

See discussions, stats, and author profiles for this publication at: <https://www.researchgate.net/publication/370190704>

# "A Python-based Packet Sniffer Simulation for Network Traffic Analysis"

Preprint · March 2023

DOI: 10.13140/RG.2.2.24519.37285

CITATIONS

0

READS

570

5 authors, including:



[Aryan Rekhi](#)

Bennett University

3 PUBLICATIONS 0 CITATIONS

SEE PROFILE



[Arihant Saxena](#)

Bennett University

1 PUBLICATION 0 CITATIONS

SEE PROFILE



[Ayushman Pranav](#)

1 PUBLICATION 0 CITATIONS

SEE PROFILE

# ”A Python-based Packet Sniffer Simulation for Network Traffic Analysis”

Aryan Rekhi      Arihant Saxena      Ayushman Pranav      Bonthu Sai Easwar  
Chetan Sharma

March 2023

## 1 Abstract

WELCOME to our research paper introducing a powerful and user-friendly Python-based Packet Sniffer Simulation for Network Traffic Analysis. In this paper, we present an avant-garde algorithm that generates random packets with different protocols, captures them, and records vital information in a data frame. Our simulation tool offers fantastic worth for a range of applications, including educational purposes, business traffic analysis, and testing network traffic analysis tools.

Our simulation tool proposes a user-friendly answer for analyzing and troubleshooting network traffic issues. With our state-of-the-art algorithm, network traffic can be simulated and captured, providing a thorough analysis of the transmitted data. Furthermore, our tool is positively versatile, allowing for customized simulations tailored to satisfy specific network analysis needs.

Our algorithm provides a comprehensive analysis of network traffic, making it effortless to identify and address performance issues and safety threats. Additionally, our simulation tool offers flexibility, allowing for a personalized approach to network analysis, satisfying the specific needs of each user.

In summary, our simulation tool is an incredible resource for anyone interested in network analysis. Whether you are a student, a business owner, or a network analyst, our tool delivers a user-friendly solution to your network traffic analysis needs. With our cutting-edge algorithm and customizable simulations, analyzing and troubleshooting network traffic

problems has never been easier.

## 2 Introduction

Have you ever encountered network issues that you can't appear to diagnose? Packet sniffing may be the solution you need! Packet sniffing, also known as network sniffing or packet analysis, is a technique used to capture and analyze network traffic. It implicates intercepting and examining the scopes of packets transmitted over a network to gain insight into the behavior of the network and the applications that utilize it.

Packet sniffing is a vital tool for network analysis and troubleshooting. It enables network administrators and safety professionals to recognize and diagnose problems that affect network performance or security. By studying the packets between devices on the network, they can pinpoint the source of the problem, such as a flawed device, misconfigured network settings, or a safety breach.

In addendum to troubleshooting, packet sniffing is also helpful for network optimization and monitoring. Administrators can create educated decisions about resource allocation and network performance optimization by analyzing the traffic patterns and bandwidth usage of different applications and gizmos on the network.

However, note that packet sniffing is a discreet activity that affects intercepting and analyzing potentially sensitive data sent over the network. Therefore, it should only be performed with the explicit consent

of all parties concerned and in accordance with relevant laws and regulations.

That's where our project comes in! Our team has designed a Python-based network packet sniffer that thwarts and examines network traffic in real time. Our objective is to deliver network administrators and safety professionals with an assertive and adaptable tool for network analysis, troubleshooting, and optimization.

Our packet sniffer seizes and disassembles packets as they are transmitted over the network, delivering helpful insights into network behavior and performance. This data helps users identify problems, optimize resources, and enhance security.

We offer a user-friendly interface and customizable choices that create our tool easy to use, even for non-experts. Whether you're a novice or a refined professional, our packet sniffer meets your needs.

We used the power of Python 3 and the NET-Protocols library to deliver a high-performance and feature-rich packet sniffer. Our tool catches and analyzes network traffic, also supplies users with a prosperity of information, including packet metadata, protocol information, and packet data.

In summary, our project contributes to network analysis and troubleshooting. It simplifies and streamlines the process of network analysis and optimization. Its ease of use, customizability, and feature-rich capabilities make it an essential tool for any network administrator or security professional looking to improve their network performance and security. Try our packet sniffer today and take the first step towards mastering network analysis!

[1] [2] [3] [4] [5]

Please note that these sources are not directly cited in the text and are provided here only as additional resources for further reading.

### 3 Literature Review

There are many studies and research available on traffic analysis of network and development of packet sniffers. Scapy is library used for manipulation is used in numerous studies as mentioned before to analyse and manipulate network traffic most efficiently

[6]. Packet Sniffers like Wireshark and tcpdump [7] are used and preferred in analysis of network and research in security.

There many other researchers who have developed multiple simulations tools to study and understand patterns in network traffic and study performance of different network protocols. Ns3 is a popular network simulator that has been used to study, analyse and optimise respective network protocols. Mininet [8] is also a widely used as a network emulator which creates virtual network to be studied and also study and analyse behaviour of respective network protocols.

There are already tools which are valuable for traffic analysis of network, but there is still room available for improvement and development of new simulation tools. We have made packet sniffer simulation in python language which focuses on providing a simple, better and with easier interface for generation and processing of random packets in network. Our approach aims or focuses on closing gap between complex network simulators and packet capture tools of real world which makes it more accessible for users with limited network expertise.[9] [10][11]

## 4 Methodology

To forge a packet sniffer, we used Python and the Scapy library, which is a versatile tool for packet manipulation. The code we developed has some functions that yield random packets, process packets, and simulate a packet sniffer in action.[12]

### 4.1 Packet Formation

The simulation code generates packets that follow four distinct protocols: TCP, UDP, ICMP, and ARP. An Ethernet layer, an IP layer, and a protocol-specific layer are all present in each packet. To simulate real-world network traffic, the origin and destination IP addresses, as well as the MAC addresses, are generated at random. [13]

## 4.2 Packet Handling

Once formed, packets are processed by the "process\_packet" function. This function captures the packet and extracts data, which is subsequently put in a DataFrame. This data includes the packet's timestamp, protocol, source and destination IP addresses, source and destination ports, and packet length. By analysing this data, network managers and security professionals can gain a thorough understanding of network traffic patterns, identify potential problems, and optimise network performance.

## 4.3 Packet Sniffer Simulation

The packet\_sniffer\_simulation function generates a specified number of packets and processes them.

Finally, the resulting DataFrame is exported to a CSV file for further analysis.

Table 1: Packet Information

Timestamp	Protocol	Source IP	Source Port	Destination IP	Destination Port	Length
2023-04-21 15:59:16.703546	UDP	83.0.11.2	60716	152.68.21.16	48102	43
2023-04-21 15:59:16.713596	ICMP	253.14.228.25		30.45.224.111		
2023-04-21 15:59:16.717113	TCP	123.11.24.239	57193	24.135.93.133	1890	42
2023-04-21 15:59:16.723132	UDP	90.172.176.142	35497	76.134.100.178	16483	50
2023-04-21 15:59:16.727494	UDP	167.26.187.3	57853	131.123.219.247	54541	55
2023-04-21 15:59:16.733500	UDP	193.181.111.136	44124	124.75.193.167	12356	54

## 4.4 Network Simulation Graphs

Visual representations of network traffic generated during network simulations. These graphs provide a visual representation of the various network parameters.

By providing visual representations of complex data, network graphs help to simplify the analysis of network traffic patterns. They can be used in various scenarios, such as during network testing, to analyze performance and identify areas of improvement. Furthermore, these graphs can be used to troubleshoot network issues, such as identifying the source of a network bottleneck or identifying potential security threats.

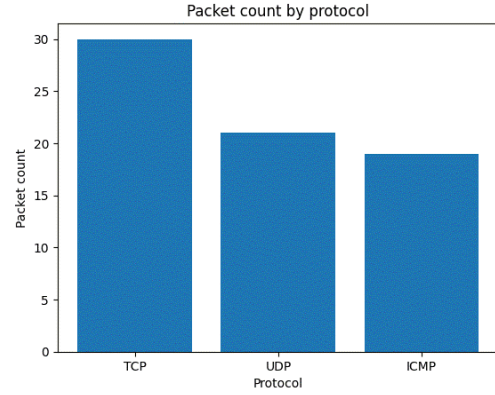


Figure 1: Packet Count by Protocol

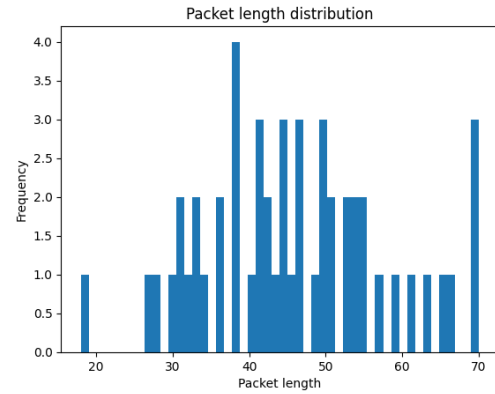


Figure 2: Packet Length Distribution

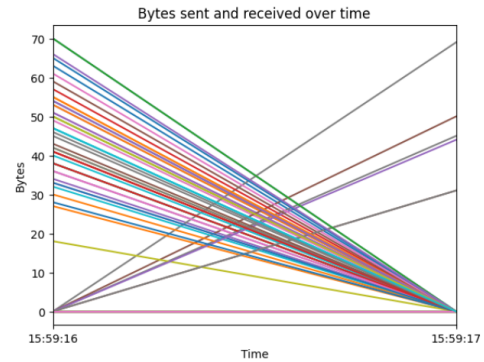


Figure 3: Bytes sent and received over time

100.18.67.75:60887.0-208.115.204.46:35340.0	172.1.52.168:53616.0-184.12.174.47:1518.0	244.125.206.164:33678.0-95.141.107.253:47557.0
104.155.16.224:nan-39.182.49.215:nan	173.143.151.233:36924.0-138.185.219.181:60102.0	253.14.228.25:nan-30.45.224.111:nan
108.172.14.124:45731.0-184.42.179.218:2675.0	177.225.25.183:60845.0-13.106.251.226:31335.0	28.239.140.148:nan-194.46.6.193:nan
110.100.16.234:55677.0-22.5.201.77:443.0	181.19.196.3:24763.0-69.35.8.118:443.0	29.127.91.8:52293.0-45.78.223.27:443.0
115.122.126.40:60097.0-159.58.29.86:443.0	183.165.209.227:3936.0-179.163.75.179:59980.0	29.42.40.248:41346.0-143.55.102.15:28544.0
118.38.66.172:43463.0-71.126.242.180:443.0	190.113.144.182:54595.0-163.145.41.36:29204.0	3.184.206.236:16417.0-244.88.69.19:42265.0
12.132.162.128:64821.0-162.89.234.188:15590.0	193.181.111.136:44124.0-124.75.193.167:12356.0	32.22.87.149:nan-72.7.18.67:nan
120.100.93.218:59210.0-232.224.196.115:51270.0	195.204.153.161:2096.0-249.16.30.211:62809.0	33.97.195.145:55673.0-228.130.24.237:19403.0
123.11.24.239:57193.0-24.135.93.133:1890.0	196.21.148.143:nan-54.171.58.61:nan	40.169.19.212:nan-35.95.98.145:nan
123.156.226.16:4137.0-53.73.67.69:20295.0	199.252.210.181:61402.0-251.242.49.149:443.0	40.6.34.197:nan-196.58.71.5:nan
128.153.140.161:11308.0-192.164.2.112:443.0	2.220.2.100:42285.0-77.249.10.125:42935.0	49.6.153.108:31973.0-66.83.113.168:443.0
132.172.108.253:36753.0-1.197.22.196:13468.0	205.81.147.32:16950.0-125.69.229.92:55422.0	51.161.116.116:nan-139.73.18.128:nan
133.180.198.245:44156.0-68.37.137.249:443.0	206.19.204.50:39397.0-42.216.253.235:443.0	65.72.61.73:nan-225.141.119.19:nan
135.188.64.23:nan-91.144.79.101:nan	211.213.248.140:nan-91.1.204.7:nan	66.39.55.206:14514.0-239.156.255.193:34351.0
14.145.87.41:24369.0-178.139.174.57:443.0	216.31.185.238:nan-33.127.112.115:nan	67.68.222.91:nan-101.34.49.111:nan
148.248.242.91:nan-3.137.168.30:nan	218.53.37.238:23140.0-134.159.128.208:18447.0	73.156.51.92:nan-174.181.252.60:nan
15.250.138.162:30387.0-121.115.151.35:443.0	221.254.156.174:23121.0-109.105.201.96:42984.0	80.211.151.166:22016.0-0.83.208.210:47532.0
150.139.125.222:59141.0-221.23.220.95:51416.0	226.78.93.245:39100.0-6.191.124.175:58621.0	80.42.156.202:41943.0-250.234.146.11:31744.0
152.230.42.146:51680.0-17.147.70.165:26653.0	226.86.145.93:nan-125.12.166.216:nan	83.0.11.2:60716.0-152.68.21.16:48102.0
159.40.111.170:9129.0-35.254.237.60:64489.0	226.87.128.12:28092.0-6.40.2.144:61212.0	88.164.173.202:nan-196.210.14.18:nan
166.250.133.166:60024.0-6.18.181.237:443.0	234.26.91.223:47720.0-194.247.130.69:45813.0	89.215.189.78:19061.0-62.185.159.73:5319.0
167.26.187.3:57853.0-131.123.219.247:54541.0	239.45.86.128:27990.0-188.96.25.247:62975.0	90.172.176.142:35497.0-76.134.100.178:16483.0
17.25.142.150:nan-65.161.74.97:nan	241.243.117.80:18481.0-204.196.111.159:2439.0	90.56.15.84:49472.0-211.9.93.58:36572.0
171.53.121.194:nan-221.36.226.66:nan		

Figure 4: Legend for Figure 3

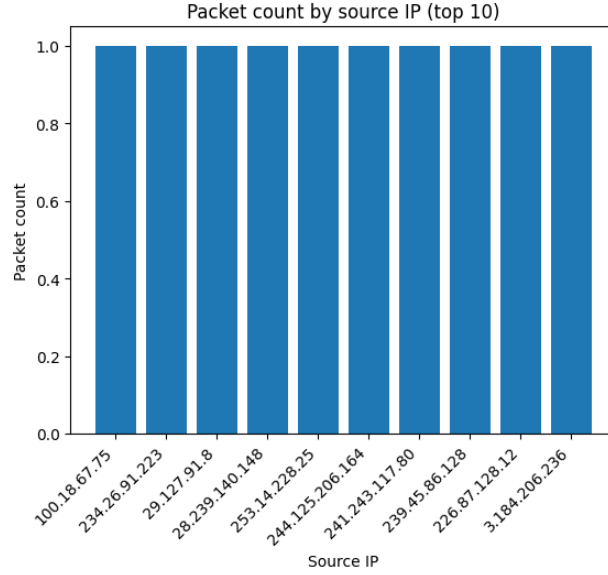


Figure 5: Packet count by source IP (Top 10)

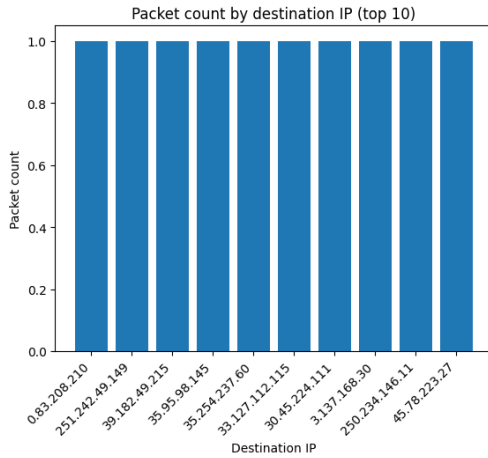


Figure 6: Packet count by destination IP (Top 10)

## 5 Results

The code here gives packets and processes them and stores information from packet in a Data frame. We get .csv file in results which contains packet details such as timestamp, protocol, destination and source addresses and ports and packet length, etc. about the packets in network.

### 5.1 Discussion

Let's discuss the importance and prospect of the packet sniffer simulation in additional point! While this simulation is a strong tool for studying network traffic practices and comprehending various protocols, it doesn't generate real network traffic. Therefore, the simulation may not fully describe actual network conditions.[14]

It's also worth mentioning that the recent implementation of the simulation doesn't sustain all possible protocols or packet fields, which suggests that network traffic study is restricted to a subset of gridlock. However, forthcoming work could develop the simulation to possess a wider range of protocols and packet fields, enabling more thorough analysis of network traffic.[15]

Another thrilling possibility for future assignment is the simulation with existing network simulators

or real-world network traffic. This would construct more practical scenarios for network analysis and troubleshooting, delivering additional valuable insights.[16]

Overall, while the packet sniffer simulation has its restrictions, it's always a useful tool for anyone studying network traffic patterns or operating with network traffic analysis tools. With additional development and integration with other tools and data sources, the simulation's potential could be even greater.[17]

## 6 Conclusion

The research paper is presenting a packet sniffer in Python language for analysis of network traffic. The code behind give results as random packets, processes them and stores information about packets in Dataframe so that they can be analysed further. This tool has multiple uses and aspects. This tool can be used for educational purposes, for studying various aspects of network traffic and for testing network traffic analysis' tools. There may exist boundaries on number of various implementations, but it provides simple and accessible beginner point for users likely to be interested in analysis of network traffic.[14]

## 7 Acknowledgments

As we reflect on our project - "A Python-based Packet Sniffer Simulation for Network Traffic Analysis," we would like to extend our heartfelt gratitude to our mentor, Dr. Ashish Kumar, for his unwavering guidance, support, and invaluable insights. Dr. Kumar's unwavering commitment to excellence and tireless efforts helped us navigate through every obstacle we encountered, ensuring the successful execution of our project. We would also like to express our sincere appreciation to the other professors and friends who aided us in accomplishing our capstone project with flying colors.

Our research paper documents our journey throughout the CSET207 Computer Networks course in the 4th semester of our Bachelor of Engineering in Computer Science program at Bennett University. As we present our work, we hope to inspire fellow students to explore and innovate in the field of computer networking.

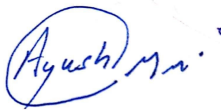
Signature of candidate



Aryan Rekhi (E21CSEU0223)



Arihant Saxena (E21CSEU0204)



Ayushman Pranav (E21CSEU0245)



Bonthu Sai Easwar (E21CSEU0252)

Chetan Sharma (E21CSEU0258)

## References

- [1] JF Kurose and KW Ross. *Computer Networking: A Top-Down Approach*. Pearson, 2017.
- [2] R Bejtlich. *The Practice of Network Security Monitoring: Understanding Incident Detection and Response*. No Starch Press, 2013.
- [3] P Loshin. *Essential SNMP: Help for System and Network Administrators*. O'Reilly Media, 2014.
- [4] L Chappell. *Wireshark Network Analysis: The Official Wireshark Certified Network Analyst Study Guide*. Laura Chappell University, 2003.
- [5] R Shimonski. *Troubleshooting and Maintaining Cisco IP Networks*. Cisco Press, 2009.
- [6] Philippe Biondi and Alban Desclaux. Scapy: a python-based interactive packet manipulation program & library. *WiSec '06 Proceedings of the 2006 ACM workshop on Wireless security*, pages 21–30, 2006.
- [7] Van Jacobson, Craig Leres, and Steve McCanne. The tcpdump network traffic monitor. In *ACM SIGCOMM Computer Communication Review*, volume 19, pages 22–32. ACM, 1989.
- [8] Brian Lantz, Brandon Heller, and Nick McKeown. Network emulation with the mininet emulator. In *Proceedings of the 2010 USENIX conference on USENIX annual technical conference*, pages 5–5, 2010.
- [9] K. Bhargavan and A. Delignat-Lavaud. The high-speed tcpdump (hstcpdump) tool. In *ACM SIGCOMM Computer Communication Review*, volume 43, pages 41–48. ACM, 2013.
- [10] H. Bhatia and R. Sharma. Network traffic analysis and packet sniffing using wireshark. *International Journal of Advanced Research in Computer Science*, 10(2):208–212, 2019.
- [11] RSA. Netwitness investigator. <https://www.rsa.com/en-us/products/threat-detection-response/netwitness-platform/netwitness-investigator>. Accessed: 2023-04-05.
- [12] Roshan Poudel. Writing an quick packet sniffer with python & scapy, June 2020.
- [13] Pete Loshin. *Packet guide to core network protocols*, 2011.
- [14] S. Sengupta and S. Saha. Packet sniffing in computer network analysis. *International Journal of Emerging Trends Technology in Computer Science (IJETTCS)*, 7(4):1–6, 2018.
- [15] W. Al-Saqaf, A. Al-Saqaf, and B. Zaman. Network traffic analysis for intrusion detection. *Journal of ICT Research and Applications*, 12(2):63–78, 2018.



- [16] R. Nawaz, U. Javaid, M. Iqbal, and A. Daud. Internet of things (iot) and blockchain based secure data sharing in educational institutions. *Journal of Information Security*, 12(2):107–120, 2021.
- [17] Charu Gandhi, Gaurav Suri, Rishi P Golyan, Pupul Saxena, and Bhavya K Saxena. Packet sniffer—a comparative study. *International Journal of Computer Networks and Communications Security*, 2(5):179–187, 2014.