**CS-521**

**Project Report**

**DNS A and PTR Records Lookup (Using Python)**

**(Harshala Yadav)**

**DNS A and PTR Records Lookup (Using Python)**

**Introduction**:

The script essentially allows users to lookup forward (A) and reverse (PTR) DNS records when required.

This is essential for network, forensic analysis where an analyst needs to query large datasets containing network logs and analyze the addresses or hostname as per business requirements.

**Implementation**:

The idea for this implementation was based on the principle of simplicity. A lot of the tools like the DIG command in unix are too technical for a naïve user to understand. A beginner interested in network analysis might feel overwhelmed with the array of tools and libraries available online for analysis. A simple command like tool can help the programmer focus on the analysis itself and worry about the complexity later or dive deeper with a more complex tool if necessary. This was my primary motivation to build this tool.

I started with getting the forward and reverse lookup parts given an appropriate input correct. Once I achieved that, proceeded to standardize their format by creating functions. Once, I finished the manual input response, I proceeded to provide the user with an added functionality of using getopt style tool.

Here are some of the libraries I used to build this tool –

**IPy–**https://pypi.org/project/IPy/  
This library helps verify an IP address both IPv4 and IPv6.

**socket –** This library is used to get the host if the host name is provided and find the host if the host address is provided as an input.

**getopt** – This library is similar to the getopt functionality of UNIX wherein a user running the program can specify the options and provide appropriate arguments to the program’s functioning

**sys –** This library is used for command line argument inputs; And exiting the program gracefully in case of failure.

There are several ways to run the program, the standard way is as follows –

* python3 Project\_CS521.py [[-h --help] [-i --ipaddress] [-m --typeoflookup] [-f --fqdn]] / [ipaddress/hostname]

**Source Code:**

from IPy import IP

import sys, getopt, socket

def get\_reverse(val):

val = IP(val)

try:

print("\tThe IP entered is: "+str(IP(val)))

print("\tThis IP is of the \""+str(val.iptype())+"\" type and is a \"IPv"+str(val.version())+"\".")

print("\tThe reverse (PTR) lookup of the IP address entered is: "+str(val.reverseNames()[0]))

try:

print("\tThe hostname for the IP entered is: "+ str(socket.gethostbyaddr(str(val))[0]))

except:

print("\tThe hostname for the IP entered is: Not found/applicable")

except:

print("\tThe IP entered is not a valid IP address")

def get\_forward(val):

print("\tThe hostname entered is: "+str(val))

# print("The forward lookup of the hostname entered is: "+str(val.reverseNames()[0]))

try:

print("\tThe IP address resolution for the hostname entered is: "+ str(socket.gethostbyname(str(val))))

except:

print("\tThe IP address for the hostname entered is not found")

def sec\_resolve(val):

if(val.replace(".","").isnumeric()):

try:

str(IP(val).int()).isnumeric()

get\_reverse(val)

except:

print("Invalid IP address")

elif(val.replace(".","").isalnum()):

get\_forward(val)

else:

print("Invalid hostname")

def resolve\_input(val):

if(not val):

val = input("Enter an IP address or a host name to resolve: ")

sec\_resolve(val)

else:

sec\_resolve(val)

def main(argv):

ipf = ""

opf = ""

fqdn = ""

opts = ""

inpt = ""

print('Project\_CS521.py -i <ipaddress> -m <typeoflookup: A or PTR> -f <FQDN>')

try:

opts, arg = getopt.getopt(argv,"hi:m:f:")

# print("opts: "+str(opts))

# print("arg: "+str(arg))

except:

print('Project\_CS521.py -i <ipaddress> -m <typeoflookup: A or PTR> -f <FQDN>')

try:

# if(not arg):

# print('Project\_CS521.py -i <ipaddress> -m <typeoflookup: A or PTR> -f <FQDN>')

# print("arg:"+str(arg))

if(not opts and not arg):

resolve\_input("")

elif(not opts and arg):

inpt = arg

# print("Hello there: "+str(inpt))

resolve\_input(inpt[0])

else:

for opt, arg in opts:

if opt == '-h':

print('Project\_CS521.py -i <ipaddress> -m <typeoflookup: A or PTR> -f <FQDN>')

sys.exit()

elif opt in ["-i", "--ipaddress"]:

ipf = arg

print("The IP Address entered is : "+str(ipf))

sec\_resolve(ipf)

# elif opt in ["-m", "--typeoflookup"]:

# opf = arg

# print("The type of lookup requested is : "+str(opf))

# sec\_resolve(opf)

elif opt in ["-f", "--FQDN"]:

fqdn = arg

print("The FQDN entered is : "+str(fqdn))

sec\_resolve(fqdn)

except:

sys.exit(2)

if \_\_name\_\_ == "\_\_main\_\_":

main(sys.argv[1:])

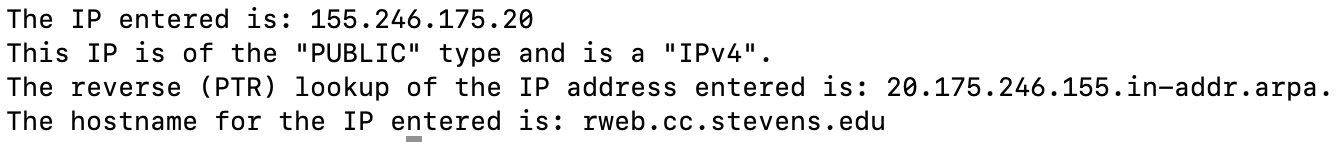
**Program Output and Test Case Analysis:**

Let us see the different ways to run the program –

1. format: python Project\_CS521.py [ipaddress]

* python Project\_CS521.py 155.246.175.20

Output:

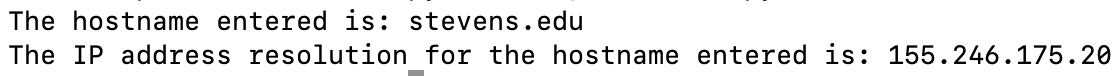


If the user who runs the program inputs the ip address as the only argument with no option, the program does a reverse lookup and returns the ip address details such as the type of the IP address entered (IPv4 or IPv6). Whether the IP address type: PUBLIC, PRIVATE

2. format: python Project\_CS521.py [hostname]

* python Project\_CS521.py stevens.edu

Output:



If the user who runs the program inputs the hostname as the only argument with no option, the program does a forward lookup and returns the ip address.

3. format: python Project\_CS521.py -h

* python Project\_CS521.py -h

Output:

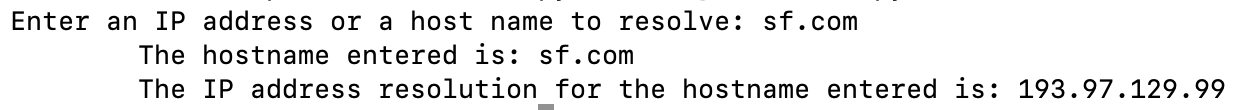


If the user who runs the program inputs the “-h” option as in the standard option to get help on a command in UNIX, the program displays the format in which one can run the program.

4. format: python Project\_CS521.py [no-input]

* python Project\_CS521.py

Output:

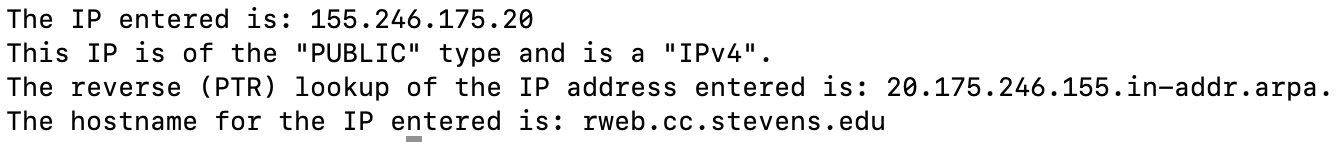


If no argument is passed along with the code, the program prompts the user to enter an IP-address or hostname to be resolved. The program detects whether a hostname or an ip address was input and provides a forward or a reverse lookup automatically.

5. format: python Project\_CS521.py [-i <ip-address>]

* python Project\_CS521.py -i 155.246.175.20

Output:

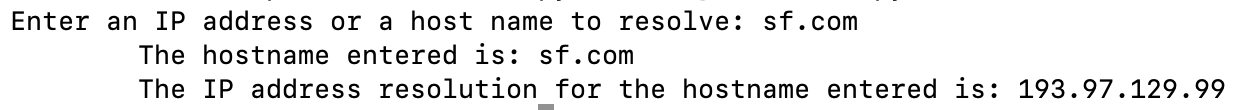


If an ip address is passed as an argument along with the -i option, the program understands that the user would like to perform a reverse lookup. The program fails gracefully and redirects the user to the correct usage if an argument is missing. Here, if an option is selected, the argument is needed.

6. format: python Project\_CS521.py [-f <FQDN>]

* python Project\_CS521.py -f sf.com

Output:

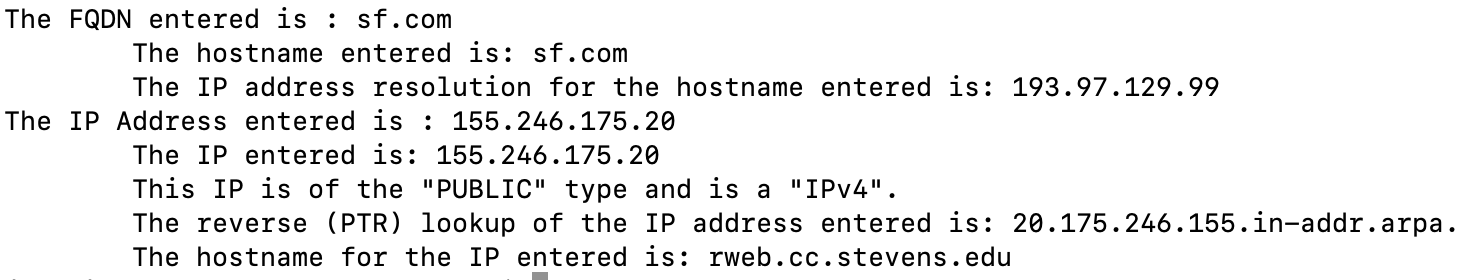


If a domain name is passed as an argument along with the -f option, the program understands that the user would like to perform a forward lookup. The program fails gracefully and redirects the user to the correct usage if an argument is missing. Here, if an option is selected, the argument is needed.

7. format: python Project\_CS521.py [-f <FQDN>] [-I <ip-address>]

* python Project\_CS521.py -f sf.com -i 155.246.175.20

Output:



If a domain name is passed as an argument along with the -f option, along with an ip-address with the -I option, the program understands that the user would like to perform both a forward lookup on the domain name and a reverse lookup on the ip-address provided. The program fails gracefully and redirects the user to the correct usage if any of the arguments are missing. Here, if an option is selected, the respective argument is needed.

**Conclusion:**

The program is succinct and fast enough for an analyst to quickly perform forward and reverse lookups on an ip-address or a domain name. The advantage of this program over existing tools is this tool is quick enough naïve user and also easy to understand. The tool can certainly be improved to include additional functionalities as needed but the idea is to keep it succinct and perform one job correctly – more like a command.