**inputScript.py**

**# -\*- coding: utf-8 -\*-**

import regex

from tldextract import extract

import ssl

import socket

from bs4 import BeautifulSoup

import urllib.request

import whois

import datetime

def url\_having\_ip(url):

**#using regular function**

# symbol = regex.findall(r'(http((s)?)://)((((\d)+).)\*)((\w)+)(/((\w)+))?',url)

# if(len(symbol)!=0):

# having\_ip = 1 #phishing

# else:

# having\_ip = -1 #legitimate

#return(having\_ip)

return 0

sss

def url\_length(url):

length=len(url)

if(length<54):

return -1

elif(54<=length<=75):

return 0

else:

return 1

def url\_short(url):

**#ongoing**

return 0

def having\_at\_symbol(url):

symbol=regex.findall(r'@',url)

if(len(symbol)==0):

return -1

else:

return 1

def doubleSlash(url):

**#ongoing**

return 0

def prefix\_suffix(url):

subDomain, domain, suffix = extract(url)

if(domain.count('-')):

return 1

else:

return -1

def sub\_domain(url):

subDomain, domain, suffix = extract(url)

if(subDomain.count('.')==0):

return -1

elif(subDomain.count('.')==1):

return 0

else:

return 1

def SSLfinal\_State(url):

try:

**#check wheather contains https**

if(regex.search('^https',url)):

usehttps = 1

else:

usehttps = 0

**#getting the certificate issuer to later compare with trusted issuer**

**#getting host name**

subDomain, domain, suffix = extract(url)

host\_name = domain + "." + suffix

context = ssl.create\_default\_context()

sct = context.wrap\_socket(socket.socket(), server\_hostname = host\_name)

sct.connect((host\_name, 443))

certificate = sct.getpeercert()

issuer = dict(x[0] for x in certificate['issuer'])

certificate\_Auth = str(issuer['commonName'])

certificate\_Auth = certificate\_Auth.split()

if(certificate\_Auth[0] == "Network" or certificate\_Auth == "Deutsche"):

certificate\_Auth = certificate\_Auth[0] + " " + certificate\_Auth[1]

else:

certificate\_Auth = certificate\_Auth[0]

trusted\_Auth = ['Comodo','Symantec','GoDaddy','GlobalSign','DigiCert','StartCom','Entrust','Verizon','Trustwave','Unizeto','Buypass','QuoVadis','Deutsche Telekom','Network Solutions','SwissSign','IdenTrust','Secom','TWCA','GeoTrust','Thawte','Doster','VeriSign']

**#getting age of certificate**

startingDate = str(certificate['notBefore'])

endingDate = str(certificate['notAfter'])

startingYear = int(startingDate.split()[3])

endingYear = int(endingDate.split()[3])

Age\_of\_certificate = endingYear-startingYear

**#checking final conditions**

if((usehttps==1) and (certificate\_Auth in trusted\_Auth) and (Age\_of\_certificate>=1) ):

return -1 **#legitimate**

elif((usehttps==1) and (certificate\_Auth not in trusted\_Auth)):

return 0 #suspicious