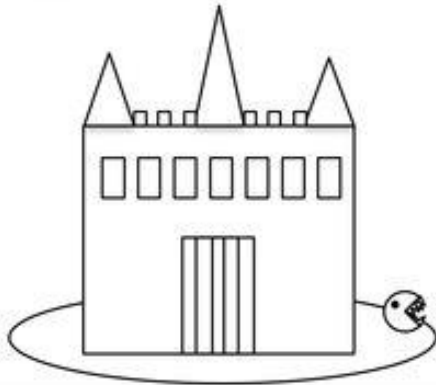
TOM PORTER, PHD

3 FEBRUARY 2018

Second Alligator

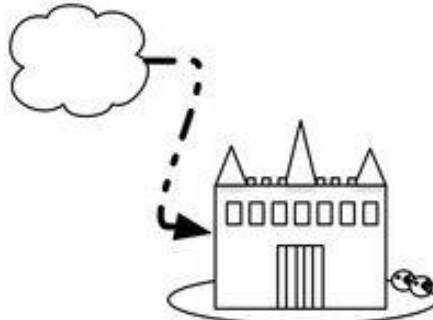
Who turned out the lights?

Sire, all is good with our layered defenses... Our moat keeps out the unauthorized thieves and the castle protects our most precious assets. We have secret rooms and passages that make it impossible to steal from us. We are secure.



Traditional Security

Sire, our plan is to put the "cloud" inside our castle to protect our assets... We'll need a second alligator to ensure we have coverage. We know how to operate this way and there is nothing different about the "cloud"... We assure you we've got it...



Traditional Security

(c) 2015 devsecops.org

Hey - who turned out the lights?

Uh - the cloud is bigger than our castle ???

Ah poop, quick someone - find your way to the assets and safeguard them with your lives...

...Ouch

Can someone help me get this alligator off my leg so we can figure out how to fix this?

...Ouch

...Ouch

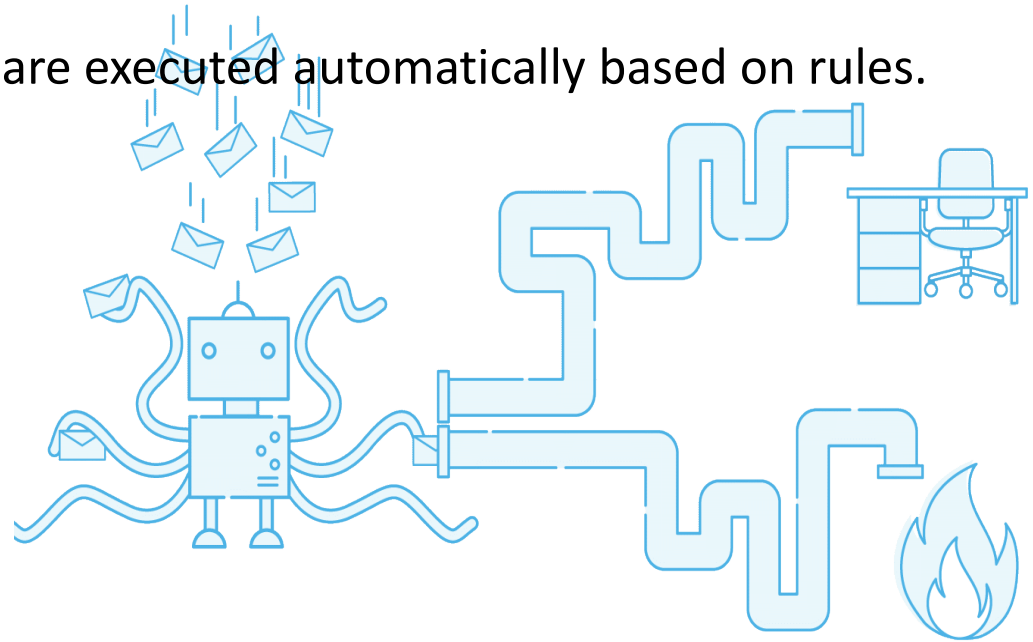
Yes sire, we were wrong. We are working to restore the lights...



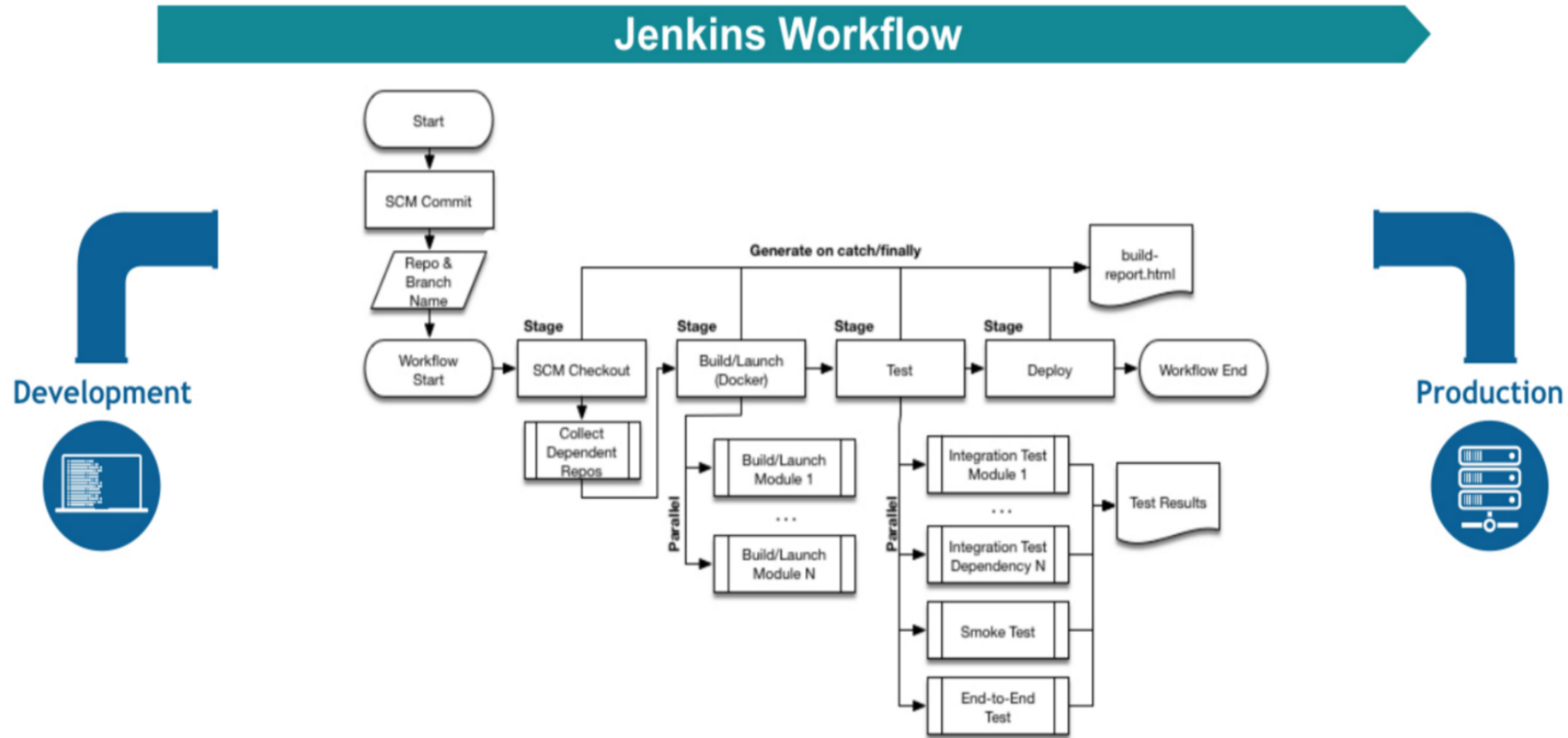
Cloud Security

The Future of Work is Workflow Automation

- A workflow is the series of steps taken to achieve a goal or result.
- A digital workflow is that same process, but occurs on a computer or device.
- Workflow automation is when those processes are executed automatically based on rules.



Sample Jenkins Workflow



The Problem: Coders versus Security Pros

Many security professionals don't code. Likewise, the majority of software developers know very little about security. How can we bridge this gap?

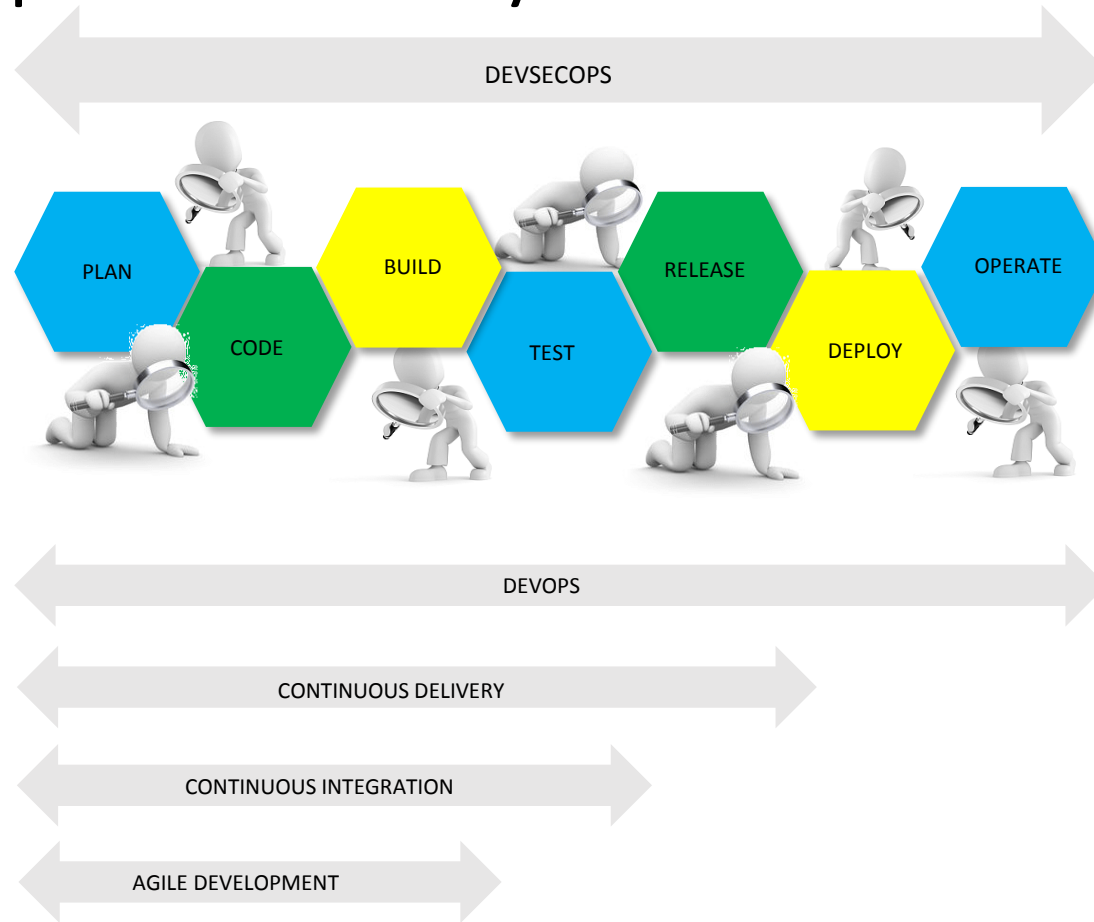


Currently, security is a one-shot affair – relying on some penetration testing to catch bugs. If we are honest, we would acknowledge that this hinders the ability to release software fast and often (a core tenet of DevOps).

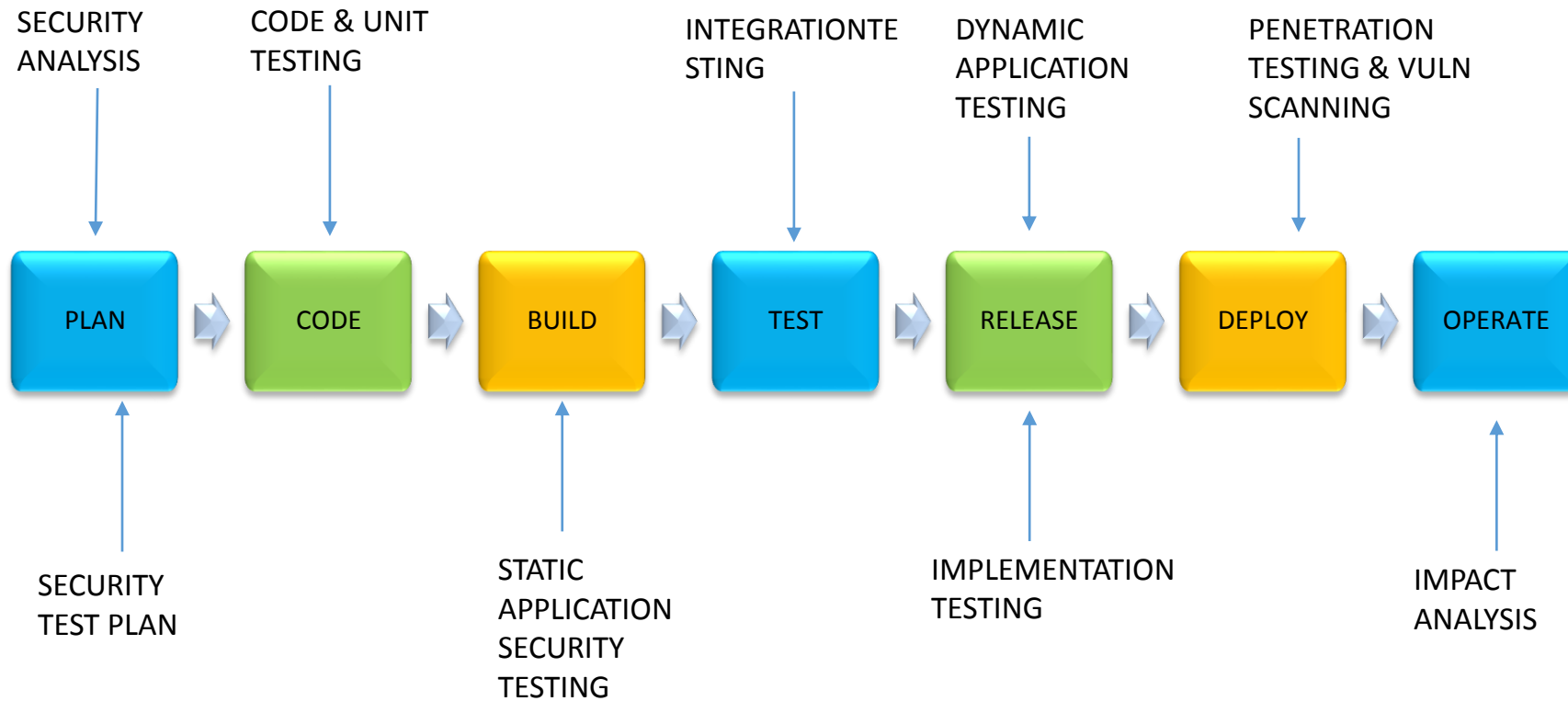
“The 2 DevSecOps Truths”

- DevSecOps: Information security architects must integrate security into DevOps workflows at multiple points and in a collaborative way that is largely transparent to developers, while preserving the teamwork, agility, speed, & mindset of DevOps and agile development environments.
- Software testing does not guarantee that software will be bug-free; rather software testing results are a means of providing an estimate of software quality to stakeholders at multiple levels.

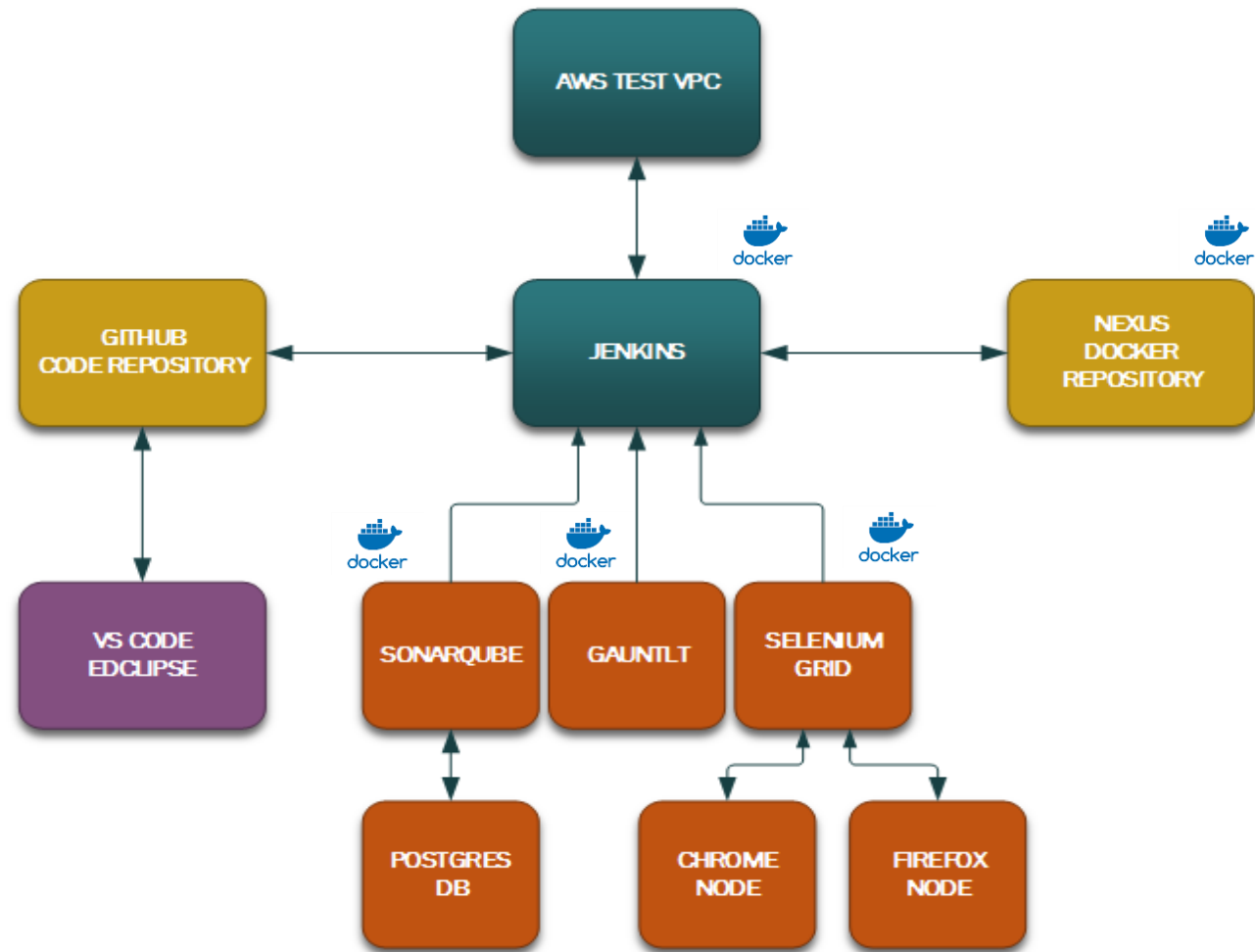
DevSecOps: Security Across the Pipeline



DevSecOps: Security Across the Pipeline



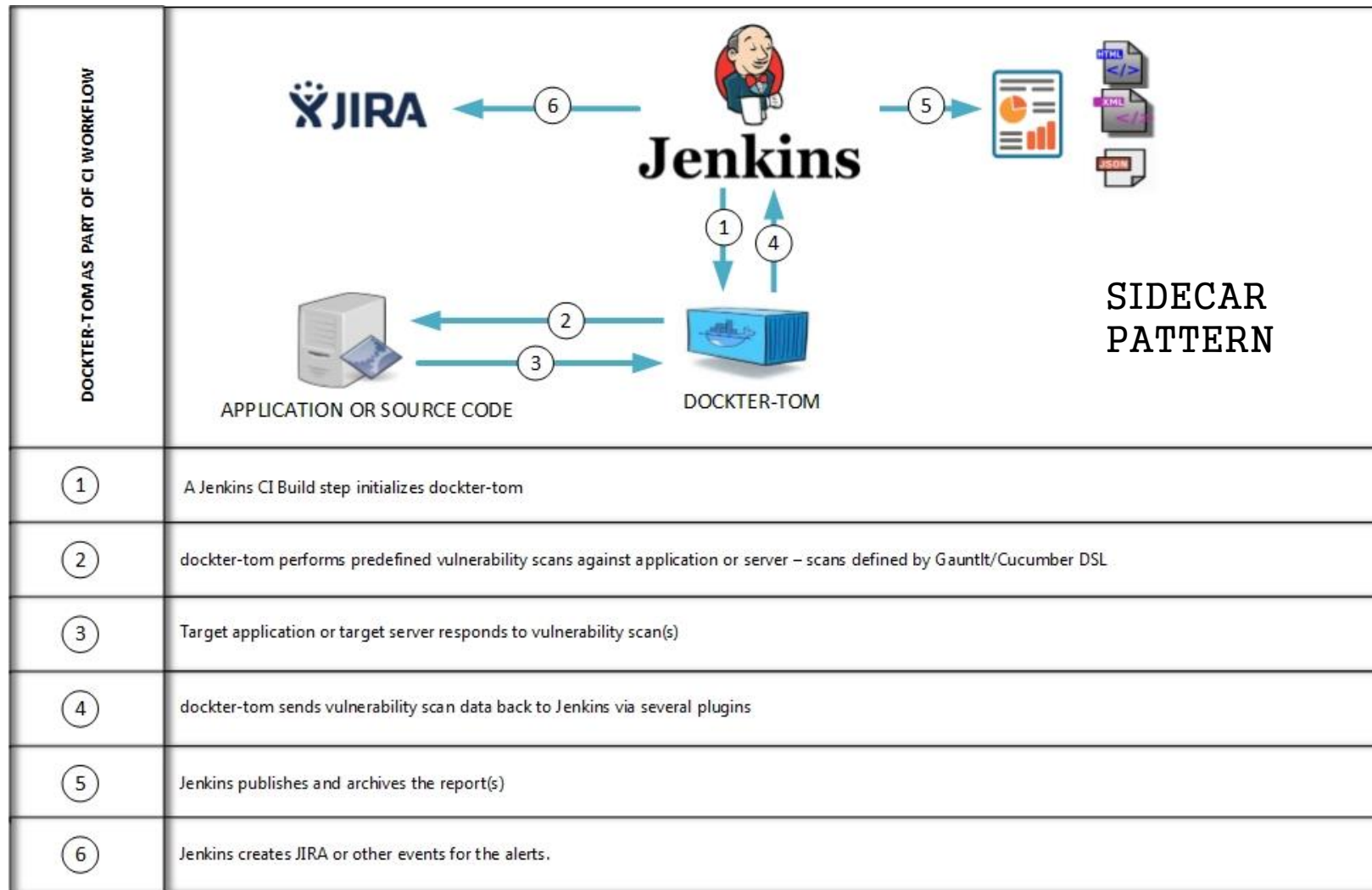
DevSecOps Tools: How to do DevSecOps at Home (an example)



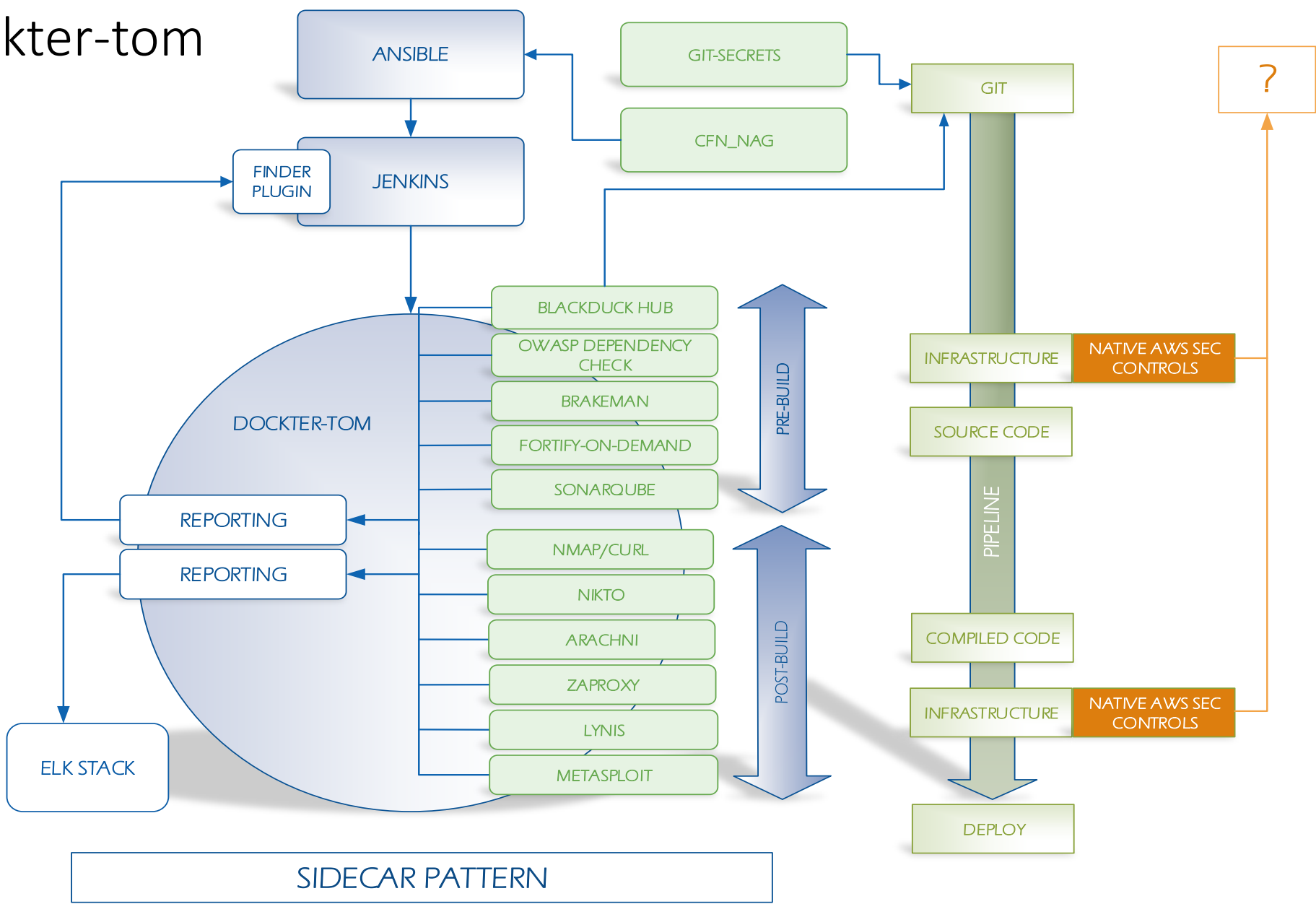
Vulnerability Scanning Definitions

- **SAST Static Application Security Testing** is a set of technologies designed to analyze application source code, byte code and binaries for coding and design conditions that are indicative of security vulnerabilities. The goal of **SAST** is to identify vulnerabilities in your source code before you deploy to production. **SAST** solutions are often programming language-specific, & analyze an application from the “inside out” in a nonrunning state. Common language-independent SAST tools include: Sonarqube, Sentry, HP Fortify SCA, Coverity, Checkmarx, & Veracode.
- **DAST Dynamic Application Security Testing** technologies are designed to detect conditions indicative of a security vulnerability in an application in its running state. Most **DAST** solutions test only the exposed HTTP, HTML, REST, & SOAP interfaces of Web-enabled applications; however, some solutions are designed specifically for non-Web protocol and data malformation such as remote procedure calls. Popular DAST tools include: nmap, Metasploit framework, Burp Suite, Nikto, Arachni, ZAPProxy, SQLMap, Lynis, & vendors -- Rapid7 & Accunetix.
- **HAST/IAST Hybrid Application Security Testing** as you would imagine, combine features of both **SAST** & **DAST** technologies. Popular HAST/IAST Vendors include: Contrast Security, Synopsis, Veracode, & Checkmarx.

Dockter-tom: Workflow



dockter-tom



CUCUMBER: Behavior Driven Development

@cwe-89

Scenario: The application should not contain SQL injection vulnerabilities

And the **SQL-Injection** policy is enabled

And the attack strength is set to **High**

And the alert threshold is set to **Low**

When the scanner is run

And the following false positives are removed

url	parameter	cweId	wascId

And the XML report is written to the file **build/zap/sql_injection.xml**

Then no **Medium** or higher risk vulnerabilities should be present

@cwe-295-auth

Scenario: Present the login form itself over an HTTPS connection

Given a new browser instance

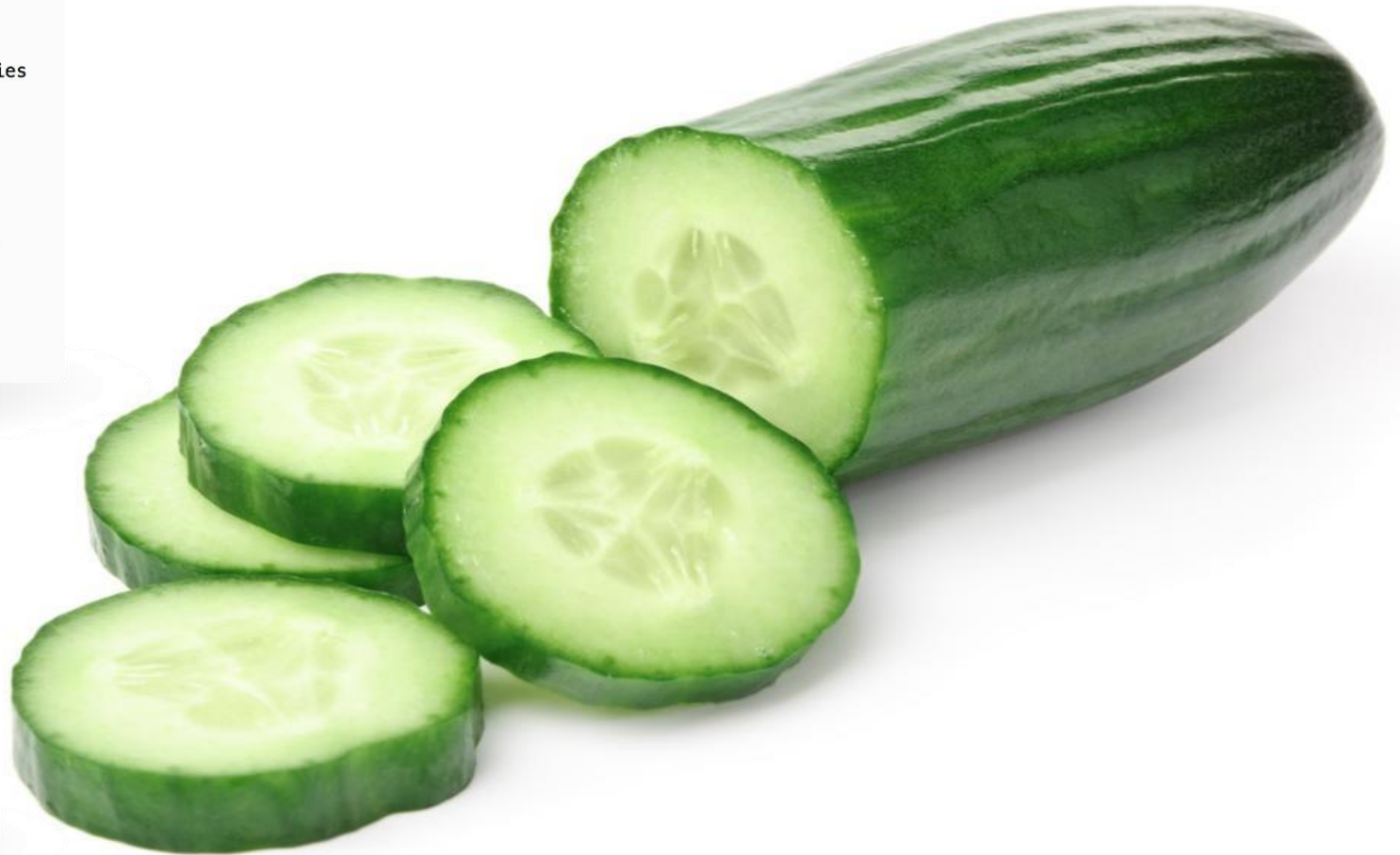
And the client/browser is configured to use an intercepting proxy

And the proxy logs are cleared

And the login page is displayed

And the HTTP request-response containing the login form

Then the protocol should be HTTPS



AWS Native Security Services

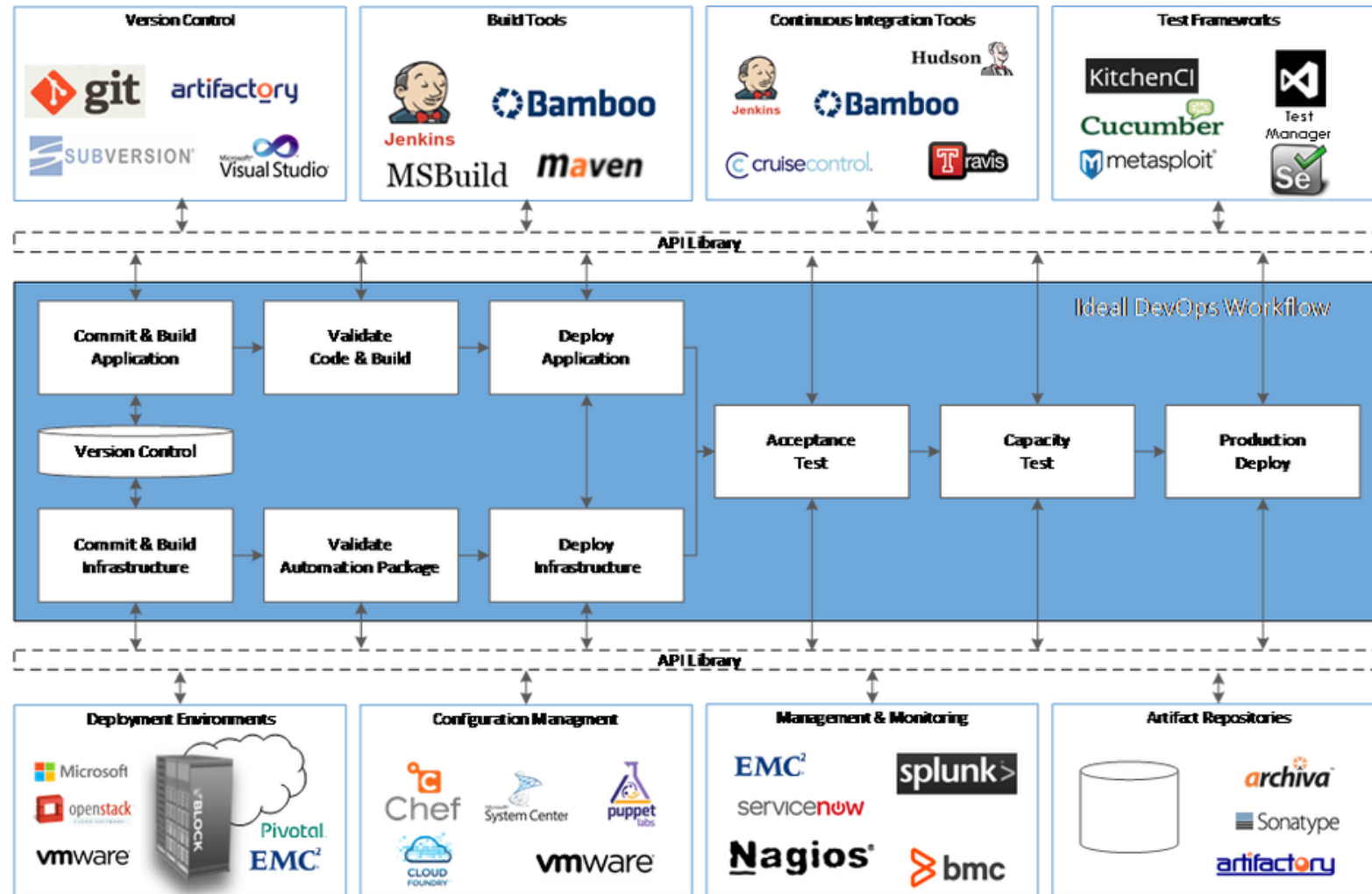
AWS Service	Summary
CloudWatch logs & logging agents	Monitor, store, and access many types of logs
CloudTrail logs	Records all API calls
IAM - Users, groups, roles, policies	Manage “least-privilege” access
VPC's	Network segregation
Security Groups	Inbound firewall
Network ACL's (NACL's)	Inbound/Outbound ACL's
VPN's & VPG's	Encrypted remote access
Selective encryption of data-at-rest	Encryption
API Gateways	Control access with IAM roles
SNS	Automated messaging to SMS
Amazon Inspector	Agent-based security assessments
IAM User Access Advisor	Monitor user, role access
Trusted Advisor	Monitor service limits
Amazon Macie	Classify and protect sensitive data in AWS
Amazon GuardDuty	Managed threat detection

BACKUPS

DevSecOps Tools – A Model I



DevSecOps Tools – A Model II



“Your sh*t is broken”

1. View actual code in stack traces, with source maps for JS, symbolication for iOS, and stack locals for Python bug tracking.
2. See the error's parameters and session information in the crash report.
3. Filter noisy stack traces with app, framework, and raw error views.