Targeting Torrents

Yash Shah
Georgia Institute of Technology
shah_yash10@gatech.edu

Abstract: Peer-to-Peer traffic is increasingly becoming one of the most dominant form of data traffic observed over the internet. A majority of which can be attributed to file sharing services. But the rise of p2p services also poses a threat to our network and the connected devices. This threat appears in the form of torrent poisoning or p2p malwares. In this research we analyze traffic data from Georgia Tech subnet aiming to identify torrent/p2p usage and document such activities so that a better localized analysis might be conducted to create preventive measures against such threats.

I. Introduction

Peer-to-peer file sharing is the distribution and sharing of digital media using peer-to-peer (p2p) networking technology [4]. P2P file sharing allows users to access media files such as books, music, movies, and games using a p2p software program that searches for other connected computers on a P2P network to locate the desired content. The nodes (peers) of such networks are usually end-user computer systems that are interconnected via the Internet.

The associated BitTorrent protocol [1] was designed to reduce the server and network impact of distributing large files. Rather than downloading a file from a single source server, the BitTorrent protocol allows users to join a "swarm" of hosts to upload/download from each other simultaneously. The protocol is an alternative to the older single source, multiple mirror sources technique for distributing data, and can work effectively over networks with lower bandwidth. Using the BitTorrent protocol, several basic computers, such as home computers, can replace large servers while efficiently distributing files to many recipients. This lower bandwidth usage also helps prevent large spikes in internet traffic in a given area, keeping internet speeds higher for all users in general, regardless of whether or not they use the BitTorrent protocol. This advantage also lead to the increasing misuse of the network mostly via the illegal distribution of copyright protected files. With strict enforcement and observation of such activities a decline in this form network was observed, predicting the fall/death of p2p networks [5].

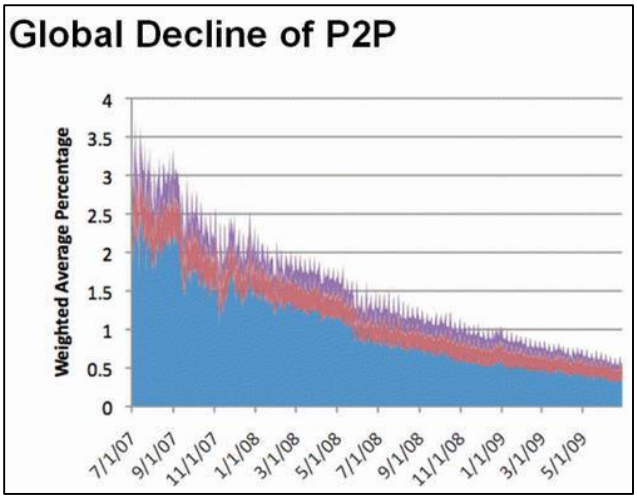


Figure 1: Initial Global Decline of p2p services

As Fig1 shows there is a clear decline in p2p traffic where the usage of this network was expected to drop to less than 0.1 by 2010. [6] But with the rise of Social Media and online content/media delivery, a growth pattern emerged showing an increasing utilization of p2p by such domains. The most popular ones being, Netflix, Spotify, YouTube, etc. [2]

	Upstream		Downstream		Aggregate	
Rank	Application	Share	Application	Share	Application	Share
1	BitTorrent	36.8%	Netflix	33.0%	Netflix	28.8%
2	HTTP	9.83%	YouTube	14.8%	YouTube	13.1%
3	Skype	4.76%	HTTP	12.0%	HTTP	11.7%
4	Netflix	4.51%	BitTorrent	5.89%	BitTorrent	10.3%
5	SSL	3.73%	iTunes	3.92%	iTunes	3.43%
6	YouTube	2.70%	MPEG	2.22%	SSL	2.23%
7	PPStream	1.65%	Flash Video	2.21%	MPEG	2.05%
8	Facebook	1.62%	SSL	1.97%	Flash Video	2.01%
9	Apple PhotoStream	1.46%	Amazon Video	1.75%	Facebook	1.50%
10	Dropbox	1.17%	Facebook	1.48%	RTMP	1.41%
	Top 10	**68.24%**	**Top 10**	**79.01%**	**Top 10**	**76.54%**

Figure 2: Top application usage in North America

Fig2 shows the network usage by applications during peak traffic hours in North America [3]. As observed BitTorrent and other media services are at a high, with the predicted growth rate of such services averaging at 20% every year.

But with the rise in p2p emerges a security threat in the form of p2p malwares and torrent poisoning [10]. The risk of such intrusions is ever present and constantly increasing with the increase in media sharing services associated with p2p networks. Numerous attack vectors exist here with varying degrees of success. Some such methods are as follows [11].

- **Decoy insertion** (or content pollution) is a method by which corrupted versions of a particular file are inserted into the network. This deters users from finding an uncorrupted version and also increases distribution of the corrupted file.
- **Index poisoning** this method targets the index found in P2P file sharing systems. The index allows users to locate the IP addresses of desired content. Thus, this method of attack makes searching difficult for network users. The attacker inserts a large amount of invalid information into the index to prevent users from finding the correct resource.
- **Spoofing** some companies that disrupt P2P file sharing on behalf of content providers create their own software in order to launch attacks. MediaDefender have written their own program which directs users to non-existent locations via bogus search results.
- **Content poisoning** Selective content poisoning (also known as proactive or discriminatory content poisoning) attempts to detect pirates while allowing legitimate users to continue to enjoy the service provided by an open P2P network. The protocol identifies a peer with its endpoint address while the file index format is changed to incorporate a digital signature. A peer authentication protocol can then establish the legitimacy of a peer when they download and upload files. Using identity based signatures, the system enables each peer to identify pirates without the need for communication with a central authority. The protocol then sends poisoned chunks to detected pirates requesting a copyright protected file only.

These and many other attacks are making p2p the desired protocol of choice and with the ever increasing usage of this network it becomes necessary to form a better understanding of p2p network and how they behave, so that a better attach prevention system may be designed to deter future attacks.

II. Data-Set

The Data set used for this experiment was initially obtained by monitoring the campus subnet for a period of 12 hours. The data set obtained was in the form of *.csv format consisting of 52 columns and 850,000+ rows. Where each row contained a basic network traffic data without the actual data segment contained within each packet, instead a 128 character raw string was extracted from the first transaction occurring when a connection is established. This raw data contained a few plaintext characters for HTTP connections, more prominently GET/POST request header, all else are unreadable ASCII string. A more comprehensive dataset could not be obtained due possible privacy and ethic law violation issues. The dataset consisted of the following headers.

1. StartDay	2. Time0	3. EndDay	4. Time1
5. Addr0	6. Port0	7. Addr1	8. Port1
9. Pkts0	10. Bytes0	11. Pkts1	12. Bytes1
13. Data0	14. Frags0	15. Data1	16. Frags1
17. Bad_frags0	18. Bad_tcp0	19. Bad_frags1	20. Bad_tcp1
21. Dups0	22. Oops0	23. Dups1	24. Oops1
25. Syn0	26. S-Ack0	27. Syn1	28. S-Ack1
29. Fin0	30. Gaps0	31. Fin1	32. Gaps1
33. Res0	34. Xr0	35. Res1	36. Xr1
37. 1_16-0	38. 64-0	39. 128-0	40. 512-0
41. 1024-0	42. 1500-0	43. 1_16-1	44. 64-1
45. 126-1	46. 512-1	47. 1024-1	48. 1500-1
49. RawData0	50. RawData1	51. HexMap	52. Protocol

Figure 3: Database headers for the obtained dataset

The headers in Fig3 are the various values obtained for each established connection. Here a connection

refers to the first communication that occurs between a client and a server.

III. Data Analysis

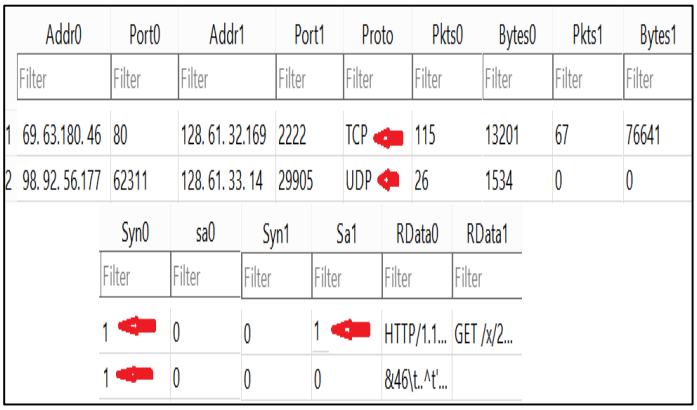
Due to the vastness of the dataset, the problem was approached in a statistical manner, to better understand the usage patterns of clients and their corresponding online behavior. To analyze the dataset, by intuition, only a small set headers were considered necessary, as the goal was not the analysis of network behavior but targeted client behavior. The headers used are:

- **Addr 0/1:** contained the IP address of the client/server pair.
- **Protocol:** Indicated whether the connection used UDP or TCP connection.
- **Port 0/1:** represents the port number over which the connection was established.
- **Bytes 0/1:** represents the total Bytes transmitted/received during the established connection.
- **RawData 0/1:** contain the first 128 character (max) data string obtained from the packets.
- **Syn 0/1, SAck 0/1:** represents whether a handshake occurred between the client/server (SYN-SYN ACK).

Here 0/1 represents the two sets of data each representing a client or a server. Using this data the following information was obtained.

a. Client-Server Identification

Since the data was from the Georgia Tech subnet, the connections could represent both outgoing connections – connections from within the subnet to an outside server, and incoming connections – connections between an outside client and a GT server. Thus it was necessary to identify and distinguish between them. For this the data in the Syn/SAck columns where used. For TCP communication if Syn0 was flagged the corresponding Addr0 represented the client for outgoing communication and if Syn1 was flagged Addr1 was the client for incoming communication. For UDP, it being a connectionless communication, the client-server identification represented a challenge initially – as doing a reverse lookup on all the IP addresses would be a time-consuming, highly inefficient task. But on deeper analysis, it was observed for each UDP communication a Syn column was flagged indicating that the corresponding 0/1 data represented the client. This was an intriguing discovery as the flag was unexpected. A packet analysis of these communications would be needed to determine whether the flag was set using packet data or by the traffic capturer to indicate the client dataset.
A sample dataset as represented in a database for the aforementioned selective headers can be seen in the following figure.



	Addr0	Port0	Addr1	Port1	Proto	Pkts0	Bytes0	Pkts1	Bytes1
	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter
1	69.63.180.46	80	128.61.32.169	2222	TCP	115	13201	67	76641
2	98.92.56.177	62311	128.61.33.14	29905	UDP	26	1534	0	0

Syn0	sa0	Syn1	Sa1	RData0	RData1
Filter	Filter	Filter	Filter	Filter	Filter
1	0	0	1	HTTP/1.1...	GET /x/2...
1	0	0	0	&46\t..^t'...	

Figure 4: Example of data used from the dataset

From the above figure we can observe that for row1, a TCP connection, Syn0 and Sa1 (SYN-ACK) has been flagged indication that Addr0 is the client while Add1 is the server. While in row2, a UDP connection, Syn0 has been flagged indication that Addr0 is the client.
Using this data it becomes a trivial task to identify the client-server pair for all communications incoming or outgoing. This is necessary to identify how each individual client's network behavior.

b. Statistical Analysis

After the identification of clients and servers, it is now possible to conduct statistical/frequency analysis on the dataset. This is done to separate usable sets of data from miscellaneous set. For this a cumulative distribution graph was plotted first to visualize how large each communications data size transaction is. The below two figures shows the CDF

of clients and servers against total bytes transmitted for each established connection.
From the graphs we observe that more than 98% of the data communicated between clients and servers are less than 0.1 MB, while the data size for an average web-traffic was ~0.8MB in 2013.

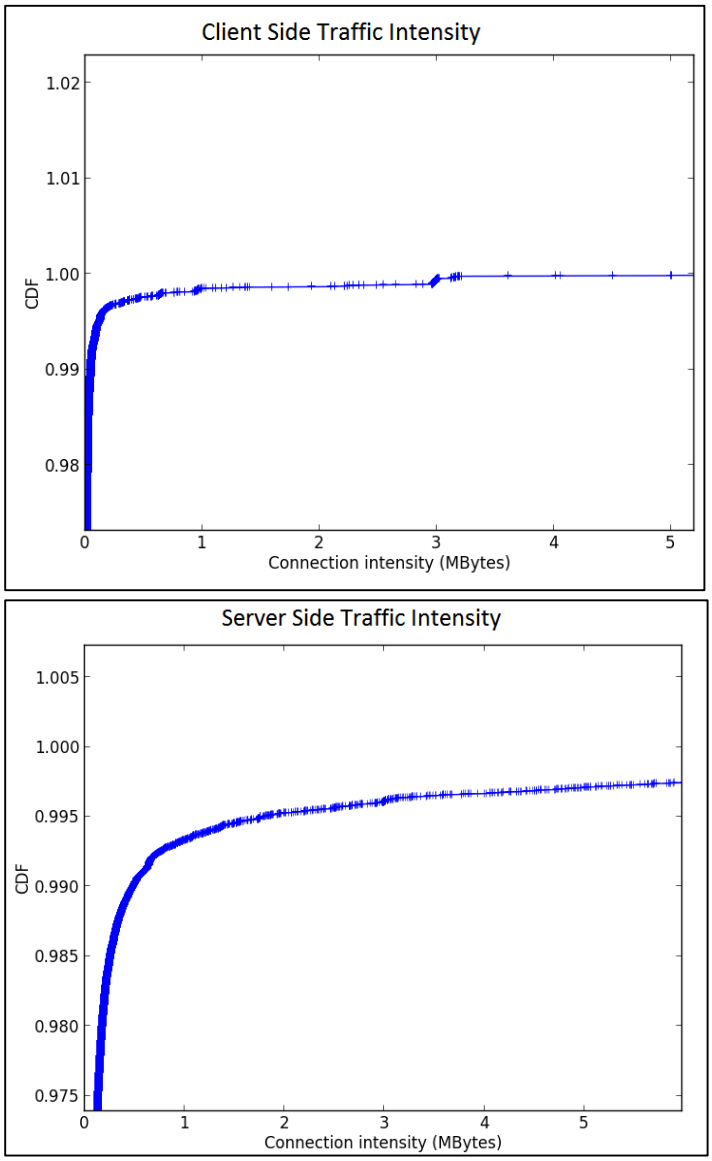


Figure 5: (a) Client side traffic intensity by bytes; (b) Server side traffic intensity by bytes

This behavior was not expected since the visualization was aimed at identifying pockets of high data transactions, while what was observed is extremely low data transaction density.
This behavior merited a deeper analysis which centered on port usage with high usage frequencies and their corresponding cumulative data usage intensity. For this a usage frequency graphs where created with corresponding cumulative data transfers for each client side and server side data sets. This can be observed from the following figures.
Fig6 (a) and (b) represents the client side top 15 ports used with their corresponding data traffic. While

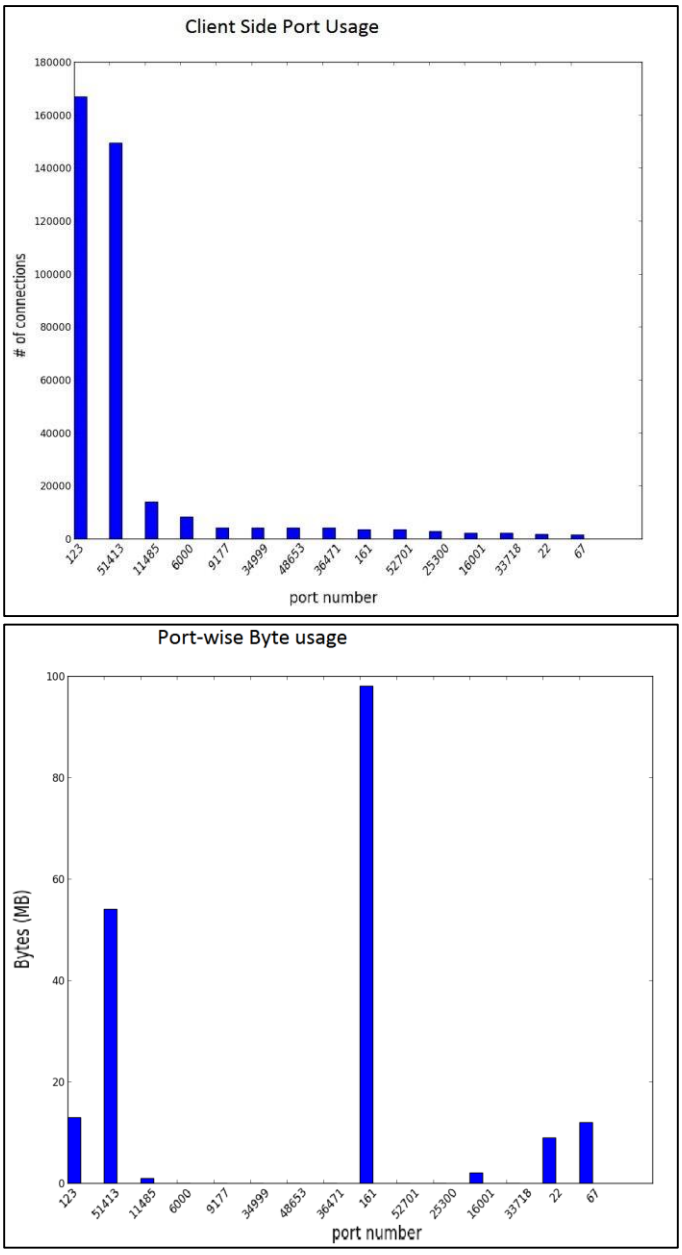


Figure 6: (a) Port with Max usage Frequency; (b) Associated port traffic

Fig6(c) represents the ports with the maximum observable traffic (most bytes transmitted).
From Fig6 (a, b) of the above graphs we observe that even though most of the top ports used to establish a connection account for ~65% of the total dataset, the corresponding cumulative data traffic for the ports are not very high.

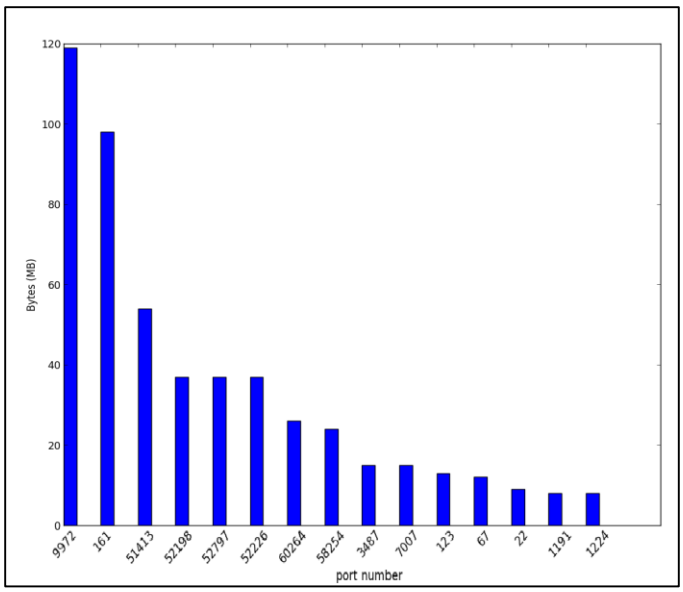


Figure 6: (c) Ports with maximum traffic

For client side nodes, the port most used is port 123 (NTP), which mostly accessed by Apple devices updating their device time periodically. While the unregistered high numbered ports used as seen in Fig6(c) (characteristic of a torrent service) can be accounted for by various internet applications who establish all outgoing connections usually using randomly assigned unregistered ports (ephemeral ports).

Therefore analyzing client port and Byte data would prove unfruitful in identifying torrent activities, as torrents also use randomly assigned unregistered ports. Due to this more emphasis was put on server side activities for the identification of torrents.

The server side analysis yielded a more clear result as most server traffic are bound by registered ports, i.e. http connects to port 80, https to port 443, etc. from the following graphs we can see that port 123 (NTP), 80 (http), 53 (DNS) are registered ports with high usage frequency, thus the results of this analysis are more commensurate with a textbook network behavior. But we still see some high numbered unregistered port being used with significant frequency and high data traffic thresholds. Initially from a general understanding of the working of the internet, they were flagged as p2p/torrent connections requiring further evaluation.

But a deeper traffic classification proved even more difficult due to emergence of Server side Ephemeral Ports.

- **Ephemeral Ports:** They are short-lived transport protocol port for Internet Protocol. [8] Such server side ports are assigned for a client-server communication after the initial handshake using a known port. This is often done to reduce high traffic bottleneck on singular ports. E.g. Microsoft Windows often utilize ports above 49152 for software updates [9].

Due to this the identification of p2p/torrent activities became even more challenging.

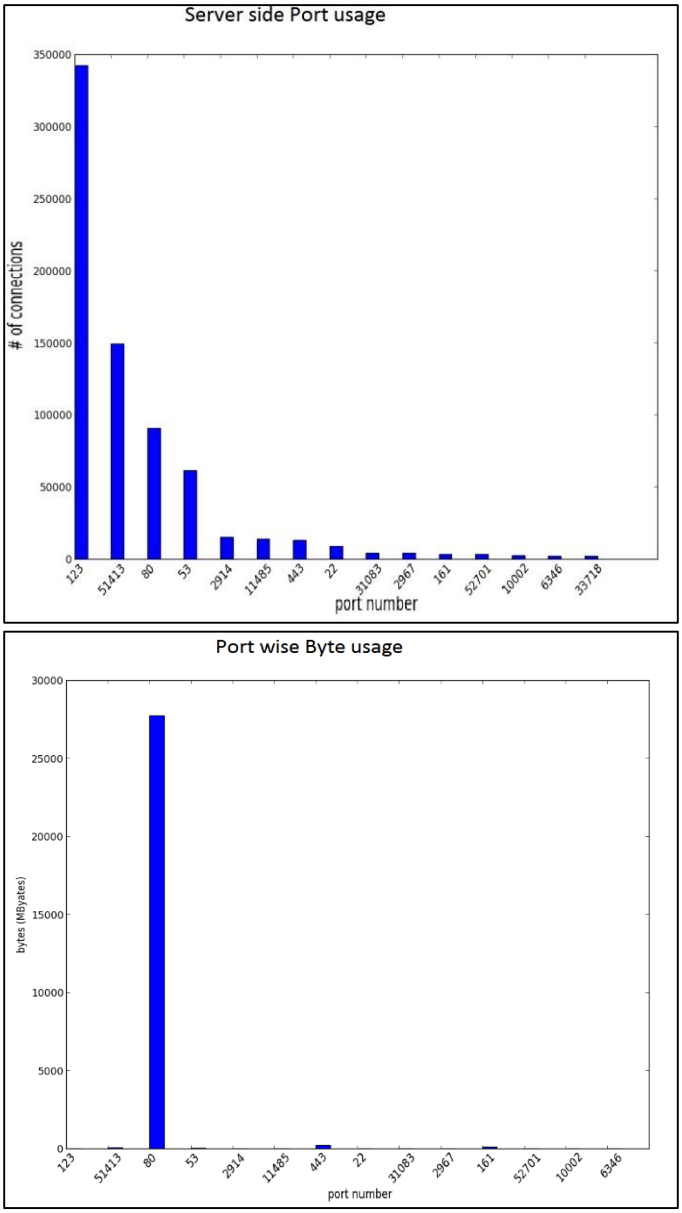


Figure 7: (a) Port with Max usage Frequency; (b) Associated port traffic

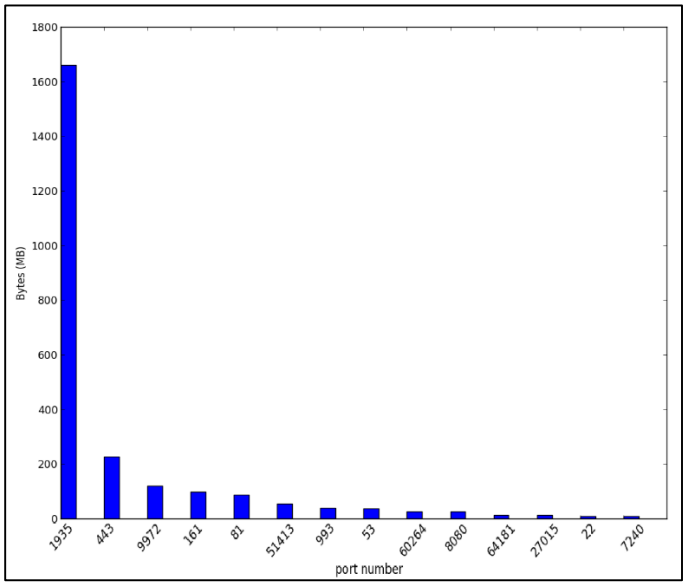


Figure 7: (c) Ports with maximum traffic

Since now even non-p2p legitimate connections also started using unregistered ports, the identification and separation of legitimate from other traffic became vital. For this a deeper understanding of server-side ephemeral ports was needed.

Fig8 shows some of the more commonly used unregistered ports and the applications associated with them [7]. On further research it was determined that due to the rise of online media delivery systems (Spotify, Netflix, YouTube) and online gaming applications (League of Legends), the use of unregistered ports has increased significantly and torrenting applications have started using ports associated with these applications to mask their activities making identifying them with the current dataset more difficult.

c. IP elimination and List creation

To solve the above problem a process of elimination in conjunction with white/grey listing of IP addresses was utilized. In the elimination process all connections with registered ports where eliminated as the probability of a torrenting application utilizing a registered port number is very miniscule. This significantly reduced the queryable dataset by more than 70%. Furthermore for unregistered ports, the RawData column held signs of regular web traffic, identified using keywords like 'GET', 'POST','HTTP', were identified and the associated

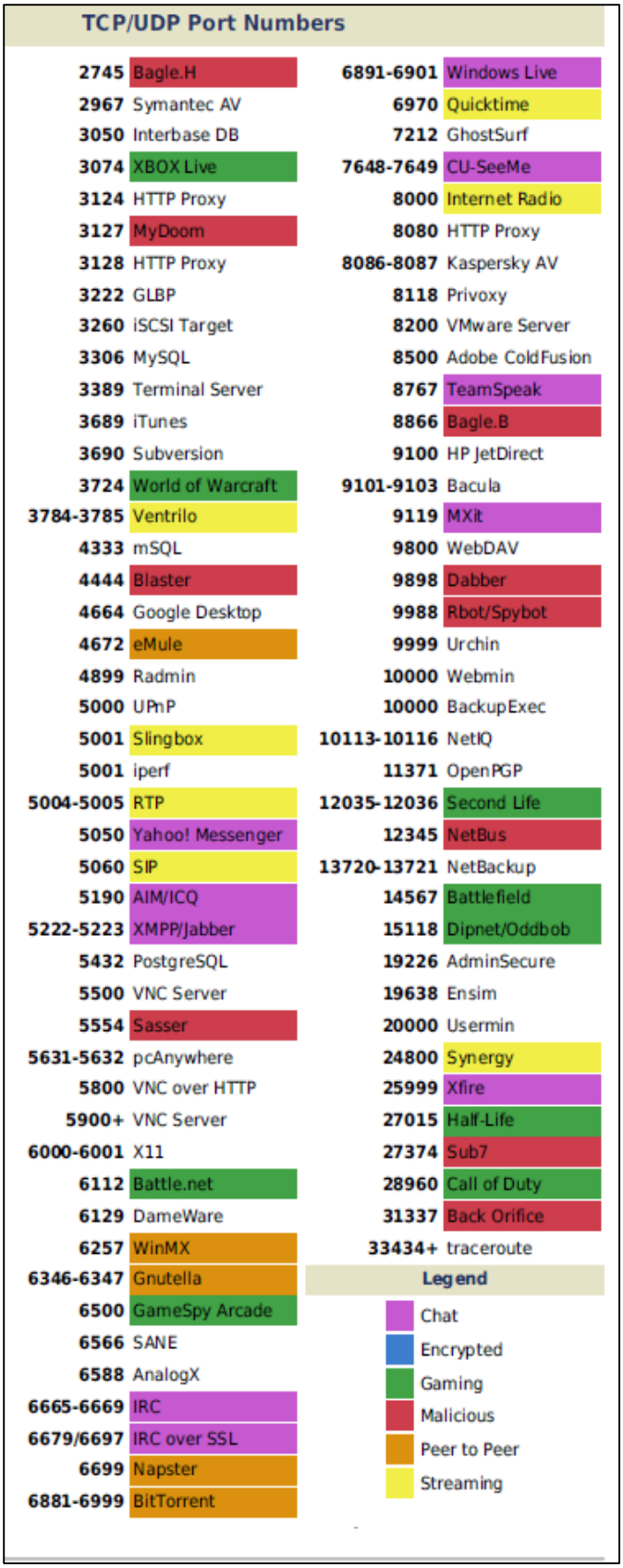


Figure 8: Commonly used unregistered port with associated applications

IP addresses whitelisted, this list is created for future reference so that the elimination of already whitelisted IP addresses becomes significantly easy. The following figure shows an example of such activities over port 65451.

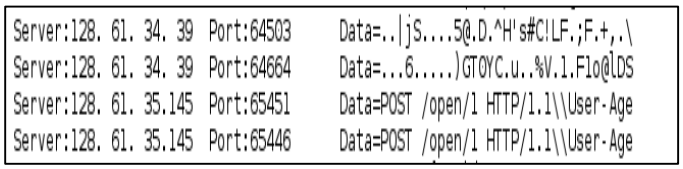


Figure 9: Example of unregistered port used for web traffic

IV. Result & Conclusion

Using such methods it became possible to reduce the data to a small enough size to make it possible to do a reverse look-up on the IP addresses without significant delay.

Once the reverse look-up was done, an IP address resolved in such a manner was whitelisted. The rest of the IP's were grey-listed as no other metric could be identified to further reduce the grey listed addresses. This grey list signifies connections that could possibly be torrent activities.

But without any means of further identifying and separating them, a grounded identification could not be conducted.

Though the size of the grey list, thus obtained, is relatively small compared the original dataset, any further analysis currently was not possible. To further reduce and correctly identify torrent activities, more data – possible the actual packets associated with these connections – would be needed.

V. Future Works

Even if direct identification is not currently possible, using the grey-list it possible to create a flow monitoring system which only monitors the grey-listed addresses or a IP based clustering. This way, once enough data is collected a further breakdown of the grey-list could be possible and the identification of torrents plausible. The efficacy of the proposed traffic monitoring system remains to be further analyzed and this system is part of research that still needs to be conducted.

VI. References:

[1] Wikipedia article: BitTorrent

[2] PCWorld: Netflix, YouTube traffic on the rise by Zach Miners

[3] TorrentFreak: BitTorrent traffic increase by Ernesto

[4] Wikipedia article: Peer-to-Peer File Sharing

[5] Deepfield study: Rise and Fall of p2p by Craig Labovitz

[6] Casey Research: Sudden Rise of p2p commerce by Doug Hornig

[7] OverAPI common Ports chart

[8] Wikipedia article: Ephemeral Ports

[9] Cymru.com : Ephemeral source port selection strategies

[10] IEEE : A first look at p2p worms: Threats and Defense by Zhou et. al.

[11] Wikipedia article: Torrent Poisoning