Some of you have asked for suggestions for your Comp Sec sessional project. I can’t share them publicly through blog post or BSADD lecture(you know why!). So, I am writing this doc to help you to understand some basic ways of implementing the attacks tools.

In almost every attacks of the given list (except: Known Password attack, Dictionary attack) one has to generate a packet using a specified protocol (eg- ARP/ TCP/ DHCP/ ICMP/ DNS). So building the protocol packet is one of the most important tasks for these network attacks.

**How to build these packets?**

To build the packets one has to know the packet format of that protocol. You can get the format from the official RFC link or from the Wikipedia. For example:

* RFC link for ICMP protocol: <https://tools.ietf.org/html/rfc792>
* Wiki link for ICMP: <https://en.wikipedia.org/wiki/Internet_Control_Message_Protocol>

You can build the raw packets from the ground following the protocol description. For example: To build raw ICMP ping message you can follow:

* In C: <https://www.geeksforgeeks.org/ping-in-c/>
* In Python: <https://github.com/samuel/python-ping/blob/master/ping.py>

Though the packets are not too complex to build from scratch (but will take time and debugging), they can be built easily using any packet crafting library. Most of the popular languages have more than one such libraries-

In C/C++:

* Libtins: <http://libtins.github.io/tutorial/>
* libpcap (the most used packet capture library- used in wireshark and tcpdump) : <https://www.tcpdump.org/pcap.html>

In Python:

* Scapy: <https://scapy.readthedocs.io/en/latest/usage.html>

For example, for sending an ICMP redirect message one just has to write:

|  |
| --- |
| eth\_h = Ether(src=src\_mac,dst=dest\_mac) ip\_h = IP(dst=dest\_ip,src=src\_ip) icmp\_h = ICMP(type=5,code=1,gw=gateway\_ip) pkt = eth\_h/ip\_h/icmp\_h  sendp(pkt,iface="eth0") |

* Impacket: <https://github.com/SecureAuthCorp/impacket/> (go through the **examples** directory to understand how to use this library. eg- <https://github.com/SecureAuthCorp/impacket/blob/master/examples/ping.py> )

I think you should be allowed to use these libraries to make the packets (they are not actually very complex libraries/ tools, yet powerful).

**Network setup:**

You can use Virtualbox images, Docker images or VPS servers for setting up the network. For the beginners, Virtualbox will be easier to configure. Please go through the Virtualbox networking modes before configuring the machines. You may like to use lightweight distributions like - Ubuntu mate, ParrotOs, Kali Linux if you are on a low configuration (memory) laptop/pc.

**Report writing:**

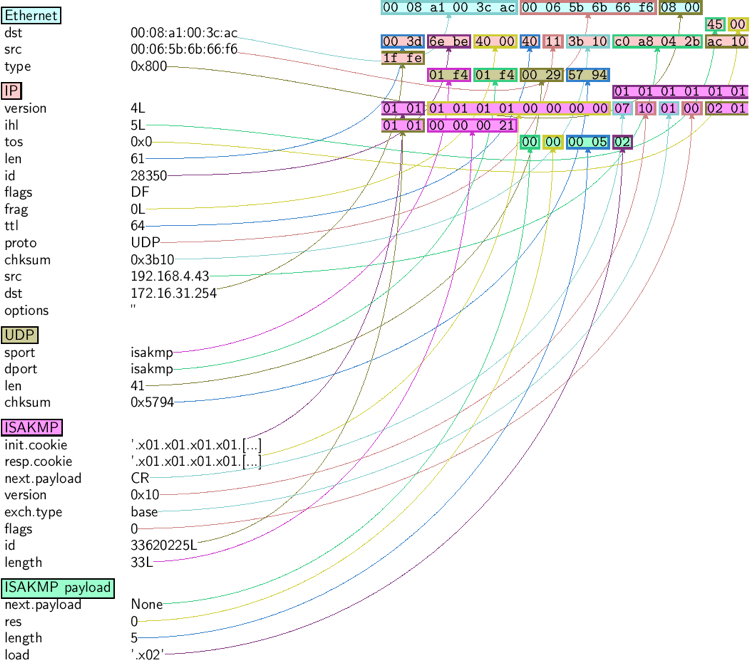
You can go through this report to get some ideas about how to approach:

<https://github.com/deepanshululla/Network-security/raw/master/Attacks%20on%20Layer2%2C3%2C4%20Protocols/projectReport.pdf>

To get **packet details** for the design report you can use a way like [this](https://scapy.readthedocs.io/en/latest/usage.html#graphical-dumps-pdf-ps):

|  |
| --- |
| pkt.pdfdump(layer\_shift=1) |

You will get the full byte by byte mapping of the created packet to add to the report:



Or you can use functions like show() or summary() to print the packet details:

|  |
| --- |
| >> p.show()  Output: ---[ IP ]--- version = 4L ihl = 5L tos = 0x0 len = 39 id = 15489 flags = frag = 0L ttl = 42 proto = ICMP chksum = 0x51dd src = 66.35.250.151 dst = 192.168.5.21 options = '' ---[ ICMP ]---  type = echo-reply  code = 0  chksum = 0xee45  id = 0x0  seq = 0x0 ---[ Raw ]---  load = 'XXXXXXXXXXX' ---[ Padding ]---  load = '\x00\x00\x00\x00' |

To get the **frame details** you can use these tools inside the virtual machines-

* [Wireshark](https://www.wireshark.org/download.html)
* [Tcpdump](https://www.tcpdump.org/#latest-releases)

**More resources:**

* Northeastern University’s CS6740 sessional submissions: <https://github.com/deepanshululla/Network-security>

This repo has following attacks implemented with the corresponding report:

1) ARP cache poisoning

2) ICMP Redirect Attack

3) SYN Flooding Attack

4) TCP RST Attacks on telnet and SSH Connections

5) TCP RST Attacks on Video Streaming Applications

6) ICMP Blind Connection-Reset and Source-Quench Attacks

7) TCP Session Hijacking

* Optimistic TCP ACK attack: <https://github.com/rameshvarun/misbehaving-receiver/blob/master/attackers/opt_ack_attacker.py>
* DHCP starvation attack:
  + <https://github.com/shreyasdamle/DHCP-Starvation-/blob/master/dhcp_starvation.py>
  + <https://github.com/avaiyang/dhcp-starvation-attack/blob/master/dhcp_starve.py>
* DHCP spoofing:
  + <https://github.com/floft/dhcp-spoof>
* ARP cache poisoning:
  + <https://medium.com/@ismailakkila/black-hat-python-arp-cache-poisoning-with-scapy-7cb1d8b9d242>
* Port Scanning:
  + <https://www.pythonforbeginners.com/code-snippets-source-code/port-scanner-in-python/>
  + <https://github.com/davidmerrick/Python-Port-Scanner/blob/master/port_scanner.py>
* MAC table flooding attack:
  + <https://github.com/WhiteWinterWolf/macof.py>
* DNS cache poisoning:
  + <https://github.com/mistahenry/DNS-Cache-Poisoning-with-Scapy>

**Acknowledgement**: I have not verified every link, so please verify them by yourselves. I may have missed some of the libraries or useful links. Please search on your own topic (specifically on Github) to get more related contents and implementations. **Sorry in advance** if there are any mistakes in this doc. **Good luck** with your projects. If any one wants to discuss or work on any Sec related topic with me you will be welcomed :).