


Improper Limitation of a Pathname to a Restricted Directory ('Path Traversal') in micronaut-core

High jameskleeh published GHSA-cjx7-399x-p2rj on Jul 16, 2021

Package

 io.micronaut:micronaut-http-server-netty (Maven)

Affected versions

< 2.5.9

Patched versions

2.5.9

Description

with a basic configuration like

```
router:
  static-resources:
    assets:
      enabled: true
      mapping: /.assets/public/**
      paths: file:/home/lstrmiska/test/
```

it is possible to access any file from a filesystem, using "/../.." in URL, as Micronaut does not restrict file access to configured paths.

Repro Steps

- create a file test.txt in /home/lstrmiska
- start micronaut
- execute command
curl -v --path-as-is "http://localhost:8080/.assets/public/../../test.txt"

Impact

Micronaut can potentially leak sensitive information.

See <https://cwe.mitre.org/data/definitions/22.html>

Patches

```
diff --git a/core/src/main/java/io/micronaut/core/io/file/DefaultFileSystemResourceLoader.java
b/core/src/main/java/io/micronaut/core/io/file/DefaultFileSystemResourceLoader.java
index 2f5a91403..19d3b7f05 100644
--- a/core/src/main/java/io/micronaut/core/io/file/DefaultFileSystemResourceLoader.java
+++ b/core/src/main/java/io/micronaut/core/io/file/DefaultFileSystemResourceLoader.java
@@ -69,6 +69,9 @@ public class DefaultFileSystemResourceLoader implements FileSystemResourceLoader
    @Override
    public Optional<InputStream> getResourceAsStream(String path) {
        Path filePath = getFilePath(normalize(path));
+       if (pathOutsideBase(filePath)) {
+           return Optional.empty();
+       }
        try {
            return Optional.of(Files.newInputStream(filePath));
        } catch (IOException e) {
@@ -79,7 +82,7 @@ public class DefaultFileSystemResourceLoader implements FileSystemResourceLoader
    @Override
    public Optional<URL> getResource(String path) {
        Path filePath = getFilePath(normalize(path));
-       if (Files.exists(filePath) && Files.isReadable(filePath) && !Files.isDirectory(filePath)) {
+       if (!pathOutsideBase(filePath) && Files.exists(filePath) && Files.isReadable(filePath) && !Files.isDirectory(filePath)) {
            try {
                URL url = filePath.toUri().toURL();
                return Optional.of(url);
@@ -117,4 +120,15 @@ public class DefaultFileSystemResourceLoader implements FileSystemResourceLoader
    private Path getFilePath(String path) {
        return baseDirPath.map(dir -> dir.resolve(path)).orElseGet(() -> Paths.get(path));
    }
+
+    private boolean pathOutsideBase(Path path) {
+        if (baseDirPath.isPresent()) {
+            Path baseDir = baseDirPath.get();
+            if (path.isAbsolute() == baseDir.isAbsolute()) {
+                Path relativePath = baseDir.relativeTo(path);
+                return relativePath.startsWith("../");
+            }
+        }
+        return false;
+    }
}
```

Workarounds

- do not use ** in mapping, use only * which exposes only flat structure of a directory not allowing traversal
- run micronaut in chroot (linux only)

References

See <https://cwe.mitre.org/data/definitions/22.html>

For more information

If you have any questions or comments about this advisory:

- Open an issue in [Github](#)
- Email us at info@micronaut.io

Severity

High 7.5 / 10

CVSS base metrics	
Attack vector	Network
Attack complexity	Low
Privileges required	None
User interaction	None
Scope	Unchanged
Confidentiality	High
Integrity	None
Availability	None

CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:N/A:N

CVE ID

CVE-2021-32769

Weaknesses

CWE-22

Credits

 strmik