

spring-boot-actuator-logview 文件包含(CVE-2021-21234)漏洞分析

t4mo / 2021-07-28 23:50:00 / 浏览数 2684

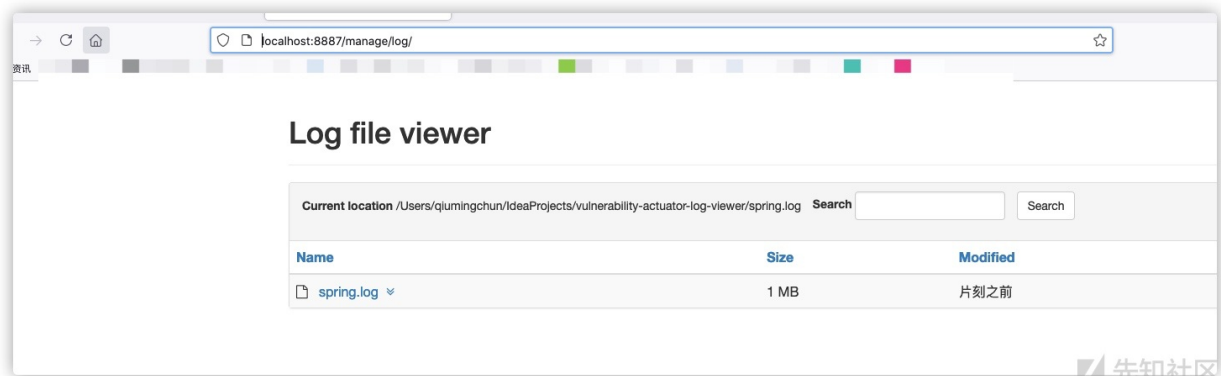
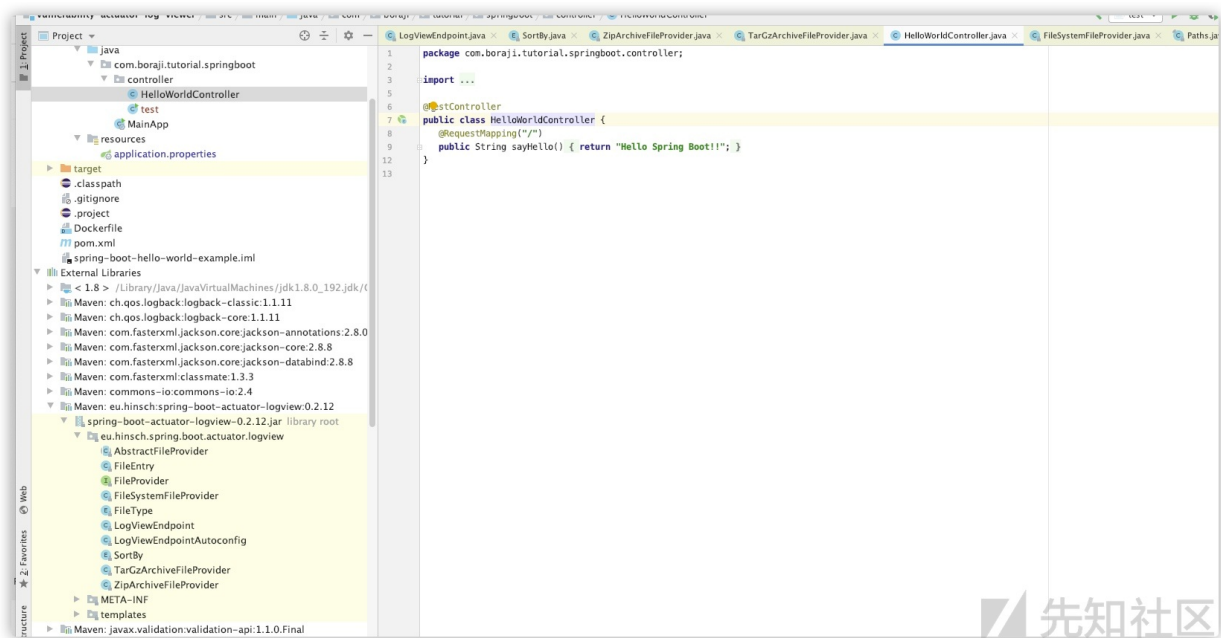
前言

日常搬砖过程中在github发现的一个CVE，<https://github.com/ARPSyndicate/kenzer-templates/blob/1f1dd550ddbde72cbe378452973b93b3e62003f5/jaeles/cvescan/medium/CVE-2021-21234.yam> 看着带了springboot就分析了下

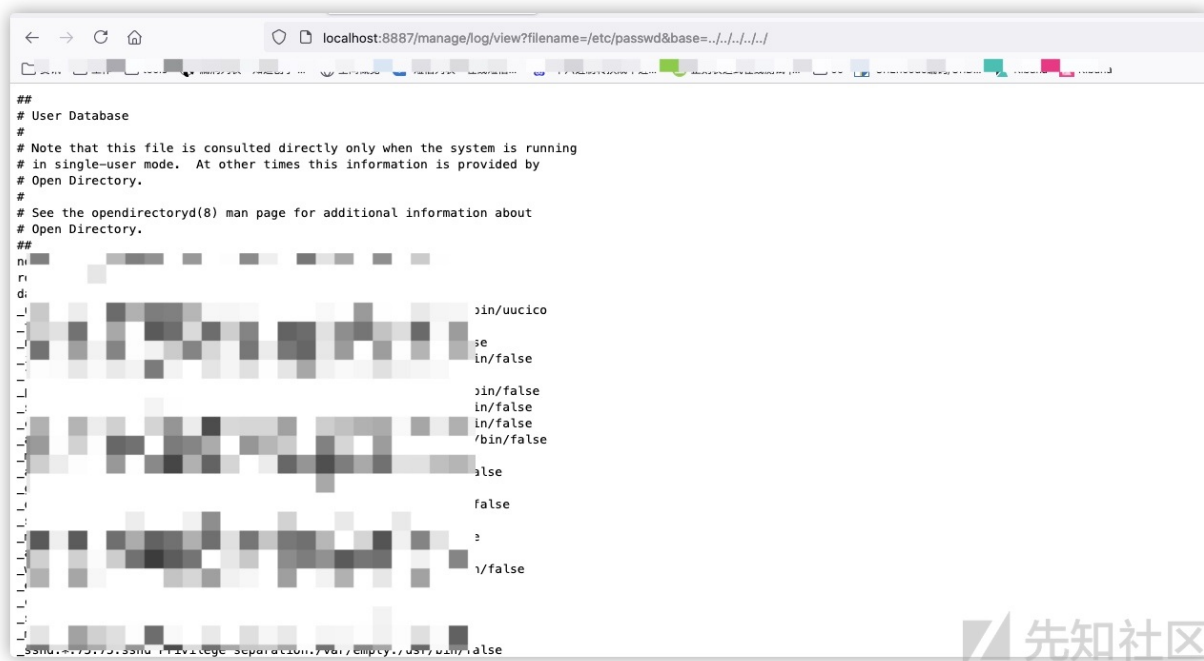
环境搭建

git clone <https://github.com/cristianeph/vulnerability-actuator-log-viewer>

启动之后访问 <http://localhost:8887/manage/log/>



漏洞复现分析



根据springboot启动日志发现/log/view 对应的方法为eu.hinsch.spring.boot.actuator.logview.LogViewEndpoint.view

```
FO 56749 --- [main] o.s.b.a.e.mvc.EndpointHandlerMapping : Mapped "{[/manage/health || /manage/health.json],methods=[GET],produces=[application/vnd.spring-boot.actuator.v1+json || application
UG 56749 --- [main] o.s.b.a.e.mvc.EndpointHandlerMapping : 4 request handler methods found on class eu.hinsch.spring.boot.actuator.logview.LogViewEndpoint: {public void eu.hinsch.spring.boot
FO 56749 --- [main] o.s.b.a.e.mvc.EndpointHandlerMapping : Mapped "{[/manage/log/view]}" onto public void eu.hinsch.spring.boot.actuator.logview.LogViewEndpoint.view(java.lang.String,java.lan
FO 56749 --- [main] o.s.b.a.e.mvc.EndpointHandlerMapping : Mapped "{[/manage/log/!]}" onto public java.lang.String eu.hinsch.spring.boot.actuator.logview.LogViewEndpoint.list(org.springframework
FO 56749 --- [main] o.s.b.a.e.mvc.EndpointHandlerMapping : Mapped "{[/manage/log/search]}" onto public void eu.hinsch.spring.boot.actuator.logview.LogViewEndpoint.search(java.lang.String,java
UG 56749 --- [main] o.s.b.a.e.mvc.EndpointHandlerMapping : Mapped "{[/manage/log || /manage/log.json]}" onto public void eu.hinsch.spring.boot.actuator.logview.LogViewEndpoint.redirect(java.l
UG 56749 --- [main] o.s.b.a.e.mvc.EndpointHandlerMapping : 1 request handler methods found on class org.springframework.boot.actuate.endpoint.mvc.EndpointMvcAdapter: {public java.lang.Obj
```

对应代码

```
@RequestMapping("/view")
public void view(@RequestParam String filename,
                @RequestParam(required = false) String base,
                @RequestParam(required = false) Integer tailLines,
                HttpServletResponse response) throws IOException {
    securityCheck(filename);
    response.setContentType(MediaType.TEXT_PLAIN_VALUE);

    Path path = loggingPath(base);
    FileProvider fileProvider = getFileProvider(path);
    if (tailLines != null) {
        fileProvider.tailContent(path, filename, response.getOutputStream(), tailLines);
    }
    else {
        fileProvider.streamContent(path, filename, response.getOutputStream());
    }
}
```

先从RequestParam获取filename参数，然后调用securityCheck进行检查，判断filename是否包含..

```
}
}

private void securityCheck(String filename) {
    Assert.doesNotContain(filename, substring: "..");
}

@Override
```

```

public static void doesNotContain(String textToSearch, String substring, String message) { textToSearch: "../etc
    if (StringUtils.hasLength(textToSearch) && StringUtils.hasLength(substring) &&
        textToSearch.contains(substring)) { textToSearch: "../etc/passwd" substring: "../"
        throw new IllegalArgumentException(message);
    }
}

```

先知社区

安全检查通过之后，将application.properties中logging.path和base拼接，返回path，base从RequestParam获取，并未经过securityCheck

```

return parent;
}

private Path loggingPath(String base) { base: ".././././././"
    return base != null ? Paths.get(loggingPath, base) : Paths.get(loggingPath); base: ".././././././" loggingPath: "spring.log"
}

```

先知社区

然后生成fileProvider 在调用 streamContent

```

securityCheck(filename); filename: "/etc/passwd"
response.setContentType(MediaType.TEXT_PLAIN_VALUE); response: ResponseFacade@5662

Path path = loggingPath(base); path: "spring.log/./././././" base: ".././././././"
FileProvider fileProvider = getFileProvider(path); path: "spring.log/./././././"
if (tailLines != null) {
    fileProvider.tailContent(path, filename, response.getOutputStream(), tailLines);
} else {
    fileProvider.streamContent(path, filename, response.getOutputStream());
}
}

```

先知社区

将path和base拼接，然后用fileinputstream打开，造成任意文件读取

```

@Override
public void streamContent(Path folder, String filename, OutputStream stream) throws IOException { folder: "spring.log/./././././" filename: "/etc/passwd" stream: Coy
    IOUtils.copy(new FileInputStream(getFile(folder, filename)), stream); folder: "spring.log/./././././" filename: "/etc/passwd" stream: CoyoteOutputStream@6687
}

private File getFile(Path folder, String filename) {
}

```

先知社区

```

Debugger Console Endpoints
Frames
"http-nio-8887-..."main": RUNNING
toFile:96, AbstractPath (sun.nio.fs)
getFile:73, FileSystemFileProvider (eu.hinsch.spring.b
Variables
this = {UnixPath@6688} "spring.log/./././././etc/passwd"
Variables debug info not available

```

先知社区

关注 | 1

点击收藏 | 2

上一篇： 记一次实战渗透(上)

下一篇： Java内存马：一种Tomcat全...

2 条回复

bluedo****



2021-07-29 10:18:48

第一个链接挂了

👍 0 回复Ta



b1gd0d

2021-07-29 15:59:09

太强了

👍 0 回复Ta

登录 后跟帖