



Local Temp Directory Hijacking Vulnerability

High waynebeaton published GHSA-g3wg-6mcf-8jj6 on Oct 22, 2020

Package	
 org.eclipse.jetty:jetty-webapp (Maven)	
Affected versions	Patched versions
<= 9.4.32.v20200930, <= 10.0.0.beta2, <= 11.0.0.beta2	9.4.33.v20201020, 10.0.0.beta3, 11.0.0.beta3
 org.mortbay.jetty:jetty-webapp (Maven)	
All	None

Description

Impact

On Unix like systems, the system's temporary directory is shared between all users on that system. A collocated user can observe the process of creating a temporary sub directory in the shared temporary directory and race to complete the creation of the temporary subdirectory. If the attacker wins the race then they will have read and write permission to the subdirectory used to unpack web applications, including their WEB-INF/lib jar files and JSP files. If any code is ever executed out of this temporary directory, this can lead to a local privilege escalation vulnerability.

Additionally, any user code uses of `WebAppContext::getTempDirectory` would similarly be vulnerable.

Additionally, any user application code using the `ServletContext` attribute for the tempdir will also be impacted.
See: <https://javaee.github.io/javaee-spec/javadocs/javax/servlet/ServletContext.html#TEMPDIR>

For example:

```
import java.io.File;
import java.io.IOException;
import javax.servlet.ServletContext;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

public class ExampleServlet extends HttpServlet {
    @Override
    protected void doGet(HttpServletRequest req, HttpServletResponse resp) throws ServletException, IOException {
        File tempDir = (File)getContext().getAttribute(ServletContext.TEMPDIR); // Potentially compromised
        // do something with that temp dir
    }
}
```

Example: The JSP library itself will use the container temp directory for compiling the JSP source into Java classes before executing them.

CVSSv3.1 Evaluation

This vulnerability has been calculated to have a **CVSSv3.1 score of 7.8/10 (AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H)**

Patches

Fixes were applied to the 9.4.x branch with:

- [53e0e0e](#)
- [9ad6beb](#)

These will be included in releases: 9.4.33, 10.0.0.beta3, 11.0.0.beta3

Workarounds

A work around is to set a temporary directory, either for the server or the context, to a directory outside of the shared temporary file system.

For recent releases, a temporary directory can be created simple by creating a directory called `work` in the `$(jetty.base)` directory (the parent directory of the `webapps` directory). Alternately the java temporary directory can be set with the System Property `java.io.tmpdir`. A more detailed description of how jetty selects a temporary directory is below.

The Jetty search order for finding a temporary directory is as follows:

- If the `WebAppContext` has a `temp directory specified`, use it.
- If the `ServletContext` has the `javax.servlet.context.tempdir` attribute set, and if directory exists, use it.
- If a `$(jetty.base)/work` directory exists, use it (since Jetty 9.1)
- If a `ServletContext` has the `org.eclipse.jetty.webapp.basetempdir` attribute set, and if the directory exists, use it.
- Use `System.getProperty("java.io.tmpdir")` and use it.

Jetty will end traversal at the first successful step.

To mitigate this vulnerability the directory must be set to one that is not writable by an attacker. To avoid information leakage, the directory should also not be readable by an attacker.

Setting a Jetty server temporary directory.

Choices 3 and 5 apply to the server level, and will impact all deployed webapps on the server.

For choice 3 just create that work directory underneath your `$(jetty.base)` and restart Jetty.

For choice 5, just specify your own `java.io.tmpdir` when you start the JVM for Jetty.

```
[jetty-distribution]$ java -Djava.io.tmpdir=/var/web/work -jar start.jar
```

Setting a Context specific temporary directory.

The rest of the choices require you to configure the context for that deployed webapp (seen as `${jetty.base}/webapps/<context>.xml`)

Example (excluding the DTD which is version specific):

```
<Configure class="org.eclipse.jetty.webapp.WebAppContext">
  <Set name="contextPath"><Property name="foo"/></Set>
  <Set name="war">/var/web/webapps/foo.war</Set>
  <Set name="tempDirectory">/var/web/work/foo</Set>
</Configure>
```

References

- [#5451](#)
- [CWE-378: Creation of Temporary File With Insecure Permissions](#)
- [CWE-379: Creation of Temporary File in Directory with Insecure Permissions](#)
- [CodeQL Query PR To Detect Similar Vulnerabilities](#)

Similar Vulnerabilities

Similar, but not the same.

- JUnit 4 - [GHSA-269g-pwp5-87pp](#)
- Google Guava - [google/guava#4011](#)
- Apache Ant - <https://nvd.nist.gov/vuln/detail/CVE-2020-1945>
- JetBrains Kotlin Compiler - <https://nvd.nist.gov/vuln/detail/CVE-2020-15824>

For more information

The original report of this vulnerability is below:

On Thu, 15 Oct 2020 at 21:14, Jonathan Leitschuh jonathan.leitschuh@gmail.com wrote:
Hi WebTide Security Team,

I'm a security researcher writing some custom CodeQL queries to find Local Temporary Directory Hijacking Vulnerabilities. One of my queries flagged an issue in Jetty.

<https://lgtm.com/query/5615014766184643449/>

I've recently been looking into security vulnerabilities involving the temporary directory because on unix-like systems, the system temporary directory is shared between all users. There exists a race condition between the deletion of the temporary file and the creation of the directory.

```
// ensure file will always be unique by appending random digits
tmpDir = File.createTempFile(temp, ".dir", parent); // Attacker knows the full path of the file that will be generated
// delete the file that was created
tmpDir.delete(); // Attacker sees file is deleted and begins a race to create their own directory before Jetty.
// and make a directory of the same name
// SECURITY VULNERABILITY: Race Condition! - Attacker beats Jetty and now owns this directory
tmpDir.mkdirs();
```

[jetty.project/jetty-webapp/src/main/java/org/eclipse/jetty/webapp/WebInfConfiguration.java](#)
Lines 511 to 518 in 1b59672

```
511 {
512     //ensure file will always be unique by appending random digits
513     tmpDir = File.createTempFile(temp, ".dir", parent);
514     //delete the file that was created
515     tmpDir.delete();
516     //and make a directory of the same name
517     tmpDir.mkdirs();
518 }
```

In several cases the `parent` parameter will not be the system temporary directory. However, there is one case where it will be, as the last fallback.

[jetty.project/jetty-webapp/src/main/java/org/eclipse/jetty/webapp/WebInfConfiguration.java](#)
Lines 467 to 468 in 1b59672

```
467 //Make a temp directory in java.io.tmpdir
468 makeTempDirectory(new File(System.getProperty("java.io.tmpdir")), context);
```

If any code is ever executed out of this temporary directory, this can lead to a local privilege escalation vulnerability.

Would your team be willing to open a GitHub security advisory to continue the discussion and disclosure there? <https://github.com/eclipse/jetty.project/security/advisories>

This vulnerability disclosure follows Google's [90-day vulnerability disclosure policy](#) (I'm not an employee of Google, I just like their policy). Full disclosure will occur either at the end of the 90-day deadline or whenever a patch is made widely available, whichever occurs first.

Cheers,
Jonathan Leitschuh

Severity

High

CVE ID

CVE-2020-27216

Weaknesses

No CWEs

Credits



JLeitschuh