International Journal of Applied Engineering Research ISSN 0973-4562 Volume 12, Number 20 (2017) pp. 10134-10137

Network Traffic Simulator from Real Time Captured Packets

Venkat Ramana Reddy[1], Mohamed Safwan[2]
[1,2] Student, Department of Computer Science and Engineering, R.V.College of Engineering, Bengaluru, India.
[1]Orcid Id: 0000-0001-7845-3120, [2]Orcid Id: 0000-0002-2831-471X

Deepamala.N[3], Shobha. G[4], Premkumar S.J [5]
[3]Associate Professor, Department of Computer Science and Engineering, R.V.College of Engineering, Bengaluru, India.
[4]Professor and Head, Department of Computer Science and Engineering, R.V.College of Engineering, Bengaluru, India.
[5]Manager, Citrix R&D India Pvt Ltd., Bengaluru, India.
[3]Orcid Id: 0000-0001-8594-2248, [4]Orcid Id: 0000-0001-8533-631X, [5]Orcid Id: 0000-0003-1046-2738

Abstract

The robust networking environments and technology makes it difficult for the analysis and debugging of network problems. There are many network testing tools that capture and replay the packets to help in debugging. But the tools that create an network environment by synchronously simulating the captured packets between two hosts which also has other features like filter, modify and save options make this tool unique.

Keywords: Network Simulator, packet synchronization, capture, replay.

INTRODUCTION

Networking involves exchange of various types of packets between communicating devices. With IOT, cloud communication, technologies like SDN etc. there is a continuous increase in number of protocols and types of packets. During the cases of network or device failure due to unexplained traffic, there is necessity of tools to recreate an environment. To recreate the environment the real time packets captured are replayed and an agent on the other end synchronously responds with response packets. This not only helps the developer to analyse the communication but also test the devices between them. The proposed tool can also generate modified flows for testing and understanding of networks.

EXISTING SYSTEM

A packet simulator or packet builder is a software that generates random packets or allows the user to construct detailed custom packets. Some of the various tools that help in debugging the network are OFRewind [1], wundsam et. al. present a tool that is scalable, multi-granular, record and relay the packets. NS-3 [2] is a popular network simulator. OMNet++ [3] is simulator for networks, queuing networks, performance evaluation etc. Ya Ku et. al. [4] explain packet generation and environment creation for wireless LAN. There are various tools that can capture and replay packets like Scapy [5], PyPacker [6], Libcrafter [7], Nping [8] that are used for packet generation and also spoofing. A comparison of these tools is as shown below in Table 1.

Table 1. Comparison of packet capture and replay tools

Features	**Scapy [5]**	**PyPacker [6]**	**Libcrafter [7]**
Language	Python 2.x/3.x	Python 3.x	C/C++
Version	V2.x	-	V0.3
Protocols supported (HTTP, FTP etc.)	All basic protocols are supported and has the capability to add user defined protocols.	Ethernet, IP, ICMP, TCP, UDP, HTTP, ARP, STP, OSPF, PPP, PPPoE, STP, VRRP, AH, ESP, IGMP, IPX, PIM, AIM, NTP, DHCP, RIP, SCTP, RTP, SIP, TFTP	Ethernet, SLL (Linux cooked-mode capture), ARP, DHCP, DHCP options, IP, IPv6, ICMP, ICMPv6, ICMP extensions, ICMPv6 extensions, TCP, TCP options, UDP and DNS
Scalability	-	Packet parsing from raw bytes is about 50 times faster when compared to scapy.	-

Modification of packets	IP & mac address, TCP, UDP payload	IP & mac address, TCP, UDP payload	IP & mac addresses, UDP payload
Dynamic adaptability: to change network packet	Yes	Yes	Yes
Usage	Python API	Python API	Implemented by using header files in C
Specialities	Support to create and add new protocols	Simple to use than scapy	Multithreading, similar interface to scapy
Problems	May miss packets under heavy load(As listed on scapy homepage)	Timestamp problems	802.11 currently not supported
Operating System/Environment	Windows, Linux based OS	Unix-based OS	Linux based system with autoconf and libtool installed on system
Source code	https://bitbucket.org/secdev/scapy/wiki/Home	https://github.com/mike01/pypacker	https://github.com/pellegre/libcrafter
Licensing	GPLv2	BSD	BSD

The drawback of these tools are:

1. These tools can only replay the given list of packets without considering the sequence of the packets.
2. They do not consider if the packet being sent is request or response.
3. They neither handle the absence of other end nor wait for response
4. The packet sender does not wait for response before sending the next packet in the sequence.
5. They do not allow the user to make modification in the captured packets.

The objective of the proposed work is to capture a real time transaction in the form of log files and use the same while debugging. The captured file is loaded into the proposed tool and the packets are replayed synchronously between the sender and the receiver.

Proposed System

The proposed simulator sends packets synchronously with the receiver. The receiver is an agent running in the receiving end which also has the same packet capture file loaded. Other major functionalities of the proposed tool are:

- The tool has a GUI for all the configuration.
- GUI support to upload trace file and support selection of a particular transaction in the trace file.
- Summary of the trace file uploaded is displayed in the GUI.
- Packet details present in the uploaded file can be edited if required. Modification of options like Source IP, Destination IP, port numbers and some fields in the TCP Header are supported.
- The output file after modification is in the form of modified trace file which can be saved and replayed.

Communication between Sender and receiver is as shown in Figure 1.

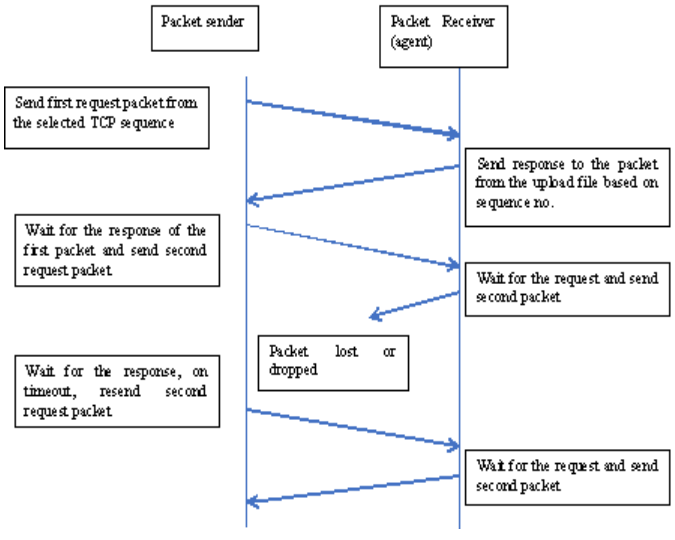

Figure 1: Sequence of packet communication synchronously.

Scapy [5] is chosen as the tool to generate traffic based on uploaded capture file.

Graphical User Interface

A web based interface has been developed for user interaction. The stack runs on Django, a python framework for web applications. MySQL is used for storing packet data contents for editing. Other software used for the development of the simulator are:

- Django – 1.9.1
- Python – 2.7
- Scapy
- MySQL-python
- MySQL-server
- MySQL-client
- Any web browser

The web application has the following features:

- Upload of any number of trace files.
- Interface to open a trace file and display its contents. The trace file contents are displayed in a tabular manner.
- The interface allows editing of fields of packets like IP, Port number etc.
- The interface allows editing an existing .pcap file (trace file) and saving a modified trace file.
- Functionality to replay the packets between two hosts.
- The interface displays the packets that are sent and received in a tabular form.

The figure 2 below shows the screen shot of the home page of the tool. Figure 3 shows content of the selected trace file. The table displays the Source IP address, Destination IP address, Protocol and Length of every packet in the trace file. Each packet is provided with two options in the action column: Edit packet and Delete Packet. When edit packet is clicked for a packet, a new page opens which displays all the editable fields of the packet as shown in figure 4. This figure shows Source MAC address, Destination MAC address, Source IP address and Destination IP address that can be edited.

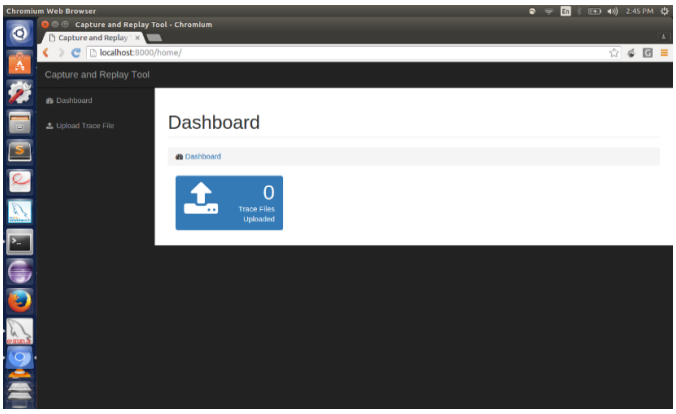

Figure 2: Homepage of Simulation Tool

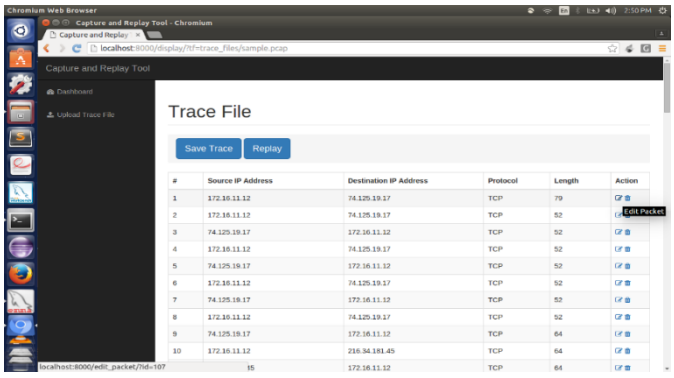

Figure 3: Screenshot of display of trace file contents

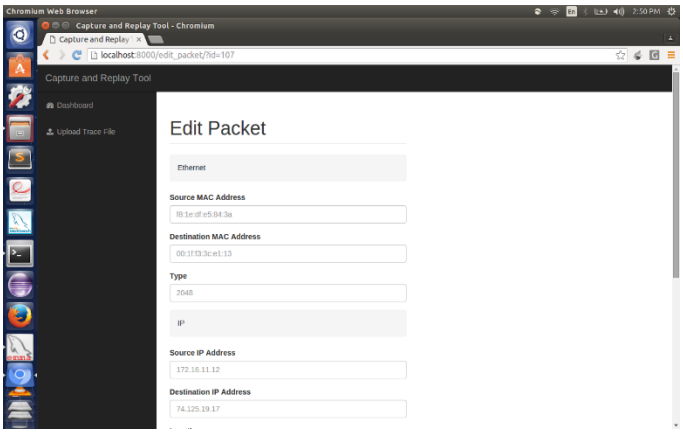

Figure 4: Screenshot of editing a packet

Packet Replay and Synchronization

The following steps were adopted to achieve synchronization of replayed packets between client and server (sender and receiver):

1. Load the trace file into the GUI. E.g. test.pcap
2. Split the loaded trace file (test.pcap) into multiple files based on sessions
3. Select the session that has to be replayed.

4. Load the selected session pcap file into Sender and Receiver (with agent)
5. Configure IP1 of session in host 1 and IP2 of session in Host 2
6. Send packet 1 from host 1 to host 2 from the loaded pcap file
7. Host 2 wait (sniff) for packet 1 and replays packet 2 from loaded pcap file
8. Host 1 sniff packet from Host 2 and replay next packet as shown in Figure 1.

RESULTS

The developed simulation tool was tested for various .pcap files with various kinds of packets. Some were downloaded from the internet and some were provided by Citrix Systems. The tool successfully generated requests and response from agent for all capture files. The screenshot in figure 5 shows how the packets are sent by the sender. The status specifies status of the packet being sent. Source IP and Destination IP has to be specified to distinguish between the source and the destination.

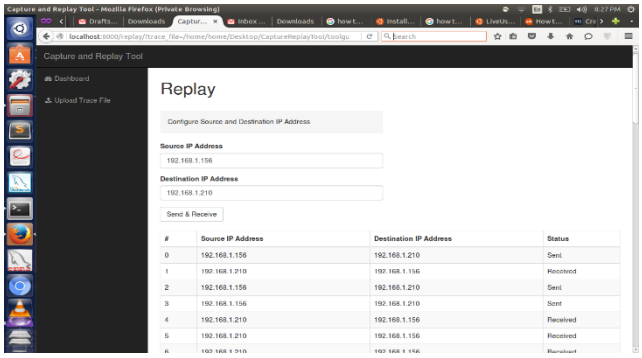

Figure 5: Screenshot of packet replay on Host 1

The screenshot figure 6 shows the receiver receiving requests and sending response packets synchronously to the sender.

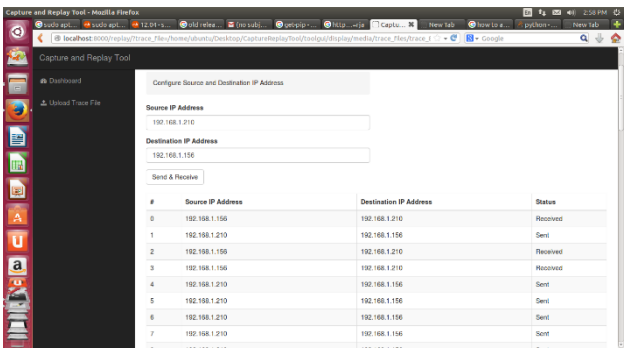

Figure 6: Screenshot of packet replay on Host 2

CONCLUSION

Testing and Debugging are the most challenging part of any Network environment. Simulation tools help the developers and testers create an environment which can be analyzed later. The simulation tool developed not only replay the captured real time packets but also create the session environment by synchronously replaying the response by a agent in the receiver. All the fields in the packet can be modified and saved for future use. This makes the tool robust. The GUI helps the user with ease of configuration.

REFERENCES

[1] A. Wundsam, D. Levin, S. Seetharaman, and A. Feldmann, “OFRewind: enabling record and replay troubleshooting for networks,” in Proceedings of the 2011 USENIX conference on USENIX annual technical conference, ser. USENIXATC’11. Berkeley, CA, USA: USENIX Association, 2011, pp. 29–29.

[2] Riley, George F., and Thomas R. Henderson. "The ns-3 network simulator." Modeling and tools for network simulation (2010): 15-34.

[3] Varga, Andras. "Omnet++ user manual." OMNeT++ Discrete Event Simulation System. Available at: http://www. omnetpp. org/doc/manual/usman. html (2010).

[4] Ku, Chia-Yu, et al. "Real traffic replay over wlan with environment emulation." Wireless Communications and Networking Conference (WCNC), 2012 IEEE. IEEE, 2012.

[5] Biondi, P. "Scapy, a powerful interactive packet manipulation program." (2010).

[6] https://pypi.python.org/pypi/pypacker/2.8 (23/9/2017)

[7] https://code.google.com/p/libcrafter/ (23/9/2017)

[8] Binnie, Chris. "Nping." Linux Server Security: Hack and Defend: 49-58.

[9] Cheng, Yuchung, Urs Hölzle, Neal Cardwell, Stefan Savage, and Geoffrey M. Voelker. "Monkey See, Monkey Do: A Tool for TCP Tracing and Replaying." In USENIX Annual Technical Conference, General Track, (2004) 87-98.