Scaling the Security Researcher to Eliminate OSS Vulnerabilities Once and for All

- Jonathan Leitschuh -
- Patrick Way -
Hello!

- Jonathan Leitschuh -

Software Engineer & Security Researcher
Dan Kaminsky Fellowship @ HUMAN Security
GitHub Star & GitHub Security Ambassador
Twitter: @JLLeitschuh
GitHub: JLLeitschuh
Hello!

- Patrick Way -
Senior Software Engineer
OpenRewrite Team @ Moderne
Twitter: @WayPatrick
GitHub: pway99
Disclaimer
Supported by
The
Dan Kaminsky Fellowship
at
HUMAN Security
It Started

With a Simple Vulnerability
// build.gradle

maven {
    setUrl("http://dl.bintray.com/kotlin/ktor")
}

HTTP Download of Dependencies in the Java Ecosystem
Why is HTTPS important?
<!-- Compiler & Test Dependencies -->

<repositories>
  <repository>
    <id>example-id</id>
    <name>Example insecure repository</name>
    <url>http://[SOME URL HERE]</url>
  </repository>
</repositories>

HTTP Download of Dependencies in the Java Ecosystem
<!-- Artifact upload - Credentials!! -->

<distributionManagement>
  <repository>
    <id>example-id</id>
    <name>Example insecure repository</name>
    <url>http://[SOME URL HERE]</url>
  </repository>
</distributionManagement>

HTTP Download of Dependencies in the Java Ecosystem
This Vulnerability was Everywhere!
Who else was vulnerable?
“25% of Sonatype Maven Central downloads are still using HTTP”

- Sonatype June 2019 -
How do we fix this?
Decommissioning HTTP Support

On or around January 15th, 2020

- Maven Central (Sonatype)
- JCenter (JFrog)
- Spring (Pivotal)
- Gradle Plugin Portal (Gradle)
“20% of Sonatype Maven Central Traffic is STILL using HTTP”

- Sonatype January 2020 -
You can imagine what happened...
January 15th, 2020
BROKEN SOFTWARE
BROKEN SOFTWARE EVERYWHERE
We stopped the bleeding
What about the other repositories?
Only the *most commonly* used repositories

- Maven Central (Sonatype)
- JCenter (JFrog)
- Spring (Pivotal)
- Gradle Plugin Portal (Gradle)
How do we fix the rest?
Bulk Pull Request Generation!
How?
import java
import semmle.code.xml.MavenPom

private classDeclaredRepository extendsPomElement {
    DeclaredRepository() {
        this.getName() = "repository" or
        this.getName() = "snapshotRepository" or
        this.getName() = "pluginRepository"
    }

    string getUrl() { result = getAChild("url").(PomElement).getValue() }

    predicate isInsecureRepositoryUsage() {
        getUrl().matches("http://") or
        getUrl().matches("ftp://")
    }
}

from DeclaredRepository repository
where repository.isInsecureRepositoryUsage()
select repository,
"Downloading or uploading artifacts over insecure protocol (eg. http or ftp) to/from repository " + repository.getUrl()
CodeQL scans 100Ks of OSS Projects
import java
import semmle.code.xml.MavenPom

private class DeclaredRepository extends PomElement {
    DeclaredRepository() {
        this.getName() = "repository" or
        this.getName() = "snapshotRepository" or
        this.getName() = "pluginRepository"
    }

    string getUrl() { result = getAChild("url").(PomElement).getValue() }

    predicate isInsecureRepositoryUsage() {
        getUrl().matches("http://") or
        getUrl().matches("ftp://")
    }
}

from DeclaredRepository repository where repository.isInsecureRepositoryUsage()
select repository, "Downloading or uploading artifacts over insecure protocol (eg. http or ftp) to/from repository " + repository.getUrl()
Pull Request Generator
Version 1

- Python Based
- Wrapper over ‘hub’ CLI
- One Nasty Regular Expression
- Bouncing off GitHub’s rate limiter
from vulnerability_fix_engine import VulnerabilityFixModule

@dataclass()
class PenVulnerabilityFixModule(VulnerabilityFixModule):

def __init__(self, branch_name: str = 'fix/ALL/use_httpS_to_resolve_dependencies',
             clone_repo: str = 'cloned_repos',
             data_base_dir: str = 'insecure_pom_data',
             save_point: str = 'save_points',
             pr_message_file: str = f'{str(pathlib.Path().absolute())}/PR_MESSAGE.md',
             commit_message: str = textwrap.dedent('"Use HTTPS instead of HTTP to resolve dependencies"'),
             ) -> None:
    """
    This fixes a security vulnerability in this project where the 'pom.xml'
    files were configuring Maven to resolve dependencies over HTTP instead of
    HTTPS.
    """

    Signed-off-by: Jonathan Leitschuh <Jonathan.Leitschuh@gmail.com>  

    self.p_fix_regex = re.compile(r'\(\{\{\{repository\}\}\}{{\{\{pluginRepository\}\}{{\{\{snapshotRepository\}\}{{\{\{repository\}\}{{\url\}http://([^\}]+}\}}}}{\url\}http://([^\}]+)}',
                                  re.IGNORECASE + re.MULTILINE + re.DOTALL)

    self.replacement = r'\2https://\1\4'

    async def de_fix_vulnerable_file(self, *, project_name: str, file: str, expected_fix_count: int) -> int:
        async with siefiles.open(file, newline='') as vulnerable_file:
            contents = await vulnerable_file.read()

            new_contents, count = self.p_fix_regex.subn(self.replacement, contents)
            if count != expected_fix_count:
                logging.warning('Fix for `%s` did not match expected fix count: (expected: %d, actual: %d)',
                                project_name, expected_fix_count, count)

        async with siefiles.open(file, 'w', newline='') as vulnerable_file:
            await vulnerable_file.write(new_contents)

        return count
p_fix_regex = \n    re.compile(\n        r'(?:(?:<repository>)(?:<pluginRepository>)(?:<snapshotRepository>)(?:!(?:repository>))*)((?:url:|s*)http://(:s*)(/s*)\s*/\s*))',
        re.IGNORECASE + re.MULTILINE + re.DOTALL
    )

replacement = r'1\2https:\3\4'
I had a problem so I used regular expressions

Now I have two problems!
It worked!
<table>
<thead>
<tr>
<th>Username</th>
<th>Project Name</th>
<th>Description</th>
<th>Status</th>
<th>Created Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>01Sharphooter/Social</td>
<td>[SECURITY] Use HTTPS to resolve dependencies in Maven Build</td>
<td>#1 opened on Feb 11 by JLLeitschuh</td>
<td>Open</td>
<td></td>
</tr>
<tr>
<td>4thine/cling</td>
<td>[SECURITY] Use HTTPS to resolve dependencies in Maven Build</td>
<td>#236 opened on Feb 11 by JLLeitschuh</td>
<td>Open</td>
<td></td>
</tr>
<tr>
<td>1000Memories/photon-core</td>
<td>[SECURITY] Use HTTPS to resolve dependencies in Maven Build</td>
<td>#4 opened on Feb 11 by JLLeitschuh</td>
<td>Open</td>
<td></td>
</tr>
<tr>
<td>2xei/spring-boot-tiles</td>
<td>[SECURITY] Use HTTPS to resolve dependencies in Maven Build</td>
<td>#1 opened on Feb 11 by JLLeitschuh</td>
<td>Open</td>
<td></td>
</tr>
<tr>
<td>weamylady2/iOS_remote</td>
<td>[SECURITY] Use HTTPS to resolve dependencies in Maven Build</td>
<td>#23 opened on Feb 11 by JLLeitschuh</td>
<td>Open</td>
<td></td>
</tr>
<tr>
<td>yishen/zzzobspk</td>
<td>[SECURITY] Use HTTPS to resolve dependencies in Maven Build</td>
<td>#1 opened on Feb 11 by JLLeitschuh</td>
<td>Open</td>
<td></td>
</tr>
<tr>
<td>wlu-mstr/hbase-ornlite</td>
<td>[SECURITY] Use HTTPS to resolve dependencies in Maven Build</td>
<td>#1 opened on Feb 11 by JLLeitschuh</td>
<td>Open</td>
<td></td>
</tr>
<tr>
<td>zhangdaiscott/jeeqc</td>
<td>[SECURITY] Use HTTPS to resolve dependencies in Maven Build</td>
<td>#50 opened on Feb 11 by JLLeitschuh</td>
<td>Open</td>
<td></td>
</tr>
<tr>
<td>wsc2/carbon-device-mgt-plugins</td>
<td>[SECURITY] Use HTTPS to resolve dependencies in Maven Build</td>
<td>#577 opened on Feb 11 by JLLeitschuh - Review required</td>
<td>Open</td>
<td></td>
</tr>
<tr>
<td>wsc2/product-lots</td>
<td>[SECURITY] Use HTTPS to resolve dependencies in Maven Build</td>
<td>#56 opened on Feb 11 by JLLeitschuh - Review required</td>
<td>Open</td>
<td></td>
</tr>
<tr>
<td>xauth/xs2jh4net</td>
<td>[SECURITY] Use HTTPS to resolve dependencies in Maven Build</td>
<td>#50 opened on Feb 11 by JLLeitschuh</td>
<td>Open</td>
<td></td>
</tr>
<tr>
<td>xzer/run-jetty-run</td>
<td>[SECURITY] Use HTTPS to resolve dependencies in Maven Build</td>
<td>#57 opened on Feb 11 by JLLeitschuh</td>
<td>Open</td>
<td></td>
</tr>
<tr>
<td>yanghus/banyan</td>
<td>[SECURITY] Use HTTPS to resolve dependencies in Maven Build</td>
<td>#5 opened on Feb 11 by JLLeitschuh</td>
<td>Open</td>
<td></td>
</tr>
</tbody>
</table>
<repository>
    <id>marandombox</id>
    <url>http://repo.omarandombox.com/content/groups/public</url>
</repository>

<repository>
    <id>spigot</id>
    <url>https://hub.spigotmc.org/nexus/content/groups/public</url>
</repository>

<repository>
    <id>vaultr</id>
    <url>https://hub.spigotmc.org/nexus/content/groups/public</url>
</repository>

<repository>
    <id>nexus.hc.to</id>
    <url>http://nexus.hc.to/content/repositories/pub_releases</url>
</repository>

<!-- Has a copy of metrics RB-SNAPSHOT! -->
<repository>
    <id>elmakers</id>
    <url>http://maven.elmakers.com/repository/</url>
</repository>

<pluginRepositories>
    <pluginRepository>
        <id>doodlerepo</id>
        <name>DoodleProject Maven 2 Repository</name>
        <url>http://doodlerepo.sourceforge.net/maven2/release/</url>
    </pluginRepository>
</pluginRepositories>

<distributionManagement>
    <repository>
        <id>repo</id>
    </repository>
</distributionManagement>
HTTP Download of Dependencies
1,596 Pull Requests
~40% Merged or Accepted
$4,000

Thanks to the GitHub Security Lab!
I got hooked on Bulk Pull Request Generation
I have a Problem
ADHD
HIGHWAY TO HEY LOOK A SQUIRREL!
I was finding too many security vulnerabilities!
I was finding too many security vulnerabilities!
I was finding too many security vulnerabilities!

I needed automation!
Automated Accurate Transformations at a Massive Scale
Abstract Syntax Tree (AST)

```java
/** myMethod */
void m()
{
    if (c > 2) {
        // c is more than 2
        x = c;
    }
    while (c < 10) { // increment x
        x += p();
        c++;
    }
}
```
Abstract Syntax Tree (AST)

```java
/** myMethod */
void m(){
    if (c > 2) {
        // c is more than 2
        x = c;
    }
    while(c < 10) { // increment x
        x += p();
        c++;
    }
}
```

```java
void m(){if(c>2){x=c;}while(c<10){x+=p();c++;}}
```
Format Preserving AST

Whitespace and comments are preserved
Generated code matches the Surrounding Formatting

**Spaces**

```java
String name = entry.getName();
Path path = dir.resolve(name);
if (!path.normalize().startsWith(dir)) {
    throw new RuntimeException("Bad zip entry");
}
OutputStream os = Files.newOutputStream(path);
```

**Tabs**

```java
Path path = dir.resolve(name);
if (!path.normalize().startsWith(dir)) {
    throw new RuntimeException("Bad zip entry");
}
OutputStream os = Files.newOutputStream(path);
```

**Braces on new line**

```java
String name = entry.getName();
Path path = dir.resolve(name);
if (!path.normalize().startsWith(dir)) {
    throw new RuntimeException("Bad zip entry");
}
OutputStream os = Files.newOutputStream(path);
```
Accurate Transformations Require Fully Type-attributed ASTs

\[
\text{log.info("...");}
\]

Is that log4j, slf4j, LogBack?
The OpenRewrite AST is both Syntactically and Semantically aware.

Syntax alone

With type attribution and formatting
Even simple code produces complex AST
if (!path.normalize().startsWith(dir)) {
    throw new RuntimeException("Bad zip entry");
}
final JavaTemplate noZipSlipPathStartsWithWithPathTemplate =
JavaTemplate.builder(this::getCursor, code: "if (!#{any(java.nio.file.Path)}::normalize() +
  .startsWith(#{any(java.nio.file.Path)})) {
throw new RuntimeException("Bad zip entry");
}").build();
```java
final JavaTemplate noZipSlipPathStartsWithWithPathTemplate =
    JavaTemplate.builder(this::getCursor, code: "" +
    "if (!#{any(java.nio.file.Path)}.normalize()" +
    "  .startsWith(#{any(java.nio.file.Path)}) ) {{\n" +
    "    throw new RuntimeException("Bad zip entry");\n" +
    "}" ).build();

return b.withTemplate(
    noZipSlipPathStartsWithWithPathTemplate,
    zipSlipSimpleInjectGuardInfo.statement.getCoordinates().after(),
    zipSlipSimpleInjectGuardInfo.zipEntry,
    zipSlipSimpleInjectGuardInfo.parentDir
);
```
public class MyZipHelper {
    public void m1(ZipEntry entry, Path dir) throws Exception {
        String name = entry.getName();
        Path path = dir.resolve(name);
        OutputStream os = Files.newOutputStream(path);
    }
}
What is possible now?
What other vulnerabilities can we fix?
Three Vulnerabilities

1. Temporary Directory Hijacking
2. Partial Path Traversal
3. Zip Slip
Vulnerability #1
Temporary Directory Hijacking
Temporary Directory on Unix-Like Systems is Shared between All Users
Temporary Directory Hijacking - Vulnerable

File f = File.createTempFile("prefix", "suffix");
f.delete();
f.mkdir();
ASK STACK OVERFLOW

GET VULNERABILITIES
Temporary Directory Hijacking - Vulnerable

```java
File f = File.createTempFile("prefix", "suffix");
f.delete();
f.mkdir();
```
Temporary Directory Hijacking - Vulnerable

```java
File f = File.createTempFile("prefix", "suffix");
f.delete();
// 🟩 Race condition
f.mkdir(); // Returns `false`
```
Temporary Directory Hijacking - Imperfect Fix

```java
File f = File.createTempFile("prefix", "suffix");
f.delete();
if (!f.mkdir())
    throw new IOException("Error");
```
Temporary Directory Hijacking - Fix

// Since Java 1.7
File f =
    Files
    .createTempDirectory("prefix")
    .toFile();
Temporary Directory Hijacking - CVEs

- CVE-2022-27772 - Spring Boot
- CVE-2021-20202 - Keycloak
- CVE-2021-21331 - DataDog API
- CVE-2020-27216 - Eclipse Jetty
- CVE-2020-17521 - Apache Groovy
- CVE-2020-17534 - Apache netbeans-html4j
Temporary Directory Hijacking

Pull Request Statistics
Temporary Directory Hijacking

64 Pull Requests!
Temporary Directory Hijacking - Pull Requests
Temporary Directory Hijacking - Putting it all together

```java
import javax.sound.midi.SysexMessage;
import java.io.File;
import java.io.IOException;
import java.nio.file.Files;

/**
 * FileUtil.copyURLToFile(
 *   getClass().getClassLoader().getResource("maven.zip"),
 *   zip);
 * File bin = File.createTempFile("maven","bin");
 * bin.delete();
 * bin.mkdirs();
 * File bin = Files.createTempDirectory("maven" + "bin").toFile();
 * Process unzip = new ProcessBuilder("unzip", zip.getAbsolutePath())
 *   .directory(bin).redirectErrorStream(true).start();
```
Temporary Directory Hijacking - Putting it all together

```java
private static File getTempFolder() throws IOException {
    if (tempFolder == null) {
        tempFolder = File.createTempFile("temp", Long.toString(System.nanoTime()));
        if (!tempFolder.delete()) {
            throw new IOException("Could not delete temp file: " + tempFolder.getAbsolutePath());
        }
        if (!tempFolder.mkdir()) {
            throw new IOException("Could not create temp directory: " + tempFolder.getAbsolutePath());
        }
        tempFolder = Files.createTempDirectory("temp" + Long.toString(System.nanoTime())).toFile();
        tempFolder.deleteOnExit();
    }
    return tempFolder;
```
Vulnerability #2

Partial Path Traversal
Partial Path Traversal

"/user/sam"
Partial Path Traversal

"/user/sam"

"/user/samantha"
Allows an attacker access to a sibling directory with the same prefix
Partial Path Traversal

"/user/sam"

Allows an attacker access to a sibling directory with the same prefix
Partial Path Traversal

"/user/sam"

Allows an attacker access to a sibling directory with the same prefix

"/user/samantha"
Partial Path Traversal

\[ /user/sam \]

Allows an attacker access to a sibling directory with the same prefix

\[ /user/samantha \]
Partial Path Traversal - Vulnerability

File `dir = new File`

```java
    parent, userControlled()
);;

if (!dir.getCanonicalPath()
    .startsWith(parent.getCanonicalPath())) {
    throw new IOException(
        "Detected path traversal attack!"
    );
}
```
new File("/user/sam/")
new File("/user/sam/")

File.getCanonicalPath()
new File("/user/sam/")

File.getCanonicalPath()

"/user/sam"
new File("/user/sam/")

File.getCanonicalPath()

"/user/sam"
Partial Path Traversal - Vulnerability

```java
File dir = new File(
    parent, userControlled()
);

if (!dir.getCanonicalPath().startsWith(parent.getCanonicalPath())) {
    throw new IOException("Detected path traversal attack!"
);
}
```
Partial Path Traversal - Vulnerability

File dir = new File(
    "/user/sam/", userControlled()
);

if (!dir.getCanonicalPath()
    .startsWith("/user/sam")) {
    ...
}
Partial Path Traversal - Vulnerability

File `dir = new File(`

```
"/user/sam/", 
"../samantha/baz"
```
)

```java
if (!dir.getCanonicalPath().startsWith("/user/sam")) {
    ...
}
```
Partial Path Traversal - Vulnerability

File  \texttt{dir} = \texttt{new File(} \\
\hspace{1cm} \texttt{"/user/sam/"}, \texttt{"../samantha/baz"}) \; \\
\texttt{)};

\textbf{if} (!\texttt{"/user/samantha/baz"} \\
\hspace{1cm} .\texttt{startsWith(} \texttt{"/user/sam"})) \{ \\
\hspace{2cm} \ldots \\
\}
File `dir = new File(`

```
    "/user/sam/", 
    ".../samantha/baz"
```

`);`

if (!`"/user/samantha/baz"

    .startsWith(`"/user/sam"`)) {

    throw new IOException(
        
        "Detected path traversal attack!"
        
    );

}
Partial Path Traversal Fix!
Partial Path Traversal - Vulnerability

```java
File dir = new File(
        parent, userControlled()
    );
if (!dir.getCanonicalPath().startsWith(parent.getCanonicalPath())) {
    throw new IOException(
        "Detected path traversal attack!"
    );
}
```
Partial Path Traversal - Vulnerability

```java
if (!dir.getCanonicalPath().startsWith(parent.getCanonicalPath())) {
    ...
}
```
Partial Path Traversal - Fix #1

```java
if (!dir.getCanonicalPath().startsWith(parent.getCanonicalPath() + File.separatorChar)) {
    ...
}
```
Partial Path Traversal - Fix #2

```java
if (!dir.getCanonicalFile() .toPath().startsWith(
    parent.getCanonicalFile().toPath())) {
    ...
}
```
if (!dir.getCanonicalFile().toPath().startsWith(
    parent.getCanonicalFile().toPath())) {
    ...
}
How do we find this vulnerability?
File dir = new File(
    parent, userControlled()
);
if (!dir.getCanonicalPath().startsWith(parent.getCanonicalPath())) {
    throw new IOException(
        "Detected path traversal attack!"
    );
}
if (!dir.getCanonicalPath()
    .startsWith(parent.getCanonicalPath())) {
    ...
}
Partial Path Traversal - Vulnerability

```java
if (!dir.getCanonicalPath().startsWith(parent.getCanonicalPath())) {
    ...
}
```
Partial Path Traversal - Safe

```java
if (!dir.getCanonicalPath().startsWith(parent.getCanonicalPath() + File.separatorChar)) {
    ...
}
```
It can’t be that easy, can it?
if (!dir.getCanonicalPath().startsWith(parent.getCanonicalPath())) {
    ...
}
String \textit{dirCanonical} = \textit{dir}.getCanonicalPath();

\textbf{if} (!\textit{dirCanonical}.startsWith(\textit{parent}.getCanonicalPath())) {
  ...
}

Partial Path Traversal - Vulnerability

```java
String dirCanonical = dir.getCanonicalPath();
String pCanonical = parent.getCanonicalPath();

if (!dirCanonical
    .startsWith(pCanonical)) {
    ...
}
```
String dirCanonical = dir.getCanonicalPath();
String pCanonical = parent.getCanonicalPath() + File.separatorChar;
if (!dirCanonical
    .startsWith(pCanonical)) {
    ...
}
We need Data Flow Analysis
String dirCanonical = dir.getCanonicalPath();
String pCanonical = parent.getCanonicalPath() + File.separatorChar;

if (!dirCanonical.startsWith(pCanonical)) {
    ...
}

Partial Path Traversal - Data Flow

String dirCanonical = dir.getCanonicalPath();
String pCanonical = parent.getCanonicalPath() + File.separatorChar;

if (!dirCanonical.startsWith(pCanonical)) {
    ...
}
Partial Path Traversal - Data Flow

```java
String dirCanonical = dir.getCanonicalPath();
String pCanonical = parent.getCanonicalPath() + File.separatorChar;

if (!dirCanonical.startsWith(pCanonical)) {
    ...
}
```
Partial Path Traversal - Data Flow

```java
String dirCanonical = dir.getCanonicalPath();
String pCanonical = parent.getCanonicalPath() + File.separatorChar;

if (!dirCanonical.startsWith(pCanonical)) {
    ...
}
```
Partial Path Traversal - Data Flow

String dirCanonical = dir.getAbsolutePath();

String pCanonical = parent.getAbsolutePath() + File.separatorChar;

String pCanonical2 = pCanonical;

if (!dirCanonical.startsWith(pCanonical2)) {
    ...
}


Data Flow

Uncovers hard to find Vulnerabilities
and prevents
False Positives
Data Flow Analysis

class GetCanonicalPathToStartsWithLocalFlow extends LocalFlowSpec<J.MethodInvocation, Expression> {

    @Override
    public boolean isSource(J.MethodInvocation methodInvocation, Cursor cursor) {
        return new MethodMatcher("java.io.File getCanonicalPath()")
            .matches(methodInvocation);
    }

    @Override
    public boolean isSink(Expression expression, Cursor cursor) {
        return InvocationMatcher
            .fromMethodMatcher(
                new MethodMatcher("java.lang.String startsWith(java.lang.String)"
            )
            .advanced()
            .isSelect(cursor);
    }
}

Partial Path Traversal - Putting it all together
Example Case: AWS Java SDK CVE-2022-31159
private boolean leavesRoot(File localBaseDirectory, String key) {
    try {
        return !new File(localBaseDirectory, key).getCanonicalPath().startsWith(localBaseDirectory.getCanonicalPath());
    } catch (IOException e) {
        throw new RuntimeException("Unable to canonicalize paths", e);
    }
}
if (leavesRoot(destinationDirectory, s.getKey())) {
    throw new RuntimeException("Cannot download key " + s.getKey() + ", its relative path resolves outside the parent directory.");
}
Vulnerability Disclosure Drama!
Aside: Email with AWS Security Team

AWS: We’d like to award you a bug bounty, however you’d need to sign an NDA.
Aside: Email with AWS Security Team

AWS: We’d like to award you a bug bounty, however you’d need to sign an NDA.

Jonathan: I don’t normally agree to NDA’s. Can I read it first before potentially agreeing?
Aside: Email with AWS Security Team

AWS: We’d like to award you a bug bounty, however you’d need to sign an NDA.

Jonathan: I don’t normally agree to NDA’s. Can I read it first before potentially agreeing?

AWS: We’re unable to share the bug bounty program NDA since it and other contract documents are considered sensitive by the legal team.
AMAZON WEB SERVICES
used LEGALESE!
AMAZON WEB SERVICES
used LEGALESE!

It hurt itself in its confusion!
Vulnerability #3

Zip Slip
Zip Slip

Path Traversal Vulnerability while Unpacking Zip File Entries
void zipSlip(File destination, ZipFile zip) {
    Enumeration<? extends ZipEntry> entries = zip.entries();
    while (entries.hasMoreElements()) {
        ZipEntry e = entries.nextElement();
        File f = new File(destination, e.getName());
        IOUtils.copy(
            zip.getInputStream(e),
            new FileOutputStream(f)
        );
    }
}
Zip Slip

```java
ZipEntry e = entries.nextElement();
File f = new File(destination, e.getName());
IOUtils.copy(
    zip.getInputStream(e),
    new FileOutputStream(f)
);
```
Zip Slip is Complicated
Zip Slip

```java
ZipEntry e = ...;
File f = new File(destination, e.getName());

IOUtils.copy(
    zip.getInputStream(e),
    new FileOutputStream(f)
);
```
Zip Slip

ZipEntry e = ...
File f = new File(destination, e.getName());
if (!f.toPath().startsWith(destination.toPath())) {
    throw new IOException("Bad Zip Entry!");
}

IOUtils.copy(
    zip.getInputStream(e),
    new FileOutputStream(f)
);
The Problem with Zip Slip
Zip Slip

ZipEntry e = ...
File f = new File(destination, e.getName());
if (!f.toPath().startsWith(destination.toPath())) {
    throw new IOException("Bad Zip Entry!");
}
IOUtils.copy(
    zip.getInputStream(e),
    new FileOutputStream(f)
);
Zip Slip

ZipEntry e = ...
File f = new File(destination, e.getName());
if (f.toPath().startsWith(destination.toPath())) {
    IOUtils.copy(
        zip.getInputStream(e),
        new FileOutputStream(f)
    );
}
Control Flow Analysis
File \( f \) = new File(\textit{destination}, \textit{e}.getName());
IOUtils.copy(
    zip.getInputStream(e),
    new FileOutputStream(f));

if (!f.toPath().startsWith(destination.toPath())){
    throw new IOException("Bad Zip Entry!");
}

IOUtils.copy(
    zip.getInputStream(e),
    new FileOutputStream(f));
abstract class Test {
    abstract int start();
    int test() {
        int x = start();
        x++;
        if (x > 5 || x < 3) {
            return 2;
        }
        return 5;
    }
}
Zip Entry e = ...

File f = new File(destination, e.getName());

if (!f.toPath().startsWith(destination.toPath())) {
    throw new IOException("Bad Zip Entry!");
}

IOUtils.copy(
    zip.getInputStream(e),
    new FileOutputStream(f)
);
```java
File f = new File(destination, e.getName());
if (!f.toPath().startsWith(destination.toPath())) {
    throw new IOException("Bad Zip Entry!");
}
IOUtils.copy(zip.getInputStream(e), new FileOutputStream(f));
```
### Zip Slip - Putting it all together

```java
while (entries.hasMoreElements()) {
    ZipEntry e = entries.nextElement();
    File f = new File(tmpZipDirectory.toFile(), e.getName());
    if (!f.toPath().normalize().startsWith(tmpZipDirectory.toFile().toPath())) {
        throw new RuntimeException("Bad zip entry");
    }
    InputStream is = zip.getInputStream(e);
    Files.copy(is, f.toPath(), StandardCopyOption.REPLACE_EXISTING);
}
```
Zip Slip - Putting it all together

```java
byte[] buffer = new byte[1024];

while ((entry = zis.getNextEntry()) != null) {
    File outputFile = new File(outputFolder.getCanonicalPath() + File.separatorChar + entry.getName());
    File outputParent = new File(outputFile.getParent());
    outputParent.mkdirs();
    if (!outputFile.toPath().normalize().startsWith(outputFolder.getCanonicalPath())) {
        throw new RuntimeException("Bad zip entry");
    }
}
```
Pull Request Generation!
GOT SECURITY VULNERABILITIES?

YOU GET A PULL REQUEST!
YOU GET A PULL REQUEST!
EVERYBODY GETS A PULL REQUEST!!!
Problems with Pull Request Generation
How fast can we generate Pull Requests?
Pull Request Generation Steps

File IO  Git Operation  GitHub API
Pull Request Generation Steps

1. Checkout (ie. Download) code Repository
Pull Request Generation Steps

1. Checkout (ie. Download) code Repository
2. Branch, Apply Diff, & Commit

File IO  Git Operation  GitHub API
Pull Request Generation Steps

1. **Checkout (ie. Download) code Repository**
2. **Branch, Apply Diff, & Commit**
3. **Fork Repository on GitHub**
Pull Request Generation Steps

1. Checkout (ie. Download) code Repository
2. Branch, Apply Diff, & Commit
3. Fork Repository on GitHub
4. Rename Repository on GitHub

File IO  Git Operation  GitHub API
Pull Request Generation Steps

1. Checkout (ie. Download) code Repository
2. Branch, Apply Diff, & Commit
3. Fork Repository on GitHub
4. Rename Repository on GitHub
5. Push changes

File IO  Git Operation  GitHub API
Pull Request Generation Steps

1. Checkout (ie. Download) code Repository
2. Branch, Apply Diff, & Commit
3. Fork Repository on GitHub
4. Rename Repository on GitHub
5. Push changes
6. Create Pull Request on GitHub

File IO    Git Operation    GitHub API
Pull Request Generation Steps

1. Checkout (ie. Download) code Repository
2. Branch, Apply Diff, & Commit
3. Fork Repository on GitHub
4. Rename Repository on GitHub
5. Push changes
6. Create Pull Request on GitHub

File IO  Git Operation  GitHub API
Pull Request Generation Steps

1. Checkout (ie. Download) code Repository

2. Branch, Apply Diff, & Commit

3. Fork Repository on GitHub

4. Rename Repository on GitHub

5. Push changes

6. Create Pull Request on GitHub

File IO  Git Operation  GitHub API
IF YOU COULD STOP RATE LIMITING YOUR API

THAT WOULD BE GREAT
We’ve made it this far

- Vulnerabilities Detected
- Style Detected
- Code Fixed & Diff Generated
- Rate Limit Bypassed
We’ve made it this far

- Vulnerabilities Detected
- Style Detected
- Code Fixed & Diff Generated
- Rate Limit Bypassed

How do we do this for all the repositories?
Moderne

- Free for Open Source Projects!
- ~7,000 Repositories indexed
- Run Open Rewrite Transformations at Scale
- Generates and Updates Pull Requests
800+ OpenRewrite Recipes including complete Framework Migrations
Bulk Pull Request Generation - public.moderne.io
<table>
<thead>
<tr>
<th>Status</th>
<th>Repository</th>
<th>Branch</th>
<th>Total results</th>
<th>Files searched</th>
<th>Actions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Finished</td>
<td>broadinstitute/picard</td>
<td>master</td>
<td>4</td>
<td>6,501</td>
<td>DIFF</td>
</tr>
<tr>
<td>Finished</td>
<td>jenkinsci/google-oauth-plugin</td>
<td>develop</td>
<td>3</td>
<td>218</td>
<td>DIFF</td>
</tr>
<tr>
<td>Finished</td>
<td>andyglick/jentails4java</td>
<td>master</td>
<td>3</td>
<td>278</td>
<td>DIFF</td>
</tr>
<tr>
<td>Finished</td>
<td>salesforce/imageOptimization</td>
<td>master</td>
<td>3</td>
<td>244</td>
<td>DIFF</td>
</tr>
<tr>
<td>Finished</td>
<td>sonatype/plexus-archiver</td>
<td>master</td>
<td>3</td>
<td>580</td>
<td>DIFF</td>
</tr>
<tr>
<td>Finished</td>
<td>zeroturnaround/zt-zip</td>
<td>master</td>
<td>3</td>
<td>230</td>
<td>DIFF</td>
</tr>
<tr>
<td>Finished</td>
<td>jenkinsci/acceptance-test-narines</td>
<td>master</td>
<td>3</td>
<td>786</td>
<td>DIFF</td>
</tr>
<tr>
<td>Finished</td>
<td>jenkinsci/jacoco-plugin</td>
<td>master</td>
<td>3</td>
<td>333</td>
<td>DIFF</td>
</tr>
<tr>
<td>Finished</td>
<td>smacke/jaydio</td>
<td>master</td>
<td>2</td>
<td>154</td>
<td>DIFF</td>
</tr>
<tr>
<td>Finished</td>
<td>jaltekruse/OpenNotebook</td>
<td>master</td>
<td>2</td>
<td>438</td>
<td>DIFF</td>
</tr>
<tr>
<td>Finished</td>
<td>nariivelib4java/lt/idJ</td>
<td>master</td>
<td>2</td>
<td>827</td>
<td>DIFF</td>
</tr>
<tr>
<td>Finished</td>
<td>jenkinsci/backend-jpi-create</td>
<td>master</td>
<td>2</td>
<td>134</td>
<td>DIFF</td>
</tr>
<tr>
<td>Finished</td>
<td>libgdx/libgdx</td>
<td>master</td>
<td>2</td>
<td>3,718</td>
<td>DIFF</td>
</tr>
<tr>
<td>Finished</td>
<td>koraktor/mavanagaiata</td>
<td>master</td>
<td>2</td>
<td>75</td>
<td>DIFF</td>
</tr>
<tr>
<td>Finished</td>
<td>Gray/libg2/JadConfig</td>
<td>master</td>
<td>2</td>
<td>357</td>
<td>DIFF</td>
</tr>
<tr>
<td>Finished</td>
<td>jbossas/jboss-vfs</td>
<td>master</td>
<td>2</td>
<td>279</td>
<td>DIFF</td>
</tr>
</tbody>
</table>
Commit title: Temporary Directory Hijacking or Information Disclosure

Commit messages:
This fixes either Temporary Directory Hijacking, or Temporary Directory Local Information Disclosure.

Reported-by: Jonathan Leitschuh Jonathan.Leitschuh@gmail.com Signed-off-by: Jonathan Leitschuh Jonathan.Leitschuh@gmail.com

Bug-tracker: https://github.com/JLLeitschuh/security-research/issues/10

<table>
<thead>
<tr>
<th>Status</th>
<th>Repository</th>
<th>Modified</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>COMPLETED</td>
<td>sanilytahir</td>
<td>about 6 hours ago</td>
<td>View commit</td>
</tr>
<tr>
<td>COMPLETED</td>
<td>broadinstitute/picard</td>
<td>about 6 hours ago</td>
<td>View commit</td>
</tr>
<tr>
<td>COMPLETED</td>
<td>AnukenArc</td>
<td>about 5 hours ago</td>
<td>View commit</td>
</tr>
<tr>
<td>COMPLETED</td>
<td>talmo-ict/umidioclet</td>
<td>about 5 hours ago</td>
<td>View commit</td>
</tr>
<tr>
<td>COMPLETED</td>
<td>jenkinsci/jenkins-test-harness</td>
<td>about 5 hours ago</td>
<td>View commit</td>
</tr>
<tr>
<td>COMPLETED</td>
<td>search/jasmine-maven-plugin</td>
<td>about 5 hours ago</td>
<td>View commit</td>
</tr>
<tr>
<td>COMPLETED</td>
<td>vert-x3/vertx-amp-bridge</td>
<td>about 5 hours ago</td>
<td>View commit</td>
</tr>
<tr>
<td>COMPLETED</td>
<td>reactor/reactor-netty</td>
<td>about 5 hours ago</td>
<td>View commit</td>
</tr>
<tr>
<td>COMPLETED</td>
<td>lbqds/lbqdx</td>
<td>about 5 hours ago</td>
<td>View commit</td>
</tr>
<tr>
<td>COMPLETED</td>
<td>Karlatemp/PublicationSign</td>
<td>about 5 hours ago</td>
<td>View commit</td>
</tr>
</tbody>
</table>
But there are more than just 7,000 repositories in the world.

How do we find the other vulnerable projects?
CodeQL
CodeQL

100k+ OSS Projects Indexed
35k+ OSS Java Projects
### jenkins-ingest / repos.csv

<table>
<thead>
<tr>
<th>Repository</th>
<th>Branch</th>
<th>Pr</th>
<th>Owner</th>
</tr>
</thead>
<tbody>
<tr>
<td>Offiz/gpr-for-gradle</td>
<td>master</td>
<td>8</td>
<td>gradle</td>
</tr>
<tr>
<td>Dopstajjpspdbjutil</td>
<td>master</td>
<td>8</td>
<td>maven</td>
</tr>
<tr>
<td>105932013072/avaporser</td>
<td>master</td>
<td>8</td>
<td>gradle</td>
</tr>
<tr>
<td>157409611jimpAop</td>
<td>master</td>
<td>8</td>
<td>gradle</td>
</tr>
<tr>
<td>183824883285/IBasePutter</td>
<td>master</td>
<td>8</td>
<td>maven</td>
</tr>
<tr>
<td>tianl1/cosmo</td>
<td>master</td>
<td>8</td>
<td>maven</td>
</tr>
<tr>
<td>tianl1/reactive</td>
<td>master</td>
<td>8</td>
<td>maven</td>
</tr>
<tr>
<td>tc-syntaxis/sols-dev-tools</td>
<td>develop</td>
<td>8</td>
<td>gradle</td>
</tr>
<tr>
<td>275893469/study</td>
<td>master</td>
<td>8</td>
<td>maven</td>
</tr>
<tr>
<td>Zidgulja/AndroidTagGroup</td>
<td>master</td>
<td>8</td>
<td>gradle</td>
</tr>
<tr>
<td>2purj CodeDesign-HomeWork1</td>
<td>master</td>
<td>8</td>
<td>maven</td>
</tr>
<tr>
<td>38live/oldid-gradle-plugin</td>
<td>master</td>
<td>8</td>
<td>gradle</td>
</tr>
<tr>
<td>Jasi/net-net-plugin</td>
<td>master</td>
<td>8</td>
<td>gradle</td>
</tr>
<tr>
<td>Jasi/gihversion-plugin</td>
<td>master</td>
<td>8</td>
<td>gradle</td>
</tr>
</tbody>
</table>

Visit: [https://github.com/moderneinc/jenkins-ingest](https://github.com/moderneinc/jenkins-ingest)
Finally!

Let’s generate some Open Source Software Pull Requests!
## Bulk Pull Request Generation Statistics

<table>
<thead>
<tr>
<th>Project</th>
<th>PR Generator</th>
<th>Pull Requests</th>
<th>Merge Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>HTTP Download of Dependencies</td>
<td>Python Bot</td>
<td>1,596</td>
<td>40%</td>
</tr>
<tr>
<td>CVE-2019-16303: JHipster RNG Vulnerability</td>
<td>Python Bot + Moderne</td>
<td>3,467</td>
<td>2.3%</td>
</tr>
<tr>
<td>CVE-2020-8597: rhostname array overflow</td>
<td>Python Bot</td>
<td>1,885</td>
<td>7.6%</td>
</tr>
<tr>
<td>Temporary Directory Hijacking</td>
<td>Moderne</td>
<td>64</td>
<td>TBD</td>
</tr>
<tr>
<td>Partial Path Traversal</td>
<td>Moderne</td>
<td>32</td>
<td>TBD</td>
</tr>
<tr>
<td>Zip Slip</td>
<td>Moderne</td>
<td>100</td>
<td>TBD</td>
</tr>
</tbody>
</table>
## Bulk Pull Request Generation Statistics

<table>
<thead>
<tr>
<th>Project</th>
<th>PR Generator</th>
<th>Pull Requests</th>
<th>Merge Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>HTTP Download of Dependencies</td>
<td>Python Bot</td>
<td>1,596</td>
<td>40%</td>
</tr>
<tr>
<td>CVE-2019-16303: JHipster RNG Vulnerability</td>
<td>Python Bot + Moderne</td>
<td>3,467</td>
<td>2.3%</td>
</tr>
<tr>
<td>CVE-2020-8597: rhostname array overflow</td>
<td>Python Bot</td>
<td>1,885</td>
<td>7.6%</td>
</tr>
<tr>
<td>Temporary Directory Hijacking</td>
<td>Moderne</td>
<td>64</td>
<td>TBD</td>
</tr>
<tr>
<td>Partial Path Traversal</td>
<td>Moderne</td>
<td>32</td>
<td>TBD</td>
</tr>
<tr>
<td>Zip Slip</td>
<td>Moderne</td>
<td>100</td>
<td>TBD</td>
</tr>
</tbody>
</table>

**New Pull Requests Generated in 2022: 590+**
## Bulk Pull Request Generation Statistics

<table>
<thead>
<tr>
<th>Project</th>
<th>PR Generator</th>
<th>Pull Requests</th>
<th>Merge Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>HTTP Download of Dependencies</td>
<td>Python Bot</td>
<td>1,596</td>
<td>40%</td>
</tr>
<tr>
<td>CVE-2019-16303: JHipster RNG Vulnerability</td>
<td>Python Bot + Moderne</td>
<td>3,467</td>
<td>2.3%</td>
</tr>
<tr>
<td>CVE-2020-8597: rhostname array overflow</td>
<td>Python Bot</td>
<td>1,885</td>
<td>7.6%</td>
</tr>
<tr>
<td>Temporary Directory Hijacking</td>
<td>Moderne</td>
<td>64</td>
<td>TBD</td>
</tr>
<tr>
<td>Partial Path Traversal</td>
<td>Moderne</td>
<td>32</td>
<td>TBD</td>
</tr>
<tr>
<td>Zip Slip</td>
<td>Moderne</td>
<td>100</td>
<td>TBD</td>
</tr>
</tbody>
</table>

**Personally Generated: 5,200+ Pull Requests**
[SECURITY] Fix Zip Slip Vulnerability
#149 opened 2 minutes ago by JLLeitschuh

[SECURITY] Fix Partial Path Traversal Vulnerability
#148 opened 6 hours ago by JLLeitschuh updated 6 hours ago

[SECURITY] Fix Temporary Directory Hijacking or Information Disclosure Vulnerability
#147 opened 2 days ago by JLLeitschuh

ProTip! Find all pull requests that aren’t related to any open issues with `-linked:issue`. 
Best Practices for Bulk Pull Request Generation
Messaging!
All Software Problems are People Problems In Disguise
Lesson 1

Sign off all Commits

--signoff
Sign off on Commits

Signed-off-by: Jonathan Leitschuh <Jonathan.Leitschuh@gmail.com>
Sign off on Commits

Why?!
Sign off on Commits

“It was introduced in the wake of the SCO lawsuit, (and other accusations of copyright infringement from SCO, most of which they never actually took to court), as a Developers Certificate of Origin. It is used to say that you certify that you have created the patch in question, or that you certify that to the best of your knowledge, it was created under an appropriate open-source license, or that it has been provided to you by someone else under those terms.”

- Stack Overflow
TL;DR
Lawyers
This push was rejected by Herald push rule H149.
Lesson 2

Be a good commitizen
Lesson 2

Be a good commitizen

GPG Sign your Commits
vuln-fix: Use HTTPS instead of HTTP to resolve deps CVE-2021-26291... Verified be60905
Instructions on masquerading as other users in git:

```bash
export GIT_AUTHOR_NAME="Linus Torvalds"
export GIT_AUTHOR_EMAIL="torvalds@linux-foundation.org"
export GIT_COMMITTER_NAME="$GIT_AUTHOR_NAME"
export GIT_COMMITTER_EMAIL="$GIT_AUTHOR_EMAIL"

git commit -m "Enjoy!"
```
Lesson 3
SECOM
Commit Format
vuln-fix: subject/header containing summary of changes in ~50 characters (Vuln-ID)

Detailed explanation of the subject/header in ~75 words.
(what) Explain the security issue(s) that this commit is patching.
(why) Focus on why this patch is important and its impact.
(how) Describe how the issue is patched.

[For Each Weakness in Weaknesses:]
Weakness: weakness identification or CWE-ID.
Severity: severity of the issue (Low, Medium, High, Critical).
CVSS: numerical representation (0-10) of the vulnerability severity.
Detection: method used to detect the issue (Tool, Manual, Exploit).
Report: http://link-to-report/
Introduced in: commit hash.

[End]

Reported-by: reporter name 1 <reporter-email@yourhost.com>
Reported-by: reporter name 2 <reporter-email@yourhost.com>
Signed-off-by: your name <your-email@yourhost.com>

[If you use an issue tracker, add reference to it here:]
[If external issue tracker:]
Bug-tracker: https://link-to-bug-tracker/id

[If github used as issue tracker:]
Resolves: #123
See also: #456, #789
Lesson 4

There are risks using your personal GitHub Account
Anyone here familiar with GitHub’s Angry Unicorn?
This was my GitHub Profile Page for most of 2020
Lesson 5
Coordinate with GitHub
Before Attempting

Reach out to GitHub!

SecurityLab@github.com
Lesson 5
Consider the Implications
Is this responsible disclosure?

#11 opened 4 hours ago  updated 35 minutes ago
Conclusion
As Security Researchers
We have an obligation to society
We know these vulnerabilities are out there
“For every 500 developers you have one security researcher.”

- GitHub 2020
"We can fix it. We have the technology. OK. We need to create the technology. Alright. The policy guys are mucking with the technology. Relax. WE'RE ON IT.

- Dan Kaminsky (1979 – 2021)
Sound Bytes

- Learn CodeQL! Seriously! It’s an incredibly powerful language!
- Contribute to OpenRewrite! Deploy your security fixes at scale!
- Join the GitHub Security Lab & OpenRewrite Slack Channels!
Thanks

Lidia Giuliano

Shyam Mehta