AMERICAN UNIVERSITY OF BEIRUT

EECE 655
Internet Security

Assignment #1

Topic: IPv4 Fragment Overlap

Submitted to:
Prof. Imad El Hajj

Prepared by:
Sammy Alawar - 202202056
Omar Hijazy - 202672924

I) Detailed Tools Description:

Attacker:

- **Language/libs:** Python3 & Scapy to control IPV4 fragments manually.
- **Modes:** Raw UDP/IP datagrams, fragmented datagrams, and fragments with overlaps.
- **Purpose:** Sending raw UDP datagrams, then properly fragmented ones, and finally testing intentionally overlapping fragments to observe the victim's response.

Detector:

- **Language/libs:** Python3 & Scapy.
- **Purpose:** inspects incoming IPv4 fragments, and raises alerts when overlapping fragments arrive.

Utility / Analysis Scripts:

- **udp_echo.py**: a simple UDP listener that shows the received payloads.
- **Detect_overlap_verbose.py**: Run the detector on the victim to display every IPv4 fragment seen including its: source, destination, IPID, start, end, offset, and the MF flag.
- **ip_fragment_overlap_attack.py:** Attacker script that crafts and sends overlapping fragments.

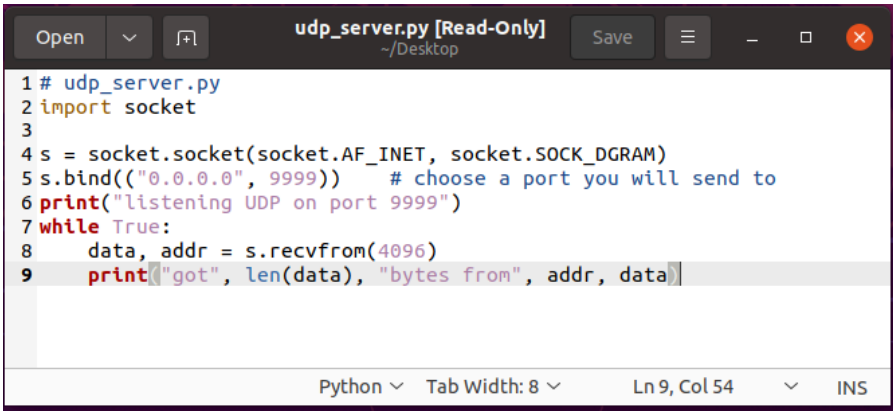
II) How to Run:

Victim (open **two** terminals):

1. **UDP listener** : sudo ./udp_echo.py --port 9999
2. **Overlap detector**: sudo ./detect_overlap_verbose.py --iface enp0s3 --port 9999

Attacker:

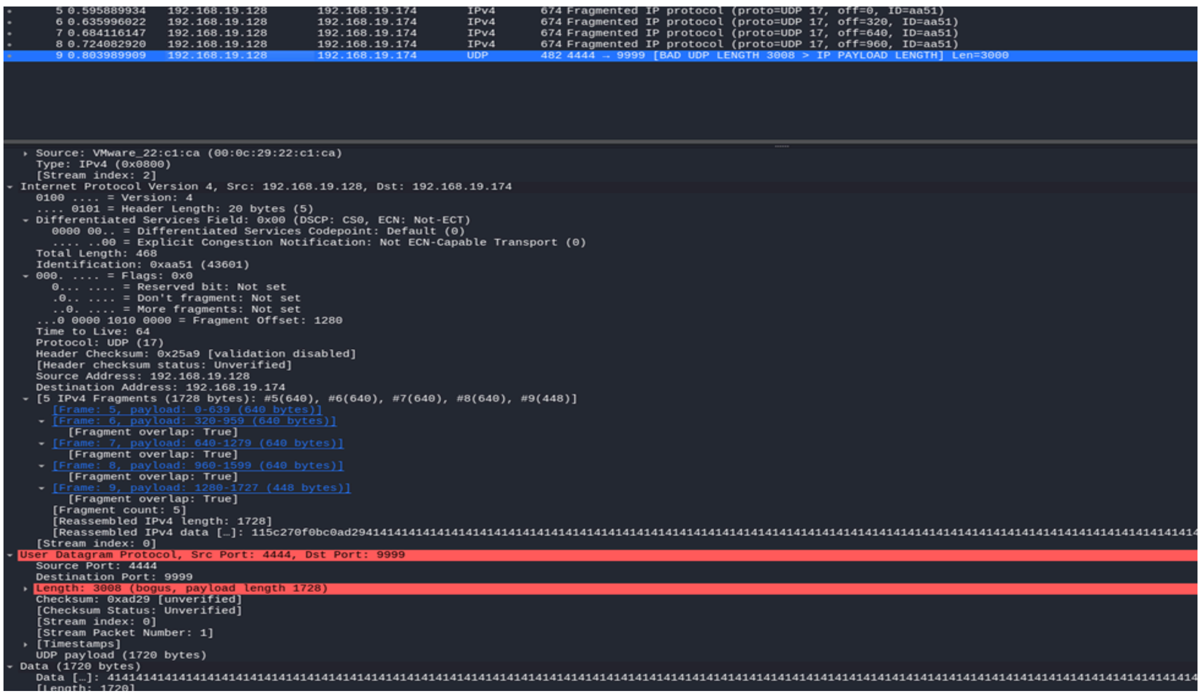
Network Setup: Kali Linux (IP 192.168.19.128) Attacker, and Ubuntu Linux (IP 192.168.19.174) Victim with a running UDP server.



For Brevity, please note that we will only showcase the Multiple Overlapping Fragment attack. Please refer to the README file on our GitHub repo for instructions on how to configure your setup and use the other features/attacks.

Overlapping fragmentation (attack):
Multiple Overlapping Fragments:
chmod +x ip_fragment_overlap_attack.py

sudo ./ip_fragment_overlap_attack.py --dst [Victim IP] --count 1 --payload "$(python3 -c 'print("A"*3000)')" --multipleoverlap --fragsize 640 --id 43601



Note how the 5 fragments (Frames 5 through 9) overlap the right and left half of the two fragments they're in between.

III) Results:

We successfully built an environment consisting of an attacker VM that sends IPv4 packets to the victim VM. Those sent packets can be either normal UDP datagrams, properly fragmented datagrams, or overlapping ones. In all cases, we were able to detect and observe manually "or using Wireshark" the contents of those sent packets, and check if any overlapping was present. We also observed how the OS can react to those overlapping cases where it drops the entire packet down and the UDP server doesn't receive them at all in that case.

IV) Team Contributions:

Sammy Alawar:

- Designed and implemented the **attacker** tool's architecture, wrote the manual fragmentation logic to produce both correct reassembly and intentional overlap, added validation and PCAP output, and verified behavior using a local UDP server and Wireshark.

Omar Hijazy:

- Designed and implemented the **detector** (detector_overlap_verbose.py) to inspect incoming IPv4 fragments.
- Designed a victim-side UDP server (udp_echo.py) to observe successfully received packets.
- Demonstrated the tools in action on a short video presentation.