基于java 字节码的灰盒动态漏洞检测

杭州安恒-吴卓群

OWASP 中国

The Open Web Application Security Project

OWASP 中国
The Open Web Application Security Project

About Me

- 目前就职于杭州安恒信息技术有限公司，任信息安全服务部副总监、研究院安全分院负责人、高级安全研究员。
- 从事多年的web应用安全领域研究。擅长漏洞发掘、代码审计、安全测试。

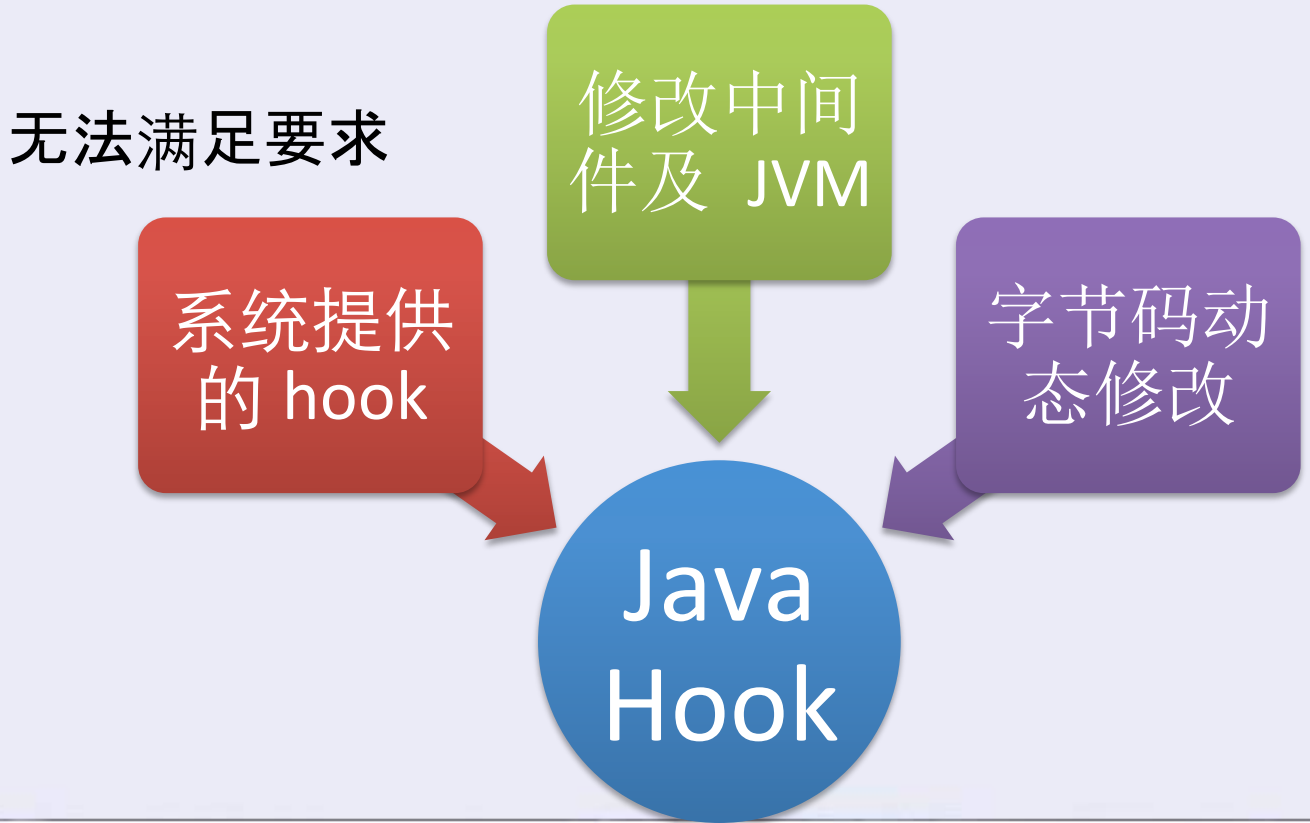
DBAPPSecurity

- 目前常用的WEB应用自动化测试程序
  - 白盒测试(源码审计系统)
    - 误报率太高
    - 逻辑顺序关联的问题无法测试
  - 黑盒测试(web扫描器)
    - 很多漏洞无法检测,如
      - 存储跨站
      - 页面无变化的注入（ update, insert 等）
      - 大部分的代码注入
      - 很多文件操作相关的漏洞

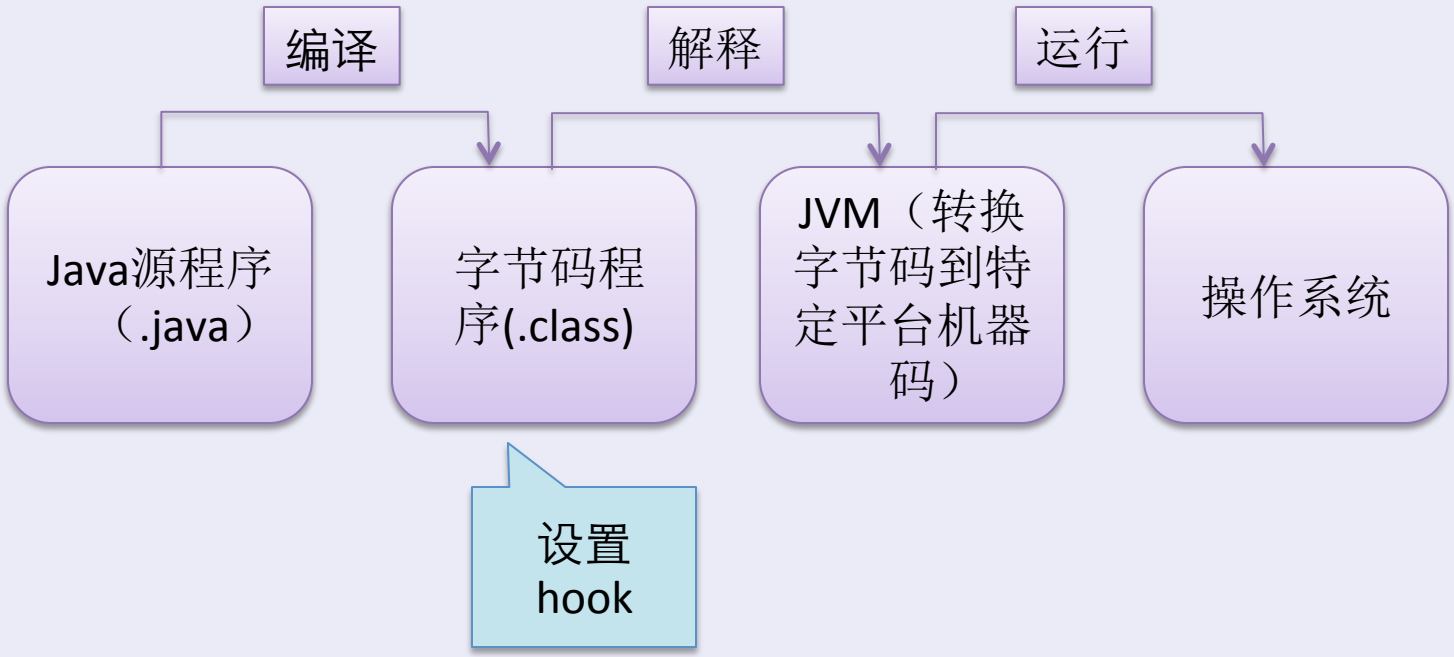
- 目前做的内容
  - Java平台的下的灰盒fuzzing测试
  - 劫持关键的操作函数（使用hook的方式）

- Java hook 的方式

太过繁琐，兼容性不好



- JAVA 编译执行代码的过程



- Javaagent 实现运行时动态修改
  - 在启动和运行期都可以加载agent代理，在启动的时候可通过-javaagent参数来执行agent代理，而在运行期就是通过attach这种机制动态load了

- 修改 MANIFEST.MF 文件中增加启动

Manifest-Version: 1.0

Sealed: true

Main-Class: JagentMain

Premain-Class: com.jagent.Jagent


```
public byte[] transform(java.lang.ClassLoader loader, String className,
        @SuppressWarnings("rawtypes")
        Class redefiningClass, ProtectionDomain domain, byte[] paramArrayOfByte)
        throws IllegalClassFormatException{
        try{
            String classFullName = className.replace("/", ".");
            ClassPool pool = ClassPool.getDefault();
            pool.insertClassPath(new ByteArrayClassPath(classFullName, paramArrayOfByte));


            for(int i = 0; i < classInjects.length; i ++){
                if(className.equals(classInjects[i].getPath()))
                {

                    JagentInjectMethod[] methods = classInjects[i].getMethods();

                    CtClass ctClass = pool.get(classFullName);

                    for(int j = 0; j < methods.length; j++){
                        try{
                            ctClass = ClassTools.classChange(ctClass, methods[j].getMethod(), methods[j].g
                                    methods[j].getCallBackFunc(), paramArrayOfByte, methods[j].getCal Back
                        }catch(Exception e){
                            JagentLogger.printLog(JagentLogger.ERROR, "inject function failed", e);
                        }
                    }
                    return ctClass.toBytecode();
```

- 强大的 Javassist
  - Javassist 是一个开源的分析、编辑和创建Java字节码的类库。Javassist 是 jboss 的一个子项目，其主要的优点，在于简单，而且快速。直接使用java编码的形式，而不需要了解虚拟机指令，就能动态改变类的结构，或者动态生成类。
  - 利用 javassist 对目标函数动态注入字节码代码

- Javassist 实现代码动态修改

```
    if(argNum == 0){
        body = String.format("{" +
                    "%s myclass = new %s();" +
                    "return myclass.%s((Object)this, \"%s\", null);" +
                    "}", callbackClass, callbackClass, callBackFunc, methodName);

    }else{
         body = String.format("{" +
                    "%s myclass = new %s();" +
                    "Object[] objs = $args;" +
                    "return myclass.%s((Object)this, \"%s\", objs);" +
                    "}", callbackClass, callbackClass, callBackFunc, methodName);
    }

}

ctMethod.insertBefore(body);
return ctClass;
```

OWASP 中国
The Open Web Application Security Project

- Tomcat 为例，劫持的关键函数
  - Request请求初始化函数
  - Request销毁函数
  - 数据库查询函数
  - 页面输出函数
  - ……

- Request请求初始化函数
  - 只需要能在执行执行其他劫持函数前获得request 请求的函数都可以
  - Org.apache.catalina.connector.Request 类
  - setRequestedSessionId函数

- Request请求销毁函数
  - 其他函数执行结束后request销毁前执行的函数都可以
  - org.apache.catalina.connector.Request 类
  - recycle

- 数据库查询函数
  - 各种 jdbc 的 class 库中的执行 sql 语句的函数
  - 如:

    com.mysql.jdbc.StatementImpl 类
    executeQuery 函数

    可检测存储跨站或注入漏洞

- 页面输出函数
  - Org.apache.jasper.runtime.JspWriterImpl write 函数

检测跨站脚本、信息泄露等漏洞

- 使用反射解决变量类型问题

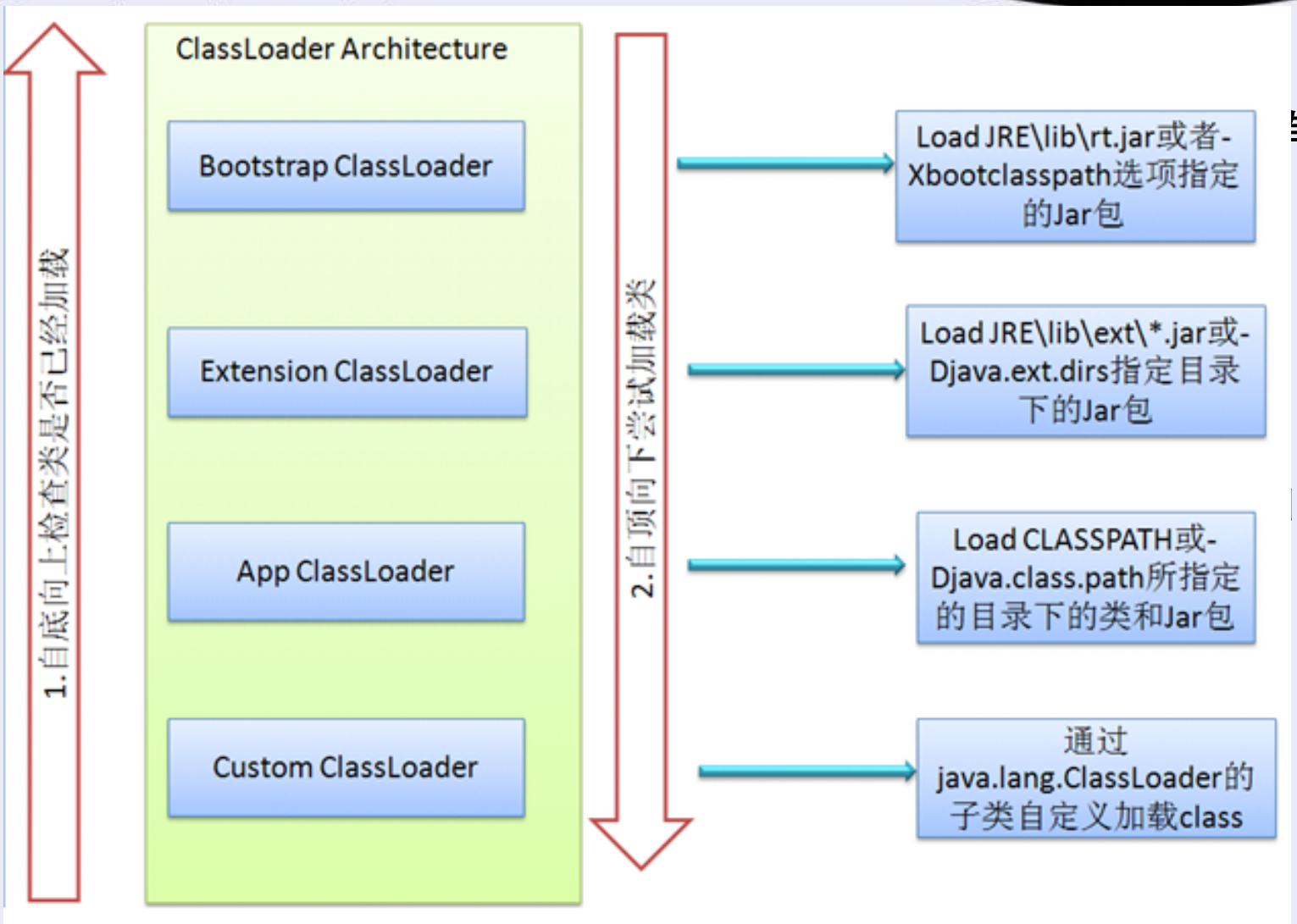
```
public String getRequestURL(){
    Method m = getMethod("getRequestURL");
    if(m == null) return null;
    try{
        StringBuffer sb = (StringBuffer)m.invoke(this.request);
        return sb.toString();
    }catch(Exception e){

        return null;
    }


}
public String getParameter(String name){
    Method m = getMethod("getParameter", name.getClass());
    if(m == null) return null;
    try{
        return (String)m.invoke(this.request, name);
    }catch(Exception e){
        e.printStackTrace();
        return null;
    }
}
```

- 系统函数的 HOOK
  - 部分漏洞的操作实习并非中间件，如
    - 文件操作的漏洞
  - 无法通过hook中间件实现，只能hook系统函数完成

- Java.lang 包的处理
  - JVM启动是加载Runtime，File等类加载优先于 premain 函数，所以无法劫持
  - Java.lang 中的 class 出于安全考虑无法 redefine 或重新加载

- Xbootclass 和 SecurityManager
  - -Xbootclasspath:bootclasspath 让 jvm 从指定路径（可以是分号分隔的目录、jar、或者zip）中加载bootclass，用来替换jdk的rt.jar
  - SecurityManager，java的安全管理器（沙盘）

ClassLoader加载流程

- 

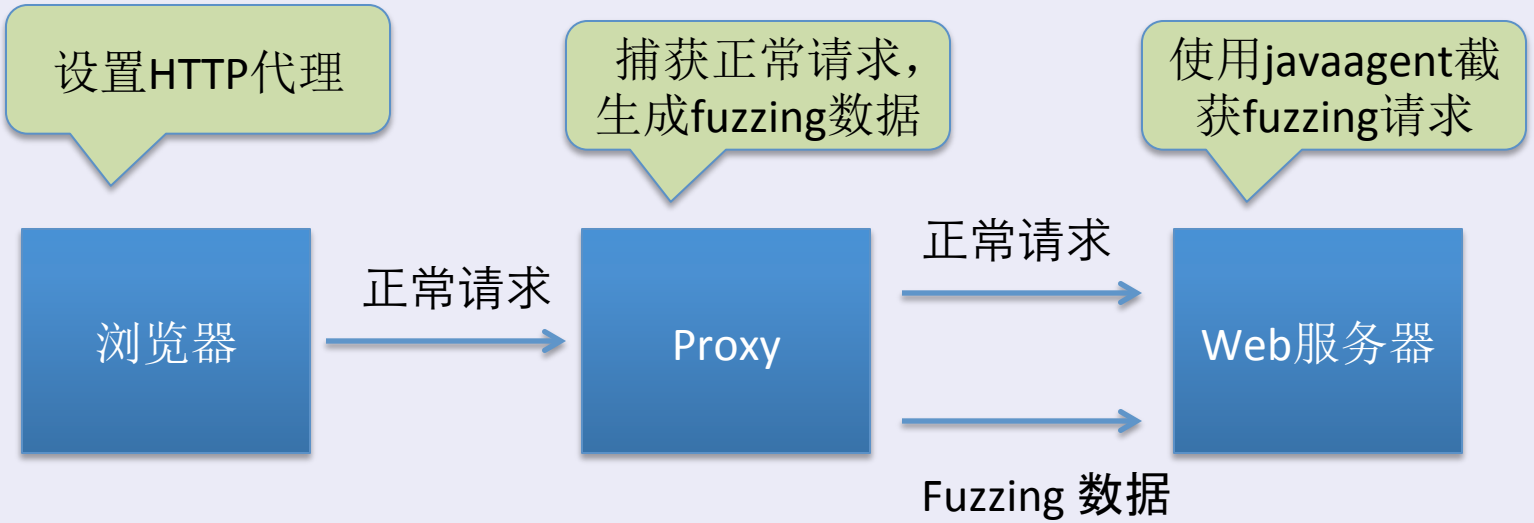
与到

果

- SecurityMananger

```
public static void premain(String paramString,
        Instrumentation paramInstrumentation){
    public class JagentSecurityManager extends SecurityManager {

        public void checkAccess(Thread paramThread){
            //System.out.println("checkAccess: " + paramThread.toString() );
        }
        public void checkRead(String file){
            FileSecCallBack.checkRead(file);
            //System.out.println("checkRead: " + file );
        }
        public void checkRead(String paramString, Object paramObject){
            FileSecCallBack.checkRead(paramString, paramObject);
            //System.out.println("checkRead: " + paramString );
        }
}
```

- 如何获得测试用例


设置HTTP代理
捕获正常请求，
生成fuzzing数据
使用javaagent截
获fuzzing请求
浏览器
正常请求
Proxy
正常请求
Web服务器
Fuzzing 数据

ProxyTest

代理端口 8080 目标站点 10.37.129.2 连接

Fuzzing

目录树

```
⊟ 10.37.129.2:8080
    /
    tomcat.gif
    tomcat-power.gif
    asf-logo-wide.gif
    RELEASE-NOTES.txt
  ⊟ examples
    ⊟ jsp
        /
      ⊟ images
          code.gif
          return.gif
          execute.gif
      ⊟ jsp2
        ⊞ el
    ⊟ servlets
        /
      ⊟ images
          code.gif
          return.gif
          execute.gif
      ⊟ servlet
          SessionExample
          CookieExample
          RequestInfoExample
          HelloWorldExample
  ⊟ docs
```

漏洞信息

```
⊟ SQL注入
  ⊟ http://10.37.129.2:8080/test.jsp
      args_name: name, poc: xxxx'--
  ⊟ http://10.37.129.2:8080/test.jsp
      args_name: name, poc: 10/3412
⊟ 跨站脚本
  ⊟ http://10.37.129.2:8080/test.jsp
      args_name: name, poc: <script>sfeufw82</script>
```

信息

```
Add a new request to fuzzing
Add a new request to fuzzing
not support https, CONNECT urs.microsoft.com:443 HTTP/1.0
not support https, CONNECT urs.microsoft.com:443 HTTP/1.0
not support https, CONNECT urs.microsoft.com:443 HTTP/1.0
not support https, CONNECT urs.microsoft.com:443 HTTP/1.0
not support https, CONNECT urs.microsoft.com:443 HTTP/1.0
not support https, CONNECT urs.microsoft.com:443 HTTP/1.0
not support https, CONNECT urs.microsoft.com:443 HTTP/1.0
not support https, CONNECT urs.microsoft.com:443 HTTP/1.0
not support https, CONNECT urs.microsoft.com:443 HTTP/1.0
not support https, CONNECT urs.microsoft.com:443 HTTP/1.0
not support https, CONNECT urs.microsoft.com:443 HTTP/1.0
```

测试队列: 0 已测试: 4

```
MacBook:b
Using CAT
Using CAT
Using CAT
Using JRE                                                                  me
Using CLA
[debug]lo
[debug]lo
[debug]lo
[debug]lo
[debug]lo
[debug]lo
[debug]lo
[debug]lo
[debug]lo
[debug]lo
[debug]lo
[debug]lo
[info]ini
2013-6-30
??Ш: The                                                                   ction enviror
tem/Libra
2013-6-30 15:28:59 org.apache.coyote.http11.Http11Protocol init
??Ш: Initializing Coyote HTTP/1.1 on http-8080
```

- 扩展
  - 劫持框架的关键函数，对中间件的安全进行测试
  - 劫持所有应用函数，判断瓶颈的函数
  - 劫持函数进行攻击阻断
  - Php的灰盒， apache模块扩展
  - .net的灰盒测试， Profiling API

OWASP 中国
The Open Web Application Security Project

THINK YOU