Tomcat 信息泄露漏洞复现和分析—【CVE-2017-12616】

xxlegend / 2017-09-26 13:13:00 / 浏览数 5041 技术文章 技术文章 顶(0) 踩(0)

官方报告：

Important: Information Disclosure CVE-2017-12616
When using a VirtualDirContext it was possible to bypass security constraints and/or view the source code of JSPs for resources served by the VirtualDirContext using a specially crafted request.
This was fixed in revision 1804729.
This issue was identified by the Tomcat Security Team on 10 August 2017 and made public on 19 September 2017.
Affects: 7.0.0 to 7.0.80

复现步骤

配置server.xml 如下：

```
<Host name="localhost" appBase="webapps"
                unpackWARs="true" autoDeploy="true">
              <Context path="/test" docBase="D:\code\tomcatsrc-master\tomcatsrc-master\lunch\webapps\examples" debug="1" relo
                  <Resources className="org.apache.naming.resources.VirtualDirContext" extraResourcePaths="/=D:\code\tomcatsr
                  <Loader className="org.apache.catalina.loader.VirtualWebappLoader" virtualClassPath="/tmp/class" />
                  <JarScanner scanAllDirectories="true" />
              </Context>
```

这个配置文件在VirtualDirContext的源码中可以看到，从tomcat wiki看到这个配置`extraResourcePaths=" /WEB-INF/ /=D:\code\tomcatsrc-master\tomcatsrc-master\leaked"`是以前的写法，到了tomcat7配置得更新为`extraResourcePaths="/=D:\code\tomcatsrc-mas`
在Tomcat中，Host，Context都是容器类，一个Host可以包含多个Context，可以理解为一个Context是一个应用，对应着一个web.xml。而Context则包含多个Wrapper

调试分析

Tomcat对于每一个请求，首先是交给connector处理，处理的过程中就会通过mapper寻找对应的资源，也就是通过org.apache.naming.resources下面的类来完成，从ser

继续跟进去
doLookup是调用其父类FileDirContext的doLookup方法，具体如下：

```
protected Object doLookup(String name) {

   Object retSuper = super.doLookup(name);
   if (retSuper != null || mappedResourcePaths == null) {
       return retSuper;
   }
```

doLookup是调用其父类FileDirContext的doLookup方法，具体如下：

```
protected Object doLookup(String name) {
   Object result = null;
   File file = file(name);

   if (file == null)
       return null;

   if (file.isDirectory()) {
       FileDirContext tempContext = new FileDirContext(env);
       tempContext.setDocBase(file.getPath());
       tempContext.setAllowLinking(getAllowLinking());
       result = tempContext;
   } else {
       result = new FileResource(file);
   }

   return result;

}
```

继续跟进file方法，在这里tomcat实现了一些校验，会判断文件是否存在和可读标记。这样就完成了一个资源映射供后续使用，从调试信息中也可以看到，请求的是0.jsp.，
File()接口调用完之后却是存在的，这就是漏洞点。

这个映射关系会存到cache中，供后续请求使用。tomcat本身的逻辑也是，第一次请求先存cache，然后再传递给Servlet处理。存完cache之后会从cache中去取文件。
这块的逻辑可以从ProxyDirContext中cachedLoad体现出来，具体如下：

继续跟进FileDirContext.streamContent()，实现如下：从图中我们可以看到已经获取到0.jsp的内容

把这些cache全部存好之后，tomcat会将请求转发DefaultServlet处理，大致流程就是Http11接收，传递给Enginer->Host->Context->Wrapper->Servlet，一种典型的责

在这里会涉及到为啥0.jsp/在这里不能直接获取jsp的内容，跟进DefaultServlet.java中doGet()方法--》serveResource，在serveResource下有对后缀/和\的控制，这就明白

总结

这个漏洞的本质原因就是File接口和FileInputStream接口可以接收test.jsp.而错误认为是test.jsp，这应该是算JAVA API的缺陷。

参考文献：
http://mp.weixin.qq.com/s/sulJSg0Ru138oASiI5cYAA

上一篇：Tomcat漏洞分析—【CVE-2... 下一篇：浅谈Java反序列化漏洞修复方案
1. 0 条回复
   - 动动手指，沙发就是你的了！