



ORACLE

Towards Intelligent Application Security

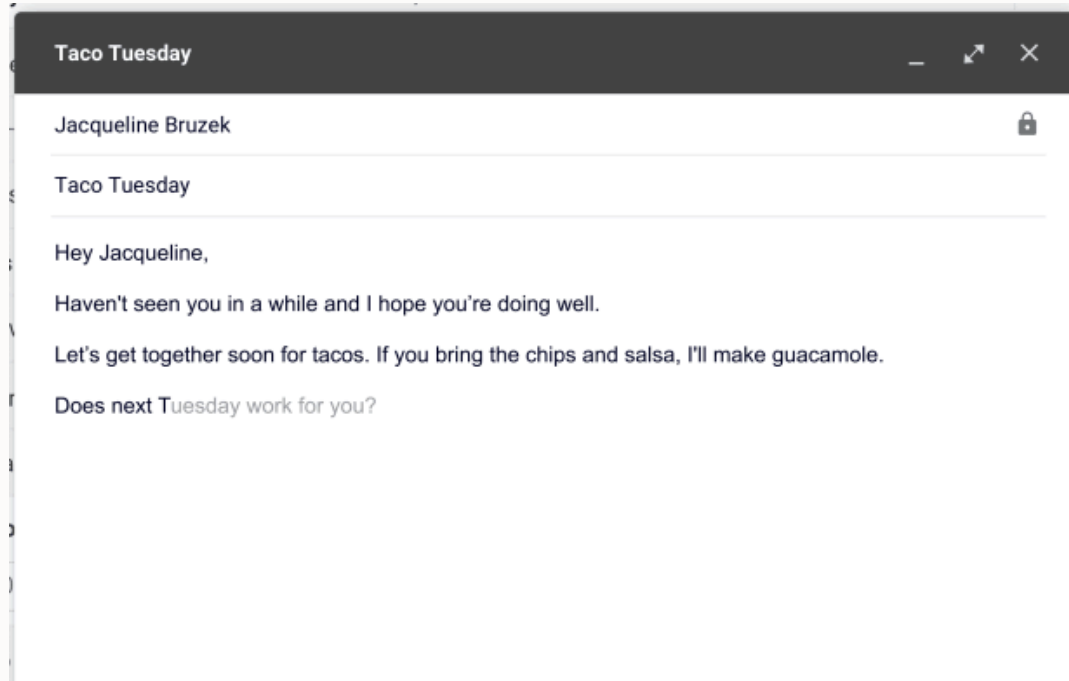
Cristina Cifuentes

Oracle Software Assurance

April 27th, 2023

We Are Living With Intelligent Applications

Gmail's Smart Compose



iOS's Predictive Text



Microsoft's Visual Studio IntelliCode

Code completion suggestions based on 1,000s of open source projects

```
127 → // use the code formatter
128 → String lineDelim = TextUtilities.getDefaultLineDelimiter(document);
129 → String replacement = CodeFormatterUtil.format(CodeFormatter.K_CLASS_BODY_DECLARATIONS,
130
131 → // remove line delimiters
132 → if (replacement.endsWith(lineDelim)) {
133 →     int endIndex = replacement.length() - lineDelim.length();
134 →     replacement = replacement.
135 → }
136
137 → return replacement;
138 → }
139 }
140
```

- ★ substring(int beginIndex, int endIndex) : St...
- ★ length() : int
- ★ endsWith(String suffix) : boolean
- ★ charAt(int index) : char
- ★ substring(int beginIndex) : String
- concat(String str) : String
- intern() : String
- replace(CharSequence target, CharSequence replacem...
- replace(char oldChar, char newChar) : String
- replaceAll(String regex, String replacement) : Str...
- replaceFirst(String regex, String replacement) : S...
- toLowerCase() : String



Facebook's Aroma

Code-to-code search and recommendation tool



```
1 final BitmapFactory.Options options = new BitmapFactory.Options();
2 options.inSampleSize = 2;
3 // ...
4 Bitmap bmp = BitmapFactory.decodeStream(is, null, options);
```

```
1 Bitmap bitmap = BitmapFactory.decodeStream(input);
```

```
1 try {
2     InputStream is = am.open(fileName);
3     image = BitmapFactory.decodeStream(is);
4     is.close();
5 } catch (IOException e) {
6     // ...
7 }
```



GitHub Copilot, Powered by OpenAI Codex

Suggests 10-15 lines of code recommendations based on code or comments

- 1 In your JetBrains IDE, create a new Java (*.java) file.
- 2 To prompt GitHub Copilot to suggest an implementation of a function in the Java file, type the following lines.

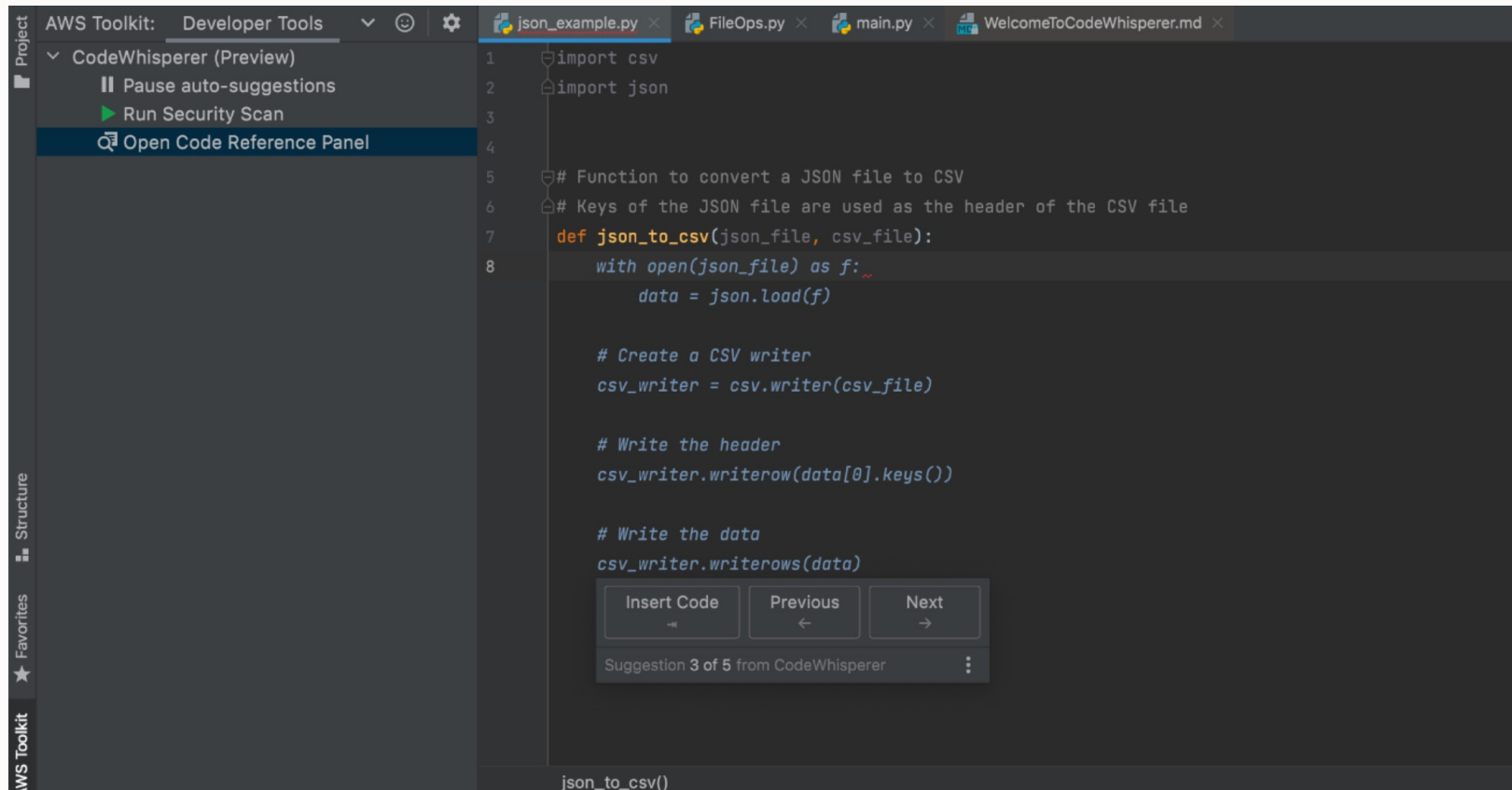
Java



```
// find all images without alternate text  
// and give them a red border  
void process () {
```

Amazon's CodeWhisperer

Generates 10-15 lines of code recommendations based on IDE comments or prior code




Amazon's CodeGuru Reviewer

Identifies critical issues and hard-to-find performance bugs and suggests ways to fix them

```
HelloWorldFunction/src/main/java/helloworld/App.java
```


```
56 + item_values.put("location", new AttributeValue(ipv4));
57 + item_values.put("date", new AttributeValue(now));
58 +
59 + final AmazonDynamoDB ddb = AmazonDynamoDBClientBuilder.defaultClient();
```

 **danilop** 3 minutes ago Author Owner 😊 ⋮

Recommendation generated by Amazon CodeGuru Reviewer. Leave feedback on this recommendation by replying to the comment or by reacting to the comment using emoji.

This code is written so that the client cannot be reused across invocations of the Lambda function. To improve the performance of the Lambda function, consider using static initialization/constructor, global/static variables and singletons. It allows to keep alive and reuse HTTP connections that were established during a previous invocation.

Learn more about [best practices for working with AWS Lambda functions](#).

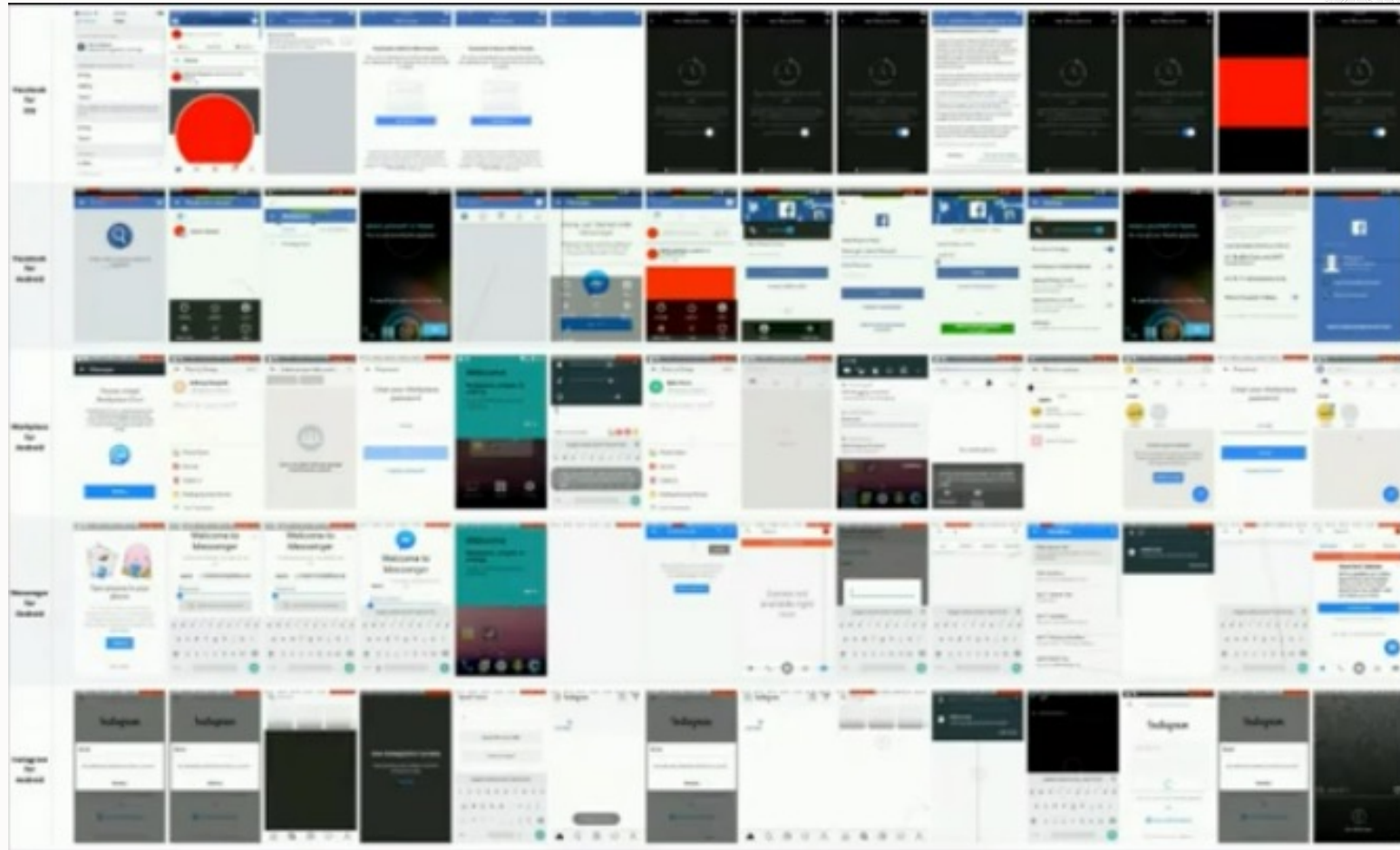
 Reply...

Resolve conversation



Facebook's Sapienz

Automatic generation of tests for Android applications based on system testing



Facebook's GetAFix

Finds fixes for bugs and offers them to engineers

```
20 public boolean onBackPressed() {
21     ApplicationContext ctx = this.getContext();
22     return ctx.onBackPressed();
23 }
24
```

phabricatorlinter suggested changes to line 22

The value of `ctx` in the call to `onBackPressed()` could be null. (Origin: call to `getContext()` at line 21).
Questions about this suggested fix? Post in [Getafix Feedback](#)
Lint code: INFER
Lint name: Null Method Call

```
+   if (ctx == null) {
+       return false;
+   }
    return ctx.onBackPressed();
```

10 minutes ago • Like • Reply • Resolve

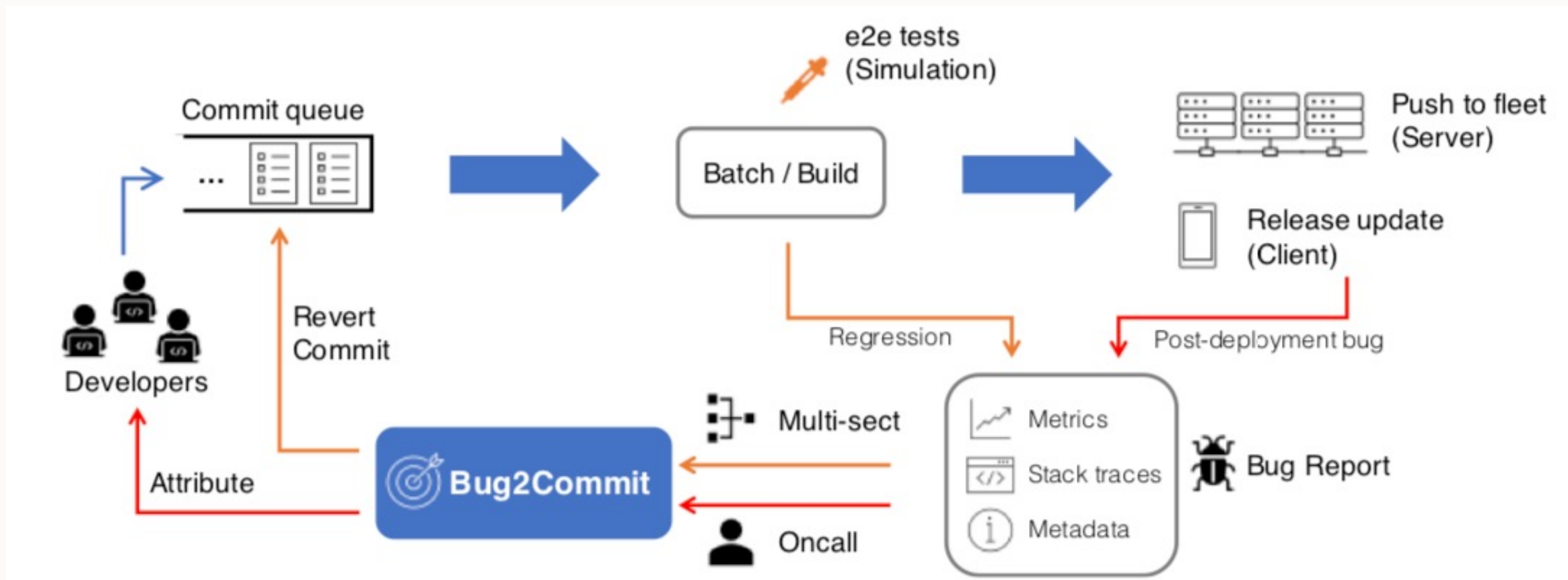
Accept • Reject

A code fix to a **lint** error



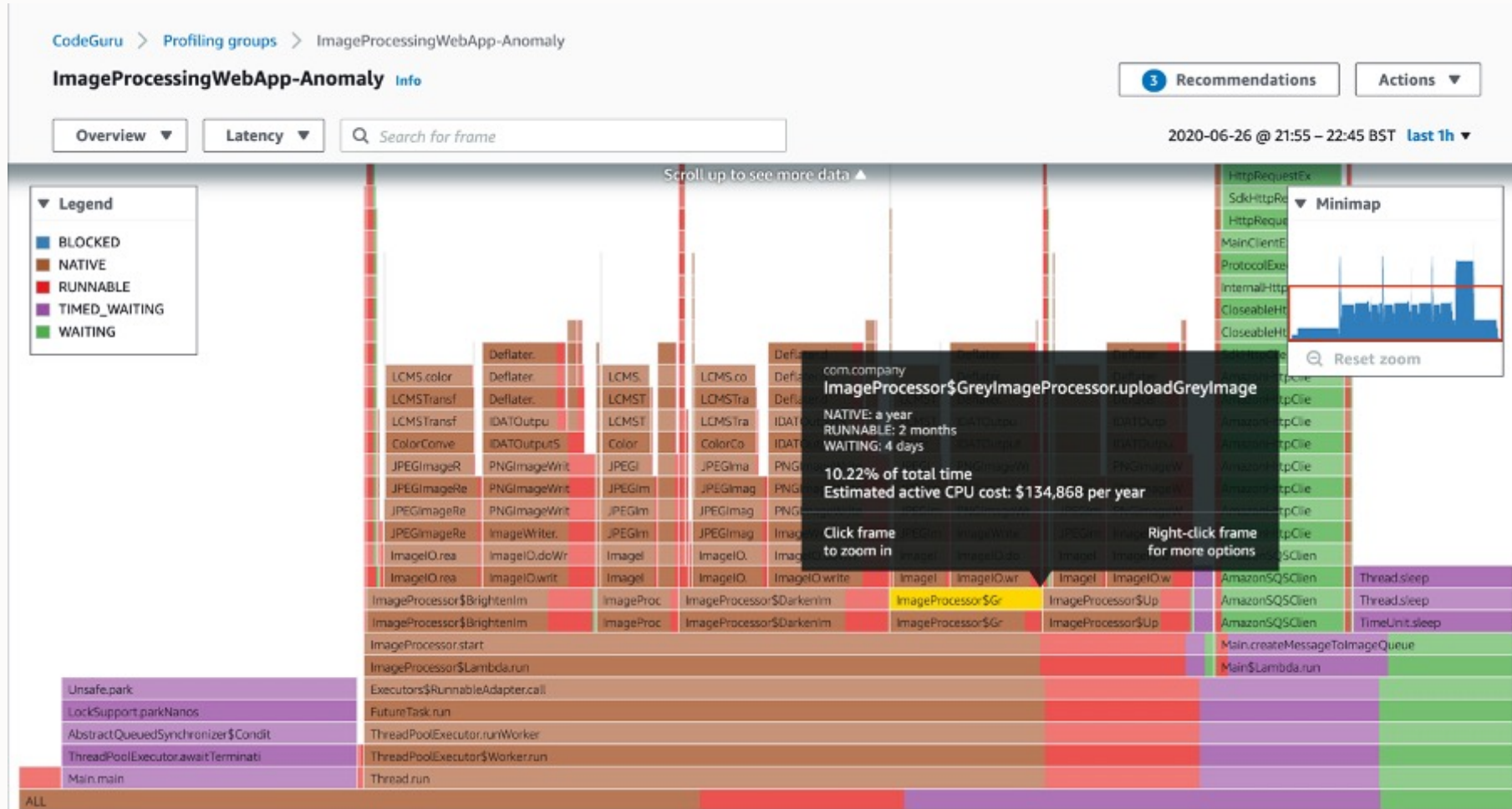
Facebook's Bug2Commit

Finds commit that exposes performance regressions in simulations or server, and crashes on mobile apps

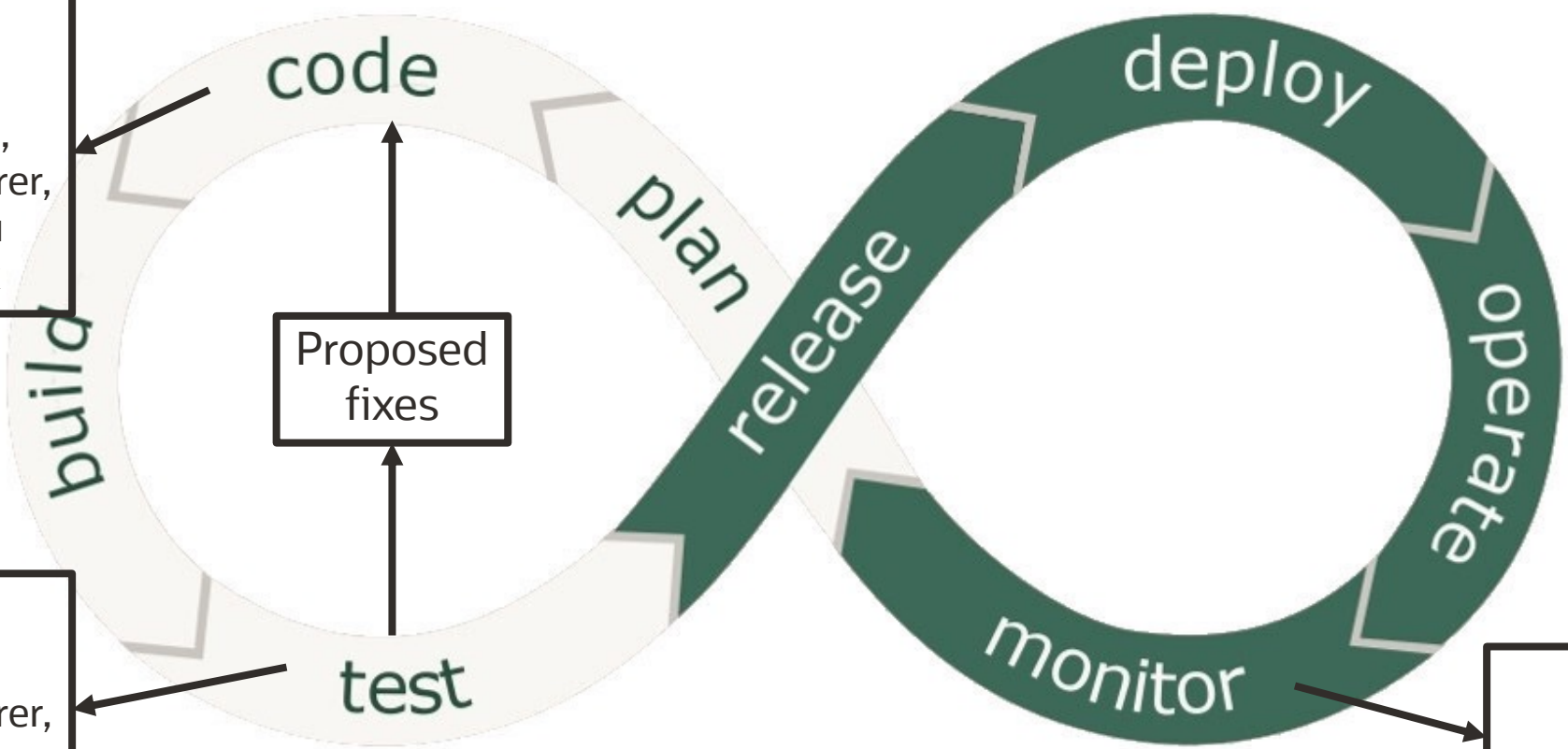
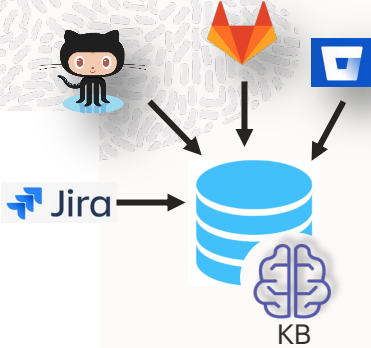


Amazon CodeGuru Profiler

Finds most expensive lines of code and recommends improvements



What Do These AI/ML Advances Mean for DevOps?



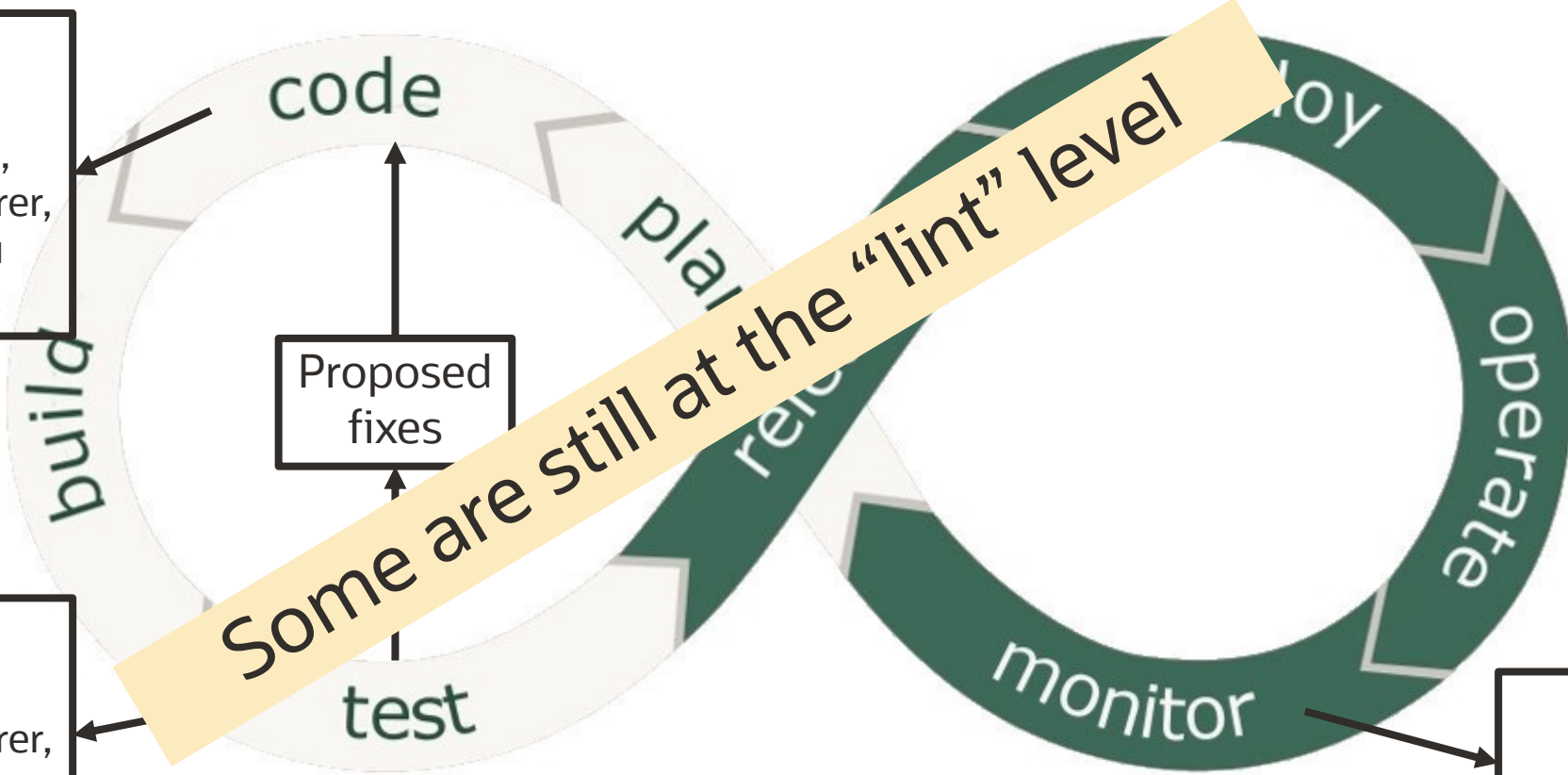
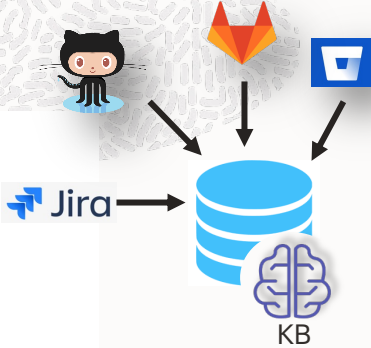
Intelligent coding
e.g., VS IntelliCode, Aroma, CodeWhisperer, Copilot, CodeGuru Reviewer, GetAFix

Intelligent testing
e.g., CodeWhisperer, Sapienz, GetAFix, Bug2Commit

Intelligent monitoring
e.g., CodeGuru Profiler



What Do These AI/ML Advances Mean for DevOps?



Intelligent coding
e.g., VS IntelliCode, Aroma, CodeWhisperer, Copilot, CodeGuru Reviewer, GetAFix

Proposed fixes

Intelligent testing
e.g., CodeWhisperer, Sapienz, GetAFix, Bug2Commit

Intelligent monitoring
e.g., CodeGuru Profiler

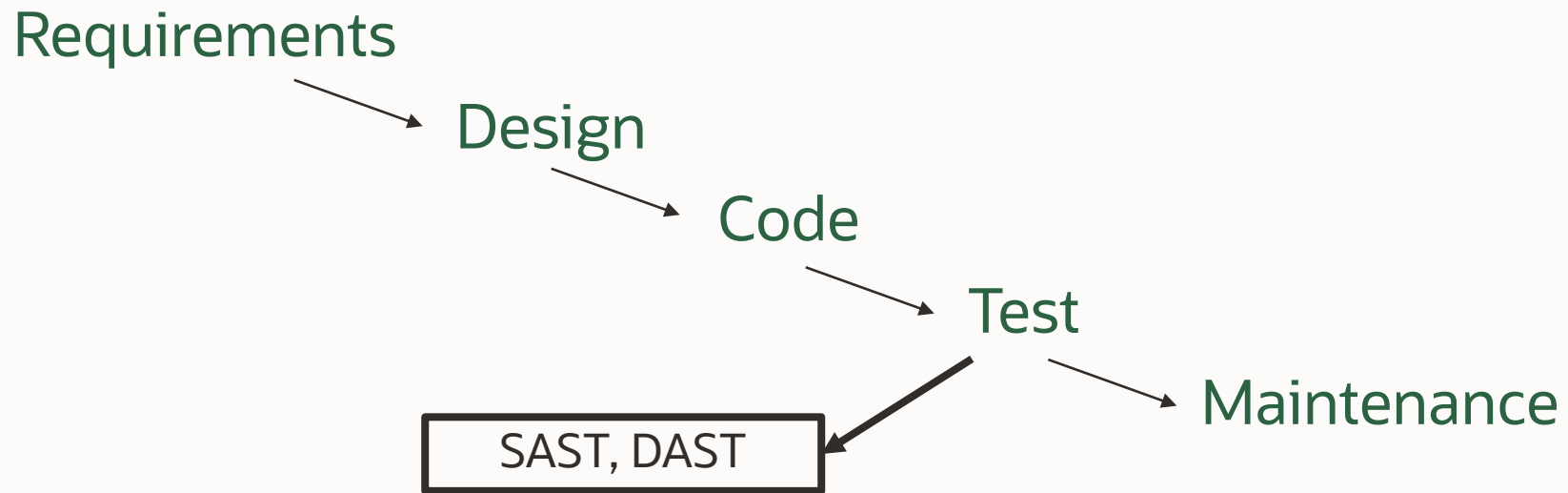


Advances in Application Security Testing Over the Past 15 Years

An Oracle Labs Perspective

Yesteryear – Application Security Testing

Application Security Testing in the Late 2000s

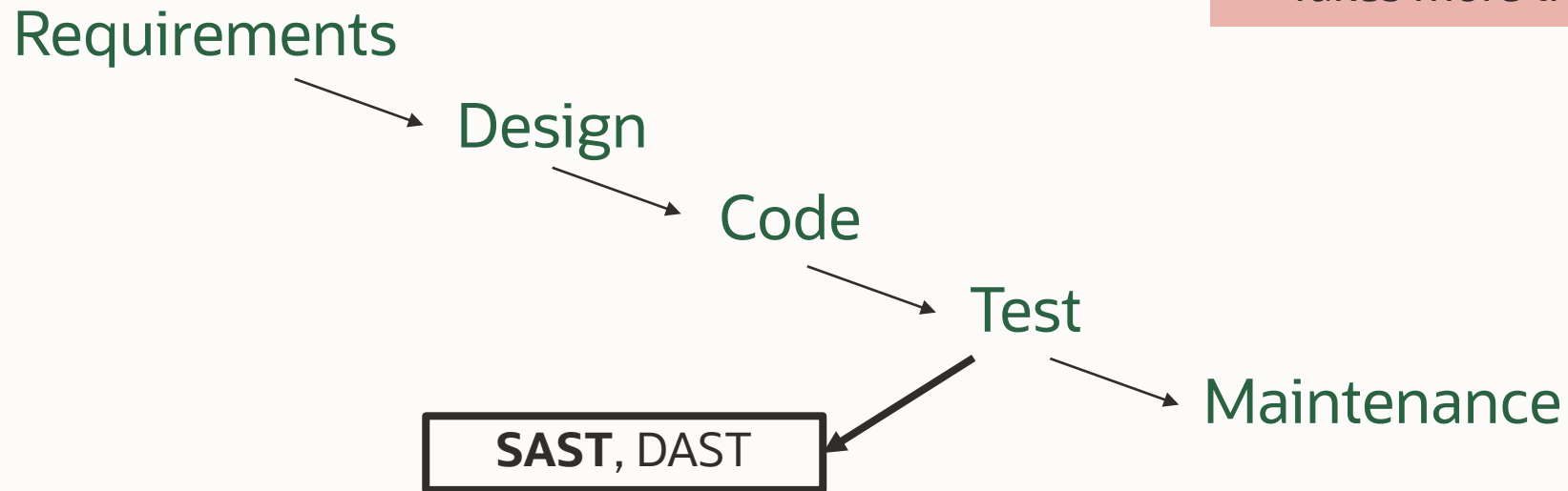


Yesteryear – Application Security Testing

Static Application Security Testing in the Late 2000s

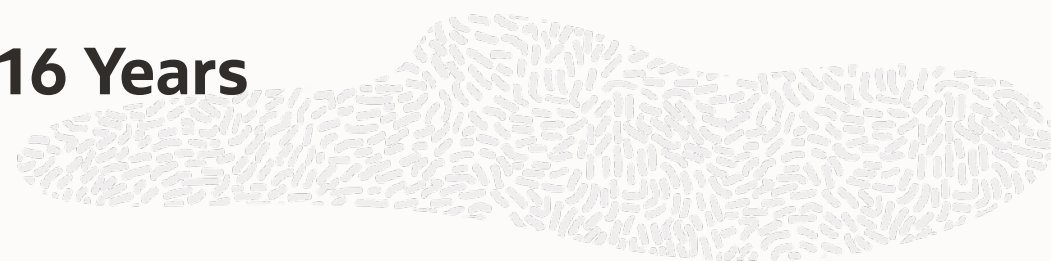
Challenges with SAST:

- Too many false positives
- Takes more than a nightly to run





During the Past 16 Years



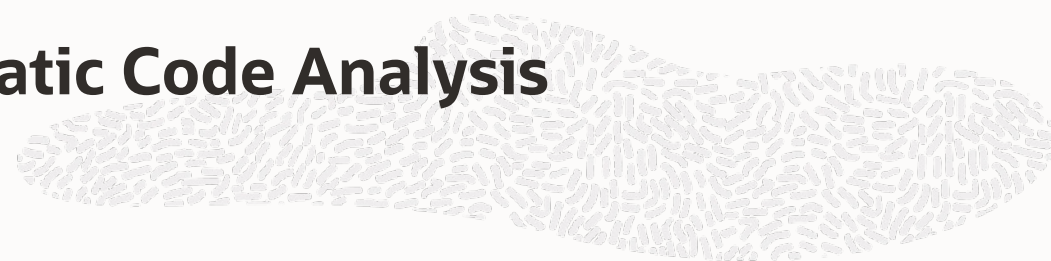
Build ←———— Test

- Efficient analysis of full codebase
 - Used to be nightly runs
 - Now part of Continuous Integration





Parfait – Scalable, Deep Static Code Analysis



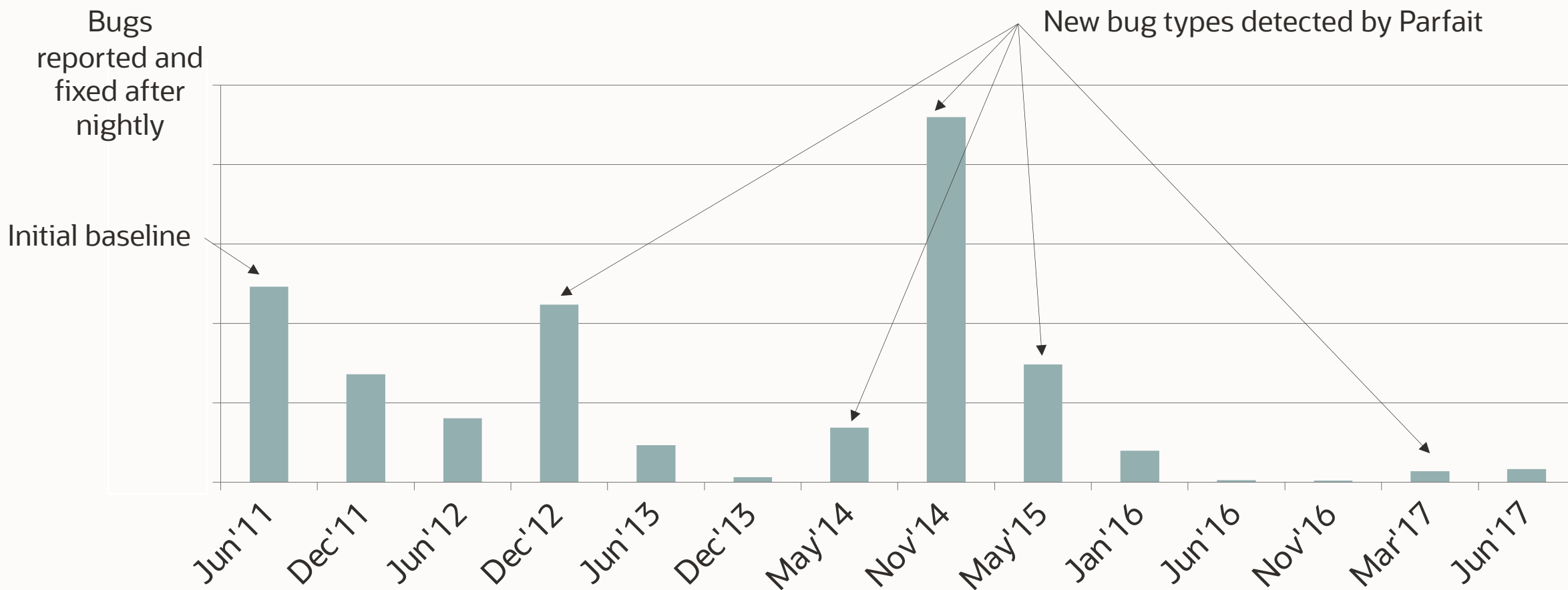
| Codebase | Non Commented Lines of Code | Number of Bug Types | Analysis runtime | Runtime in KLOC/min |
|-----------------------|-----------------------------|---------------------|------------------|---------------------|
| Oracle Linux Kernel 5 | 16,586,325 C | 34 | 19 m 20s | 858 KLOC/min |
| Cloud service | 1,216,168 Java | 5 | 7 m 2 s | 173 KLOC/min |





Parfait – Precise, Deep Static Code Analysis

Bugs fixed by developers once **baseline** had been established



Path-sensitive data flow analysis simplified, ICFEM 2013.

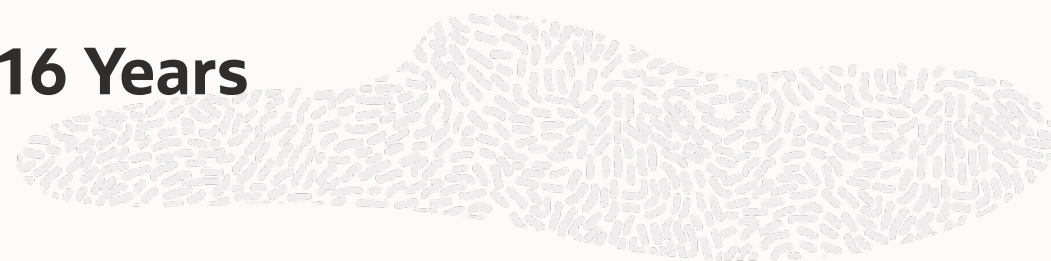
Internal deployment of the Parfait static code analysis tool at Oracle, Invited talk, APLAS 2013.

The Parfait static code analysis framework – Lessons learnt. DECAF workshop 2016.





During the Past 16 Years



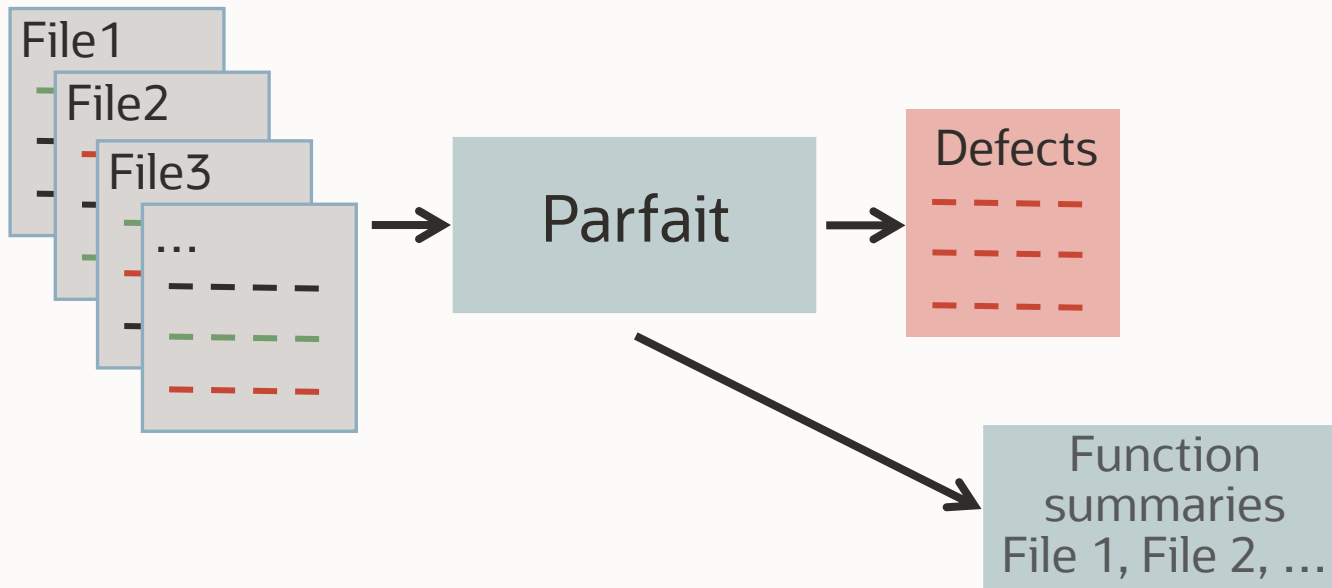
- Efficient analysis of full codebase
 - Used to be nightly runs
 - Now part of Continuous Integration
- Efficient analysis of changeset
 - Prevent bugs from being introduced into the codebase
 - Can be hooked into the commit, push, pull request or merge request



Analysis of Full Codebase vs Analysis of Commit/Push/Pull/Merge Request



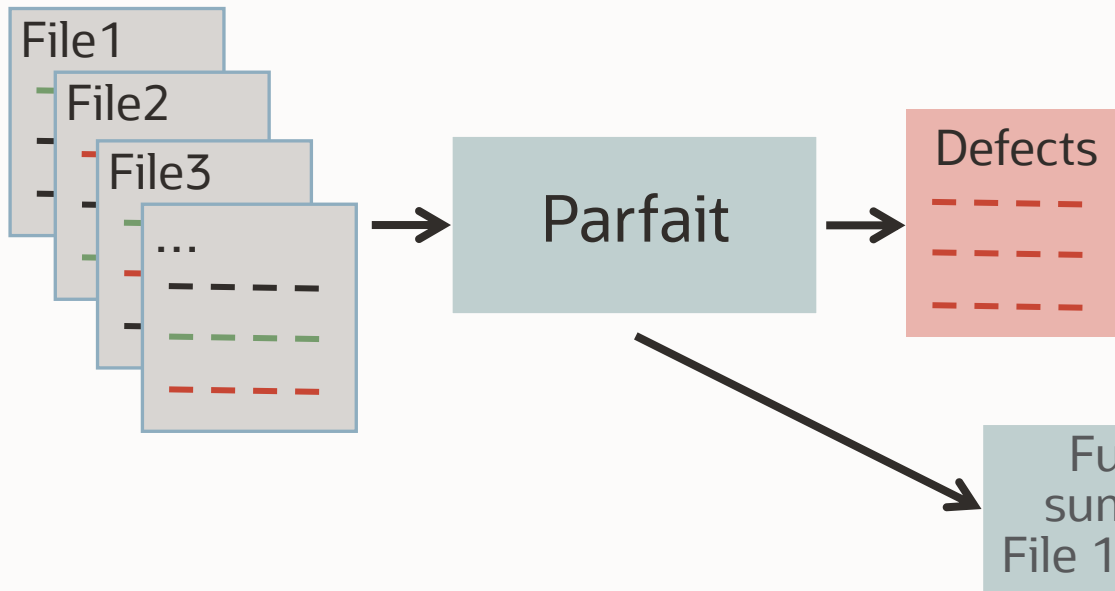
Analysis of full codebase



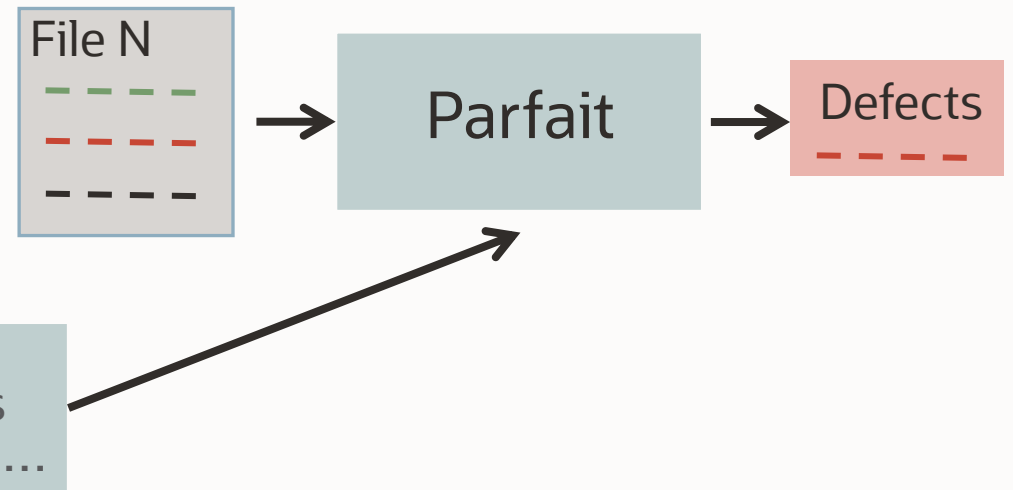
Analysis of Full Codebase vs Analysis of Commit/Push/Pull/Merge Request



Analysis of full codebase



Analysis of changeset



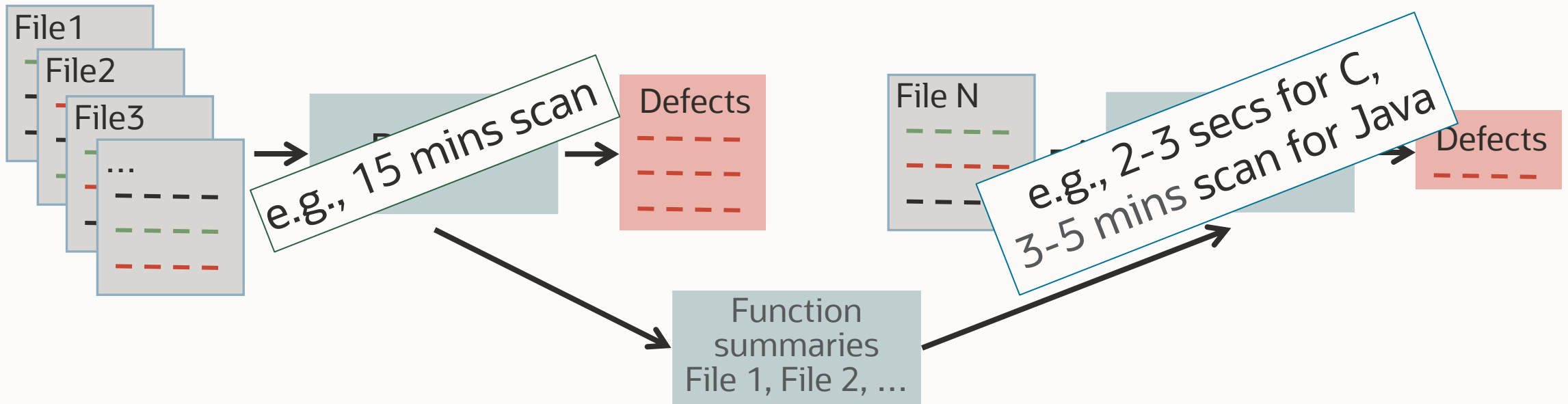


Analysis of Full Codebase vs Analysis of Commit/Push/Pull/Merge Request



Analysis of full codebase

Analysis of changeset

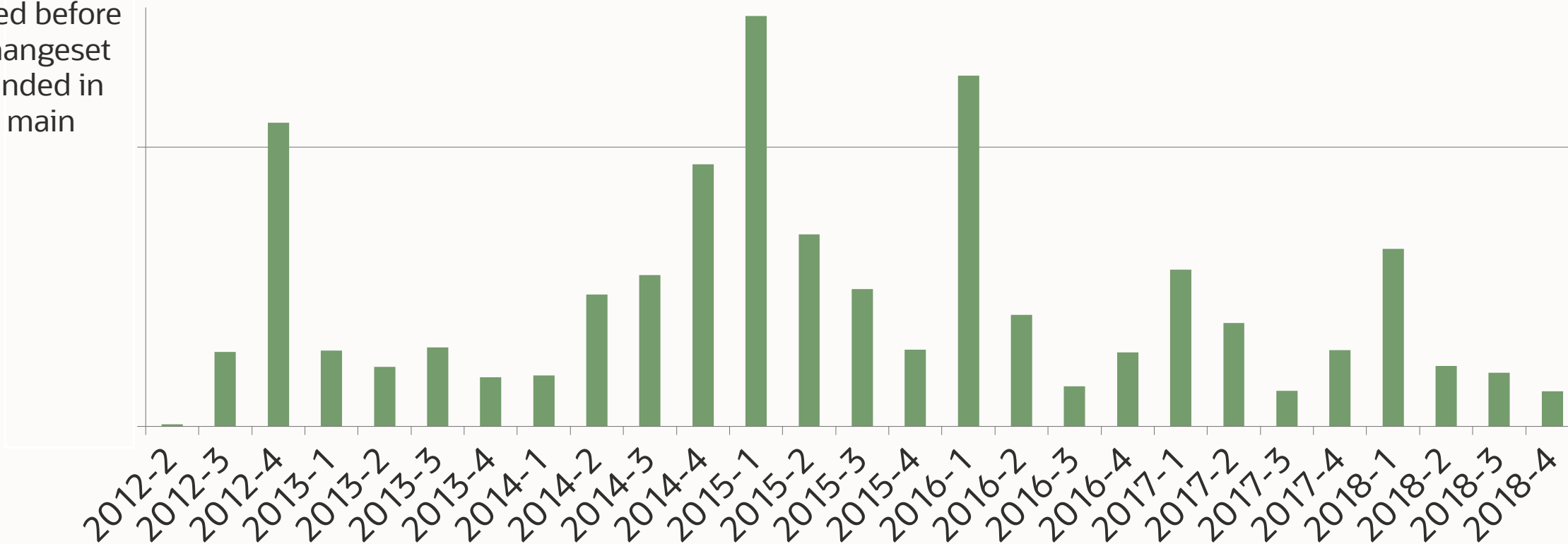




Bugs Prevented from Being Introduced into the Codebase

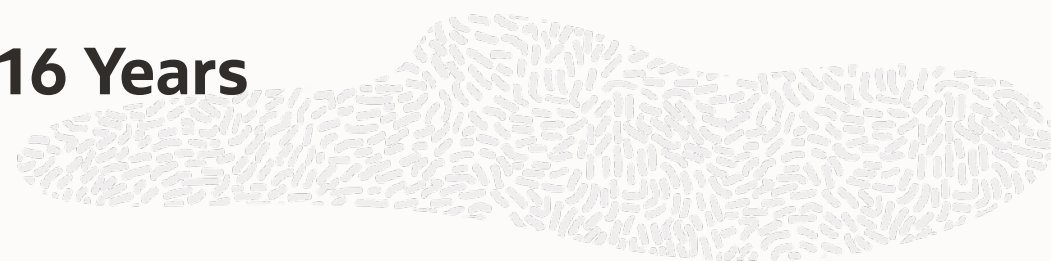
Changeset analysis prevents **80%** of new bugs (compared to baseline)

Bugs reported and fixed before changeset landed in main





During the Past 16 Years



- **Efficient analysis of full codebase**
 - Used to be nightly runs
 - Now part of Continuous Integration
- **Efficient analysis of changeset**
 - Prevent bugs from being introduced into the codebase
 - Can be hooked into the commit, push, pull request or merge request

Innovations for SAST via Parfait:

- Precise results
- Scalable, can integrate early in the development cycle



Yesteryear – Application Security Testing in the 2000s

Dynamic Web Application Security Testing in 2015

Requirements



Design



Coding



Testing



Maintenance



Challenges with DAST:

- Automation lacking for effective use of the tool
- Inefficient use of compute resources; misses vulnerabilities





During the Past 6 Years

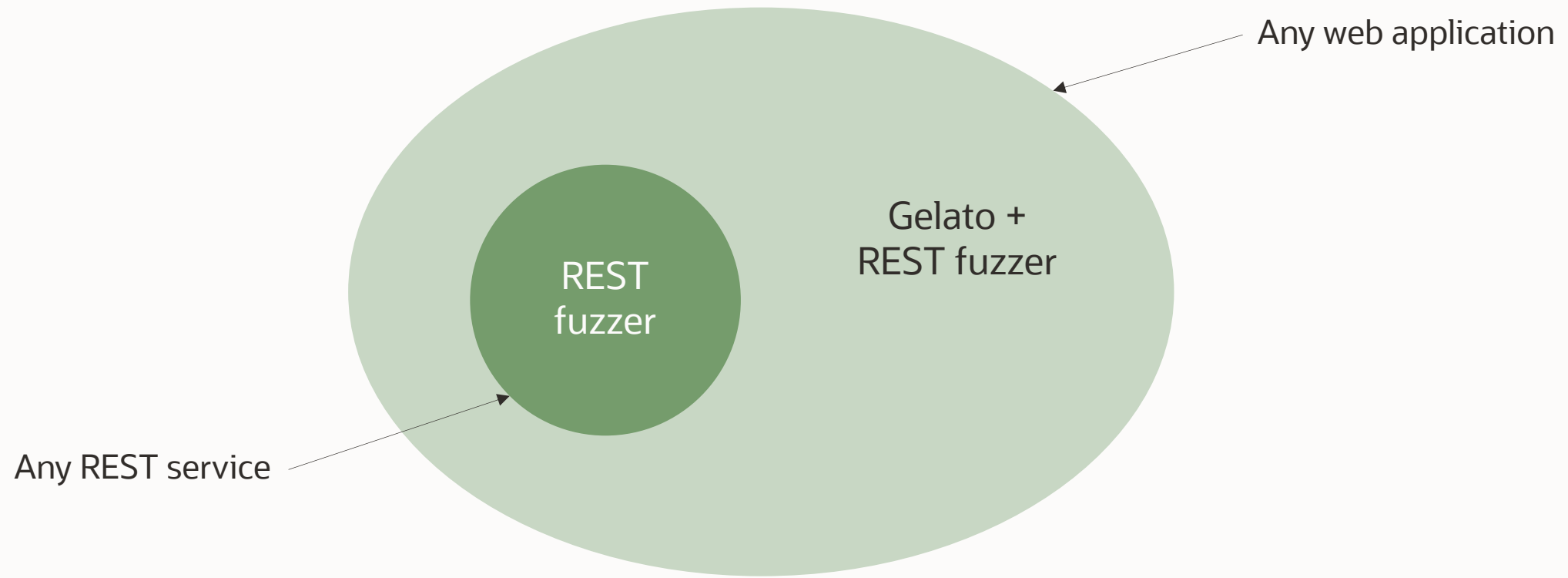
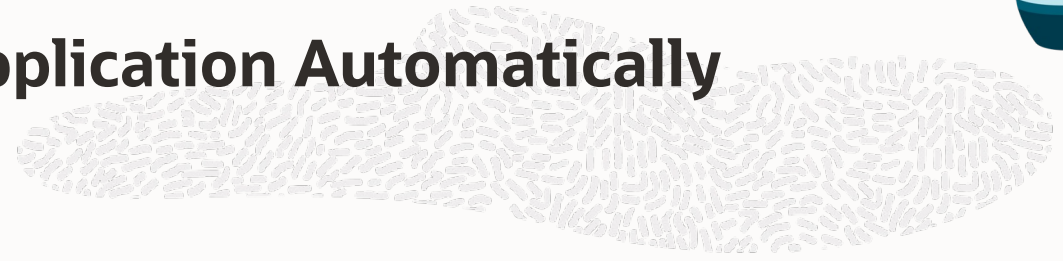


Test

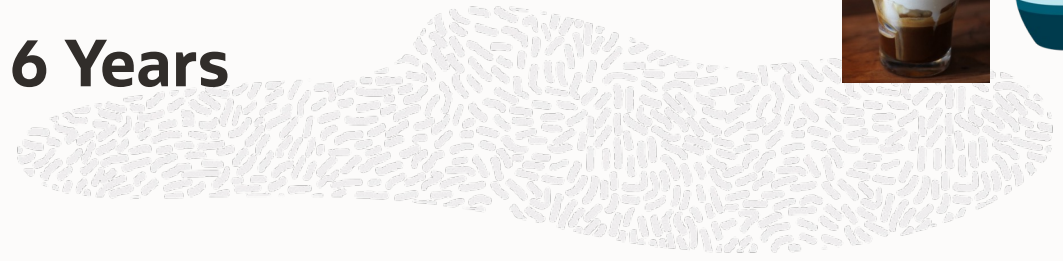
- Automatic, high coverage detection of endpoints
 - Automated – no input from pentester or developer needed
 - Generation of Swagger/Open API spec to drive inputs into existing blackbox REST fuzzer



Enabling Fuzzing of Any Web Application Automatically



During the Past 6 Years



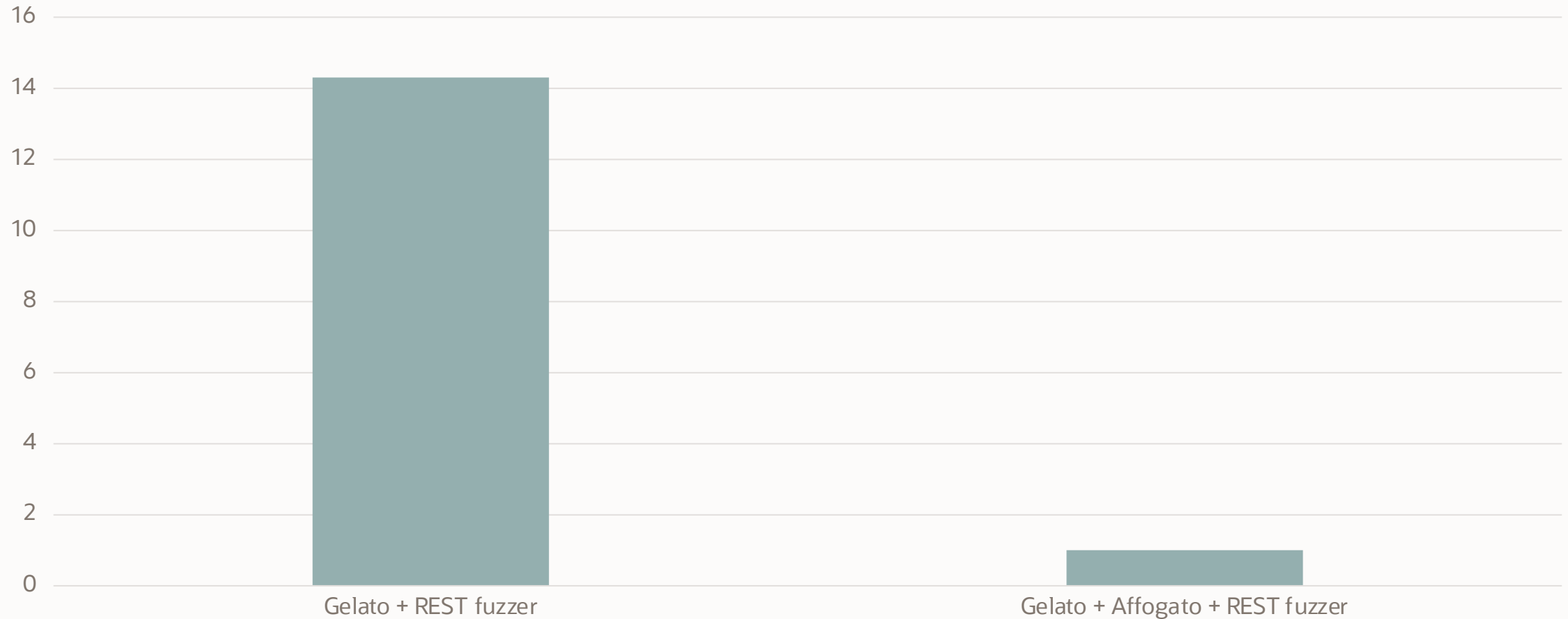
Test

- Automatic, high coverage detection of endpoints
 - Automated – no input from pentester or developer needed
 - Generation of Swagger/Open API spec to drive inputs into existing blackbox REST fuzzer
- Integrate greybox solution
 - Greybox approach provides context to drive efficiency into existing blackbox solutions

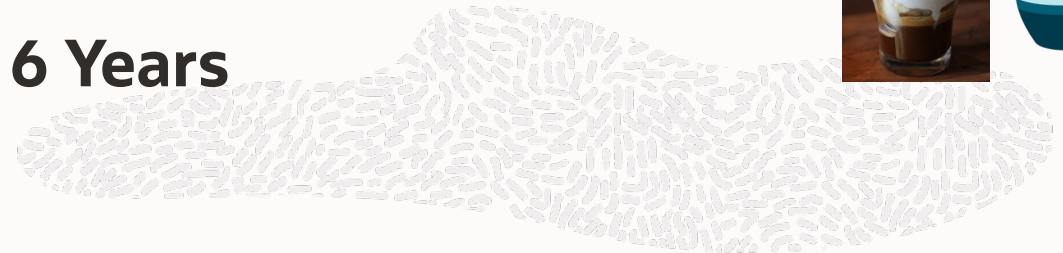


Efficient Use of Compute Resources

Comparative runtime to find 0-days in web applications



During the Past 6 Years



Test

Innovations for DAST via Gelato and Affogato

- Automated attack surface detection
- Efficient use of compute resources finds more 0-days

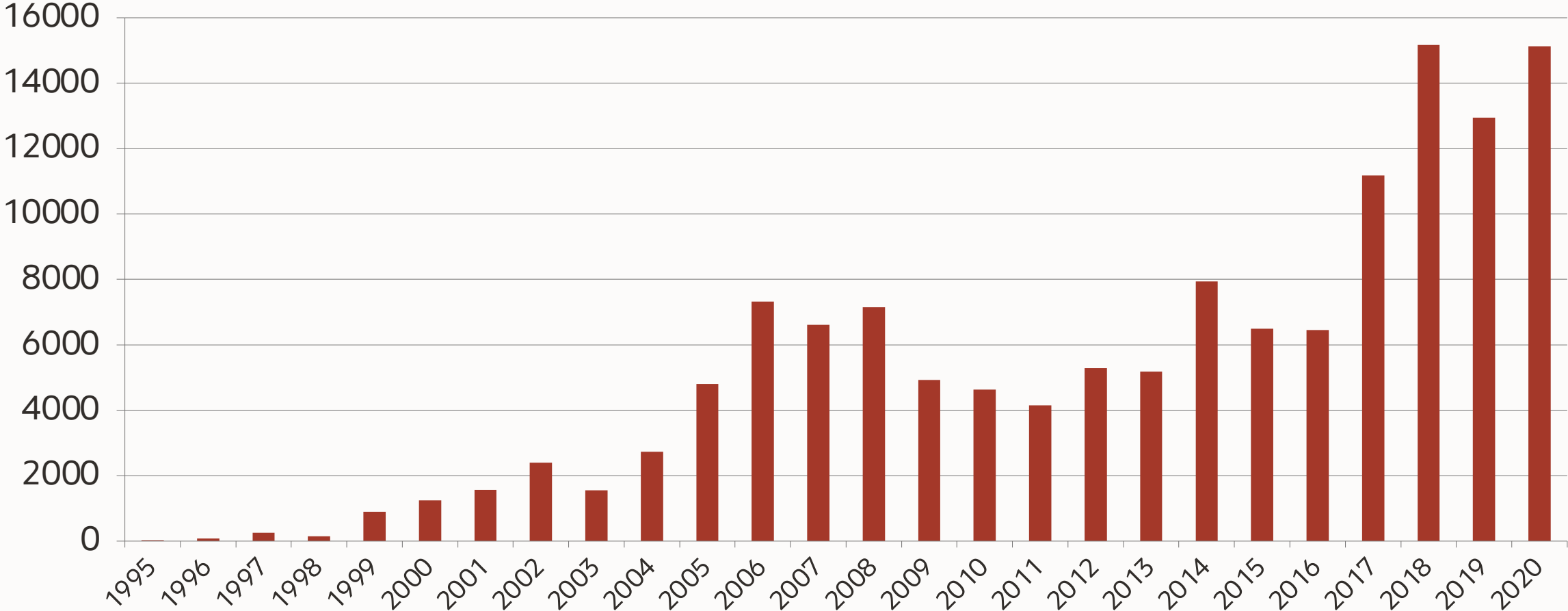
- Automatic, high coverage detection of endpoints
 - Automated – no input from pentester or developer
 - Generation of Swagger spec to drive inputs into existing blackbox REST fuzzer
- Integrate greybox solution
 - Greybox approach provides context to drive efficiency into existing blackbox solutions



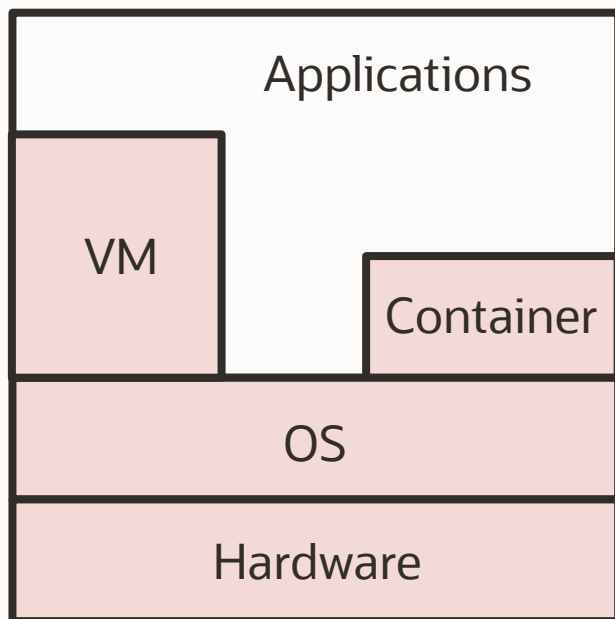
We Can Build an Intelligent Application Security Future

#ias

Exploited CVE Vulnerabilities Per Year



Vulnerabilities in the Systems and Applications Stack



SQL injection, XSS, Unsafe deserialisation, ...

Unguarded Caller-Sensitive Method call, Unsafe deserialisation, ...

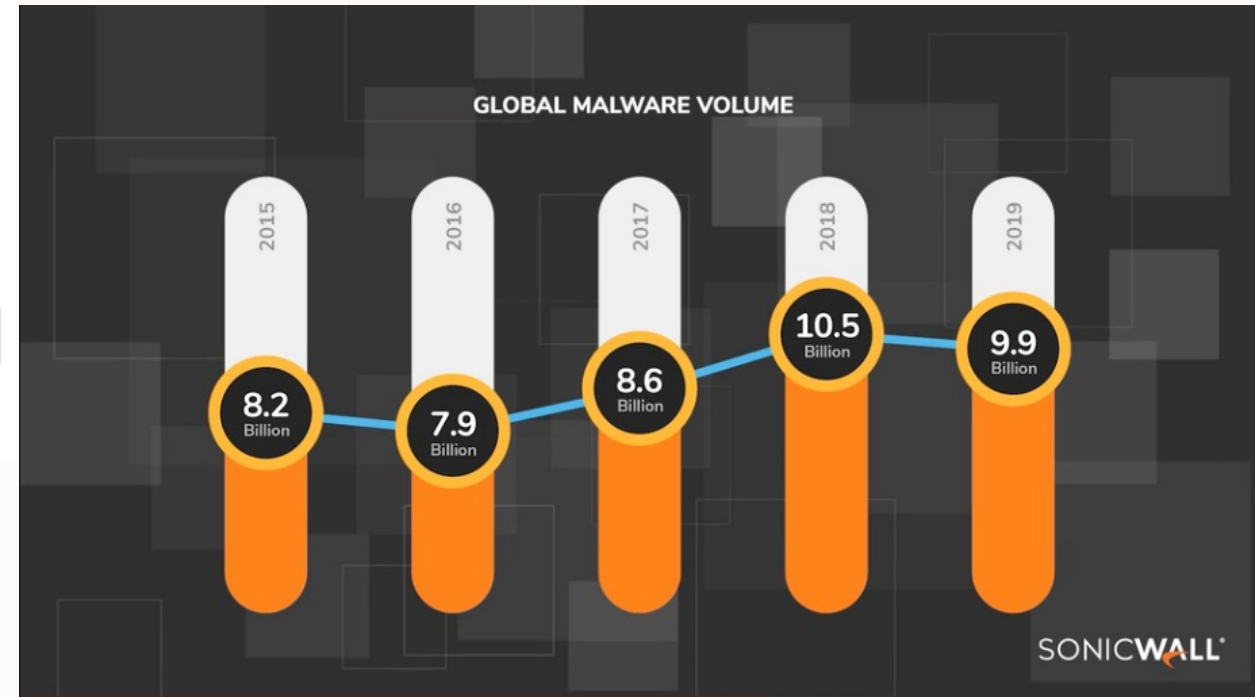
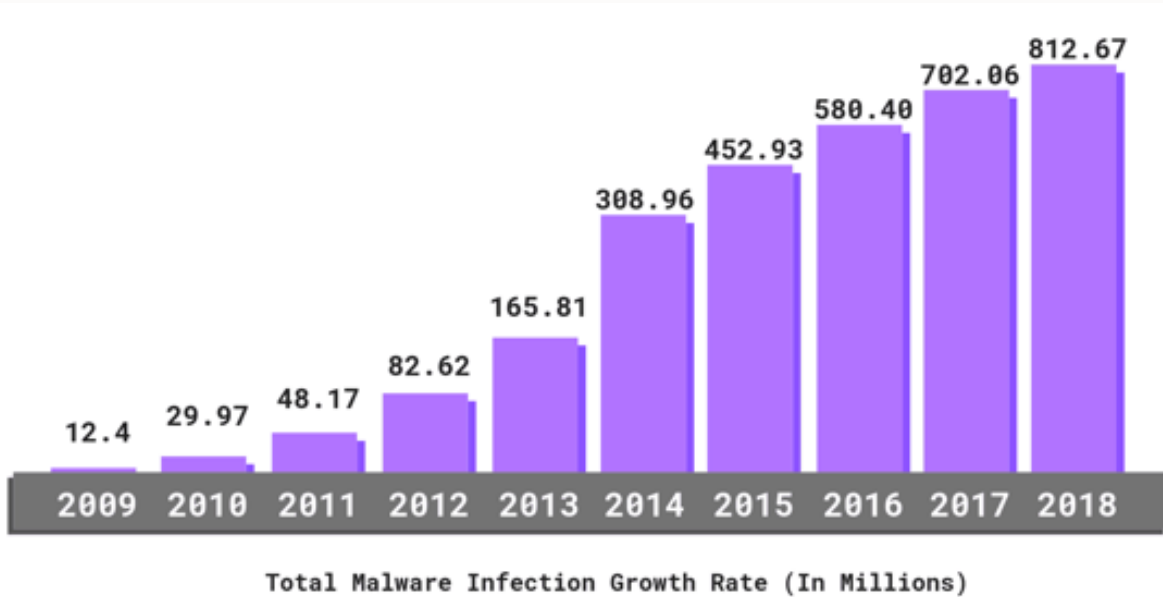
OS + VM + Application vulnerabilities

Buffer overflow, use after free, ...

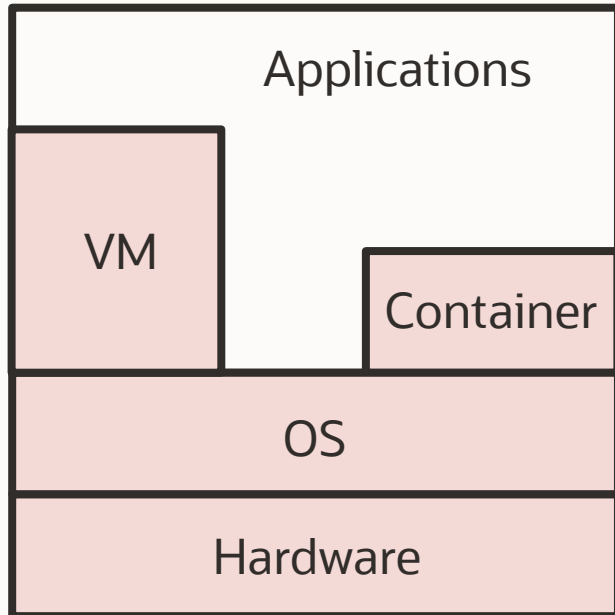
Spectre, Meltdown, L1TF, ...



The Rise of Malware



Sample Systems and Applications Software Affected by Malware



SolarWinds Orion Platform, Dec 2020

ESLint Scope, Jul 2018

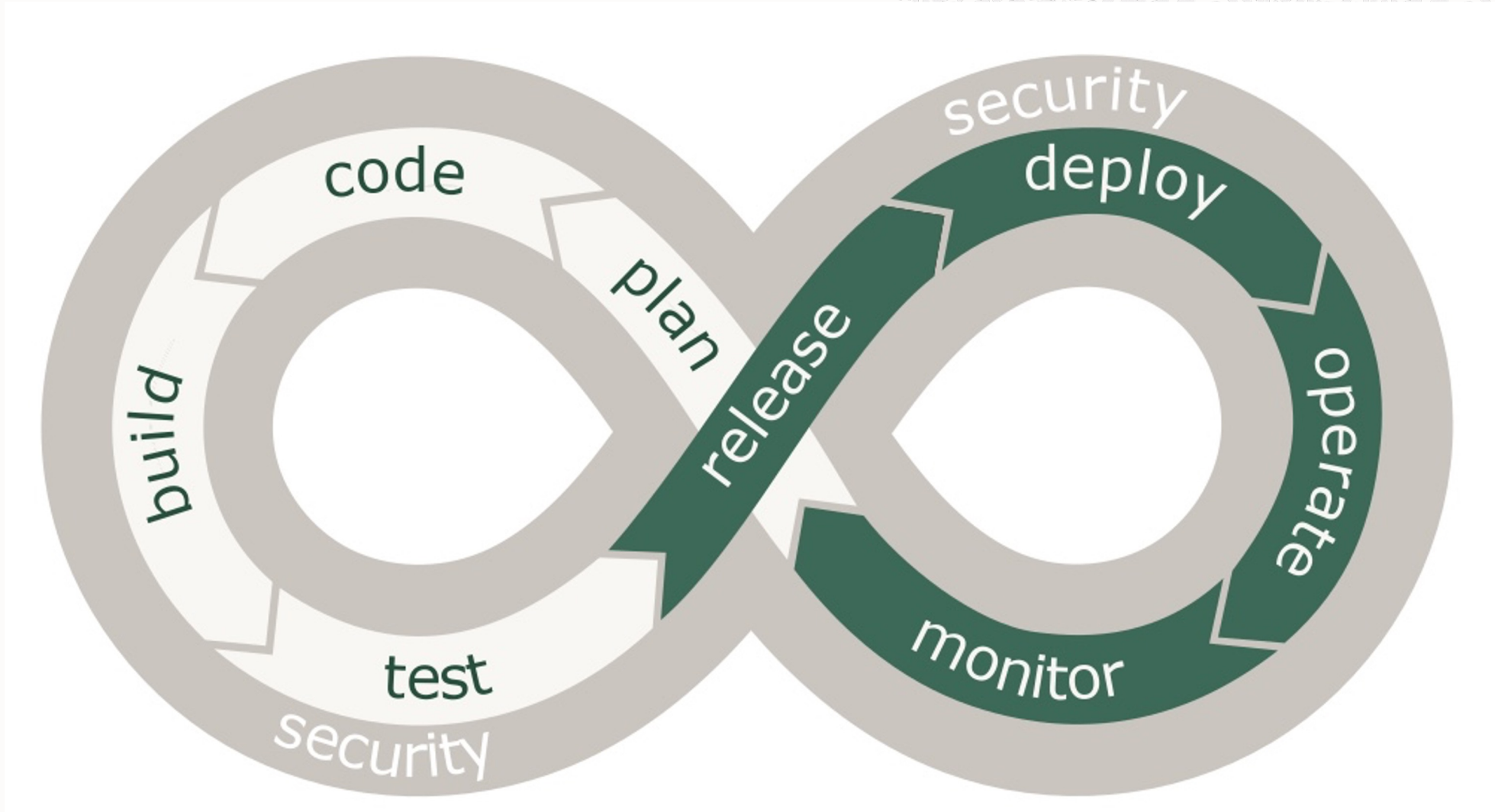
Cryptominer in Community Amazon Machines image, Aug 2020

XCodeGhost (iOS), Sep 2015

Apple's M1 chip, Feb 2021



Today – DevSecOps – Integrating Security Into DevOps

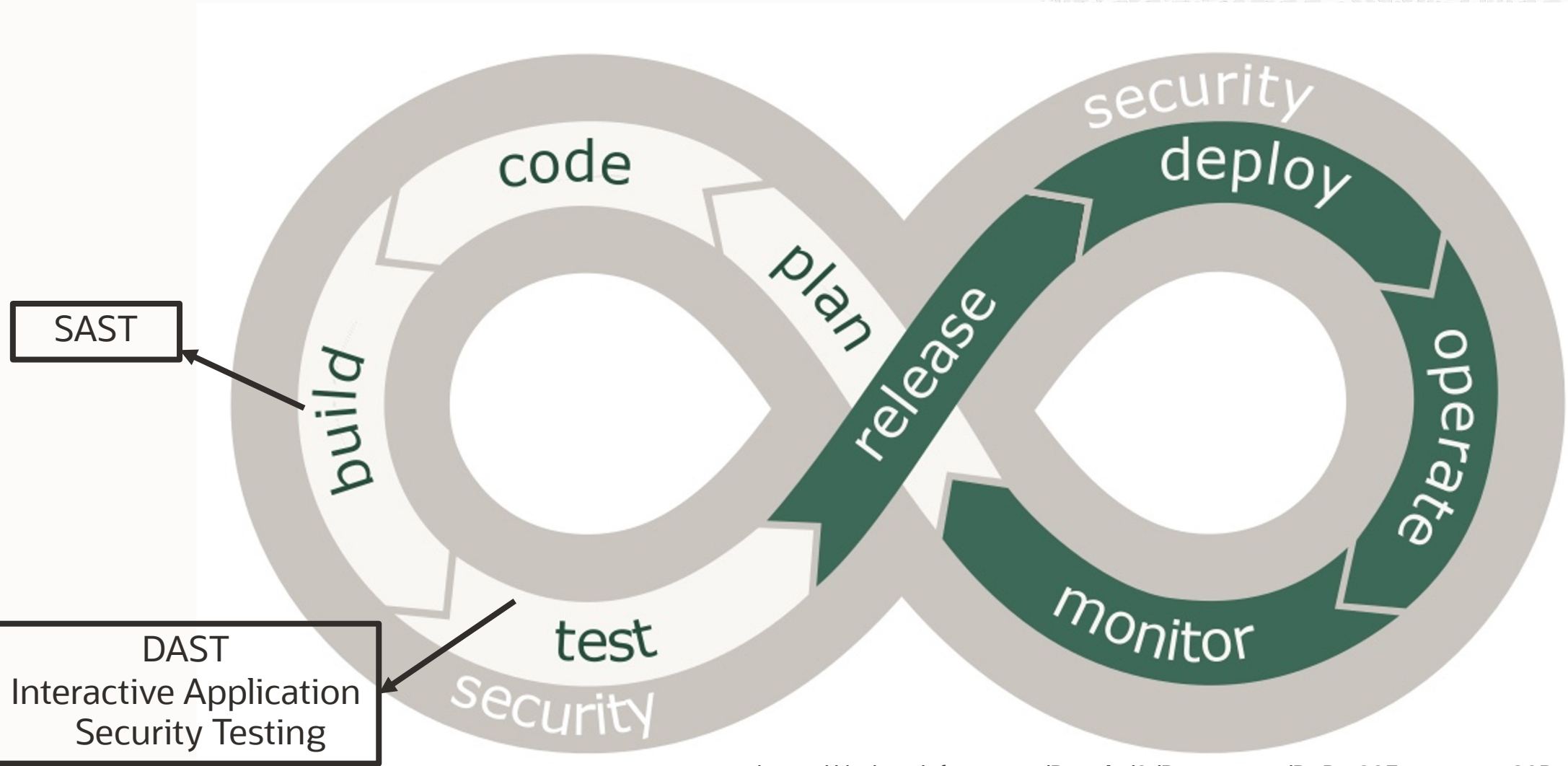


“DevSecOps is an organizational software engineering culture and practice that aims at **unifying software development (Dev), security (Sec) and operations (Ops)**. The main characteristic of DevSecOps is **to automate, monitor, and apply security at all phases** of the software lifecycle: plan, develop, build, test, release, deliver, deploy, operate, and monitor. In DevSecOps, **testing and security are shifted to the left** through automated unit, functional, integration, and security testing - this is a key DevSecOps differentiator since security and functional capabilities are tested and built simultaneously.”

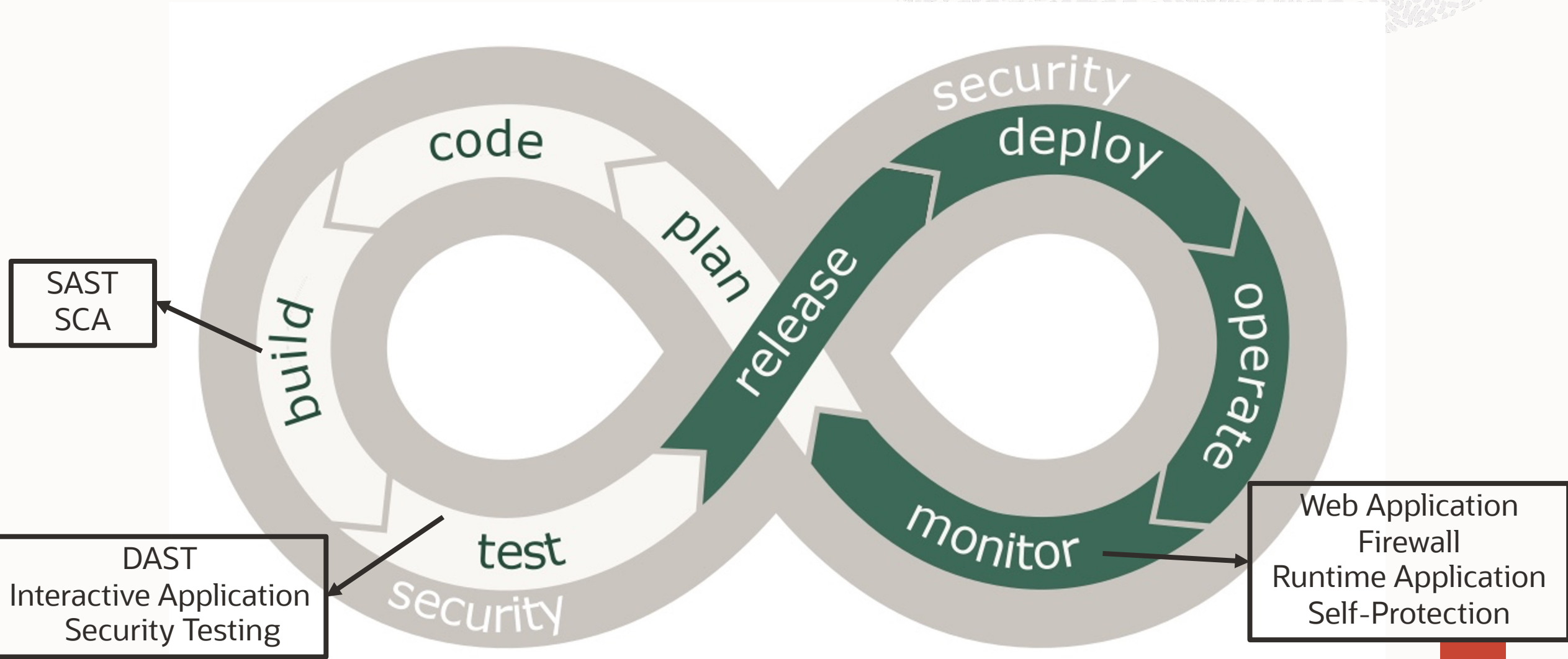
DoD Enterprise DevSecOps Reference Design

12 August 2019

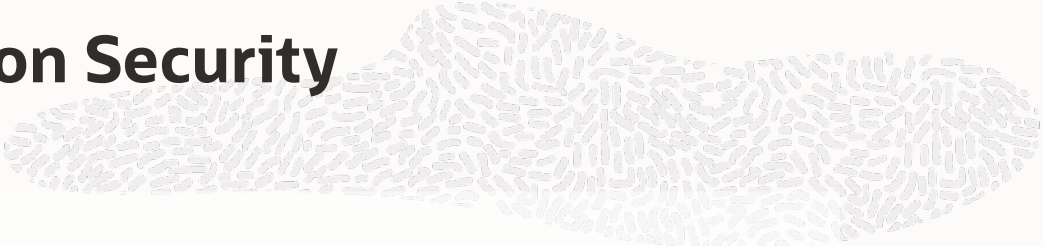
Application Security in DoD's DevSecOps Reference Design



Today – Application Security Testing ~~X~~ in the DevSecOps Model

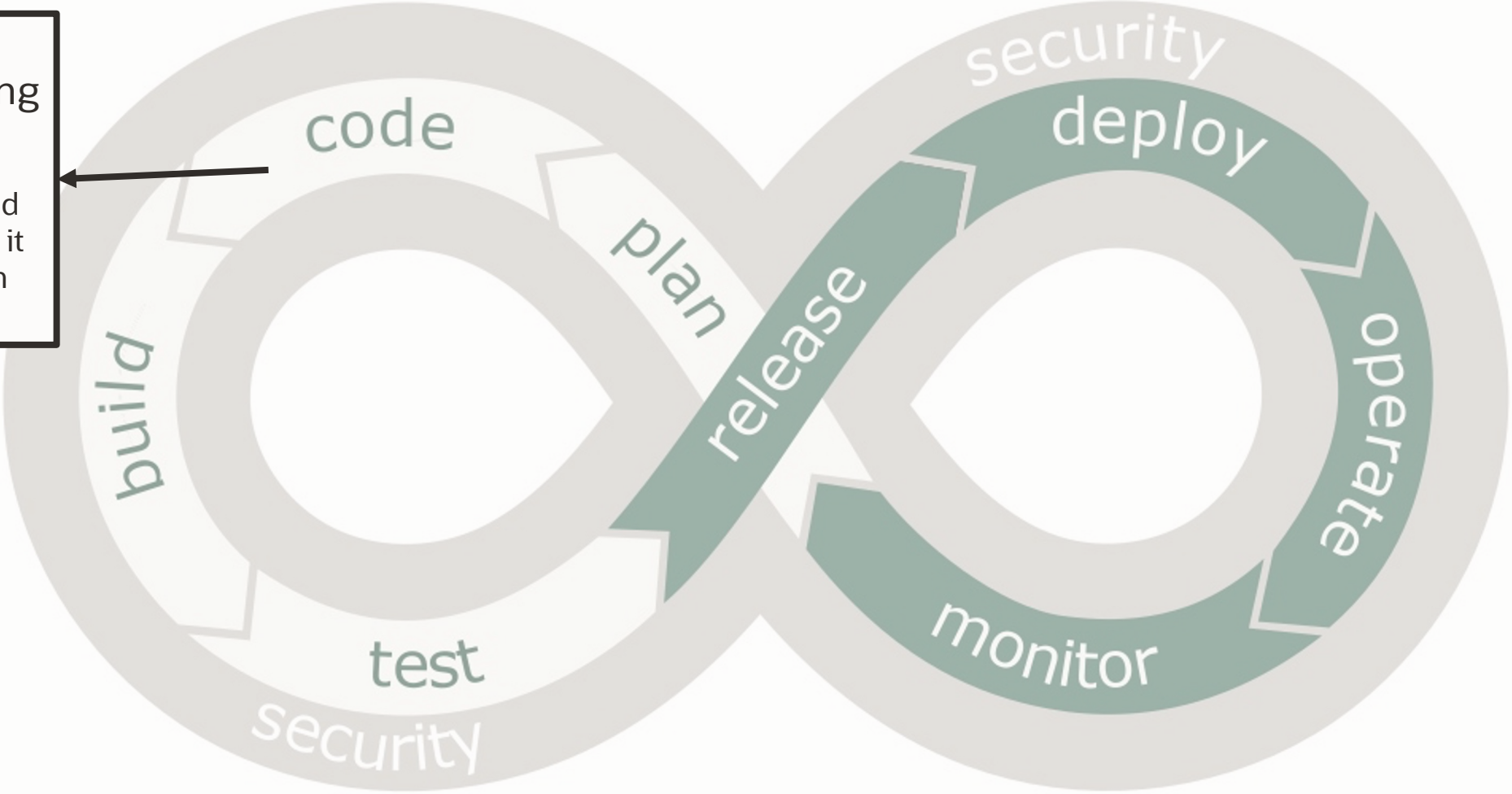


Intelligent Application Security

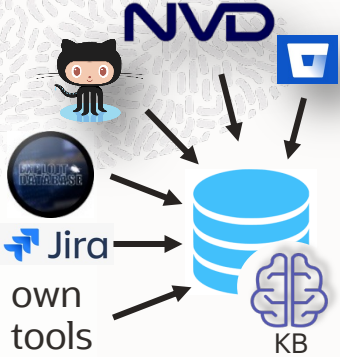
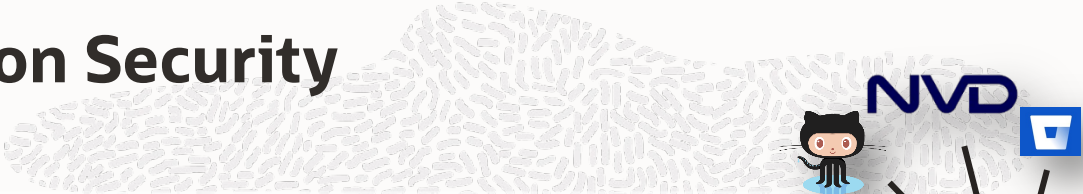


Intelligent security coding

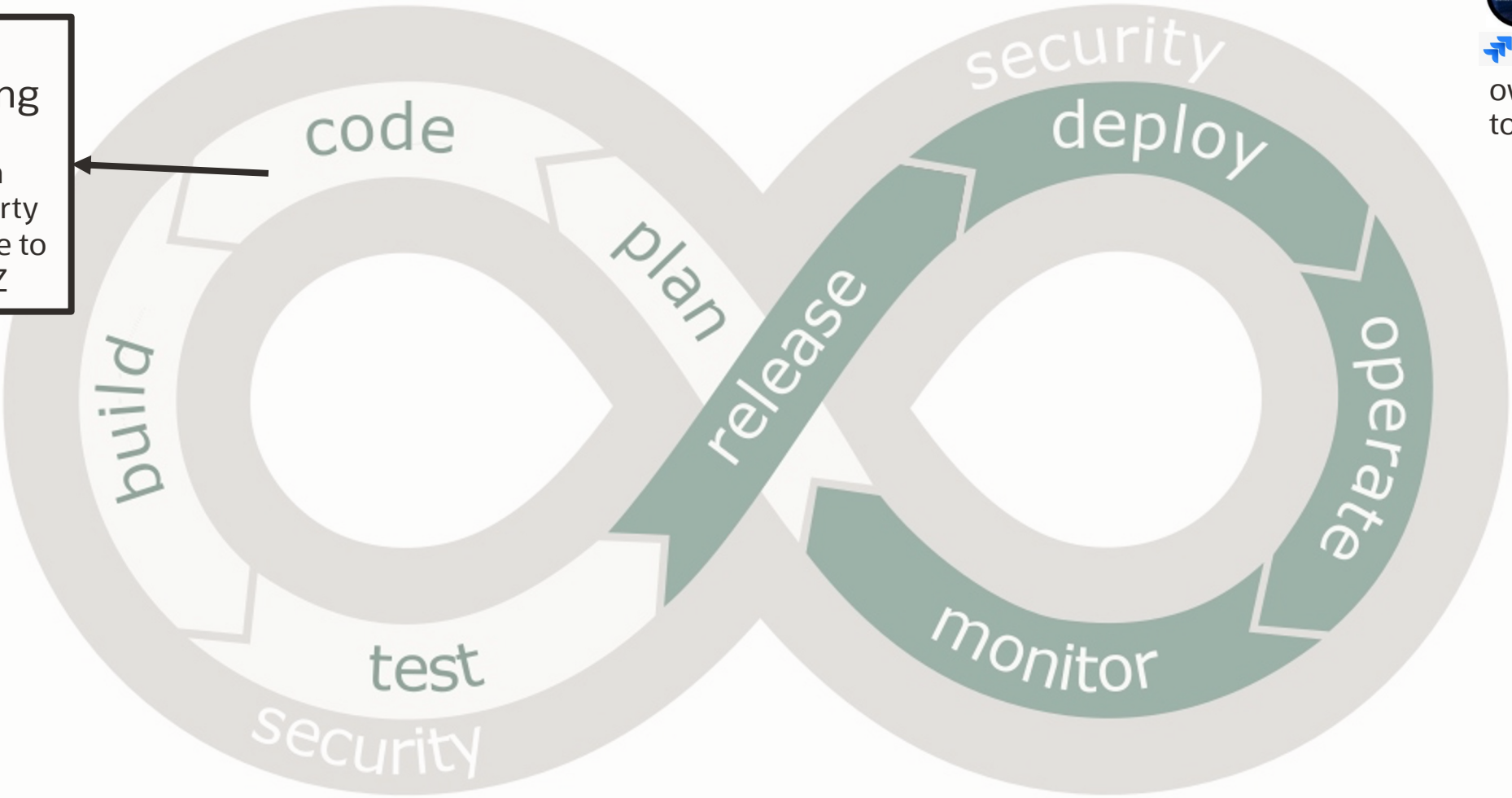
You have a bug at line X and here's how to fix it [before you can commit]



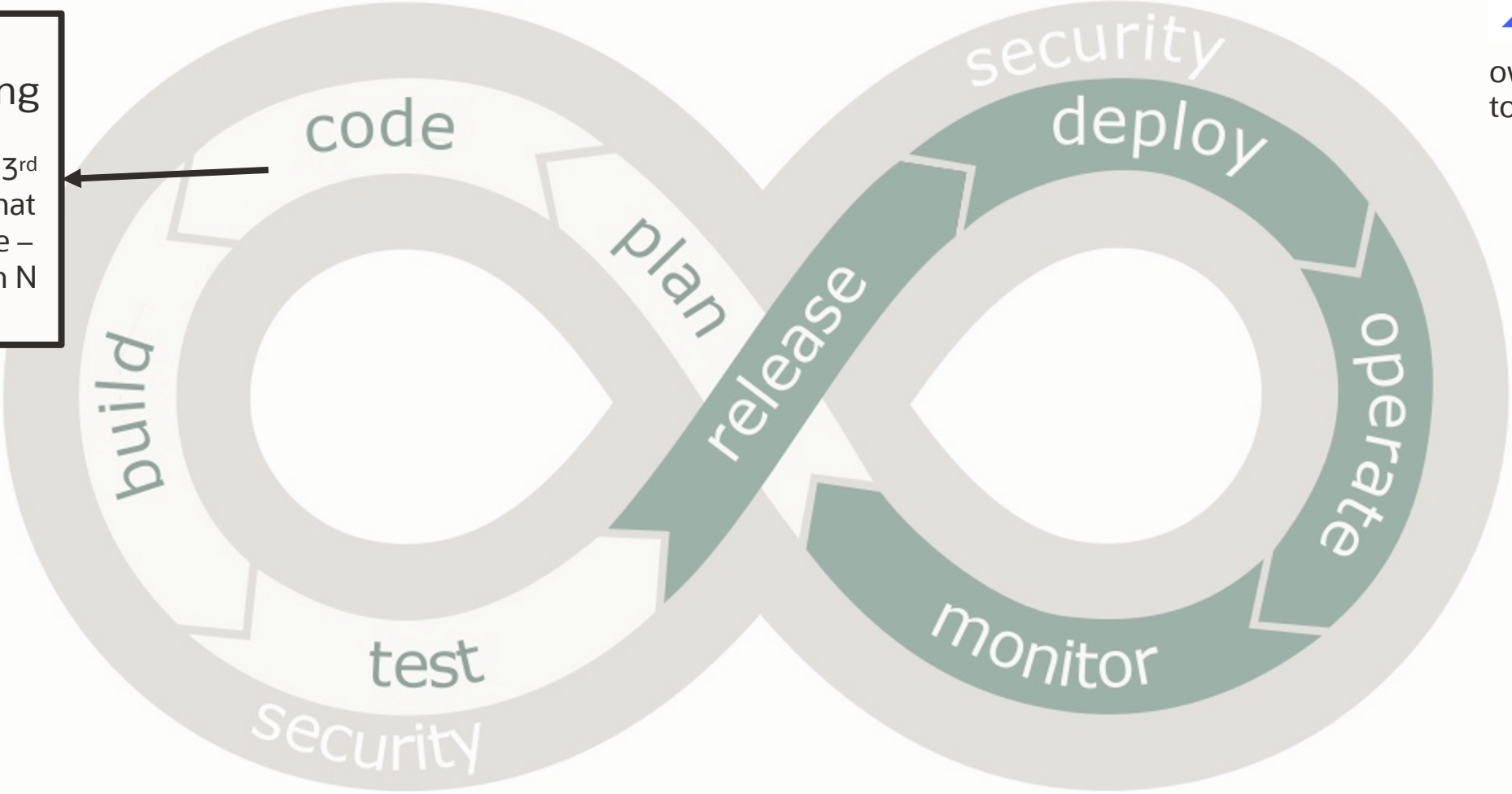
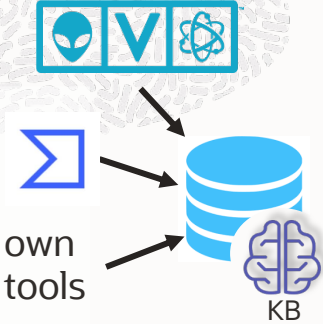
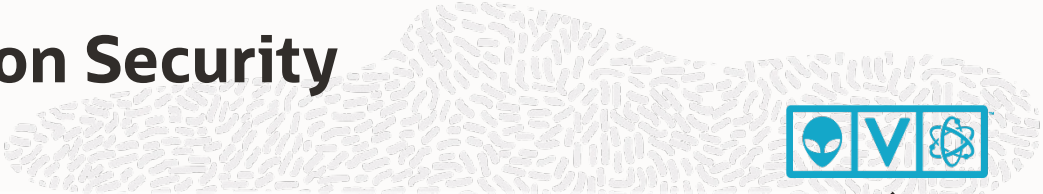
Intelligent Application Security



Intelligent security coding
You depend on vulnerable 3rd party library Y – upgrade to clean version Z



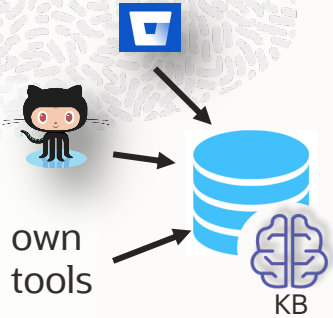
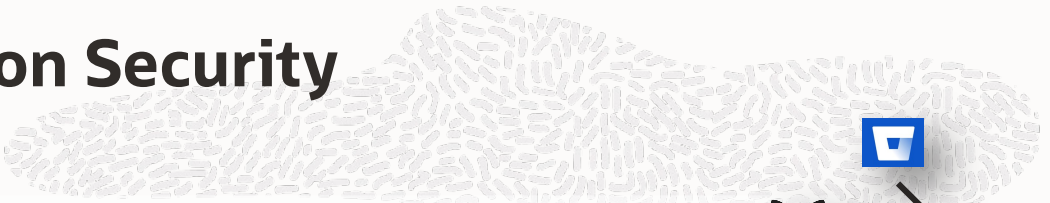
Intelligent Application Security



Intelligent security coding
You depend on a 3rd party library M that contains malware – use clean version N instead



Intelligent Application Security

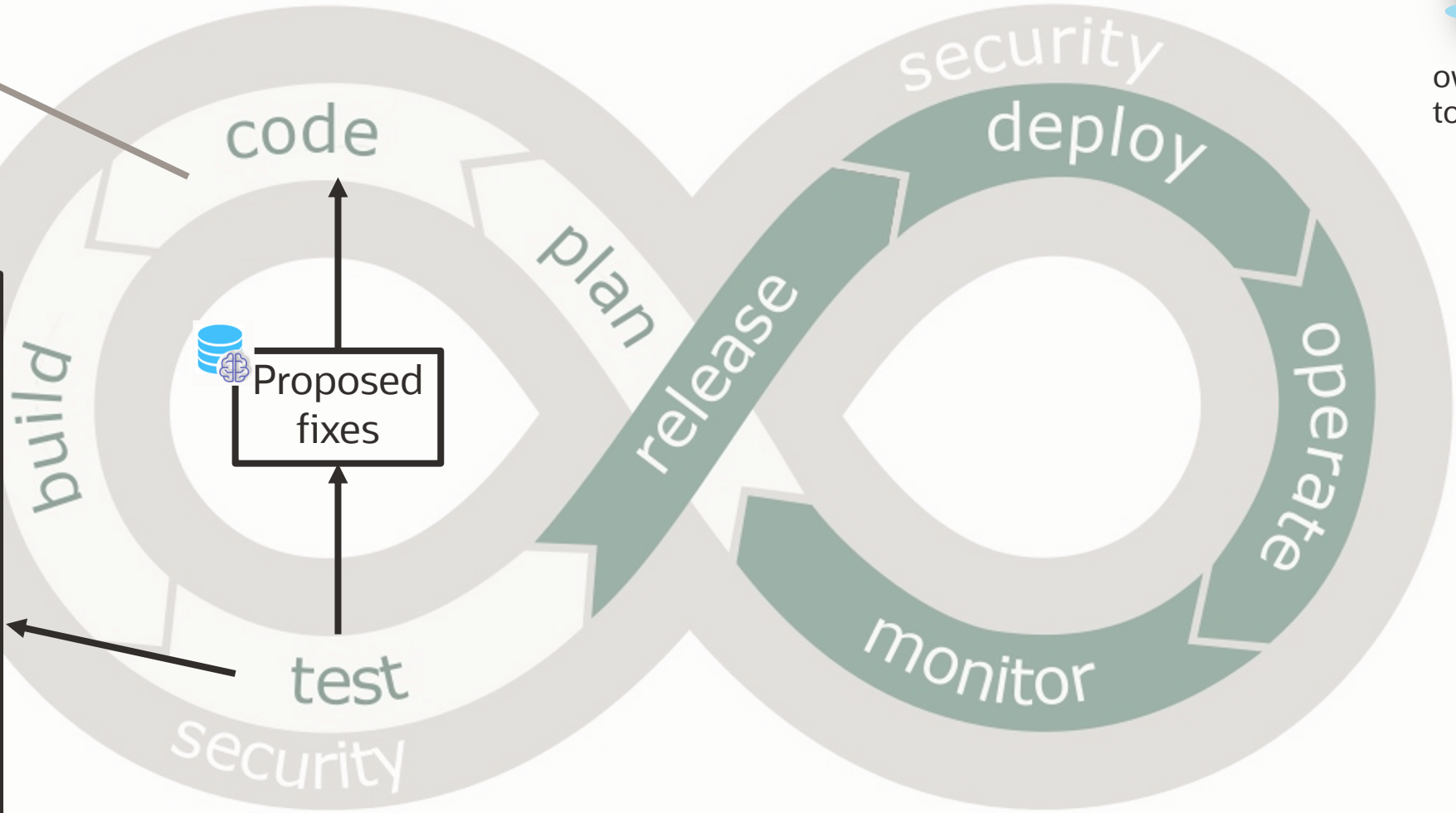


Intelligent security coding

Intelligent security testing
e.g., Automatically generate tests for security

Automatically propose bug fixes to failing security tests

Automatically check deployment containers and suggest non-malware version updates



©, Oracle and/or its affiliates



How do we check for malware imported via a 3rd party dependency or contained in a container?

“A decompiler is a program that reads a program written in a machine language – the source language – and translates it into an equivalent program in a high-level language – the target language.”

Cristina Cifuentes

“Reverse Compilation Techniques”, PhD Thesis,
Queensland University of Technology, July 1994

Decompilation to Reverse Engineer Malware

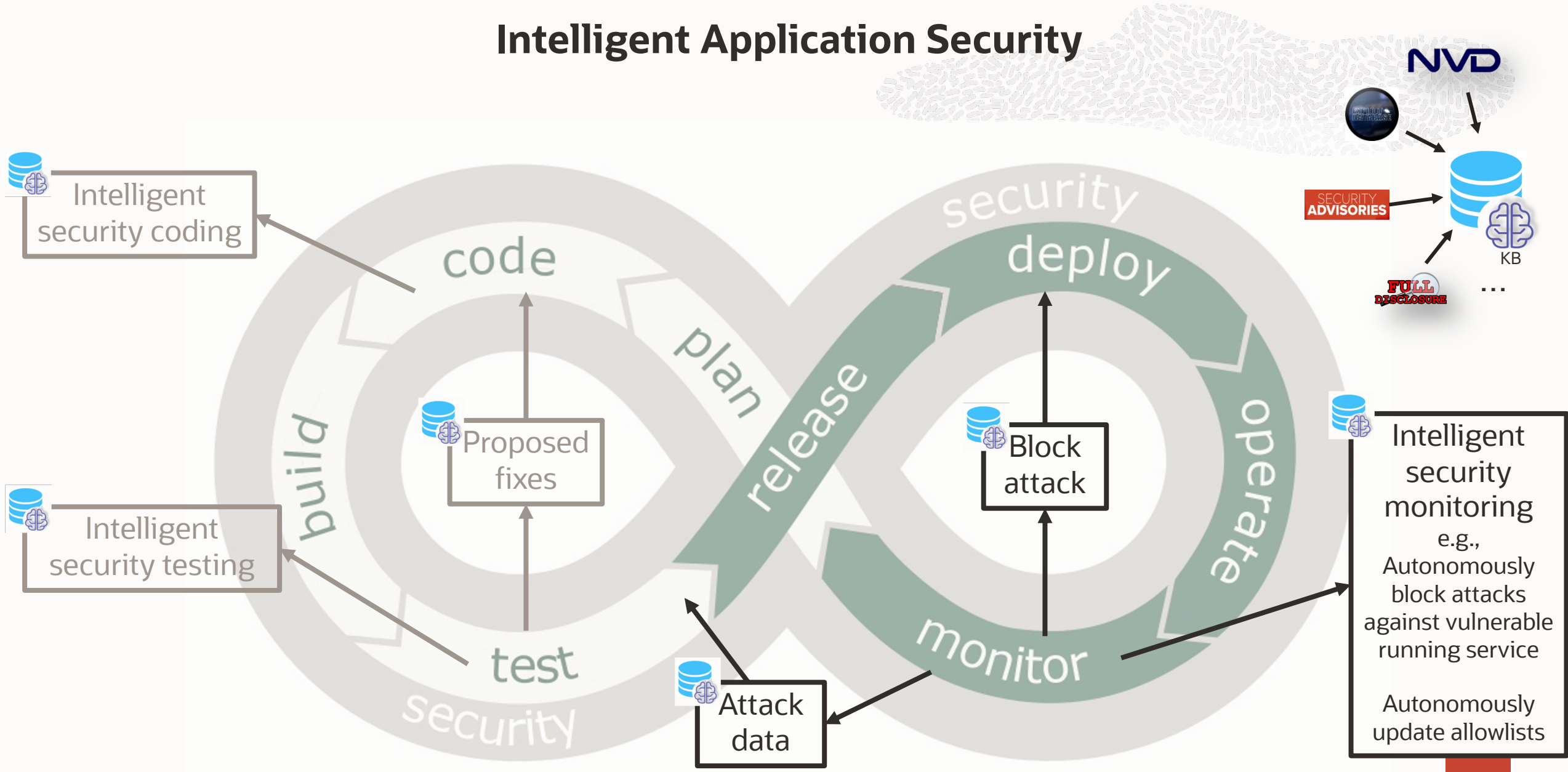
Ghidra's decompilation of WannaCry

```
int entry(undefined4 param_1,int param_2,undefined4 param_3)

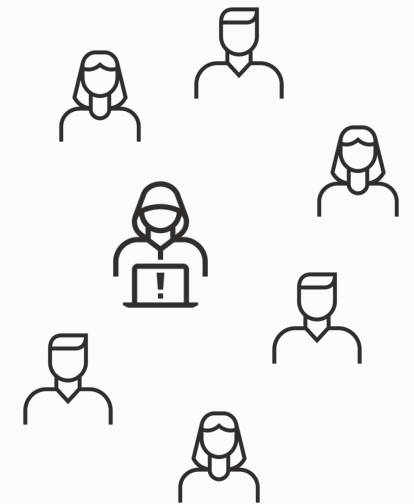
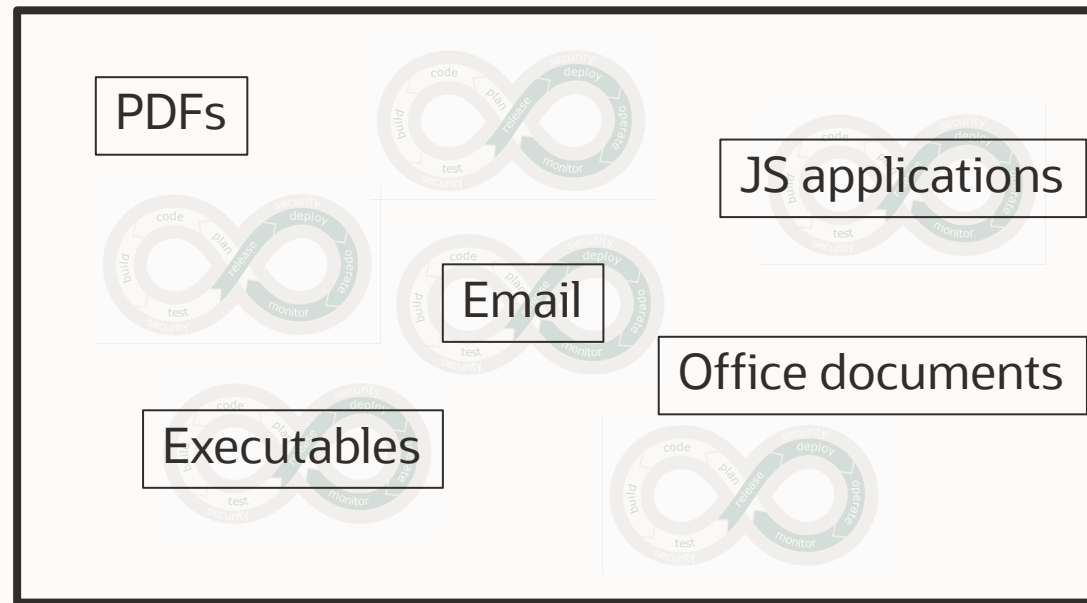
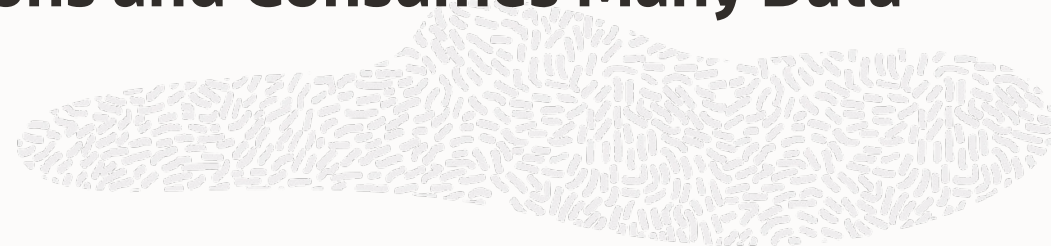
int iVar1;
int iVar2;
int iVar3;
undefined4 uVar4;

iVar1 = param_2;
iVar2 = _DAT_10003140;
if (param_2 != 0) {
    if ((param_2 != 1) && (param_2 != 2)) goto LAB_10001231;
    if ((DAT_10003150 != (code *)0x0) &&
        (iVar2 = (*DAT_10003150)(param_1,param_2,param_3), iVar2 == 0)) {
        return 0;
    }
    iVar2 = FUN_1000113e(param_1,param_2,param_3);
}
if (iVar2 == 0) {
    return 0;
}
AB_10001231:
iVar2 = FUN_10001000(param_1,param_2,param_3);
if (param_2 == 1) {
    if (iVar2 != 0) {
```

Intelligent Application Security



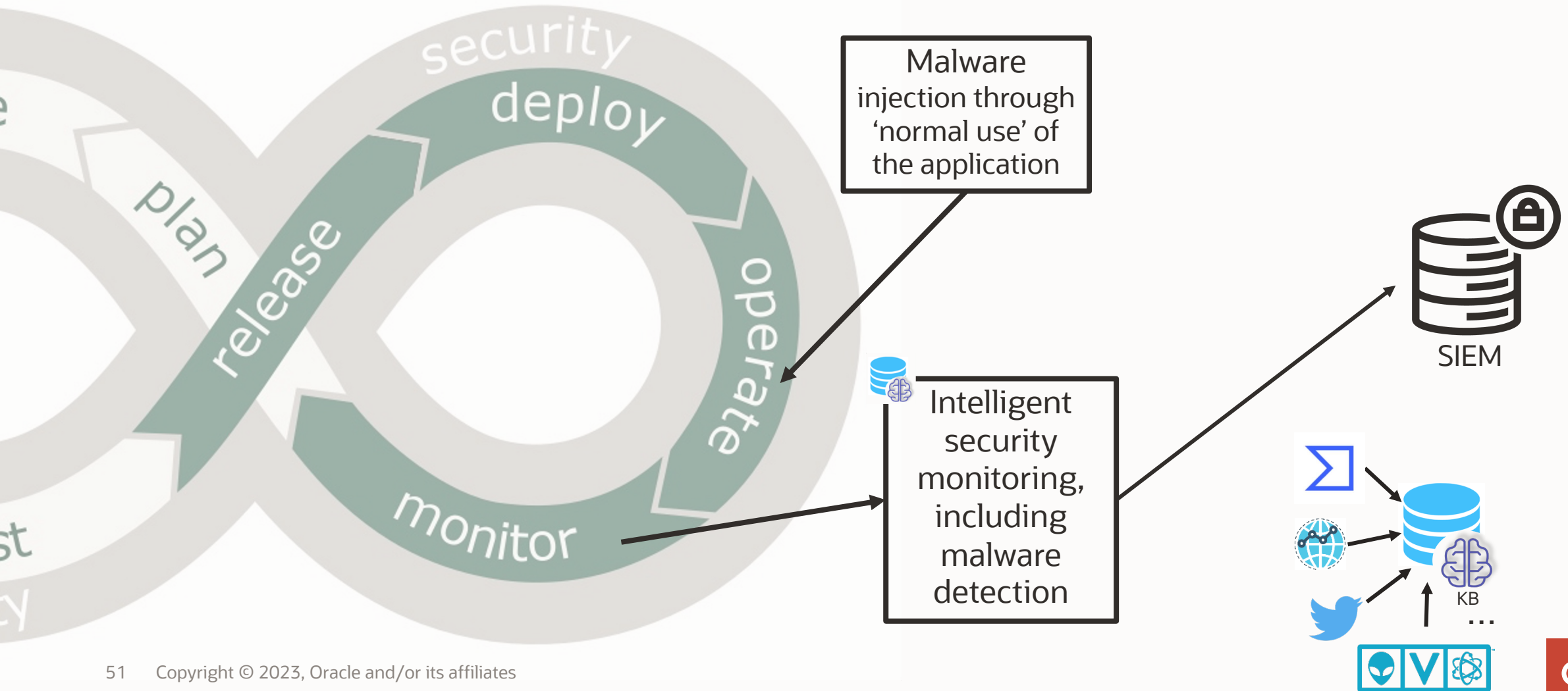
An Organisation Runs Many Applications and Consumes Many Data Formats



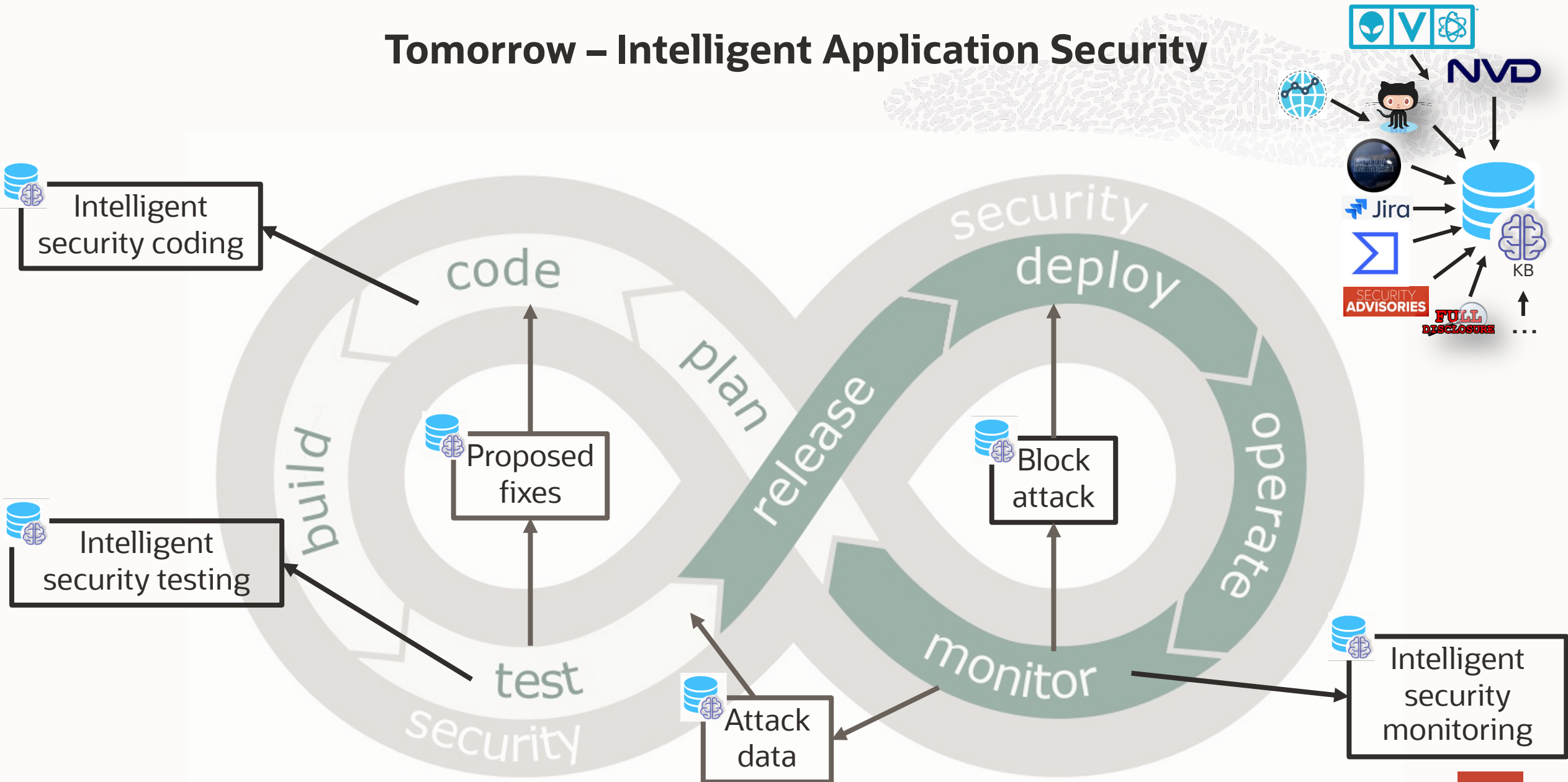
Office and PDF malicious files account for 33% of new malware detection. Exe accounts for 15.8%.

Mid-Year Update – 2020 SonicWall Cyber Threat Report
July 2020

Ingesting Data and Accessing Websites



Tomorrow – Intelligent Application Security





“Intelligent Application Security aims to provide an automated approach to integrate security into all aspects of application development and operations, at scale, using learning techniques that incorporate signals from the code and beyond, to provide actionable intelligence to developers, security analysts, operations staff, and autonomous systems.”

Cristina Cifuentes

October 2020



Closer integration of data produced by code analysis, build integration, testing, operations, and external source is needed.

Closer integration of data produced by code analysis, build integration, testing, operations, and external source is needed.

As well as teams providing and consuming data.

Recap

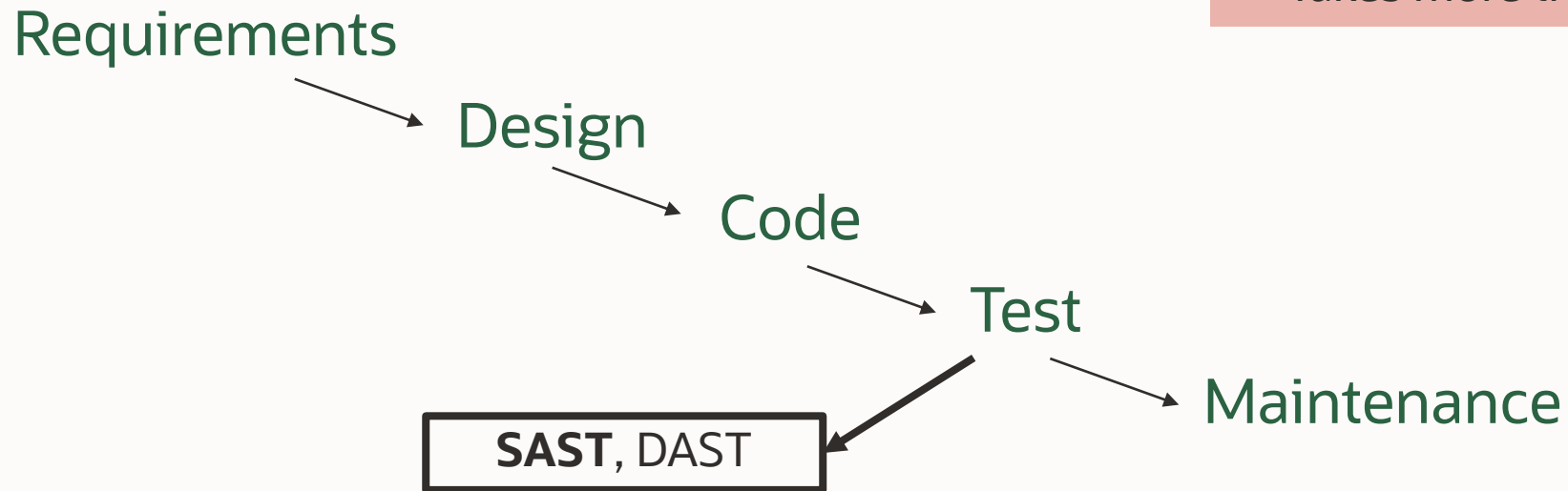
#ias

Yesteryear – Application Security Testing

Static Application Security Testing in the Late 2000s

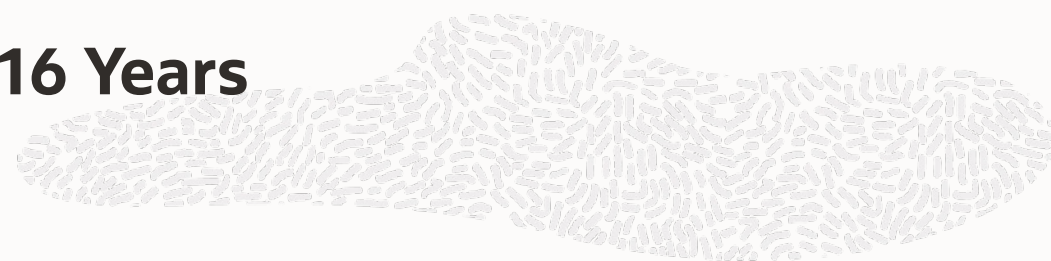
Challenges with SAST:

- Too many false positives
- Takes more than a nightly to run





During the Past 16 Years



- **Efficient analysis of full codebase**
 - Used to be nightly runs
 - Now part of Continuous Integration
- **Efficient analysis of changeset**
 - Prevent bugs from being introduced into the codebase
 - Can be hooked into the commit, push, pull request or merge request

Innovations for SAST:

- Precise results
- Scalable, can integrate early in the development cycle



Yesteryear – Application Security Testing in the 2000s

Dynamic Web Application Security Testing in 2015

Requirements



Design



Coding



Testing



Maintenance

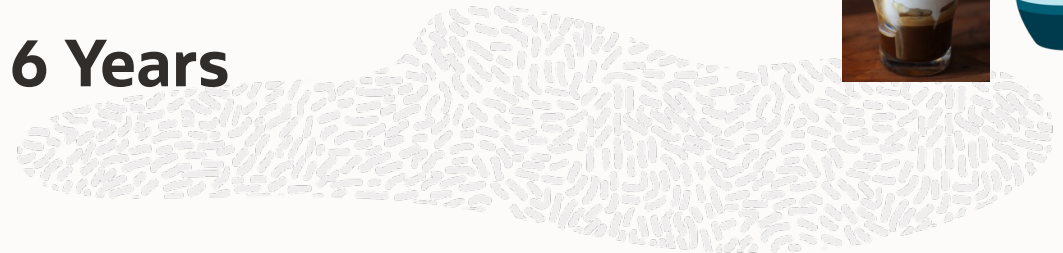


Challenges with DAST:

- Automation lacking for effective use of the tool
- Inefficient use of compute resources; misses vulnerabilities



During the Past 6 Years



Test

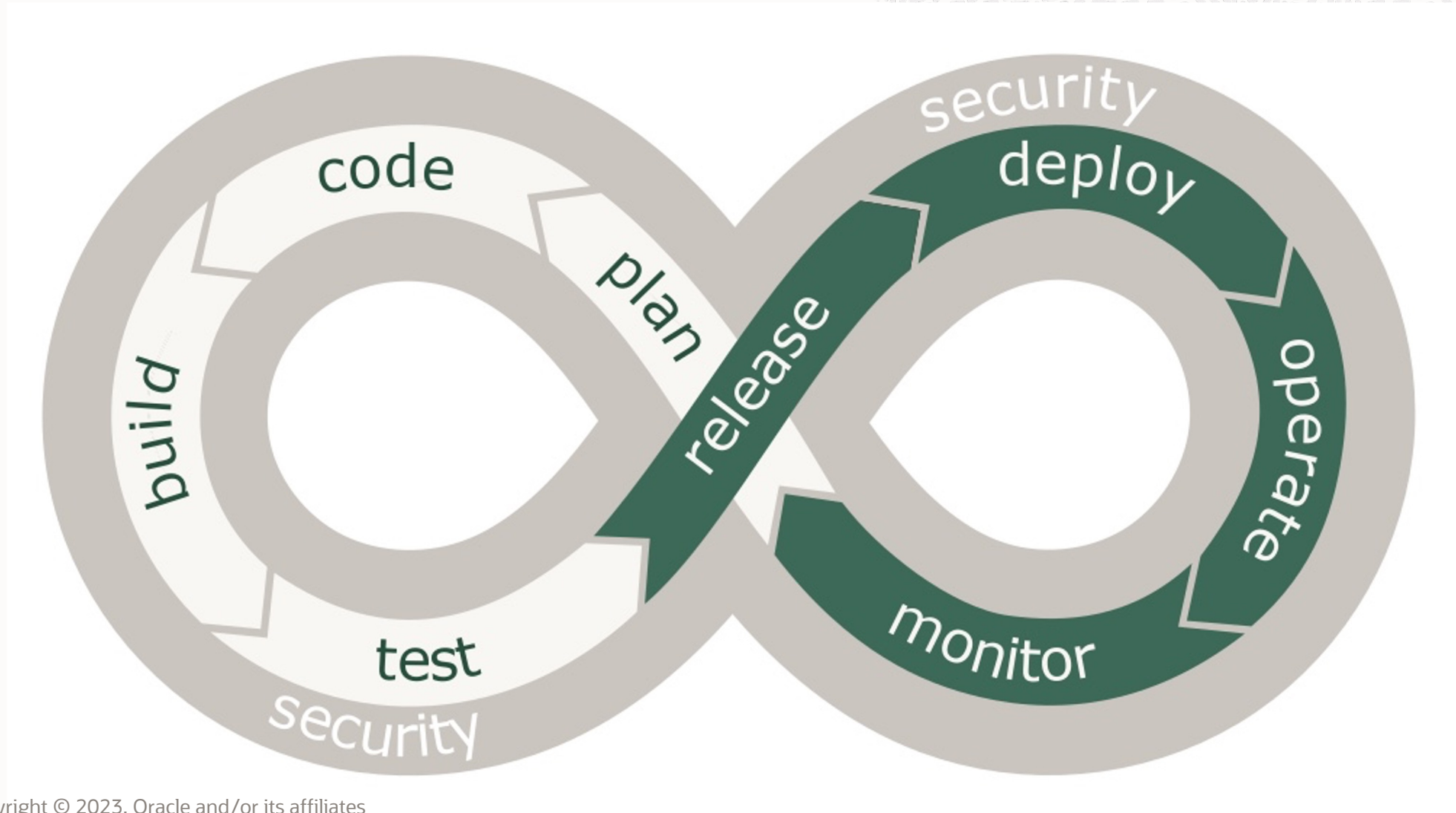
- Automatic, high coverage detection of endpoints
 - Automated – no input from pentester or developer
 - Generation of Swagger spec to drive inputs into existing blackbox REST fuzzer
- Integrate greybox solution
 - Greybox approach provides context to drive efficiency into existing blackbox solutions

Innovations for DAST:

- Automated attack surface detection
- Efficient use of compute resources finds more 0-days



Today – DevSecOps – Integrating Security Into DevOps



Learning-based technologies have ripen
over the past 10 years

Security is not just for expert developers

Automation is key

It's time to combine program analysis,
learning-based techniques and
data analytics to make
Intelligent Application Security,
at scale, a reality



ORACLE

cristina.cifuentes@oracle.com

<http://labs.oracle.com/locations/australia>

Twitter: @criscifuentes

LinkedIn: drcristinacifuentes



Our mission is to help people see
data in new ways, discover insights,
unlock endless possibilities.



ORACLE