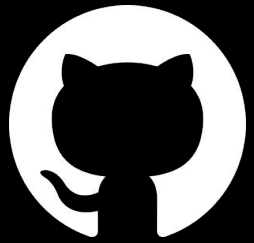
GitHub Security Meetup

securitylab.github.com/get-involved



No More Whack-a-Mole:

How to Find and Prevent Entire Classes of Security Vulnerabilities

Presented by Sam Lanning - GitHub

 @s0

 @samlanning

About Me

- Developer Advocate for GitHub
  - (formerly Semmle, acquired by GitHub)
  - (formerly core developer for LGTM.com)
- Passionate about Open Source, Security, Privacy, Cryptography, Vulnerability Research, Code Quality & Lighting.
- Twitter: **`@samlanning`** GitHub: **`@s0`**

A story of many bugs (CVE-2017-8046)

7 September 2017
Mo privately discloses vulnerability and exploit in Spring Framework

21 September 2017
Pivotal publish a patch, and make an announcement.

22 September 2017
Mo checks patch, sees it's incomplete sends updated exploit to Pivotal

26 September 2017
Pivotal sends Mo details of second attempt at fix

27 September 2017
Mo checks patch, sees it's ***still*** incomplete sends updated exploit to Pivotal

25 October 2017
Pivotal publishes a complete refactor of relevant code to hopefully prevent further occurrences.

https://blog.semmle.com/spring-data-rest-CVE-2017-8046-ql/

A story of many bugs 2

27 April 2016
S2-032 / CVE-2016-3081
RCE in Apache Struts 2 via OGNL
Nike Zheng

12 May 2016
S2-033 / CVE-2016-3087
RCE in Apache Struts 2 via OGNL
Alvaro Munoz

20 June 2016
S2-037 / CVE-2016-4438
RCE in Apache Struts 2 via OGNL
Chao Jack, Shinsaku Nomura

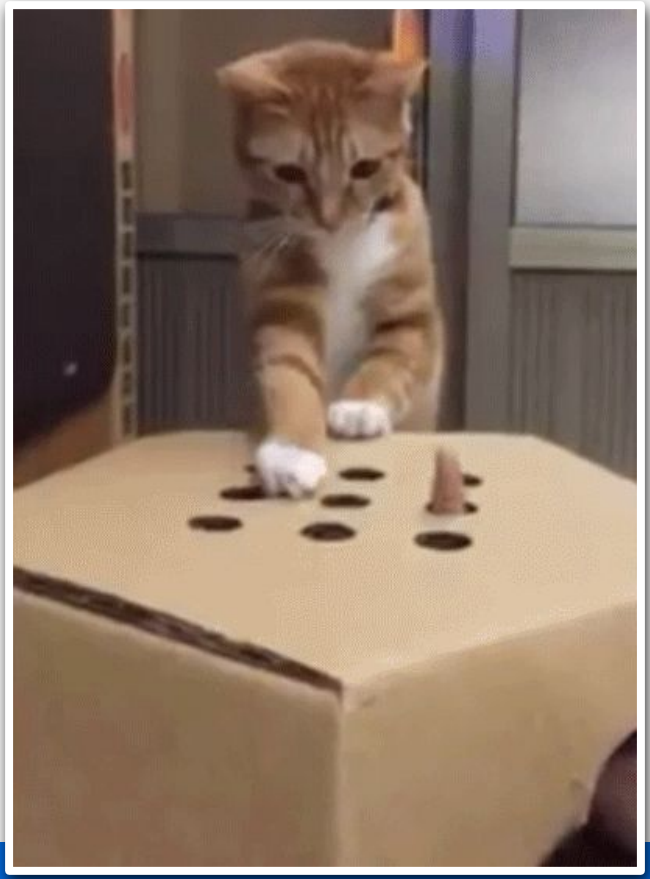
19 March 2017
S2-045 / CVE-2017-5638
RCE in Apache Struts 2 via OGNL
Nike Zheng

22 September 2017
S2-046 / CVE-2017-5638
RCE in Apache Struts 2 via OGNL
Chris Frohoff, Nike Zheng, Alvaro Munoz

24 September 2018
S2-057 / CVE-2018-11776
RCE in Apache Struts 2 via OGNL
Man Yue Mo

See Also: CVE-2012-0391, CVE-2012-0392, CVE-2012-0394, CVE-2013-1965, CVE-2013-1966, CVE-2013-2115, CVE-2013-2134, CVE-2013-2135, CVE-2016-0785, CVE-2016-3090

https://www.reddit.com/r/gifs/comments/2nyeb1/arcade_game_for_cats/



Solution:

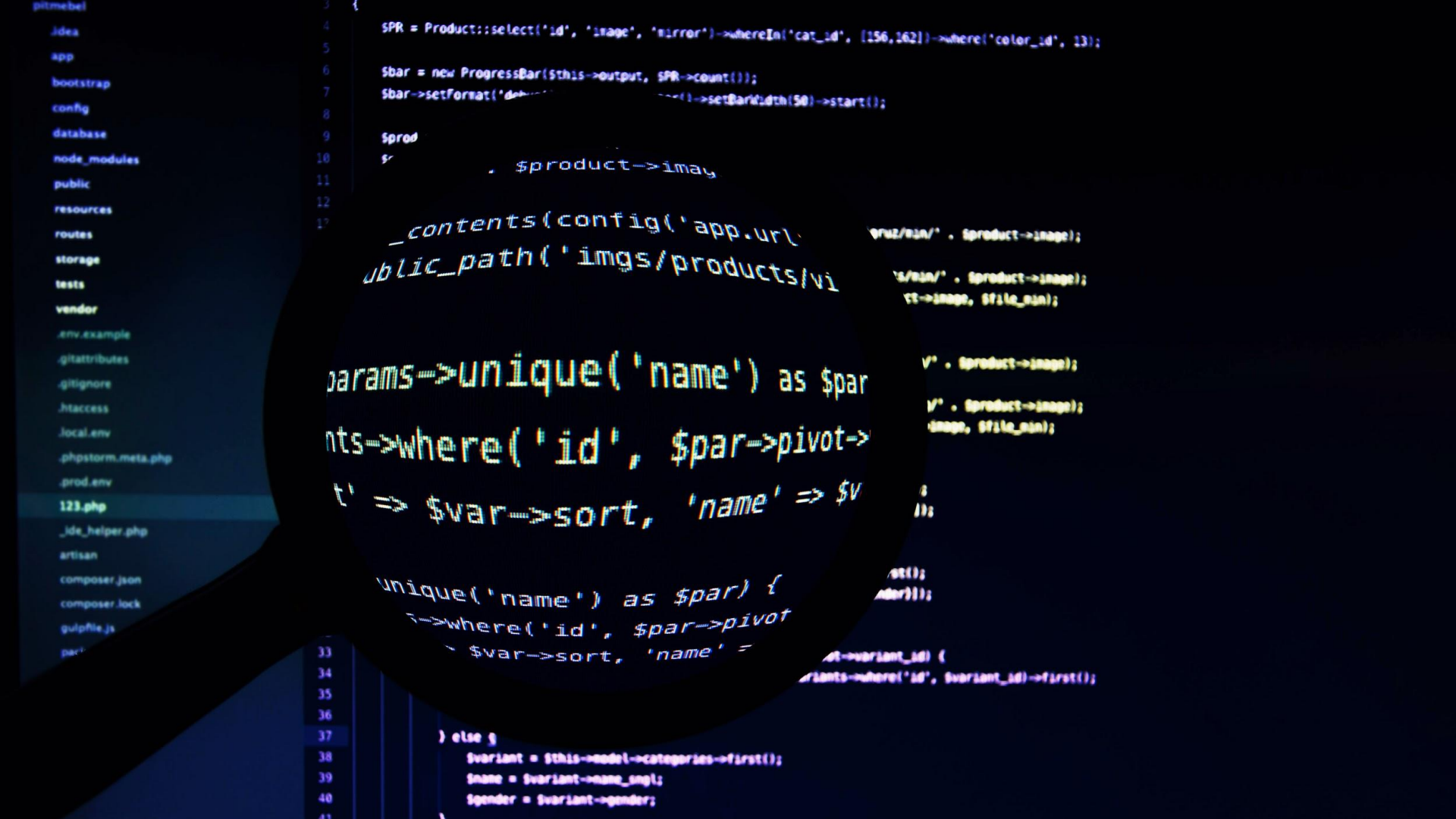
When a new mistake is discovered, try and find similar mistakes across your code base

Variant Analysis?

“After doing this [root cause analysis], our next step is **variant analysis**: finding and investigating any variants of the vulnerability. It’s important that we find all such variants and patch them simultaneously, otherwise we bear the risk of these being exploited in the wild.”

- Steven Hunter, MSRC Vulnerabilities & Mitigations team

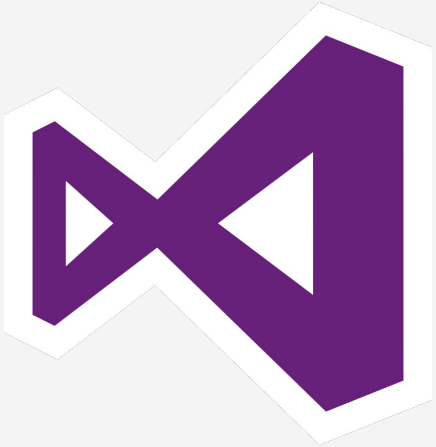
https://blogs.technet.microsoft.com/srd/2018/08/16/vulnerability-hunting-with-semmle-ql-part-1/


pitmebel
.idea
app
bootstrap
config
database
node_modules
public
resources
routes
storage
tests
vendor
.env.example
.gitattributes
.gitignore
.htaccess
.local.env
.phpstorm.meta.php
.prod.env
123.php
_ide_helper.php
artisan
composer.json
composer.lock
gulpfile.js
$PR = Product::select('id', 'image', 'mirror')->whereIn('cat_id', [156,162])->where('color_id', 13);
$bar = new ProgressBar($this->output, $PR->count());
$product->imag
_contents(config('app.url
ublic_path('imgs/products/vi
params->unique('name') as $par
nts->where('id', $par->pivot->
t' => $var->sort, 'name' => $v
unique('name') as $par) {
->where('id', $par->pivot
$var->sort, 'name'
} else {
$variant = $this->model->categories->first();
$name = $variant->name_sngl;
$gender = $variant->gender;

Code Navigation / IDE

sourcegraph

eclipse



Never enough security researchers

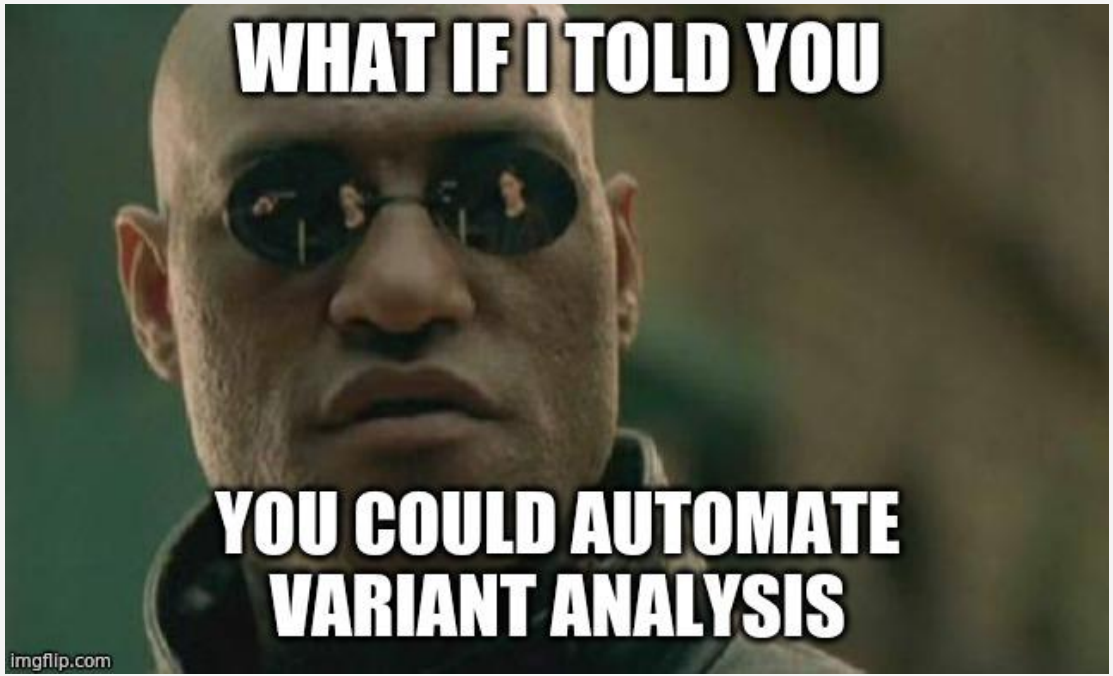
	Engineers	Security Engineers	%
facebook.	10,381	151	1.4%
Google	35,208	390	1.1%
Microsoft	39,894	492	1.2%
Bloomberg	5,015	34	0.7%
Uber	3,674	60	1.6%

Source: LinkedIn Sales Navigator

Even hi-tech leaders, can't find enough talent

The security skills gap is only expected to grow

Automating Variant Analysis



Automating Variant Analysis

An Example: Chakra

```cpp
HRESULT SomeClass::vulnerableFunction(Var* args, UINT argCount, Var* retVal)
{
    // get first argument -
    // from Chakra, acquire pointer to array
    BYTE* pBuffer;
    UINT bufferSize;
    hr = Jscript::GetTypedArrayBuffer(args[1], &pBuffer, &bufferSize);

    // get second argument -
    // from Chakra, obtain an integer value
    int someValue;
    hr = Jscript::VarToInt(args[2], &someValue);

    // perform some operation on the array acquired previously
    doSomething(pBuffer, bufferSize);
}
```

C++

https://blogs.technet.microsoft.com/srd/2018/08/16/vulnerability-hunting-with-semmle-ql-part-1/

An Example: Chakra

```cpp
HRESULT SomeClass::vulnerableFunction(Var* args, UINT argCount, Var* retVal)
{
    // get first argument -
    // from Chakra, acquire pointer to array
    BYTE* pBuffer;
    UINT bufferSize;
    hr = Jscript::GetTypedArrayBuffer(args[1], &pBuffer, &bufferSize);

    // get second argument -
    // from Chakra, obtain an integer value
    int someValue;
    hr = Jscript::VarToInt(args[2], &someValue);

    // perform some operation on the array acquired previously
    doSomething(pBuffer, bufferSize);
}
```

C++

https://blogs.technet.microsoft.com/srd/2018/08/16/vulnerability-hunting-with-semmle-ql-part-1/

An Example: Chakra

```cpp
HRESULT SomeClass::vulnerableFunction(Var* args, UINT argCount, Var* retVal)
{
    // get first argument -
    // from Chakra, acquire pointer to array
    BYTE* pBuffer;
    UINT bufferSize;
    hr = Jscript::GetTypedArrayBuffer(args[1], &pBuffer, &bufferSize);

    // get second argument -
    // from Chakra, obtain an integer value
    int someValue;
    hr = Jscript::VarToInt(args[2], &someValue);

    // perform some operation on the array acquired previously
    doSomething(pBuffer, bufferSize);
}
```

C++

https://blogs.technet.microsoft.com/srd/2018/08/16/vulnerability-hunting-with-semmle-ql-part-1/

An Example: Chakra

```cpp
HRESULT SomeClass::vulnerableFunction(Var* args, UINT argCount, Var* retVal)
{
    // get first argument -
    // from Chakra, acquire pointer to array
    BYTE* pBuffer;
    UINT bufferSize;
    hr = Jscript::GetTypedArrayBuffer(args[1], &pBuffer, &bufferSize);

    // get second argument -
    // from Chakra, obtain an integer value
    int someValue;
    hr = Jscript::VarToInt(args[2], &someValue);

    // perform some operation on the array acquired previously
    doSomething(pBuffer, bufferSize);
}
```

C++

https://blogs.technet.microsoft.com/srd/2018/08/16/vulnerability-hunting-with-semmle-ql-part-1/

An Example: Chakra

```cpp
HRESULT SomeClass::vulnerableFunction(Var* args, UINT argCount, Var* retVal)
{
    // get first argument -
    // from Chakra, acquire pointer to array
    BYTE* pBuffer;
    UINT bufferSize;
    hr = Jscript::GetTypedArrayBuffer(args[1], &pBuffer, &bufferSize);

    // get second argument -
    // from Chakra, obtain an integer value
    int someValue;
    hr = Jscript::VarToInt(args[2], &someValue);

    // perform some operation on the array acquired previously
    doSomething(pBuffer, bufferSize);
}
```

C++

https://blogs.technet.microsoft.com/srd/2018/08/16/vulnerability-hunting-with-semmle-ql-part-1/

An Example: Chakra

```javascript
var buf = new ArrayBuffer(length);
var arr = new Uint8Array(buf);

var param = {}
param.valueOf = function() {
    neuter(buf);  // neuter `buf` by e.g. posting it to a web worker
    gc();         // trigger garbage collection
    return 0;
};

vulnerableFunction(arr, param);
```

JavaScript

https://blogs.technet.microsoft.com/srd/2018/08/16/vulnerability-hunting-with-semmle-ql-part-1/

An Example: Chakra

```JavaScript
var buf = new ArrayBuffer(length);
var arr = new Uint8Array(buf);

var param = {}
param.valueOf = function() {
    neuter(buf);  // neuter `buf` by e.g. posting it to a web worker
    gc();         // trigger garbage collection
    return 0;
};

vulnerableFunction(arr, param);
```

https://blogs.technet.microsoft.com/srd/2018/08/16/vulnerability-hunting-with-semmle-ql-part-1/

An Example: Chakra

```
from Variable v, TypedArrayBufferPointer def, FunctionCall call, VariableAccess use
where
  // variable is defined at "def", and used at "use"
  v.getAnAccess() = def and v.getAnAccess() = use
  // "call" may eventually call a function that calls back into JS code
  and exists(Function g | g.hasName("MethodCallToPrimitive") and call.getTarget().calls+(g))
  // "call" happens after "def"
  and def.getASuccessor+() = call
  // "use" happens after "call"
  and call.getASuccessor+() = use
select def, call, use
```

Query

*slightly modified query from:

https://blogs.technet.microsoft.com/srd/2018/08/16/vulnerability-hunting-with-semmle-ql-part-1/

Beyond your own code

- ➔ Make your (general-purpose) queries/checks open source!
- ➔ Use external queries/checks!

ZipSlip

https://snyk.io/research/zip-slip-vulnerability

ZipSlip

Metadata	Name (path)	Data
<metadata>	lib/foo	<7731911f...
<metadata>	lib/bar	<236dbe48...
<metadata>	usr/foo	<e80b70d2...
<metadata>	usr/foo/bar	<63f30ae0...

https://snyk.io/research/zip-slip-vulnerability

ZipSlip

```
../../../.bashrc

../../../../../../../../../etc/crontab
```

https://snyk.io/research/zip-slip-vulnerability

ZipSlip

```java
Enumeration<ZipEntry> entries = zip.getEntries()
while (entries.hasMoreElements()) {
    ZipEntry e = entries.nextElement();
    File f = new File(destinationDir, e.getName());
    InputStream input = zip.getInputStream(e);
    IOUtils.copy(input, write(f));
}
```

Java

https://snyk.io/research/zip-slip-vulnerability

ZipSlip

```java
Enumeration<ZipEntry> entries = zip.getEntries()
while (entries.hasMoreElements()) {
    ZipEntry e = entries.nextElement();
    File f = new File(destinationDir, e.getName());
    InputStream input = zip.getInputStream(e);
    IOUtils.copy(input, write(f));
}
```

Java

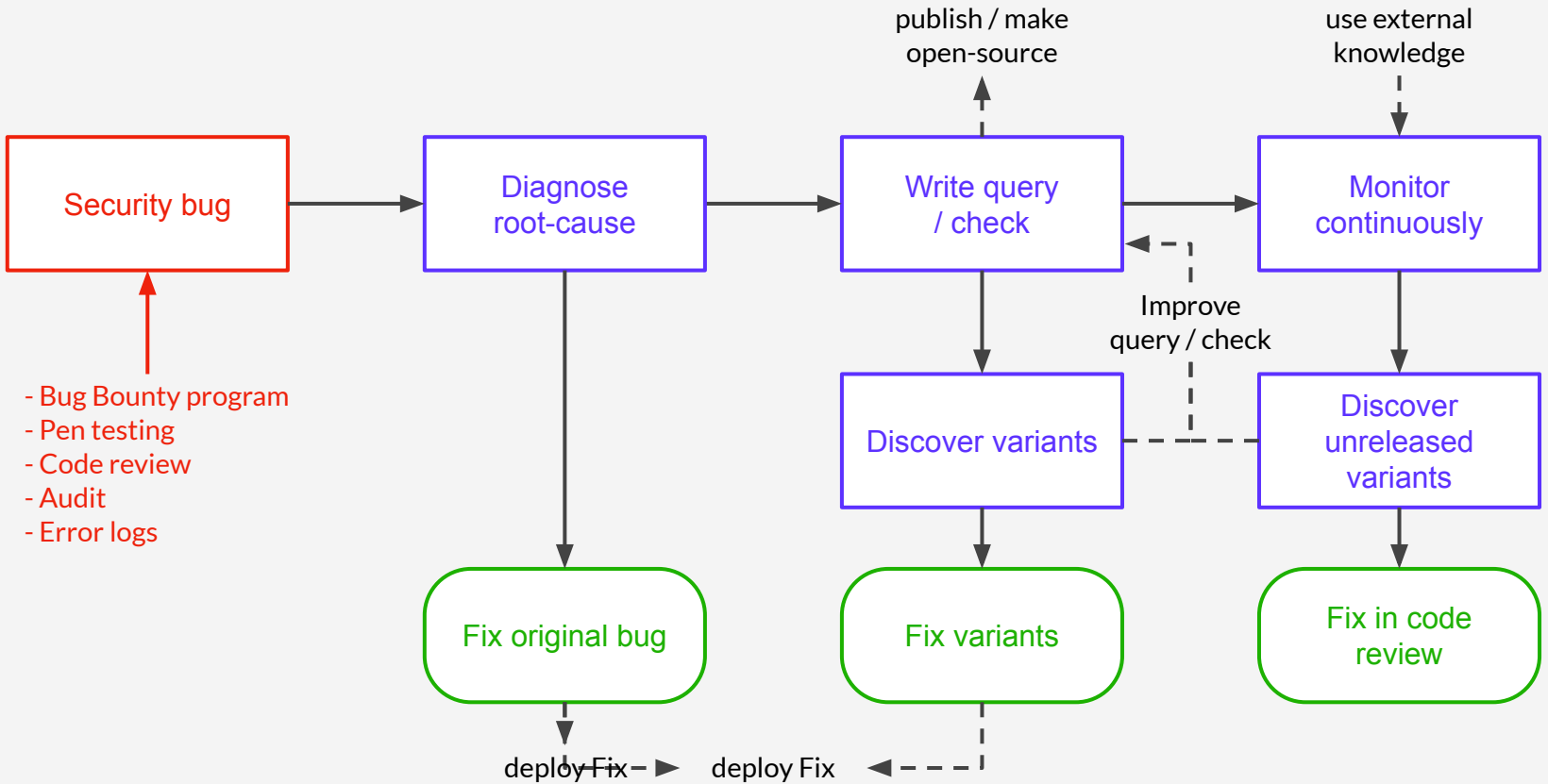
https://snyk.io/research/zip-slip-vulnerability

ZipSlip

```java
Enumeration<ZipEntry> entries = zip.getEntries()
while (entries.hasMoreElements()) {
    ZipEntry e = entries.nextElement();
    File f = new File(destinationDir, e.getName());
    if (!f.toPath().normalize().startsWith(destinationDir.toPath()))
        throw new IOException("Bad zip entry");
    InputStream input = zip.getInputStream(e);
    IOUtils.copy(input, write(f));
}
```

Java

https://snyk.io/research/zip-slip-vulnerability


publish / make
open-source
use external
knowledge
Security bug
Diagnose
root-cause
Write query
/ check
Monitor
continuously
Improve
query / check
- Bug Bounty program
- Pen testing
- Code review
- Audit
- Error logs
Discover variants
Discover
unreleased
variants
Fix original bug
Fix variants
Fix in code
review
deploy Fix
deploy Fix

No vulnerability response process?

Independent security researcher / consultant?

Get Started

- ➔ Writing / Maintaining Software?
  - ◆ Look at which tools other teams are using
  - ◆ Try out a selection, choose what works for you
- ➔ Security Researcher?
  - ◆ Experiment writing checks / queries with different technologies, see what works for you
  - ◆ Blog posts from researchers finding variants

Recap

- ➔ You should do variant analysis (if creating software)
- ➔ Better yet, you should do ***automated*** variant analysis
- ➔ Checks should be run ***continuously***, not once-off!
- ➔ ***Use*** and ***contribute*** to open-source queries / checks
- ➔ Can use variant analysis to supercharge research
- ➔ VA ***compliments*** (not replaces) other security practices

Thank You

Questions?

 @samlanning

 @s0