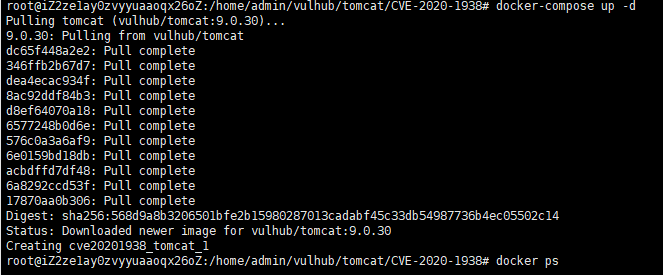
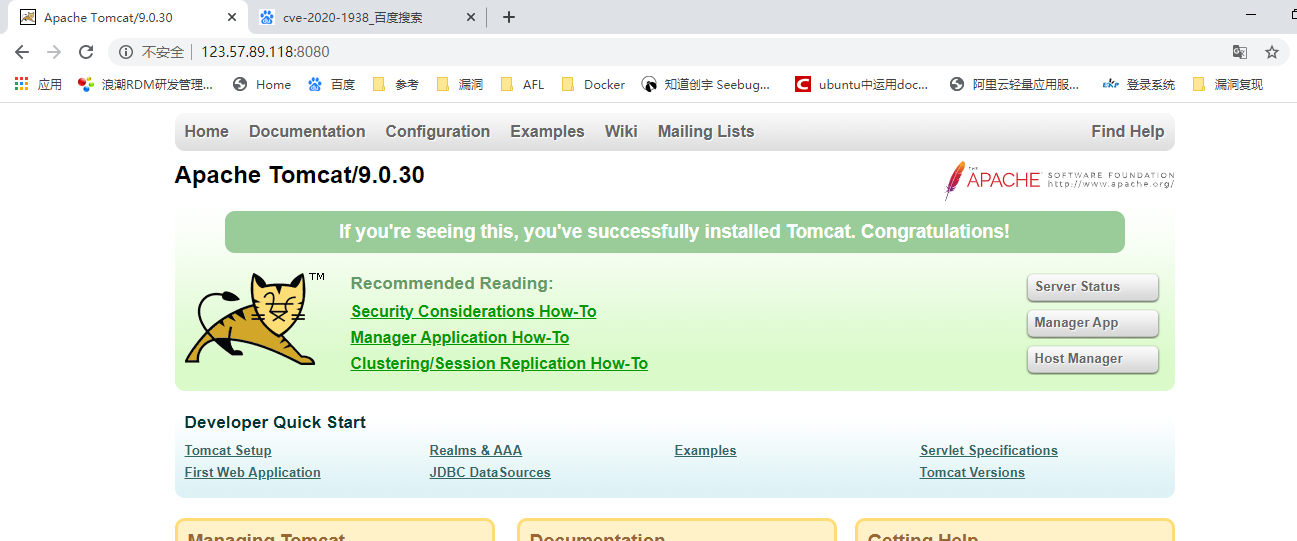
## 复现

使用docker-compose up -d命令 在CVE-2020-1938目录下自动搭建漏洞复现环境：



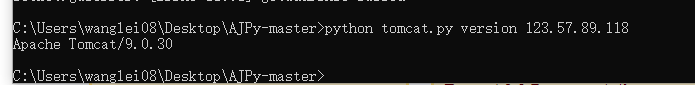
访问web页面,环境搭建成功：



这里使用<https://github.com/hypn0s/AJPy>的漏洞利用工具：



获取一下tomcat的指纹：



查看文件：（poc执行环境为python 2.x）



## 分析

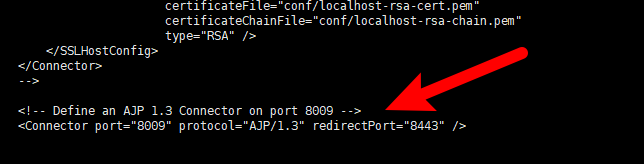
前置协议AJP: AJP（Apache JServ Protocol）是定向包协议。它的功能其实和HTTP协议相似，区别在于AJP协议使用的是二进制格式传输文本，走的是TCP协议来SERVLET容器进行通信，因此漏洞的利用就需要依赖于一个客户端，而不能依赖于浏览器或是HTTP的抓包工具。

Tomcat Connector 是 Tomcat 与外界连接的通道,使得它与 Catalina 能够接收外部的请求,传递给对应的 Web 应用程序处理，并返回对应的请求结果。

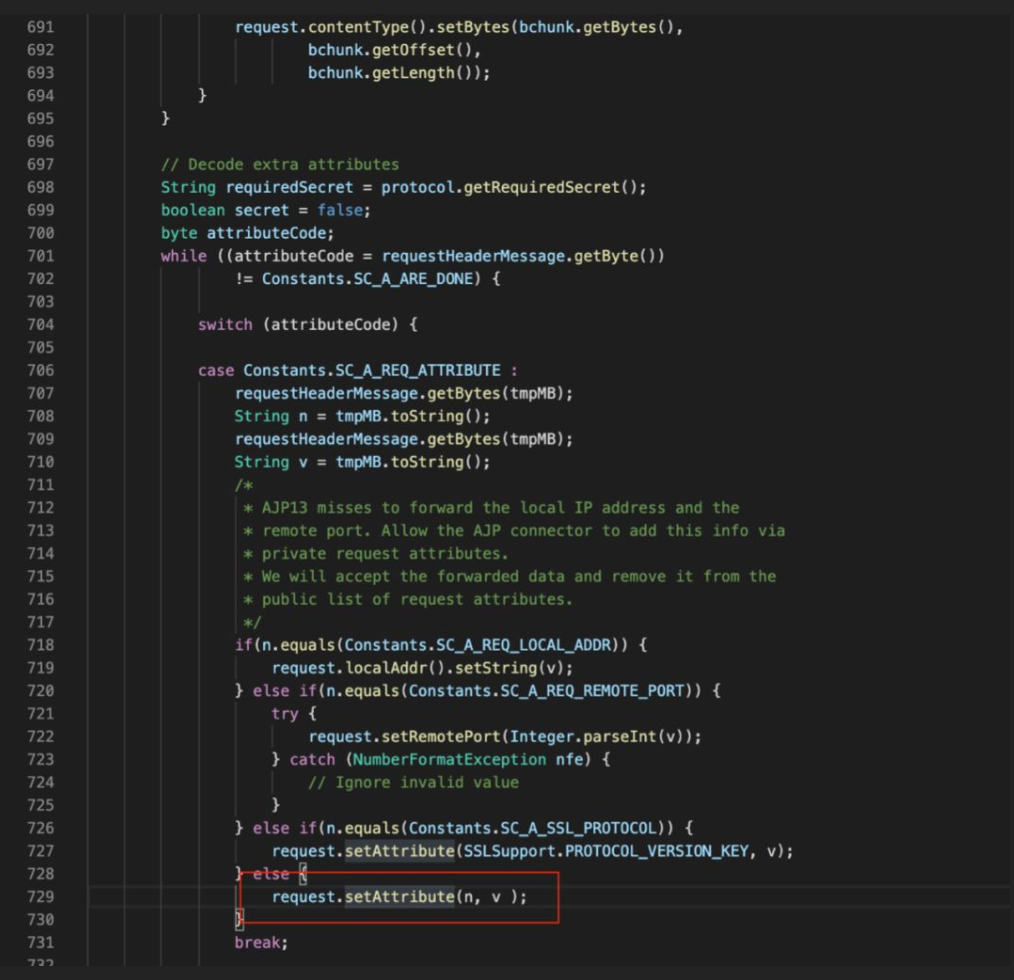
Tomcat默认的 conf/server.xml 中配置了2个 Connector，一个为8080的对外提供的HTTP协议(1.1版本)端口，默认监听地址: 0.0.0.0:8080，另外一个就是默认的8009 AJP协议(1.3版本)端口，默认监听地址为:0.0.0.0:8009，两个端口默认均监听在外网ip。

HTTP Connector就是用来提供经常用到的HTTP Web服务,而AJP Connector使用的AJP协议(Apache JservProtocol)，AJP 可以理解为HTTP协议的二进制性能优化版本，能降低HTTP请求的处理成本，所以主要在需要集群、反向代理的高并发需要高处理能力的场景中使用。

打开配置文件可以看到8009端口：



Tomcat在接收AJP请求的时候调用org.apache.coyote.ajp.AjpProcessor来处理AJP消息，prepareRequest将AJP里面的内容取出来设置成request对象的Attribute属性。



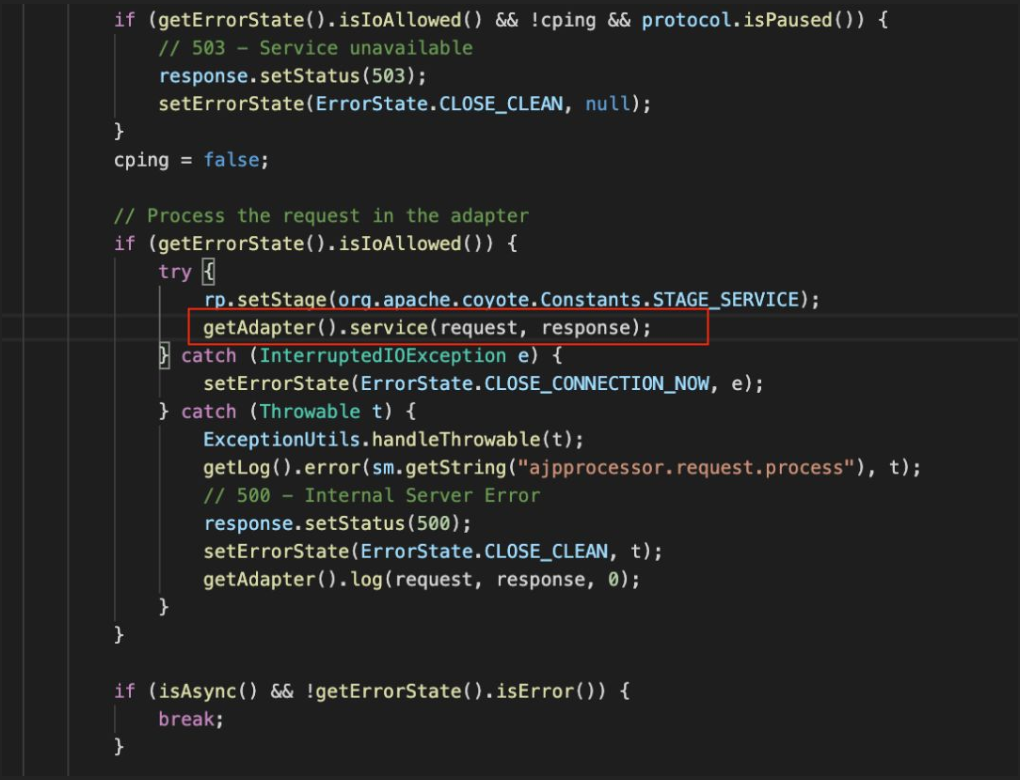
因此通过此种特性可以控制request对象的下面三个Attribute属性：

javax.servlet.include.request\_uri

javax.servlet.include.path\_info

javax.servlet.include.servlet\_path

然后封装成对应的request之后，继续走servlet的映射流程，如下图所示:

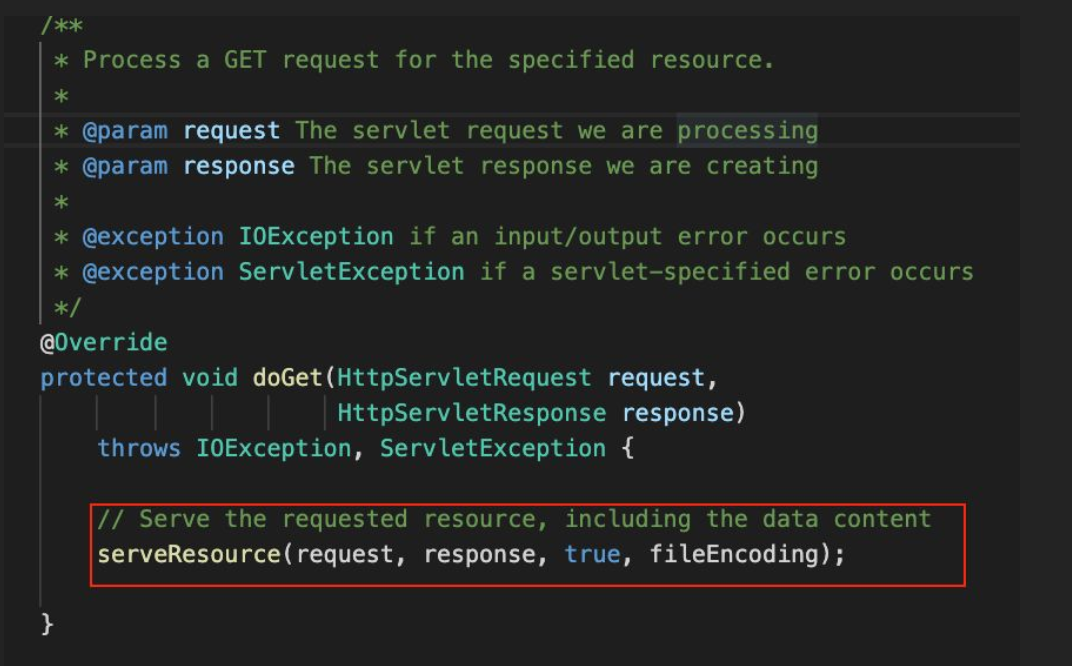


## 利用方式

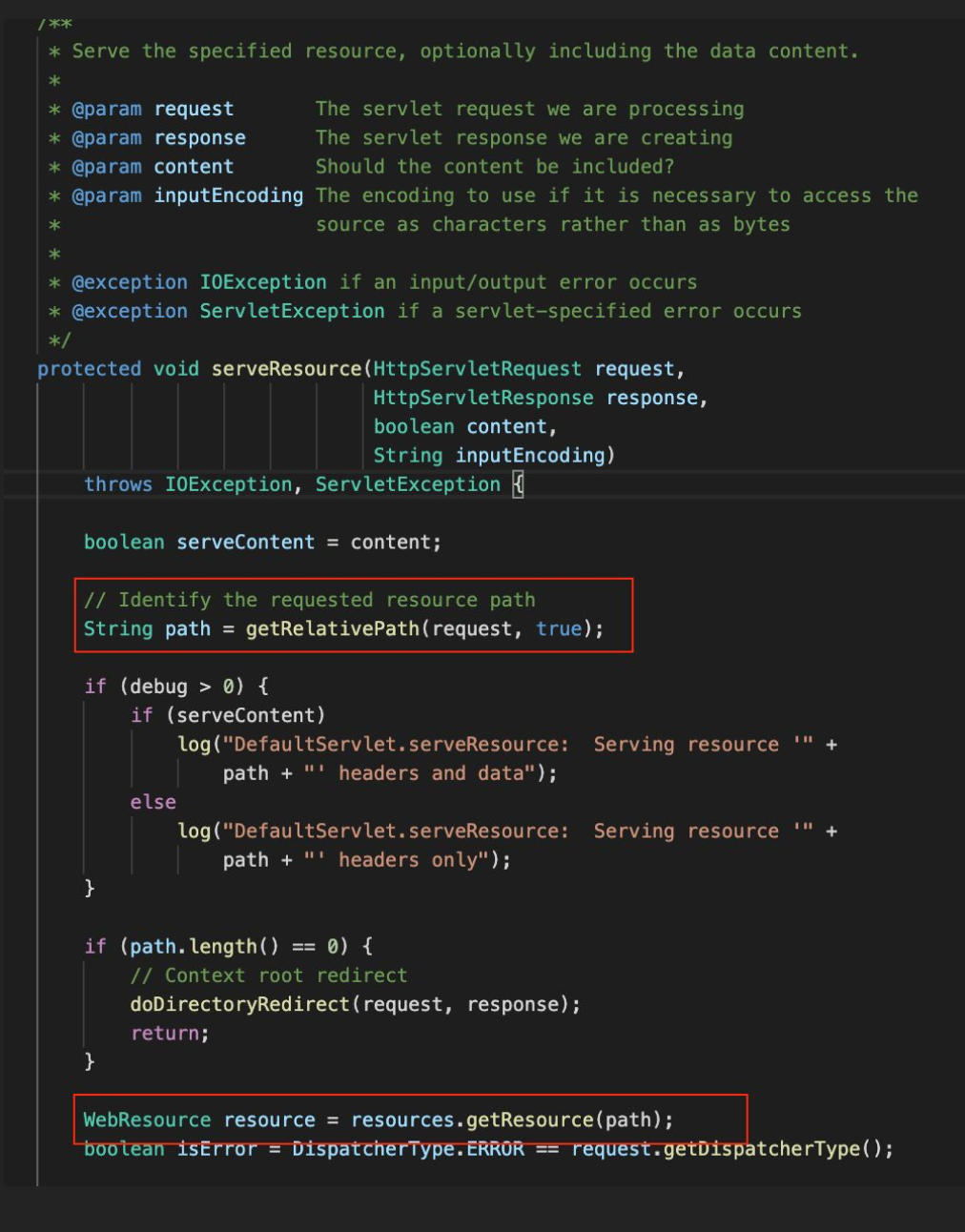
现阶段存在两种利用方式：

1. 利用 DefaultServlet 实现任意文件下载

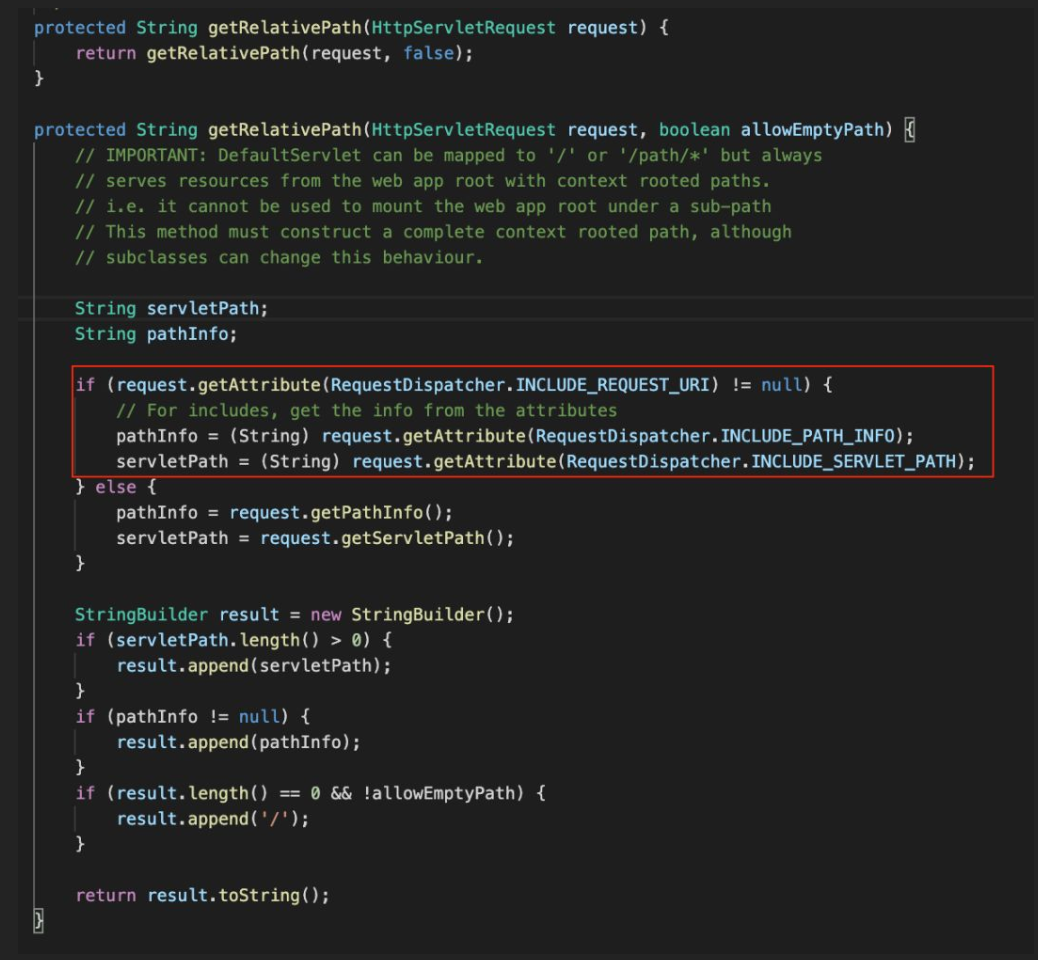
当url请求未在映射的url列表里面，则会通过Tomcat默认的DefaultServlet根据上面的三个属性来读取文件，如图：



通过serveResource方法来获取资源文件：



通过getRelativePath来获取资源文件路径：



然后通过控制AJP控制的上述三个属性来读取文件，通过操控上述三个属性可以读取到/WEB-INF下面的所有敏感文件，如网站文件、class、xml、jar等文件。

2. 通过 jspservlet 实现任意后缀文件包含

当url（如http://xxxx/xx.jsp）请求映射在org.apache.jasper.servlet.JspServlet这个servlet的时候，也可通过上述三个属性来控制访问的jsp文件，如下图：



控制路径之后就可以用jsp解析该文件，只需要一个可控文件内容的文件即可实现RCE。

## 漏洞检测

现阶段该漏洞的检测方式主要为检测Tomcat版本：

通常在Apache Tomcat官网下载的安装包名称中会包含有当前Tomcat的版本号。

Windows的Tomcat版本进入Tomcat安装目录的bin目录，输入命令version.bat后，可查看当前的软件版本号。

Linux可以进入Tomcat的bin目录下（cd /tomcat目录/bin），在执行./version.sh

## 漏洞防护

目前官方已在最新版本中修复了该漏洞，用户可通过版本升级进行防护。

7.\*分支7.0.100之前版本，建议更新到7.0.100版本

8.\*分支8.5.51之前版本，建议更新到8.5.51版本

9.\*分支9.0.31之前版本，建议更新到9.0.31版本

2. 其他安全防护措施

如果相关用户暂时无法进行版本升级，可根据实际情况采用下列防护措施：

关闭AJP

如若用不到Tomcat AJP协议，可直接关闭AJP Connector，或将其监听地址改为仅监听本机localhost。

具体操作如下

a.编辑 <CATALINABASE>/conf/server.xml，找到如下行（<CATALINABASE> 为 Tomcat 的工作目录）：<Connectorport=”8009″protocol=”AJP/1.3″ redirectPort=”8443″/>

b.然后将其注释掉或者删掉, (以下为注释的方式):<!–<Connectorport=”8009″protocol=”AJP/1.3″redirectPort=”8443″ />–>

c.保存后重新启动Tomcat，即可生效.

添加认证凭证

如若需要使用AJP协议，可根据当前版本配置协议属性设置认证凭证，主要使用配置AJP配置中的secretRequired跟secret属性来限制认证。

具体操作如下

使用Tomcat 7和Tomcat 9的用户可为AJP Connector配置secret来设置AJP协议的认证凭证。例如（注意必须将YOURTOMCATAJP\_SECRET更改为一个安全性高、无法被轻易猜解的值）：<Connectorport=”8009″protocol=”AJP/1.3″redirectPort=”8443″address=”YOUR\_TOMCAT\_IP\_ADDRESS”secret=”YOUR\_TOMCAT\_AJP\_SECRET”/>

使用Tomcat 8的用户可为AJP Connector配置requiredSecret来设置AJP协议的认证凭证。例如（注意必须将YOURTOMCATAJP\_SECRET更改为一个安全性高、无法被轻易猜解的值）：<Connectorport=”8009″protocol=”AJP/1.3″redirectPort=”8443″address=”YOUR\_TOMCAT\_IP\_ADDRESS”requiredSecret=”YOUR\_TOMCAT\_AJP\_SECRET”/>