【JSP代码审计】某商城几处漏洞审计分析

mosin / 2017-12-21 15:02:00 / 浏览数 3117 安全技术 漏洞分析 顶(0) 踩(0)

---

0x00 前言

前段时间在测试某商城系统时,发现使用了某通用CMS，本着学习的态度，于是下载下来对源码进行分析。
因为自己懒得搭环境，喜欢实战环境，所以直接找官方Demo，因为漏洞提交至今未修复，为了不涉及某商城系统，故对截图进行了打码。

0x01 漏洞分析

远程代码执行

打开源码包，看到这个，struts2架构，远古时期，包都不用解，一看就知道ST2-16。

不搭环境，带上工具，直接官网Demo

任意文件上传
struts2架构，相关页面关联信息都写在了xml里面，找起来方便
我们看看上传页面的前端文件upload.jsp

```
<script type="text/javascript">
       $(function() {
           var appType = $('#appType').val();
           var url = ctx + '/FileUpload!save.do?appType='+appType;
           var fileDesc = '';
           var fileExt = '';
           if(appType == 'magazine'){
               fileDesc = '■■■■■doc/docx/pdf/rar/zip/txt.';
               fileExt = '*.doc;*.docx;*.pdf;*.rar;*.zip;*.txt';
           }else{
               fileDesc = '■■■■:jpg/gif/jpeg/png/bmp.';
               fileExt = '*.jpg;*.gif;*.jpeg;*.png;*.bmp';
           }
           $("#fileupload").uploadify({
               'uploader'       : ctx + '/scripts/framework/uploadify/uploadify.swf',
               'script'         : url,                                          //servlet■■■■■.jsp
               'cancelImg'      : ctx + '/scripts/framework/uploadify/cancel.png',
               'fileDataName'   : 'fileupload',                                 //■■
               'queueID'        : 'fileQueue',
               'auto'           : false,                                        //■■■■■■■■■■■■■■■false
               'multi'          : false,                                        //■■■■■■■■■■■■■■■false
               'simUploadLimit' : 1,                                            //■■■■■■■■■■■■
               'sizeLimit'      : 2000000,                                      //■■■■■■■■■■■■■■■byte
               'queueSizeLimit' : 5,                                            //■■■■■■■■■■■■■■■■■■■■■■■■■■= 999■■■■■■■■
               'fileDesc'       : fileDesc,                                     //■■■■■■■■■'fileExt'■■■■■■■■■■■■■■■
               'fileExt'        : fileExt,                                      //■■■■■■
               onComplete: function (event, queueID, fileObj, response, data) {
                   var arrTemp = response.split(',');
                   var idStr = arrTemp[0];
                   var picPath = arrTemp[1];
                   if('false'==idStr){
                       idStr = '';
                       alert("■■:" + fileObj.name + "■■■■■");
                   }else{
                       alert("■■:" + fileObj.name + "■■■■■");

                       if('Good'== appType || 'GoodAlbum' == appType || 'magazine' == appType || 'Packaging' == appType || 'Pr
                           parent.upload.close(idStr,picPath);
                       }else if('GoodExtend' == appType || 'Advertise' == appType || 'magazinePic' == appType || 'Information'
                           window.returnValue = picPath;
                           window.close();
                       }
                   }
               },
               onError: function(event, queueID, fileObj) {
                   alert("■■:" + fileObj.name + "■■■■■");
```

```
            },
            onCancel: function(event, queueID, fileObj){
                //alert("■■■" + fileObj.name);
            }
        });

    });
</script>
```

文件对appType进行判断，继而进行处理，对于我们来讲，appType在这里没有实质性的作用，只是选择上传的类型目录而已。

来看下处理上传的文件，在文件FileUploadAction.class中

```
public void save(){
String folderPath = Static.APACHE_CONTEXT_PATH + Static.FILE_PATH;
Date now = new Date();
String nowStr = DateUtil.date2Str(now, "yyyyMMdd");
now = DateUtil.str2Date(nowStr, "yyyyMMdd");

folderPath = folderPath + "/" + this.appType + "/" + nowStr;

logger.info("relativePath:" + folderPath);
String idStr = "";
String imgPath = "";
String fileName = "";

boolean isOk = true;
if ((this.fileupload != null) && (this.fileupload.length > 0))
{
 logger.info("fileupload.length:" + this.fileupload.length);

 File savedir = new File(folderPath);
 if (!savedir.exists()) {
   savedir.mkdirs();
 }
 for (int i = 0; i < this.fileupload.length; i++)
 {
   fileName = this.fileuploadFileName[i];
   String postfix = fileName.substring(fileName.lastIndexOf(".") + 1);
   logger.info("uploadFileName[" + i + "]=" + fileName);

   String id = this.fileUploadService.makeId();
   idStr = idStr + (i == 0 ? id : new StringBuilder(",").append(id).toString());

   String fileNewName = id + "." + postfix;
   File savefile = new File(savedir, fileNewName);
   logger.info("save file:" + fileNewName + " to folder:" + savedir.getPath());
   try
   {
     FileUtils.copyFile(this.fileupload[i], savefile);

     FileUpload fileUpload = new FileUpload();
     fileUpload.setId(id);
     fileUpload.setAppType(this.appType);
     fileUpload.setCreateTime(now);
     fileUpload.setPostfix(postfix);
     fileUpload.setOriginalName(fileName);

     StringBuffer relativePath = new StringBuffer();
     relativePath.append(Static.FILE_PATH)
       .append("/").append(this.appType)
       .append("/").append(nowStr)
       .append("/").append(id).append(".").append(postfix);
     fileUpload.setRelativePath(relativePath.toString());
     imgPath = relativePath.toString();
     this.fileUploadService.insert(fileUpload);
   }
   catch (Exception e)
   {
     if (isOk) {
```

```
      isOk = false;
    }
    logger.error("error when copyFile,savefile:" + savefile, e);
  }
 }
}
else
{
 logger.warn("fileupload is null or fileupload.length <=0");
 isOk = false;
}
if (!isOk) {
 responseFlag(isOk);
} else if (this.appType.equals("News")) {
 responseFlag(imgPath);
} else if (this.appType.equals("OrderGood")) {
 responseFlag(idStr + ',' + fileName);
} else {
 responseFlag(idStr + ',' + imgPath);
}
```

首先对appType和目录进行了拼接，也就是上传的路径

```
folderPath = folderPath + "/" + this.appType + "/" + nowStr;
```

判断文件长度大小

```
if ((this.fileupload != null) && (this.fileupload.length > 0))
```

然后取后缀，到这里为止，文件都没有对上传的内容进行任何判断,后缀也是一样，直接读取拼接，不做判断。

加之在文件中也未发现任何的登录权限验证，所以造成了前端无限制任意文件上传。

```
String postfix = fileName.substring(fileName.lastIndexOf(".") + 1);
String fileNewName = id + "." + postfix;
```

下面就是存储过程了，最后返回上传结果。附上传成功并getshell截图。

存储型XSS
这个系统好像通篇没有过滤XSS的脚本，不知道有没有过滤文件反正我没有看到.可以在商品收货地址或商品展示处等地方插入XSS。
因为通篇XSS，所以就挑一个来说
在jsp文件edit_SysUser.jsp中，这个是用于修改个人信息的，定位源码SysUserAction.class
下面是两个重要函数
首先edit()从jsp页面获取到登录用户的信息，对信息进行修改,save()函数接收修改的信息，对用户信息进行存储更新，在文件里面，我们没有看到任何的过滤函数存在。

■■■■■■.....

```
public String edit(){
SysUser loginMan = getSessionUserInfo();
if (this.sysUser == null) {
 this.sysUser = new SysUser();
}
String id = this.sysUser.getId();
if (StringUtils.isBlank(id))
{
 super.initModel(true, this.sysUser, loginMan);
}
else
{
 this.sysUser = ((SysUser)this.sysUserService.getModel(id));

 super.initModel(false, this.sysUser, loginMan);
}
this.sysRoleList = this.sysRoleService.select(null);
if (this.sysRoleList == null) {
 this.sysRoleList = new ArrayList();
} else {
 for (int i = 0; i < this.sysRoleList.size(); i++) {
   if ("admin".equals(((SysRole)this.sysRoleList.get(i)).getCode()))
   {
     this.sysRoleList.remove(i);
```

```
     break;
   }
 }
}
return "edit_SysUser";
```

省略无关代码.....

```
public void save(){
try
{
 String id = this.sysUser.getId();

 String roleId = this.sysUser.getRoleId();
 if (StringUtils.isNotBlank(roleId))
 {
   SysRole sysRole = (SysRole)this.sysRoleService.getModel(roleId);
   this.sysUser.setRoleCode(sysRole.getCode());
   this.sysUser.setRoleName(sysRole.getName());
 }
 if (StringUtils.isBlank(id)) {
   this.sysUserService.insert(this.sysUser);
 } else {
   this.sysUserService.update(this.sysUser);
 }
 responseFlag(true);
}
catch (Exception e)
{
 responseFlag(false);
 logger.error("error occur when save model!", e);
}

}
```

测试结果，后台和前台

任意帐号密码修改

漏洞发生在app\front\action\UserManageAction.class文件中

首先在重置密码处会先进行一次帐号验证，也就是邮箱地址验证是否正确，然后会返回注册手机号码（下面会用到），代码就不贴了，这个不是重点，重点是sendEmail()这

首先会获取提交过来的手机号码和邮箱地址

```
this.customer = getSessionCustomerInfo();
String toMail = this.customer.getEmail();
String registerName = this.customer.getCode();
```

接下来,直接设置发送邮件的帐号密码，url构造随机数ID连接。
然后就是理想的发送邮件验证重置密码连接了。

```
String userName = "XXXXXX@126.com";
String password = "XXXXX";

String registerId = Math.random() * Math.random();
String url = "http://localhsot:8080/frontLogin.do?registerId=" + registerId;

MimeMessage msg = new MimeMessage(session);
msg.setFrom(from);
msg.setSubject("■■■■");
msg.setSentDate(new Date());
msg.setContent("<a href='" + url + "'>■■" + url + "</a>", "text/html;charset=utf-8");
msg.setRecipient(Message.RecipientType.TO, to);
Transport.send(msg);
```

直接构造和修改邮箱，即可修改密码。
http://www.xxx.com/sendEmail.do?customer.code=135xxxxxxx6&customer.email=xxxxx@xxx.com

0x02 最后

分析出了这几个漏洞和看了官网后，越发觉得这家公司为什么还能活着？
因为代码通用，影响旗下所有电商系统。
吐槽：一个软件卖到9000+，3年不升级。还有谁？



上一篇：【渗透技巧】搜集SRC信息中的"技... 下一篇：求推荐，nessus和nexpos...

1. 4 条回复


wooyun 2017-12-21 15:36:28

大佬给你端茶，哈哈：打开源码包，看到这个，struts2架构，远古时期，包都不用解，一看就知道ST2-16

0 回复Ta

---


hades 2017-12-21 16:44:34

2013-10月 估计已经没人维护了吧

0 回复Ta

---


ADog 2017-12-22 10:07:23

膜拜大佬~

0 回复Ta

---

bma 2017-12-26 09:15:57

重置密码的关键代码没贴出来吧？

1 回复Ta

---

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