SCAPY – A Python-based network packet analyzer and more

R. R. Maiti

Intro

- to get the scapy shell
  - $scapy
- To know all supported protocol layers
  - >>>ls()
- To know all supported user commands
  - >>>lsc()
- To know all configurable objects
  - >>>config
- Reading a pcap file
  - >>>a=rdpcap(“/spare/captures/isakmp.cap”)
- Writing in a pcap file
  - >>> wrpcap(“temp.cap”,pkts)
- Using sniff() for reading packets
  - >>> pkts = sniff(offline=”temp.cap”)
  - >>> pkts = sniff(offline=”temp.cap”,count = 1)

Intro

- >>>def arp_monitor_callback(pkt):
  - if ARP in pkt and pkt[ARP].op in (1,2): #who-has or is-at return pkt.sprintf(“%ARP.hwsrc% %ARP.psrc%”)
- >>>sniff(prn=arp_monitor_callback, filter=”arp”, store=0, iface = ‘eth0’)
- >>>sniff(prn=arp_monitor_callback, filter=”arp”, store=0, offline = ‘temp.pcap’)

- To know all the fields in a protocol
  - ls(Proto)
- traceroute
  - >>> traceroute([“www.yahoo.com”,”www.altavista.com” ,”www. wisenut.com”,”www.copernic.com”],maxttl=20)
- Saving a scapy session
  - >>>save_session("session.scapy")
- Loading a scapy session
  - >>>load_session("session.scapy")

Find tcp sessions

- help(sniff)
- sessions()
- sessions().keys()
  - Make it full duplex
  - Full_duplex = sessions()[‘a.b.c.d:p1 > w.x.y.z:p2’]
  - Full_duplex += sessions()[‘w.x.y.z:p2 > a.b.c.d:p1’]
  - sessions(full_duplex).keys()
- res, unans = traceroute(["www.voila.com"],maxttl=20)
- res.graph()
- res.trace3D()

Sniff

- #! /usr/bin/env python
- from scapy.all import *
- a=sniff(count=10)
- a.nsummary()

Sniff and analyze

- from scapy.all import *
- def arp_monitor_callback(pkt):
  - if ARP in pkt and pkt[ARP].op in (1,2):
    - #who-has or is-at
    - return pkt.sprintf("%ARP.hwsrc% %ARP.psrc%")
- sniff(prn=arp_monitor_callback, filter="arp", store=0)

Class Assignment: 1

- Write a function to compute the mean and standard deviation of lengths of the packets of a protocol taken as input and display as a list.

Home assignment 1:

- Find the mean and standard deviation of lengths of the packets of each of the protocols in Network Layer protocol present in the pcap files shared over CMS and store them in dictionary. Return the dictionary to main function.