PAKETTO
DOCCUMENTATION / GUIDE

Contents

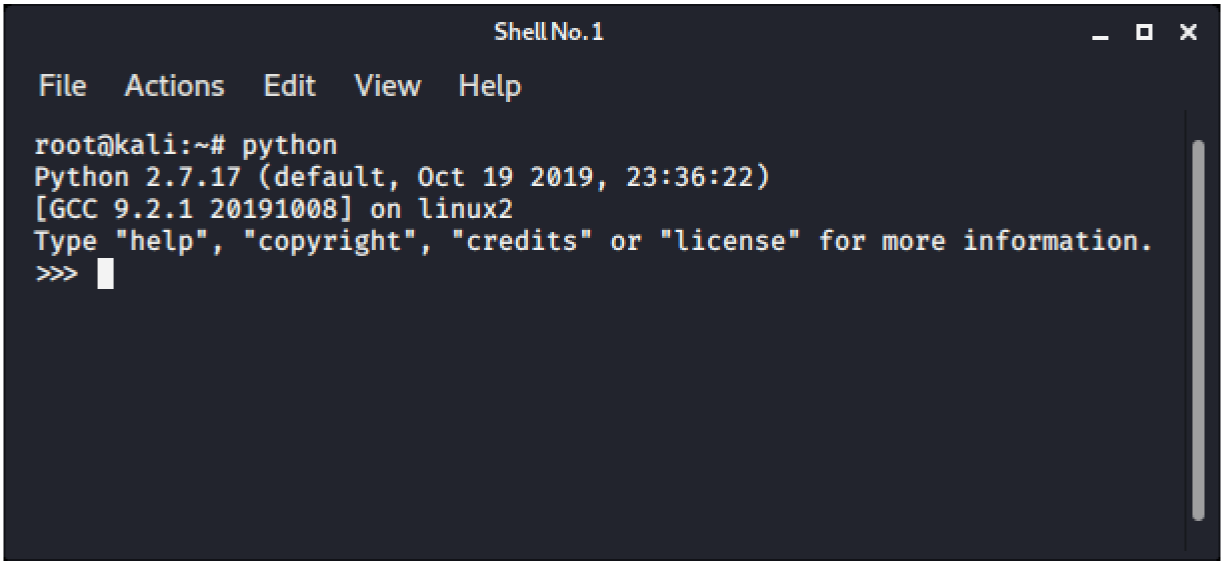
Installation

This guide, will teach you how to install our product onto a fresh kali VM (or any other Linux based machine that supports python and anaconda)

Prerequisites

Python

Make sure the machine has Python installed (higher than v2.6)

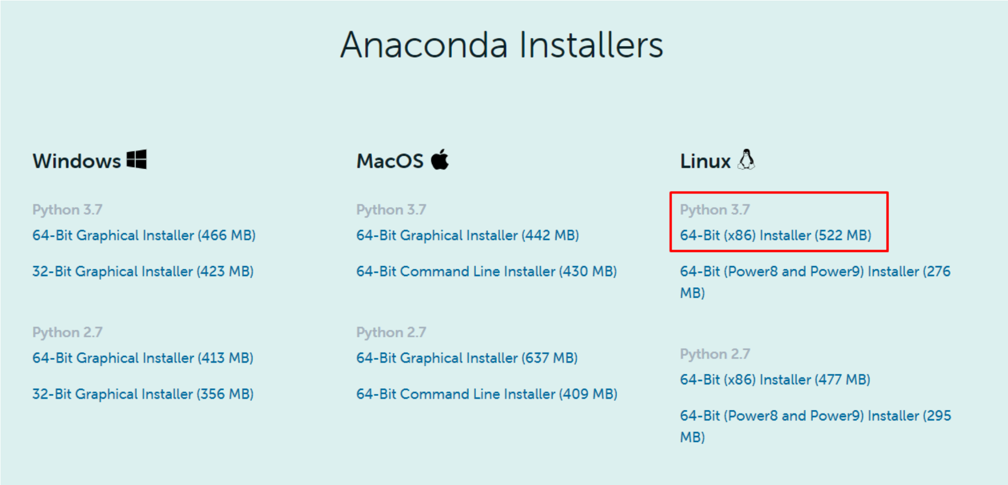


```
root@kali:~# python
Python 2.7.17 (default, Oct 19 2019, 23:36:22)
[GCC 9.2.1 20191008] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> 
```

Anaconda 64-bit (x86)

Next we have to install the anaconda framework on the VM to be able to leverage off of the ML/AI improvements of packetto

First off, get the latest anaconda individual installation from here. Then proceed to download the **64-Bit (x86) Installer** at the bottom of the webpage.



Anaconda Installers

Windows

Python 3.7
64-Bit Graphical Installer (466 MB)
32-Bit Graphical Installer (423 MB)

Python 2.7
64-Bit Graphical Installer (413 MB)
32-Bit Graphical Installer (356 MB)

MacOS

Python 3.7
64-Bit Graphical Installer (442 MB)
64-Bit Command Line Installer (430 MB)

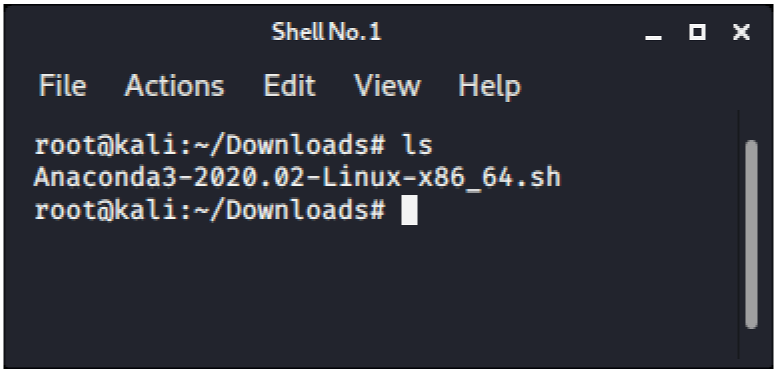
Python 2.7
64-Bit Graphical Installer (637 MB)
64-Bit Command Line Installer (409 MB)

Linux

Python 3.7
64-Bit (x86) Installer (522 MB)
64-Bit (Power8 and Power9) Installer (276 MB)

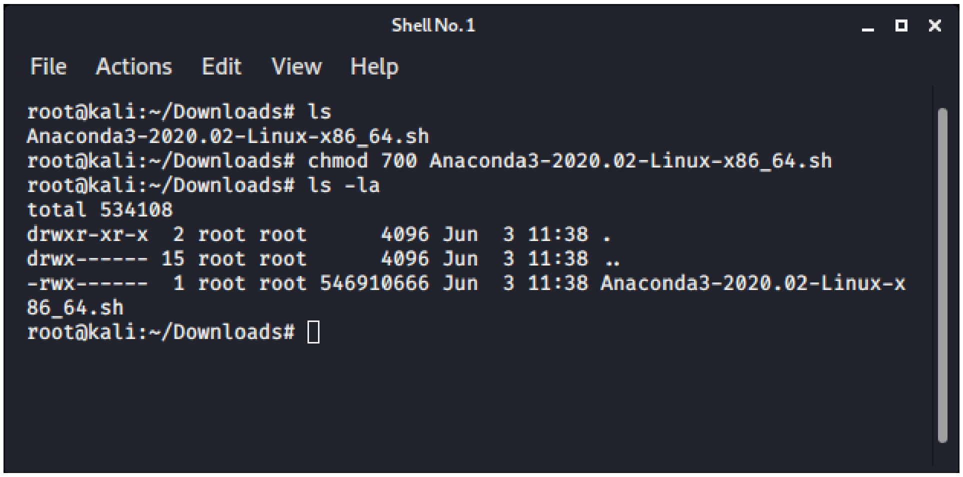
Python 2.7
64-Bit (x86) Installer (477 MB)
64-Bit (Power8 and Power9) Installer (295 MB)

Next up, download and save it onto the kali VM (in our case) and open a shell in the same location the file is saved. Use command `ls` to verify.



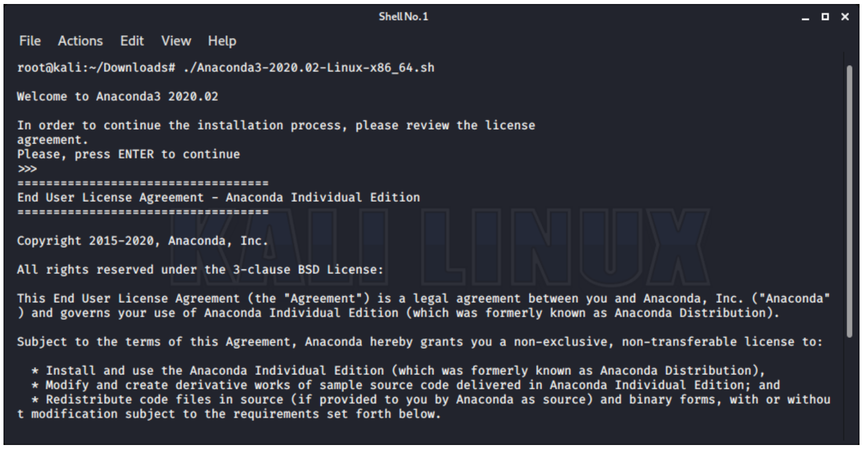
```
root@kali:~/Downloads# ls
Anaconda3-2020.02-Linux-x86_64.sh
root@kali:~/Downloads# 
```

Now we need to give this file permissions so that it is able to install Anaconda successfully

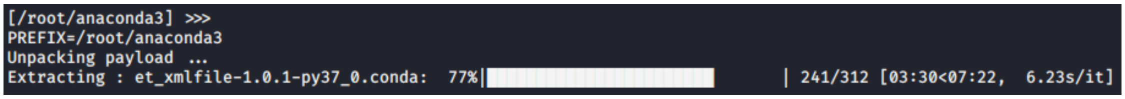


```
root@kali:~/Downloads# ls
Anaconda3-2020.02-Linux-x86_64.sh
root@kali:~/Downloads# chmod 700 Anaconda3-2020.02-Linux-x86_64.sh
root@kali:~/Downloads# ls -la
total 534108
drwxr-xr-x  2 root root      4096 Jun  3 11:38 .
drwx------ 15 root root      4096 Jun  3 11:38 ..
-rwx------  1 root root 546910666 Jun  3 11:38 Anaconda3-2020.02-Linux-x
86_64.sh
root@kali:~/Downloads# 
```

Now to install it, enter the following command.



```
root@kali:~/Downloads# ./Anaconda3-2020.02-Linux-x86_64.sh

Welcome to Anaconda3 2020.02

In order to continue the installation process, please review the license
agreement.
Please, press ENTER to continue
>>>
===================================
End User License Agreement - Anaconda Individual Edition
===================================

Copyright 2015-2020, Anaconda, Inc.

All rights reserved under the 3-clause BSD License:

This End User License Agreement (the "Agreement") is a legal agreement between you and Anaconda, Inc. ("Anaconda"
) and governs your use of Anaconda Individual Edition (which was formerly known as Anaconda Distribution).

Subject to the terms of this Agreement, Anaconda hereby grants you a non-exclusive, non-transferable license to:

  * Install and use the Anaconda Individual Edition (which was formerly known as Anaconda Distribution),
  * Modify and create derivative works of sample source code delivered in Anaconda Individual Edition; and
  * Redistribute code files in source (if provided to you by Anaconda as source) and binary forms, with or withou
t modification subject to the requirements set forth below.
```



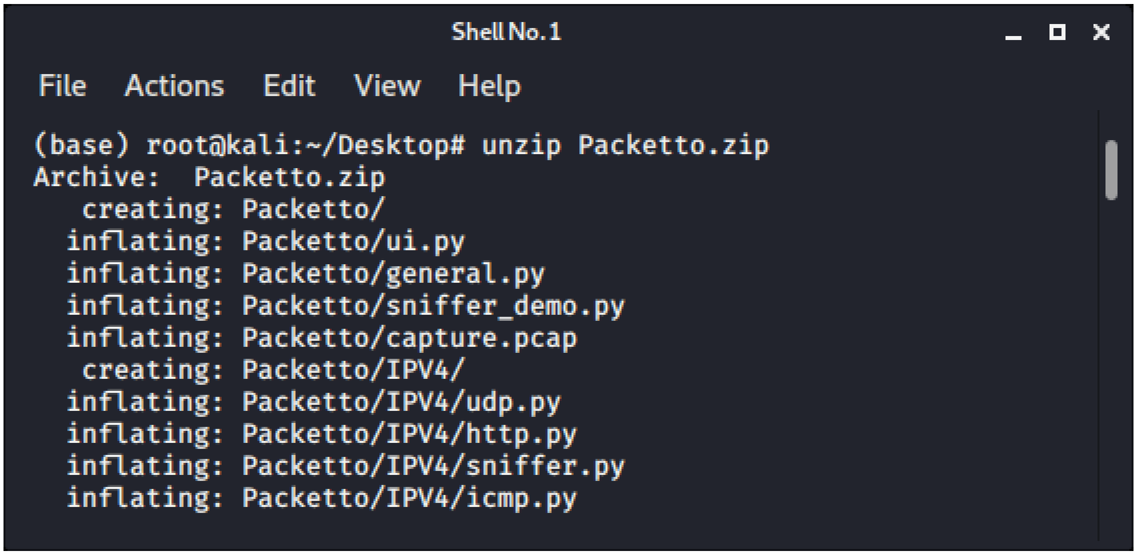
```
[/root/anaconda3] >>>
PREFIX=/root/anaconda3
Unpacking payload ...
Extracting : et_xmlfile-1.0.1-py37_0.conda:  77%|█████████████████████         | 241/312 [03:30<07:22,  6.23s/it]
```

```
root@kali:~/Downloads#  ./Anaconda3-2020.02-Linux-x86_64.sh
```

Importing and unzipping files

Packetto.zip

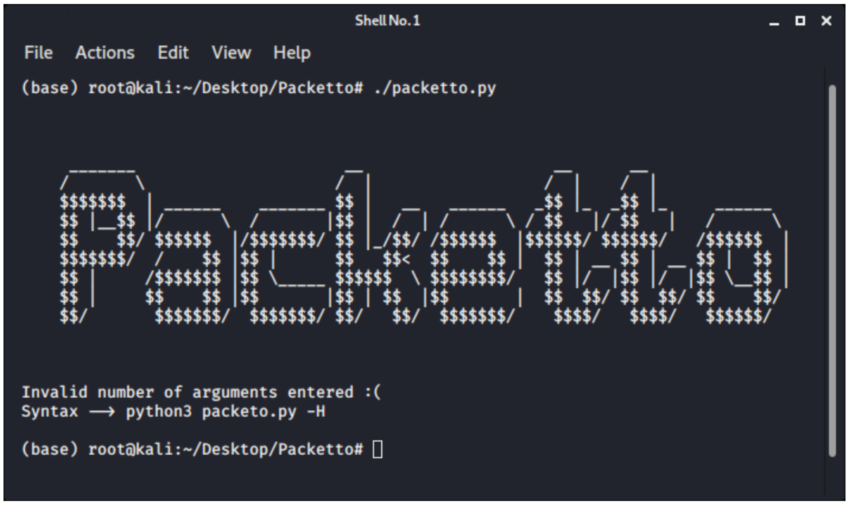
Import this zip file into the VM (Desktop). Unzip it as shown below.



```
(base) root@kali:~/Desktop# unzip Packetto.zip
Archive:  Packetto.zip
   creating: Packetto/
  inflating: Packetto/ui.py
  inflating: Packetto/general.py
  inflating: Packetto/sniffer_demo.py
  inflating: Packetto/capture.pcap
   creating: Packetto/IPV4/
  inflating: Packetto/IPV4/udp.py
  inflating: Packetto/IPV4/http.py
  inflating: Packetto/IPV4/sniffer.py
  inflating: Packetto/IPV4/icmp.py
```

```
root@kali:~/Desktop# unzip Packetto.zip
```

If everything was extracted correctly, this should be the output when the command shown below is entered.

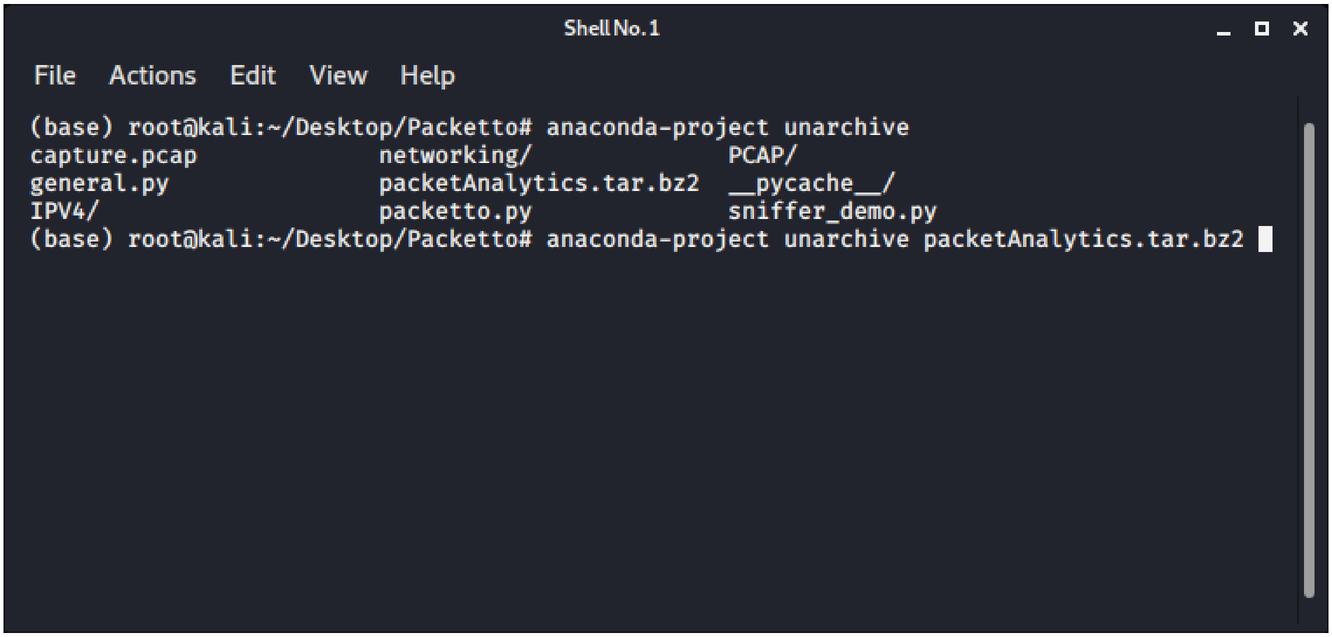


```
(base) root@kali:~/Desktop/Packetto# ./packetto.py



 _______                       __                   __       __
/       \                     /  |                 /  |     /  |
$$$$$$$  |  ______    _______ $$ |   __   ______  _$$ |_   _$$ |_    ______
$$ |__$$ | /      \  /       |$$ |  /  | /      \/ $$   | / $$   |  /      \
$$    $$/  $$$$$$  |/$$$$$$$/ $$ |_/$$/ /$$$$$$  |$$$$$$/  $$$$$$/  /$$$$$$  |
$$$$$$$/   /    $$ |$$ |      $$   $$<  $$    $$ |  $$ | __  $$ | __ $$ |  $$ |
$$ |      /$$$$$$$ |$$ \_____ $$$$$$  \ $$$$$$$$/   $$ |/  | $$ |/  |$$ \__$$ |
$$ |      $$    $$ |$$       |$$ | $$  |$$       |  $$  $$/  $$  $$/ $$    $$/
$$/        $$$$$$$/  $$$$$$$/ $$/   $$/  $$$$$$$/    $$$$/    $$$$/   $$$$$$/



Invalid number of arguments entered :(
Syntax ⟶ python3 packeto.py -H

(base) root@kali:~/Desktop/Packetto# ▯
```

```
root@kali:~/Desktop/Packetto# ./packetto.py
```

packetAnalytics.tar.bz2

A file with this name will be found in the Packetto directory. We will have to unzip that with anaconda. To do that lets open a terminal in the same directory.

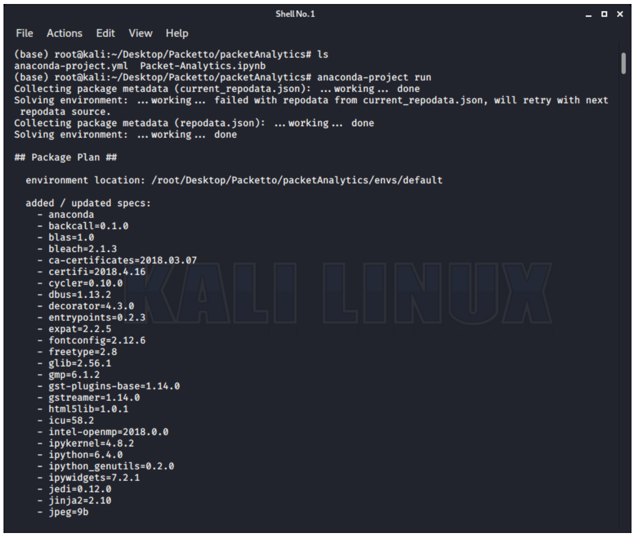


```
(base) root@kali:~/Desktop/Packetto# anaconda-project unarchive
capture.pcap              networking/               PCAP/
general.py                packetAnalytics.tar.bz2   __pycache__/
IPV4/                     packetto.py               sniffer_demo.py
(base) root@kali:~/Desktop/Packetto# anaconda-project unarchive packetAnalytics.tar.bz2
```

```
root@kali:~/Desktop/Packetto# anaconda-project unarchive packetAnalytics.tar.bz2
```

After unzipping we need to change directory into it and run the anaconda project and let it setup.

```
root@kali:~/Desktop/Packetto/packetAnalytics# anaconda-project run
```



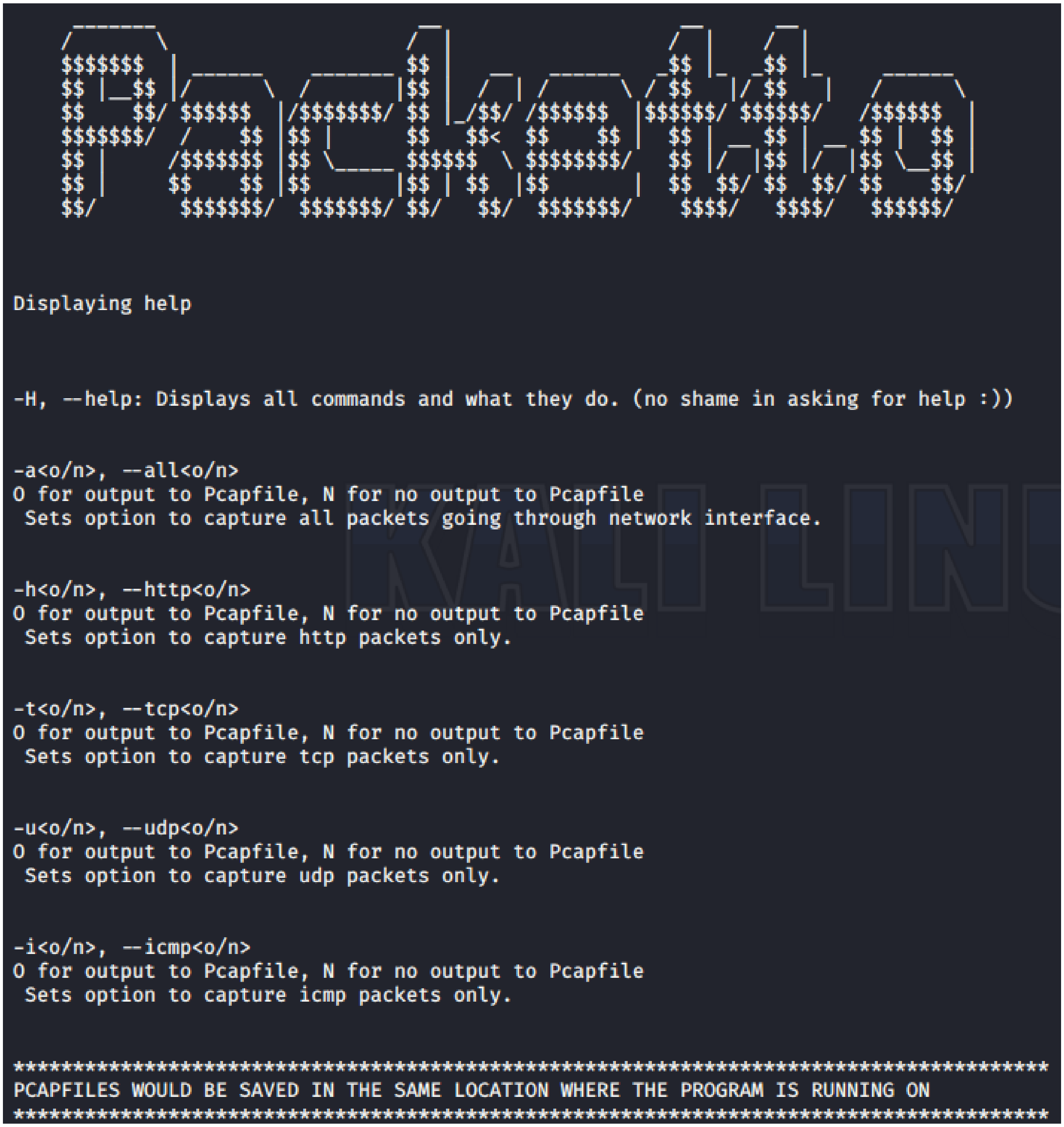
```
(base) root@kali:~/Desktop/Packetto/packetAnalytics# ls
anaconda-project.yml  Packet-Analytics.ipynb
(base) root@kali:~/Desktop/Packetto/packetAnalytics# anaconda-project run
Collecting package metadata (current_repodata.json): ...working... done
Solving environment: ...working... failed with repodata from current_repodata.json, will retry with next
 repodata source.
Collecting package metadata (repodata.json): ...working... done
Solving environment: ...working... done

## Package Plan ##

  environment location: /root/Desktop/Packetto/packetAnalytics/envs/default

  added / updated specs:
    - anaconda
    - backcall=0.1.0
    - blas=1.0
    - bleach=2.1.3
    - ca-certificates=2018.03.07
    - certifi=2018.4.16
    - cycler=0.10.0
    - dbus=1.13.2
    - decorator=4.3.0
    - entrypoints=0.2.3
    - expat=2.2.5
    - fontconfig=2.12.6
    - freetype=2.8
    - glib=2.56.1
    - gmp=6.1.2
    - gst-plugins-base=1.14.0
    - gstreamer=1.14.0
    - html5lib=1.0.1
    - icu=58.2
    - intel-openmp=2018.0.0
    - ipykernel=4.8.2
    - ipython=6.4.0
    - ipython_genutils=0.2.0
    - ipywidgets=7.2.1
    - jedi=0.12.0
    - jinja2=2.10
    - jpeg=9b
```

After that 15 minute process has completed and succeeded, we are all setup to use the product!!

Usage of Packetto

Packet sniffing with Packetto

There are various different kinds of packets that you can sniff with packetto. To get started, you can visit the help page following the commands shown below. (Open the shell in the Packetto directory first)



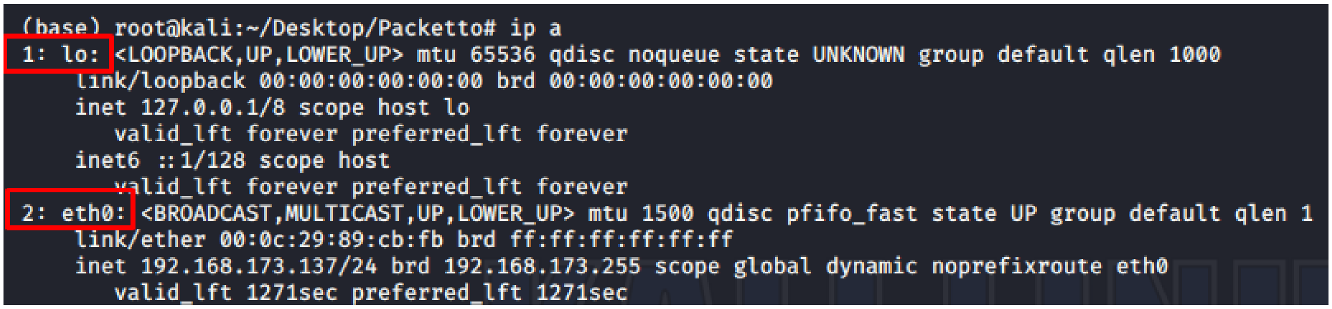
```
     _______                      __                    __      __
    /       \                    /  |                  /  |    /  |
    $$$$$$$  | ______    _______ $$ |   __   ______   _$$ |_  _$$ |_     ______
    $$ |__$$ |/      \  /       |$$ |  /  | /      \ / $$   |/ $$   |   /      \
    $$    $$/ $$$$$$  |/$$$$$$$/ $$ |_/$$/ /$$$$$$  |$$$$$$/ $$$$$$/   /$$$$$$  |
    $$$$$$$/  /    $$ |$$ |      $$   $$<  $$    $$ |  $$ | __ $$ | __ $$ |  $$ |
    $$ |     /$$$$$$$ |$$ \_____ $$$$$$  \ $$$$$$$$/   $$ |/  |$$ |/  |$$ \__$$ |
    $$ |     $$    $$ |$$       |$$ | $$  |$$       |  $$  $$/ $$  $$/ $$    $$/
    $$/       $$$$$$$/  $$$$$$$/ $$/   $$/  $$$$$$$/    $$$$/   $$$$/    $$$$$$/



Displaying help



-H, --help: Displays all commands and what they do. (no shame in asking for help :))


-a<o/n>, --all<o/n>
O for output to Pcapfile, N for no output to Pcapfile
 Sets option to capture all packets going through network interface.


-h<o/n>, --http<o/n>
O for output to Pcapfile, N for no output to Pcapfile
 Sets option to capture http packets only.


-t<o/n>, --tcp<o/n>
O for output to Pcapfile, N for no output to Pcapfile
 Sets option to capture tcp packets only.


-u<o/n>, --udp<o/n>
O for output to Pcapfile, N for no output to Pcapfile
 Sets option to capture udp packets only.


-i<o/n>, --icmp<o/n>
O for output to Pcapfile, N for no output to Pcapfile
 Sets option to capture icmp packets only.


**************************************************************************************
PCAPFILES WOULD BE SAVED IN THE SAME LOCATION WHERE THE PROGRAM IS RUNNING ON
**************************************************************************************
```

```
root@kali:~/Desktop/Packetto# ./packetto.py -H
```

In each option in our program, you are able to choose which adapter you would like to sniff incase you have more than one main adapter on your machine. Moreover, you are even able to choose if u would like to output it to a pcap file for further analysis with other software or our AI/ML driven packet analysis solution!

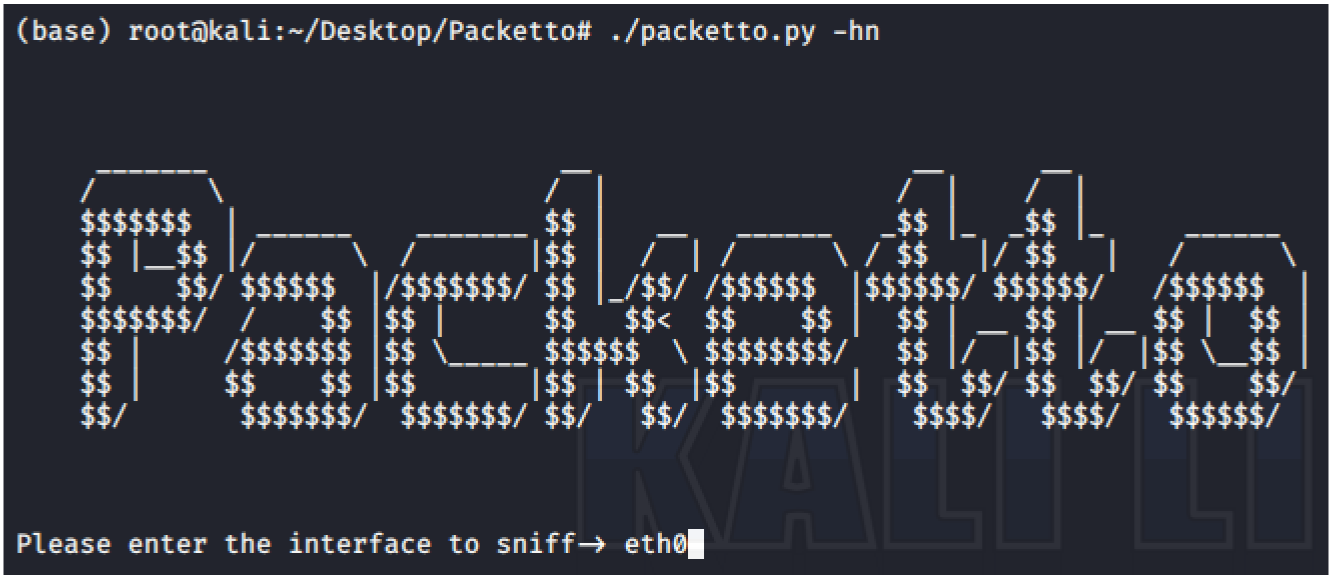
For e.g
To select the network adapter that you would like to use for the sniffing process, do the following.

```
root@kali:~/Desktop/Packetto# ip a
```



```
(base) root@kali:~/Desktop/Packetto# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1
    link/ether 00:0c:29:89:cb:fb brd ff:ff:ff:ff:ff:ff
    inet 192.168.173.137/24 brd 192.168.173.255 scope global dynamic noprefixroute eth0
       valid_lft 1271sec preferred_lft 1271sec
```

As you can see above there are 2 adapters available to us to use for sniffing. When u launch our program with the correct argument stated in the help page, you will see this question asked to you. You can enter the adapter that you would like to use and then hit enter!



```
(base) root@kali:~/Desktop/Packetto# ./packetto.py -hn



     _______                      __                  __       __
    /       \                    /  |                /  |     /  |
    $$$$$$$  |  ______    _______ $$ |   __   ______  _$$ |_   _$$ |_     ______
    $$ |__$$ | /      \  /       |$$ |  /  | /      \/ $$   | / $$   |   /      \
    $$    $$/  $$$$$$  |/$$$$$$$/ $$ |_/$$/ /$$$$$$  |$$$$$$/  $$$$$$/   /$$$$$$  |
    $$$$$$$/   /    $$ |$$ |      $$   $$<  $$    $$ |  $$ | __  $$ | __ $$ |  $$ |
    $$ |      /$$$$$$$ |$$ \_____ $$$$$$  \ $$$$$$$$/   $$ |/  | $$ |/  |$$ \__$$ |
    $$ |      $$    $$ |$$       |$$ | $$  |$$       |  $$  $$/  $$  $$/ $$    $$/
    $$/        $$$$$$$/  $$$$$$$/ $$/   $$/  $$$$$$$/    $$$$/    $$$$/   $$$$$$/



Please enter the interface to sniff→ eth0
```

If you want to filter for HTTP packets but do not want to save the output to a pcap file, you can do the following

```
root@kali:~/Desktop/Packetto# ./packetto.py -hn
```

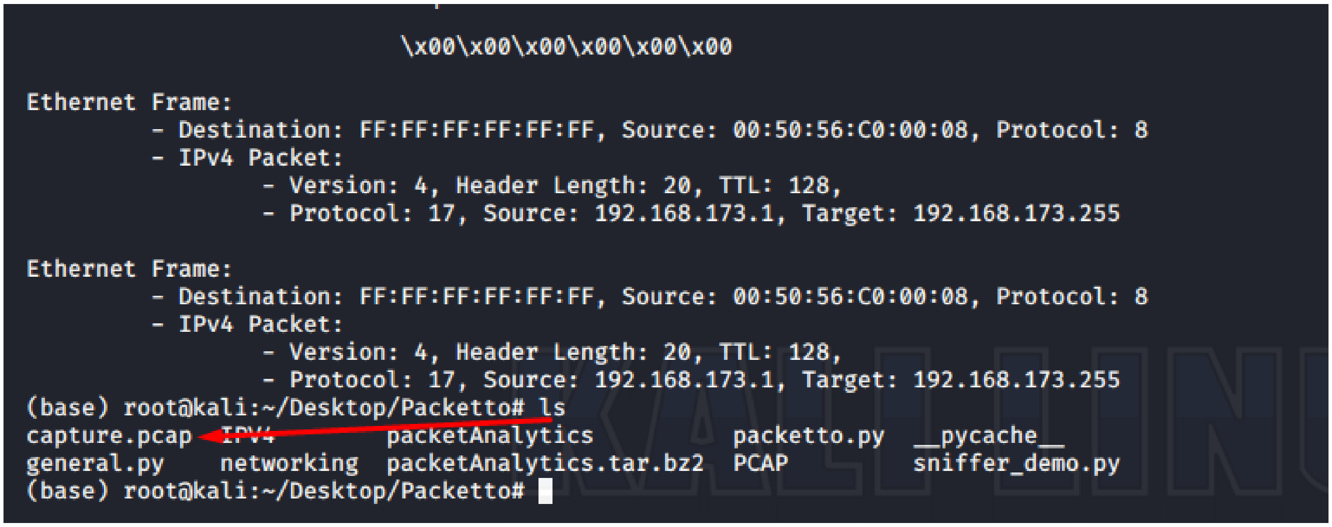
If you want to filter for HTTP packets and want to save the output to a pcap file, you can do the following

```
root@kali:~/Desktop/Packetto# ./packetto.py -ho
```

As you can see the only difference is the value passed together with the main argument. O or N.

```
O = output to pcap file

N = no output to pcap file
```

Here's how things should look if everything goes through properly.



```
                             \x00\x00\x00\x00\x00\x00

Ethernet Frame:
         - Destination: FF:FF:FF:FF:FF:FF, Source: 00:50:56:C0:00:08, Protocol: 8
         - IPv4 Packet:
                 - Version: 4, Header Length: 20, TTL: 128,
                 - Protocol: 17, Source: 192.168.173.1, Target: 192.168.173.255

Ethernet Frame:
         - Destination: FF:FF:FF:FF:FF:FF, Source: 00:50:56:C0:00:08, Protocol: 8
         - IPv4 Packet:
                 - Version: 4, Header Length: 20, TTL: 128,
                 - Protocol: 17, Source: 192.168.173.1, Target: 192.168.173.255
(base) root@kali:~/Desktop/Packetto# ls
capture.pcap  IPV4        packetAnalytics          packetto.py  __pycache__
general.py    networking  packetAnalytics.tar.bz2  PCAP         sniffer_demo.py
(base) root@kali:~/Desktop/Packetto# 
```

There should be a new file named capture.pcap

Packet Analysis (AI/ML)

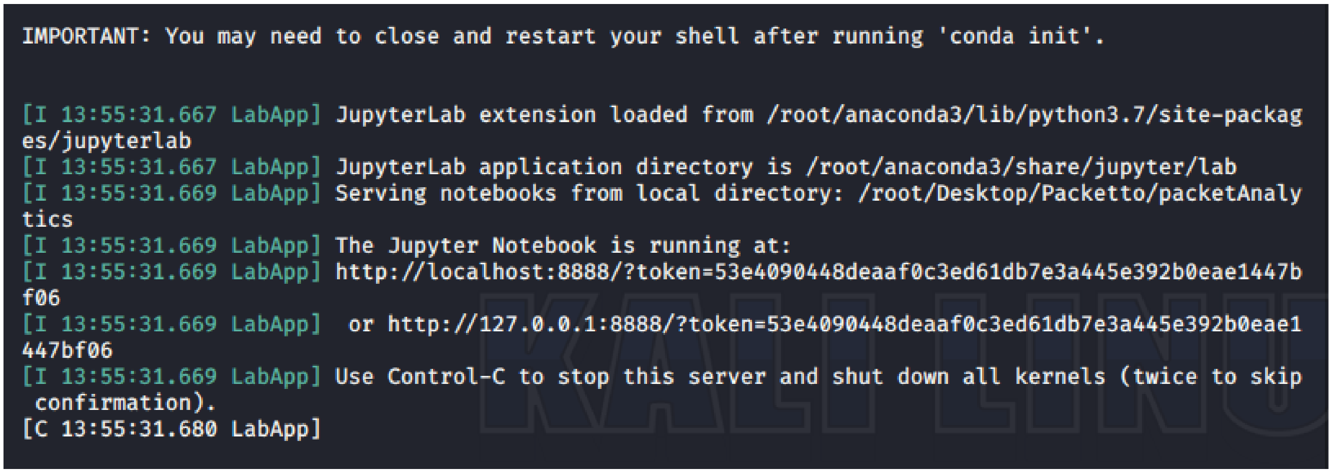
Start the tool

We first need to change directory into the packetAnalytics directory. In there is a file named `initialise.sh` This will make it much easier for the end user to import the pcap capture file into the directory and setup the anaconda environment. First enter this command the command below and then Run that file similar to how we would run any shell script.

```
root@kali:~/# conda activate /root/Desktop/Packetto/packetAnalytics/envs/default

root@kali:~/Desktop/Packetto/packetAnalytics# ./initialise.sh
```

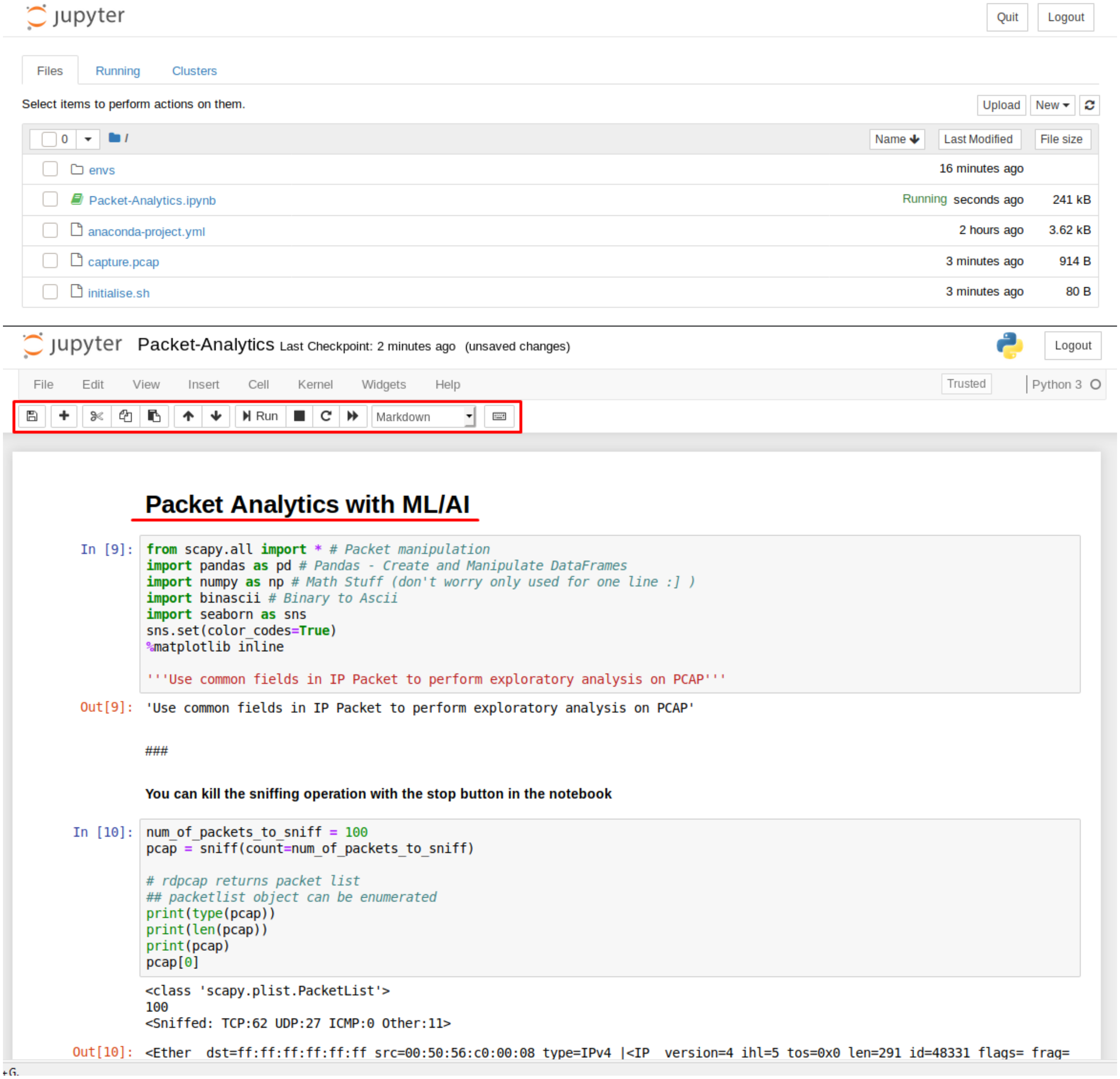
The script and the command does 4 important things.

- Removes any pcap files that exist in the directory(incase the previous user forgot to remove it)
- Brings the newly created pcap file by Packetto into this directory and preps it for data analytics
- Activates the anaconda environment with python 3 and all the dependencies that we need for this project
- Initiates jupyter lab with root permissions



```
IMPORTANT: You may need to close and restart your shell after running 'conda init'.


[I 13:55:31.667 LabApp] JupyterLab extension loaded from /root/anaconda3/lib/python3.7/site-packag
es/jupyterlab
[I 13:55:31.667 LabApp] JupyterLab application directory is /root/anaconda3/share/jupyter/lab
[I 13:55:31.669 LabApp] Serving notebooks from local directory: /root/Desktop/Packetto/packetAnaly
tics
[I 13:55:31.669 LabApp] The Jupyter Notebook is running at:
[I 13:55:31.669 LabApp] http://localhost:8888/?token=53e4090448deaaf0c3ed61db7e3a445e392b0eae1447b
f06
[I 13:55:31.669 LabApp]  or http://127.0.0.1:8888/?token=53e4090448deaaf0c3ed61db7e3a445e392b0eae1
447bf06
[I 13:55:31.669 LabApp] Use Control-C to stop this server and shut down all kernels (twice to skip
 confirmation).
[C 13:55:31.680 LabApp]
```

Disclaimer: You will only be able to access the jupyter lab environment on the web browser using the link provided in the terminal due to the token auth being switched on. This is to prevent some attack vectors.



Select items to perform actions on them.

☐ envs		16 minutes ago	
☐ Packet-Analytics.ipynb	Running	seconds ago	241 kB
☐ anaconda-project.yml		2 hours ago	3.62 kB
☐ capture.pcap		3 minutes ago	914 B
☐ initialise.sh		3 minutes ago	80 B

Packet-Analytics Last Checkpoint: 2 minutes ago (unsaved changes)

Packet Analytics with ML/AI

```
In [9]:  from scapy.all import * # Packet manipulation
         import pandas as pd # Pandas - Create and Manipulate DataFrames
         import numpy as np # Math Stuff (don't worry only used for one line :] )
         import binascii # Binary to Ascii
         import seaborn as sns
         sns.set(color_codes=True)
         %matplotlib inline

         '''Use common fields in IP Packet to perform exploratory analysis on PCAP'''

Out[9]:  'Use common fields in IP Packet to perform exploratory analysis on PCAP'
```

###

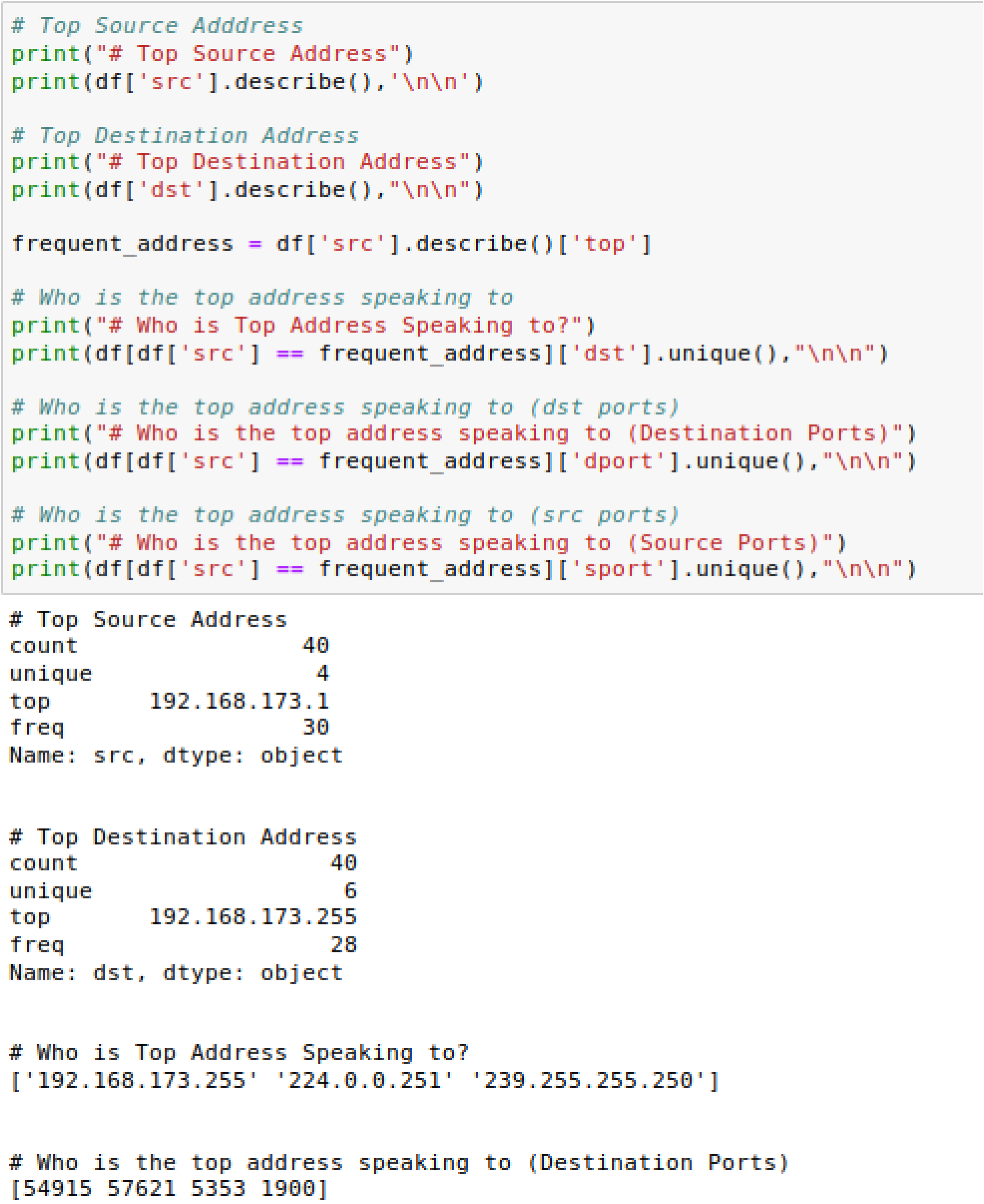
You can kill the sniffing operation with the stop button in the notebook

```
In [10]: num_of_packets_to_sniff = 100
         pcap = sniff(count=num_of_packets_to_sniff)

         # rdpcap returns packet list
         ## packetlist object can be enumerated
         print(type(pcap))
         print(len(pcap))
         print(pcap)
         pcap[0]

         <class 'scapy.plist.PacketList'>
         100
         <Sniffed: TCP:62 UDP:27 ICMP:0 Other:11>

Out[10]: <Ether  dst=ff:ff:ff:ff:ff:ff src=00:50:56:c0:00:08 type=IPv4 |<IP  version=4 ihl=5 tos=0x0 len=291 id=48331 flags= frag=
```

This is what you will be greeted with when u access the jupyter lab web environment using the web browser. Our main working file will be **Packet-Analytics.ipynb.** You can see a bunch of controls at the top of the screen. This is to run/stop/modify/delete code on the file.

Now that we have initialised the environment, let's get to using it! The codes on the file will run in snippets so that you can follow through and see the output as the pcap file is being analysed. You just have to keep pressing play [>RUN] up top and you will be able to see the output on the screen!

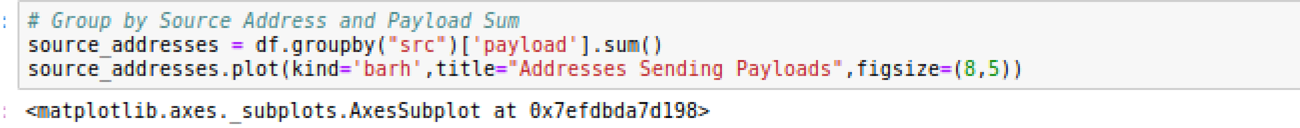
Statistics



```
# Top Source Adddress
print("# Top Source Address")
print(df['src'].describe(),'\n\n')

# Top Destination Address
print("# Top Destination Address")
print(df['dst'].describe(),"\n\n")

frequent_address = df['src'].describe()['top']

# Who is the top address speaking to
print("# Who is Top Address Speaking to?")
print(df[df['src'] == frequent_address]['dst'].unique(),"\n\n")

# Who is the top address speaking to (dst ports)
print("# Who is the top address speaking to (Destination Ports)")
print(df[df['src'] == frequent_address]['dport'].unique(),"\n\n")

# Who is the top address speaking to (src ports)
print("# Who is the top address speaking to (Source Ports)")
print(df[df['src'] == frequent_address]['sport'].unique(),"\n\n")
```

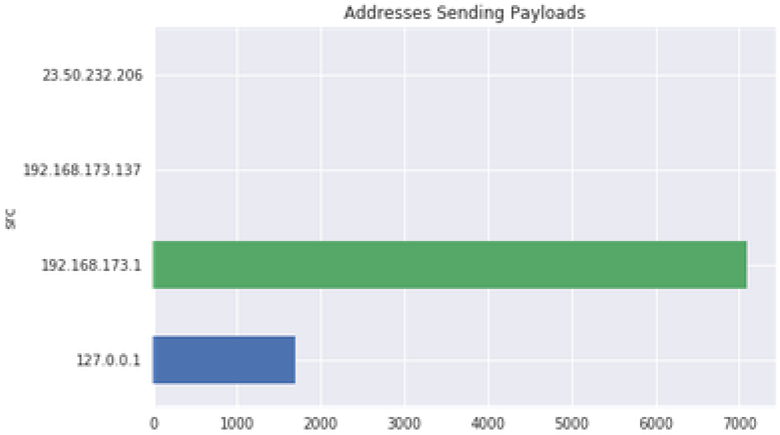
```
# Top Source Address
count                     40
unique                     4
top          192.168.173.1
freq                      30
Name: src, dtype: object


# Top Destination Address
count                       40
unique                       6
top          192.168.173.255
freq                        28
Name: dst, dtype: object


# Who is Top Address Speaking to?
['192.168.173.255' '224.0.0.251' '239.255.255.250']


# Who is the top address speaking to (Destination Ports)
[54915 57621 5353 1900]
```

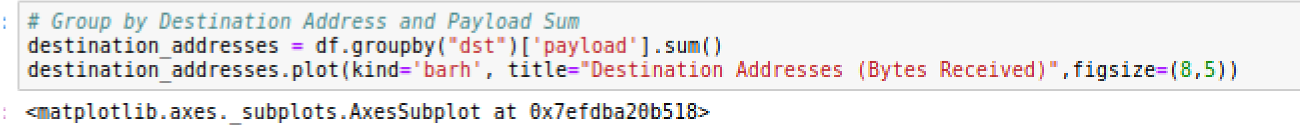
Graphing



```
# Group by Source Address and Payload Sum
source_addresses = df.groupby("src")['payload'].sum()
source_addresses.plot(kind='barh',title="Addresses Sending Payloads",figsize=(8,5))
```

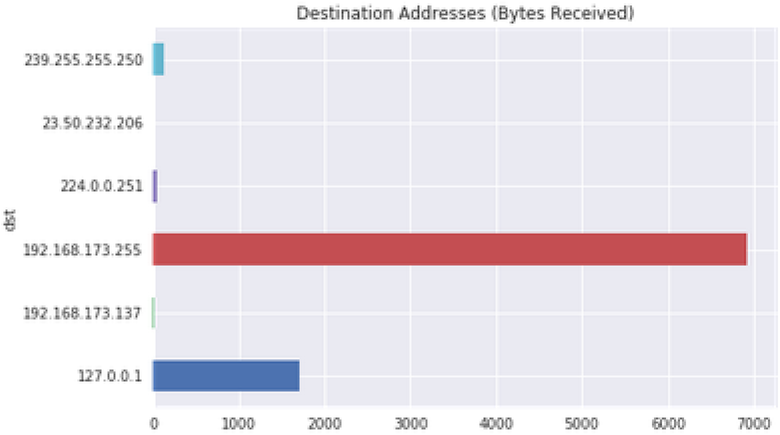
```
<matplotlib.axes._subplots.AxesSubplot at 0x7efdbda7d198>
```

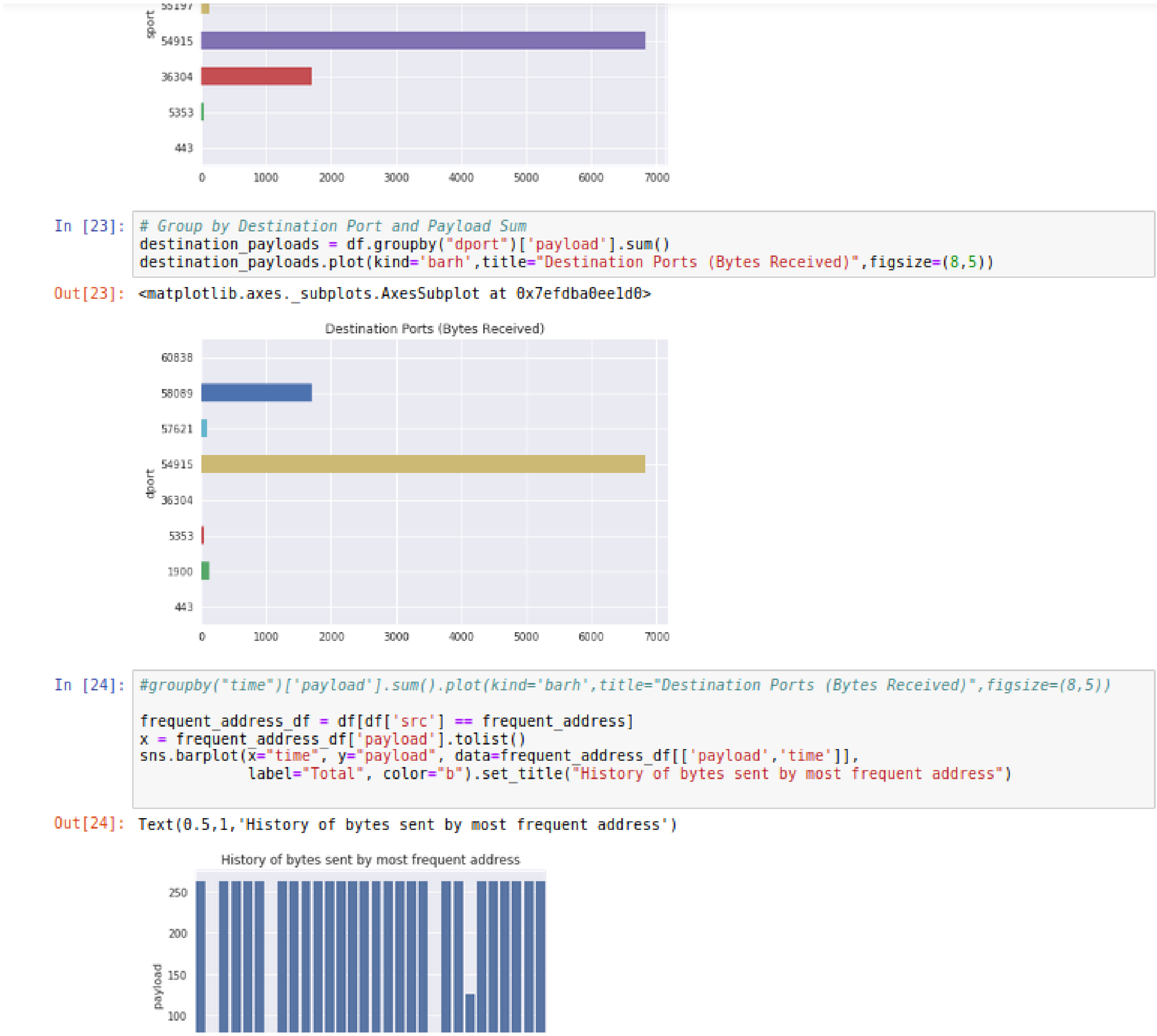




```
# Group by Destination Address and Payload Sum
destination_addresses = df.groupby("dst")['payload'].sum()
destination_addresses.plot(kind='barh', title="Destination Addresses (Bytes Received)",figsize=(8,5))
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7efdba20b518>
```





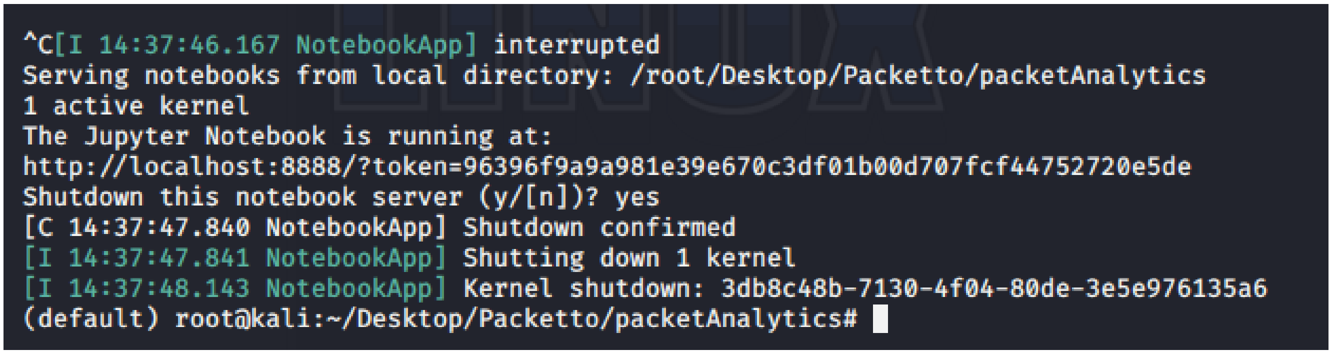
```
In [23]: # Group by Destination Port and Payload Sum
         destination_payloads = df.groupby("dport")['payload'].sum()
         destination_payloads.plot(kind='barh',title="Destination Ports (Bytes Received)",figsize=(8,5))
Out[23]: <matplotlib.axes._subplots.AxesSubplot at 0x7efdba0ee1d0>
```

```
In [24]: #groupby("time")['payload'].sum().plot(kind='barh',title="Destination Ports (Bytes Received)",figsize=(8,5))

         frequent_address_df = df[df['src'] == frequent_address]
         x = frequent_address_df['payload'].tolist()
         sns.barplot(x="time", y="payload", data=frequent_address_df[['payload','time']],
                     label="Total", color="b").set_title("History of bytes sent by most frequent address")
Out[24]: Text(0.5,1,'History of bytes sent by most frequent address')
```

Stop the tool

Head to the command prompt where the jupyter lab notebook is running and do the following.

```
Control + c;   yes.
```



```
^C[I 14:37:46.167 NotebookApp] interrupted
Serving notebooks from local directory: /root/Desktop/Packetto/packetAnalytics
1 active kernel
The Jupyter Notebook is running at:
http://localhost:8888/?token=96396f9a9a981e39e670c3df01b00d707fcf44752720e5de
Shutdown this notebook server (y/[n])? yes
[C 14:37:47.840 NotebookApp] Shutdown confirmed
[I 14:37:47.841 NotebookApp] Shutting down 1 kernel
[I 14:37:48.143 NotebookApp] Kernel shutdown: 3db8c48b-7130-4f04-80de-3e5e976135a6
(default) root@kali:~/Desktop/Packetto/packetAnalytics#
```