JOURNAL OF SCIENCE TECHNOLOGY AND EDUCATION 8(2), JUNE, 2020
ISSN: 2277-0011; Journal homepage: www.atbuftejoste.com

Analysis of UDP Traffic Norms through Packet Sniffing on Peer-to-Peer Networks

Ahmad Musa
Department of Software Engineering,
Bayero University Kano.

ABSTRACT
A significant obstacle with current peer-to-peer (p2p) traffic monitoring and analysis involves the large number of newly emerging p2p applications possessing more intricate communication structures and traffic patterns than the traditional client-server applications. The volume of traffic generated by these applications, such as uTorrent, BitTorrent, eMule, Ares, etc., is stated to be well over half of the total internet traffic. The dynamic use of port numbers, multiple sessions, and other smart features of these applications complicate the characterization of current p2p traffic. Transport-level traffic identification is a preliminary but required step towards traffic characterization, which this article mainly addresses. The traditional traffic identification methods are mostly based on well-known port numbers. These methods are not appropriate for the identification of emerging P2P applications. This paper proposes using a packet sniffer to identify the current p2p traffic. We categorized the most current used p2p network application based on their user datagram protocol (UDP) information to analyze the traffic patterns. Using this categorization, we found that UDP provided the most features that guarantee data integrity.

ARTICLE INFO
Article History
Received: January, 2020
Received in revised form: April, 2020
Accepted: May, 2020
Published online: June, 2020

KEYWORDS
UDP, Traffic, Network, Packets, Peer-to-Peer

INTRODUCTION

Over the past years, peer-to-peer (P2P) file-sharing has continuously evolved to represent a challenging component of global internet traffic. The amount of P2P traffic is sufficiently dominant on some applications to incest increased local peering among network administrators to observable yet unquantified effect on the global internet traffic and routing operation, not to mention competing for market dynamics (Liberatore et al., 2010). Despite this evolution, reliable profiling of P2P traffic remains obscure. We no longer enjoy the fleeting privilege of first-generation P2P applications, which was comparatively easily classified due to its use of specific port numbers. Emerging P2P networks tend to alter their generated traffic deliberately to circumvent both filtering firewalls and legal issues most emphatically articulated by the various government agencies. Most P2P applications now run on top of custom-designed proprietary protocols, nonstandard, but current P2P clients can readily run on any port number, even (Hypertext Transfer Protocol) HTTP's port 80 (Dabir & Matrawy, 2007).

These circumstances hint to a frustrating conclusion: robust identification of p2p traffic is only attainable by examining user information. Yet packet capture and analysis pose a set of often insuperable methodological landmines: privacy, legal, technical, and financial hindrances abound, and overcoming the challenges leaves the task to inadequate government regulations leading to a growing number of poorly documented P2P applications (Das & Tuna, 2017). While reviewing the

Corresponding author: Musa, A. asmusa.se@buk.edu.ng *Dept. of software Engineering, Bayero University, Kano, Nigeria.*

JOURNAL OF SCIENCE TECHNOLOGY AND EDUCATION 8(2), JUNE, 2020
ISSN: 2277-0011; Journal homepage: www.atbuftejoste.com

literature, we found the closest investigation to our research that analyses transmission control protocol (TCP) packets while relying on passive methods of studies such as false-positive identification, which are prone to a variety of errors (Bauer et al., 2009). To mitigate the potential for false-positive peer identification, the authors investigated the feasibility of using active methods to monitor huge BitTorrent swarms.

Another exciting study found was a brief review of the characteristics of current P2P networks. By observing these networks' behaviours, they propose some heuristic rules for identifying the first uploader of a shared file (Leong et al., 2010). The authors concluded that the rules had been demonstrated to apply to some simulated cases. The study also aimed at providing a foundation for the future development of investigations in P2P file-sharing networks. This paper uses an FPGA-based embedded software application, a forensic tool they developed, to identify and track shared contraband digital files using the BitTorrent protocol (Schrader et al., 2009). The system uses TCP to inspect each packet on the network for a BitTorrent Handshake message, which extracts the "info hash" of the file being shared, after comparing the hash against a list of known contraband files and, in the case of a match, adds the message to a log register for forensic analysis.

Further confusing network characterization attempts is the increasing tendency of P2P applications to support the sharing of encrypted messages. Admittedly, the rate with which P2P protocols are developed and upgraded renders the existing packet analysis unrealistic and glaringly inefficient. In this paper, we develop a methodology to classify UDP flows at the transport layer, i.e., based on the changing flow connection patterns of P2P traffic, and without relying on existing strategies, which is an entirely new paradigm. To the best of our knowledge, our work is the first to explore UDP traffic in understanding the complexities of p2p networks.

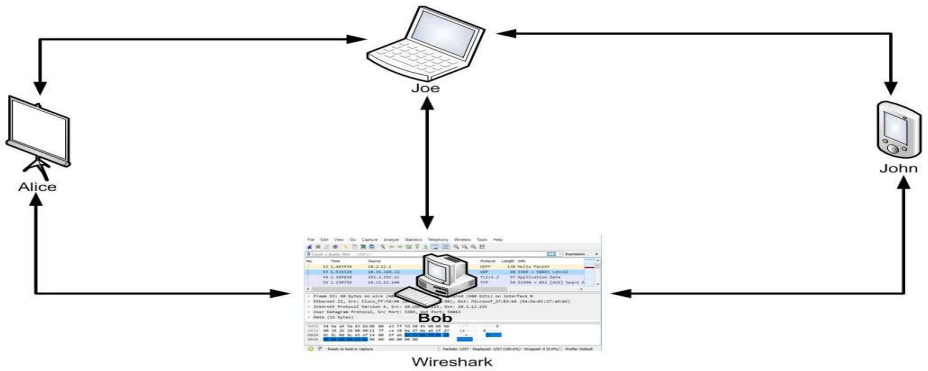

Fig. 1: Wireshark Capture Model on Peer-to-Peer Systems

EXPERIMENTAL PROCEDURE

To illustrate how the model presented in Fig. 1 work, we performed an experiment to analyze the effective usage of UDP traffic to monitor p2p networks through packet sniffing. For the analysis, we obtained a Laptop PC running Windows 10 Home 64-bit Operating System with Installed memory (RAM) of 8.00 GB, 1 TB Hard Disk Drive and Processor details: Intel (R) Core (TM) i5-3230M CPU @ 2.60GHz. We also obtained uTorrent, a BitTorrent client that allows the sharing and downloading of torrent files. A torrent is responsible for saving metadata used for BitTorrent. Then, we set up Wireshark, the

Corresponding author: Musa, A. asmusa.se@buk.edu.ng Dept. of software Engineering, Bayero University, Kano, Nigeria.

world's foremost widely used network protocol analyser, by setting up the network interface card (NIC) to initialize when prompted (Wireshark, 2016). The protocol analyser was initialised on a relatively clean computer with no running apps on the background.

Hardware Specifications

A Laptop PC running Windows 10 Home 64-bit Operating System, with Installed memory (RAM) of 8.00 GB, 1 TB Hard Disk Drive and Processor details: Intel(R) Core (TM) i5- 3230M CPU @ 2.60GHz.

Software Specifications

The leading software used in the experiment is uTorrent, version 1.8.9 Windows Installer for windows. Wireshark, version 2.6.8, was also used as an integral part of the research.

RESULTS AND DISCUSSION

For the UDP experimentation using Wireshark, we first checked the configuration of our protocol analyser. The analyser works very well, capturing all promiscuous data and Mac addressees over its secure channel for analytics (Musa et al., 2019). Consequently, we searched for the metadata of the latest movie that was just released and still showing in the cinema on the p2p tracker websites. We chose a recent movie to demonstrate the effects of file sharing on p2p applications, usability, and the novelty of our research. By mere possession of such a file, the original source uploader and all the leeches participating in the network, including us, are involved in an illegal act of copyright abuse. Although we did it for analytics and investigation purposes, the fact still remains.

The metadata was then initiated into the uTorrent client for download but not before initializing Wireshark. The integrity of the captured traffic sent over the p2p client will only be complete if the whole session was captured. Wireshark was set to scan every packet received by the NIC and then the Mac address of the received packet compared with each other for maximum integrity and validation. If the MAC address is the same, the received packet is accepted; otherwise, the packet gets filtered.

In our experiment, we focus on the traffic of the uTorrent network of p2p. Generally, the traffic of an uTorrent session is always massive (uTorrent, 2018). Still, because we employed Wireshark for our experiment, capturing the packets, analysis, and identifying the UDP flow and pattern was possible. The UDP pattern is visible on both the transport layer of the OSI and TCP/IP model (Qadeer et al., 2010). This made it easier to verify our experiment results using the UDP data flow of the OSI model and the UDP checksum of the TCP/IP model. Based on the information recovered from the transport layer, we can match the information retrieved to validate our results.

Finally, after the downloading session is done as well as the capturing over Wireshark, the packets will be processed to display only UDP packets for analysis. Table. 1 shows the statistics over IP of all the HTTP packets that were captured. It also provides us with the number of packets to each website that interacted with the session. This will help us to identify how many requests and responses to the download session recorded. The Rate in Table 1 denotes the time at which the request and response are being processed in milliseconds while the Percent row provides us with the rate of the data flow. Methods like NOTIFY, SEARCH, POST, and HTTP Request Packets denotes the processes of HTTP requests. The Table also provides us with the evidence validation that the file-sharing was done without a single error from either side of the HTTP request and response.

Corresponding author: Musa, A. asmusa.se@buk.edu.ng Dept. of software Engineering, Bayero University, Kano, Nigeria.

Table 1: HTTP Packets over IP

HTTP/Packet Counter	Count	Rate (ms)	Percent (%)
Total HTTP Packets	57	0.0007	100%
Other HTTP Packets	0	0	0.00%
HTTP Response Packets	12	0.0002	21.05%
???: broken	0	0	0.00%
5xx: Server Error	0	0	0.00%
4xx: Client Error	0	0	0.00%
3xx: Redirection	0	0	0.00%
2xx: Success	12	0.0002	100.00%
200 OK	12	0.0002	100.00%
1xx: Informational	0	0	0.00%
HTTP Request Packets	45	0.0006	78.95%
SEARCH	16	0.0002	35.56%
POST	4	0.0001	8.89%
NOTIFY	25	0.0003	55.56%

Fig. 2 shows the HTTP request and responses from the host server address and the peers that participated in it. Note that in the p2p network, each peer acts as both the client and the server while sharing computing responsibilities (Musa et al., 2018). Now that Table 1 has proven that there was no error, and the integrity of the captured data, the information in figure 2 can be accepted. The experiment is therefore relevant and useful for uTorrent client and possibly other p2p applications. The uniform resource locator (URL) addresses recovered, and the number of participations counts in figure 2 will go a long way for a network administrator that is launching an investigation into the origins of a shared file.

Topic / Item	Count
HTTP Responses by Server Address	12
4.16.75.8	3
OK	3
23.21.92.252	1
OK	1
192.168.0.7	2
OK	2
192.168.0.1	6
OK	6
HTTP Requests by Server	45
HTTP Requests by Server Address	45
4.16.75.8	3
oss-content.securestudies.com	3
239.255.255.250	41
239.255.255.250:1900	41
23.21.92.252	1
i-64.b-44994.ut.bench.utorrent.com	1
HTTP Requests by HTTP Host	45
oss-content.securestudies.com	3
4.16.75.8	3
i-64.b-44994.ut.bench.utorrent.com	1
23.21.92.252	1
239.255.255.250:1900	41
239.255.255.250	41

Fig. 2: HTTP Count

Corresponding author: Musa, A. asmusa.se@buk.edu.ng Dept. of software Engineering, Bayero University, Kano, Nigeria.

Wireshark enables the second part of the analysis after capturing the traffic over the uTorrent network. Wireshark makes the NIC card on our machine to enter into promiscuous mode. As previously stated, the Ethernet NIC is made to filter irrelevant traffic. The sniffer then observes all the traffic and thus be promiscuous. User Datagram Protocol (UDP) does not have a three-way handshaking mechanism used by TCP (Awan et al., 2006). Therefore, it is often considered as unreliable because it does not guarantee the duplicate protection or deliver ordering of packets. However, UDP is known for increase stability and fairness over the internet as it lacks congestion control over the network (Yu, 2010). UDP checksum provides data integrity apart from port numbers that are used to address different functions of sending and receiving nodes. The communicated data over the network is collected and recorded by Wireshark while the network adapter goes to promiscuous mode. Fig. 3 shows the UDP data flow of the total captured packets. The diagram shows that the destination port has received all the packets well above the threshold for the captured packets. The horizontal lines denote the total number of packets captured while the bars denote the source and destination port.

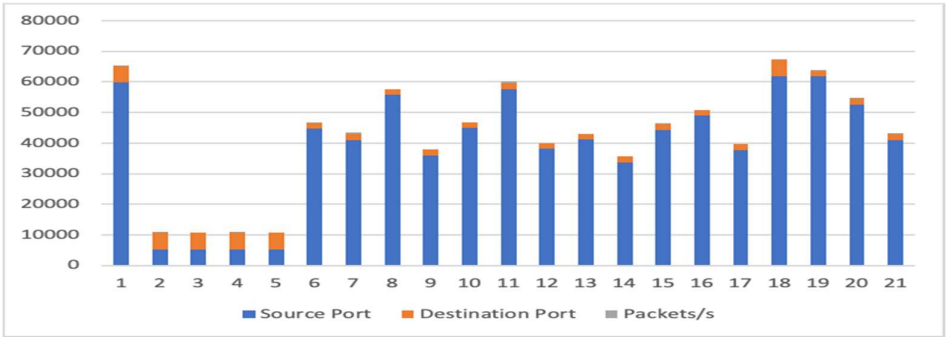

Fig. 3: UDP Data Flow

All non-TCP packets flow follow the same pattern as UDP packet flow. A checksum is the 16 bit 1's complement of the 1's complement sum of a pseudo-header of information from the IP header, the UDP header, and the data, padded with 0 at the end to make a multiple of two octets (Qadeer et al., 2010). If the calculated checksum results in the value zero, it should be sent as the 1's complement. Fig. 4 shows that the bar for bad checksum as negligible and cannot be seen, meaning only good checksum was recorded for this experiment. Good checksum = 0 and bad checksum = 1.

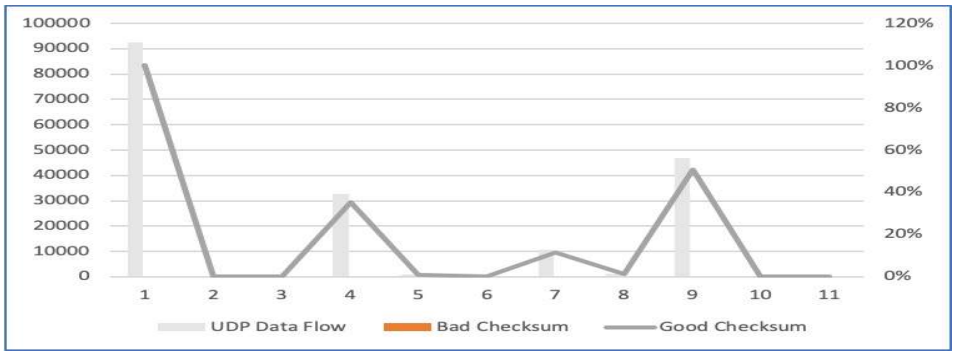

Fig. 4: UDP Checksum

Corresponding author: Musa, A. asmusa.se@buk.edu.ng *Dept. of software Engineering, Bayero University, Kano, Nigeria.*

CONCLUSION

This paper focused on the identification and monitoring of a significant and growing component of p2p traffic. Traditionally, P2P traffic has been classified by well-known port numbers, which is unique to each protocol. However, increasing concerns due to legal, connectivity, marketing, and other complications have pushed P2P networks to challenge network standards by port numbers randomization and, in general, making some attempt to disguise their activity. As a result, conventional techniques of analyses are bound to miscalculate P2P traffic, and reliable identification of P2P traffic requires examination on the transport layer.

Network traffic monitoring is one of the most critical concerns of network administrators. In this paper, we employed a traffic analysis based on a connectionless datagram-oriented protocol, UDP. We have also established that p2p networks can be effectively monitored using a packet sniffer. Furthermore, p2p systems security can move one step closer by ensuring the integrity of the captured traffic. We have demonstrated in our experiment using UDP data flow and checksum because accurate data is key to the overall success of any digital investigation

References

Awan, A., Ferreira, R. A., Jagannathan, S., & Grama, A. (2006). Unstructured peer-to-peer networks for sharing processor cycles. *Parallel Computing*, *32*(2), 115–135. https://doi.org/10.1016/j.parco.2005.09.002

Bauer, K., McCoy, D., Grunwald, D., & Sicker, D. (2009, December 1). *BitStalker: Accurately and efficiently monitoring bittorrent traffic*. IEEE Xplore. https://doi.org/10.1109/WIFS.2009.5386457

Dabir, A., & Matrawy, A. (2007, November 1). *Bottleneck Analysis of Traffic Monitoring using Wireshark*. IEEE Xplore. https://doi.org/10.1109/IIT.2007.4430446

Das, R., & Tuna, G. (2017). Packet tracing and analysis of network cameras with Wireshark. *2017 5th International Symposium on Digital Forensic and Security (ISDFS)*. https://doi.org/10.1109/isdfs.2017.7916510

Leong, R. S. C., Lai, P. K. Y., Chow, K. P., Kwan, M. Y. K., & Law, F. Y. W. (2010). *Forensic Investigation of Peer-to-Peer Networks*. Handbook of Research on Computational Forensics, Digital Crime, and Investigation: Methods and Solutions. https://www.igi-global.com/chapter/forensic-investigation-peer-peer-networks/39225

Liberatore, M., Erdely, R., Kerle, T., Levine, B. N., & Shields, C. (2010). Forensic investigation of peer-to-peer file sharing networks. *Digital Investigation*, *7*, S95–S103. https://doi.org/10.1016/j.diin.2010.05.012

Musa, A., Abubakar, A., Gimba, U. A., & Rasheed, R. A. (2019). An Investigation into Peer-to-Peer Network Security Using Wireshark. *2019 15th International Conference on Electronics, Computer and Computation (ICECCO)*. https://doi.org/10.1109/icecco48375.2019.9043236

Musa, A., Almohannadi, H., & Alhamar, J. (2018, August 1). *Malware Propagation Modelling in Peer-to-Peer Networks: A Review*. IEEE Xplore. https://doi.org/10.1109/W-FiCloud.2018.00038

Corresponding author: Musa, A. asmusa.se@buk.edu.ng Dept. of software Engineering, Bayero University, Kano, Nigeria.

Neuner, S., Schmiedecker, M., & Weippl, E. R. (2016). PeekaTorrent: Leveraging P2P hash values for digital forensics. *Digital Investigation*, *18*, S149–S156. https://doi.org/10.1016/j.diin.2016.04.011

Qadeer, M. A., Iqbal, A., Zahid, M., & Siddiqui, M. R. (2010). Network Traffic Analysis and Intrusion Detection Using Packet Sniffer. *Second International Conference on Communications, Software and Networks*, 313–317.

Schrader, K., Mullins, B., Peterson, G., & Mills, R. (2009). TRACKING CONTRABAND FILES TRANSMITTED USING BITTORRENT. In *Advances in Digital Forensics* (Vol. 306, pp. 159–173). Springer. https://link.springer.com/content/pdf/10.1007%2F978-3-642-04155-6_12.pdf

uTorrent. (2018). *µTorrent - a BitTorrent client*. Utorrent.Com. https://www.utorrent.com/

Wireshark. (2016). Wireshark.Org. https://www.wireshark.org/

Yu, B. (2010). Based on the network sniffer implement network monitoring. *IEEE Xplore*, *7*, V7–1-V7-3. https://doi.org/10.1109/ICCASM.2010.5620505

Corresponding author: Musa, A. asmusa.se@buk.edu.ng Dept. of software Engineering, Bayero University, Kano, Nigeria.