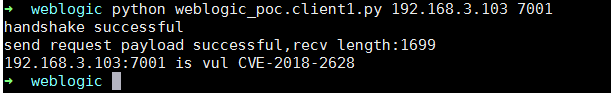
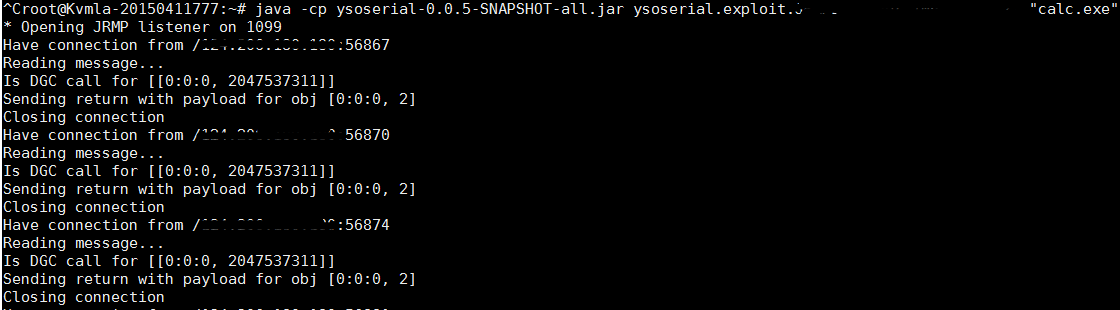
cve-2018-2628 简单复现与分析

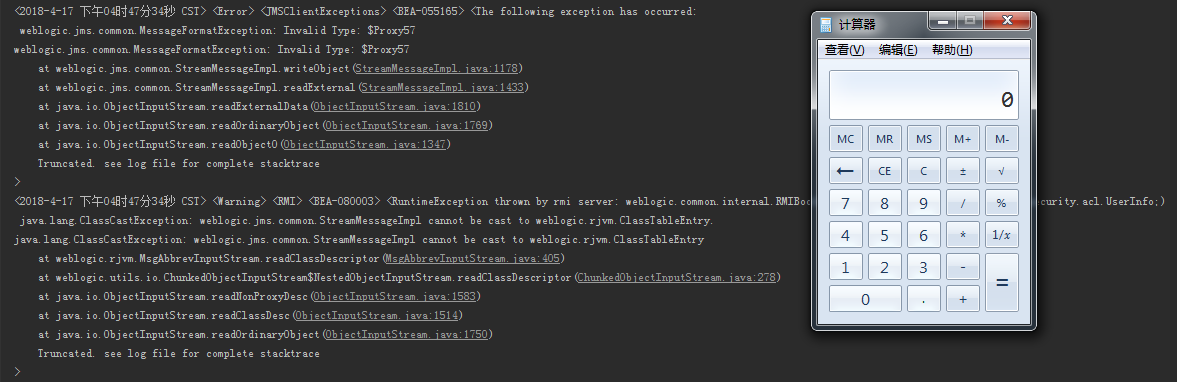
复现：第一步发送测试PoC，PoC中远程连接的服务器地址就是第二步中所使用的服务器，攻击的ip是192.168.3.103的7001端口上的T3服务，该服务会解包Object结构，通过一步步的readObject去第二步服务器上的1099端口请求恶意封装的代码，然后在本地弹出计算器。



第二步在远程服务器上启用ysoserial.exploit.JRMPListener，JRMPListener会将含有恶意代码的payload发送回请求方。



查看weblogic的日志，可以看到如下错误，此时已经弹出计算器：



分析：

Weblogic已经将互联网暴露的PoC都已经加入了黑名单，如果要绕过他的黑名单的限制就只能自己动手构造。来看看InboundMsgAbbrev中resolveProxyClass的实现，resolveProxyClass是处理rmi接口类型的，只判断了java.rmi.registry.Registry，其实随便找一个rmi接口即可绕过。

protected Class<?> resolveProxyClass(String[] interfaces) throws IOException, ClassNotFoundException {  
   String[] arr$ = interfaces;  
   int len$ = interfaces.length;  
  
   for(int i$ = 0; i$ < len$; ++i$) {  
      String intf = arr$[i$];  
      if(intf.equals("java.rmi.registry.Registry")) {  
         throw new InvalidObjectException("Unauthorized proxy deserialization");  
      }  
   }  
  
   return super.resolveProxyClass(interfaces);  
}

其实核心部分就是JRMP（Java Remote Method protocol），在这个PoC中会序列化一个RemoteObjectInvocationHandler，它会利用UnicastRef建立到远端的tcp连接获取RMI registry，加载回来再利用readObject解析，从而造成反序列化远程代码执行。

披露时间线：

2017/7/19：发现问题

2017/11/23：报告给Oracle官方

2017/11/29：Oracle官方接收

2017/11/30：Oracle官方分配bug号(S0947640)，正式进入主线版本修复

2017/11/30：索要公司域名邮箱

2018/4/14：分配CVE，CVE-2018-2628

2018/4/17：发布补丁