P2P, the Gorilla in the Cable

Alexandre Gerber, Joseph Houle, Han Nguyen, Matthew Roughan, Subhabrata Sen
AT&T Labs - Research

Abstract

There is considerable interest in Peer-to-peer (P2P) traffic because of its remarkable increase over the last few years. By analyzing flow measurements at the regional aggregation points of several cable operators, we are able to study its properties. It has become a large part of broadband traffic and its characteristics are different from older applications, such as the Web. It is a stable balanced traffic: the peak to valley ratio during a day is around 2 and the Inbound/Outbound traffic balance is close to one. Although P2P protocols are based on a distributed architecture, they don't show strong signs of geographical locality. A cable subscriber is not much more likely to download a file from a close region than from a far region.

It is clear that most of the traffic is generated by heavy hitters who abuse P2P (and other) applications, whereas most of the subscribers only use their broadband connections to browse the web, exchange e-mails or chat. However it is not easy to directly block or limit P2P traffic, because theses applications adapt themselves to their environment: the users develop ways of eluding the traffic blocks. The traffic that could be once identified with five port numbers is now spread over thousands of TCP ports, pushing port based identification to its limits. More complex methods to identify P2P traffic are not a long-term solution, the cable industry should opt for a a "pay for what you use" model like the other utilities.

INTRODUCTION

File Sharing Applications

KaZaA, Gnutella and DirectConnect are all decentralized, self-organizing file sharing systems with data and index information (metadata for searching) distributed over a set of end-peers or peers, each of which can be both a client and a server of content. Peers can join and leave frequently, and organize in a distributed fashion into an application-level overlay via point-to-point application-level connections between a peer and a set of other peers (its neighbors). By default, all the communications occur over well known ports.

The process of obtaining a file can be broadly divided into two phases – a search followed by a object retrieval. First, a peer uses the P2P protocol to search for the existence of a certain file in the P2P system, receives one or more responses, and if the search is successful, identifies one or more target peers from which to download that file. The search queries as well as the responses are transmitted via the overlay connections using protocol-specific application level routing. The details of how the signaling is propagated through the overlay is protocol-dependent. In earlier P2P protocols exemplified by Gnutella version 4.0, a peer initiates a query by flooding it to all its neighbors in the overlay. The neighboring peers in turn, flood to their neighbors, using a scoping mechanism to control the query flood. In contrast, for both KazaA and DirectConnect as well as newer versions of Gnutella, queries are forwarded to

and handled by only a subset of special peers (called SuperNodes in KazaA, Hubs in DirectConnect, and UltraPeers in Gnutella). A peer transmits an index of its content to the ``special peer'' to which it is connected. The special peer then uses the corresponding P2P protocol to forward the query to other such peers in the system.

Once search results are in, the requesting peer directly contacts the target peer, typically using HTTP (the target peer runs has a HTTP server listening by default on a known,protocol-specific port), to get the requested resource. Some newer systems, such as KazaA and Gnutella, use "file swarming" -- a file download is executed by retrieving different chunks from multiple peers.

Although the earlier P2P systems mostly used their default network ports for communication, there is substantial evidence to suggest that substantial P2P traffic nowadays is transmitted over a large number of non-standard ports. This seems to be primariliy motivated by the desirwe to circumvent firewall restrictions as well as rate–limiting actions by ISPs targeted at such applications - we shall discuss this more later in the paper.

Another recent development has been the development of tools for allowing an end-user to explicitly select the SuperNode it connects to. This appears to be an attempt to improve the quality of the best-effort search process in the P2P system, for files that may exhibit locality in storage. For instance, connecting to a SuperNode in Brazil may increase the chances of locating Samba-related content.

Data Collection

We have access to "flow-level" data at the regional aggregation points for several broadband ISPs. Flow-level data is considerably more detailed than data sets such as SNMP, and at least this level of detail if needed to perform application classification. The regional aggregation points provide the MSOs with access to the backbone for traffic between regions and to the rest of the Internet, where a region typically ranges from an extended metropolitan area to a state.

By flow, we mean a sequence of packets exchanged by two applications. More precisely we define a flow to be a series of uni-directional packets with the same IP protocol, source and destination address, and source and destination ports (in the case of TCP and UDP traffic). The flow measurements used here are called Cisco Netflow; they are implemented in many of Cisco's routers. The data collected about a flow (apart from the information above) are the duration, the number of packets, and bytes transmitted, and which header flags (SYN, ACK, …) were used in the flow. Measured flows are also constrained in time (Cisco Netflow collection sends flows from the router at 15 minute intervals), so there is a need to reconstruct the actual traffic from a single "connection". After reconstruction there will be one flow per connection – a potentially enormous volume of information.

In order to minimize any performance impact on the routers collecting the flow measurements the measurements are based on sampled packets collected on the routers, which then export the flows to aggregators. To reduce the huge data volume the aggregator further samples the flows using the smart sampling algorithm [SAMP] that is better suited for heavy tailed distribution, such as typically found in Internet flows. In addition to that there is also an uncontrolled sampling due to measurement packet losses. These three types of sampling can be estimated and corrected and don't affect our

results that are based on the weekly or monthly average traffic generated by hundreds of thousands of cable subscribers.

More precisely, we used data ranging from May 2002 to February 2003 from five different MSOs. When we were not collecting all the traffic coming from a region, we were using SNMP data to extrapolate the actual traffic. However, when we analysed the behaviour per broadband user, we selected only regional aggregation points for which we were collecting all the flow level measurements.

Identifying Applications

There are a number of ways one could go about identifying individual applications within IP traffic. However, as noted, Netflow only keeps data on some aspects of flows. The most useful of these for application breakdowns are the source and destination port numbers, and the IP protocol number. The protocol numbers used are well documented [IANA1], with TCP being protocol 6, and UDP being 17. TCP, and UDP traffic also define (16 bit) source and destination port numbers intented (in part) to for use by different applications. The port numbers are divided into three ranges: the Well Known Ports (0-1023), the Registered Ports (1024-49,151), and the Dynamic and/or Private ports (49,152-65,535).

A typical TCP connection starts with a SYN/ACK handshake from a client to a server. The client addresses its initial SYN packet to the server port for a particular application, and uses a dynamic port as the source port for the SYN. The server listens on its port for connection. UDP uses ports similarly though without connections. All future packets in the TCP/UDP flow use the same pair of ports at the client and server ends. Therefore, in principle the server port number can be used to identify the higher layer application using TCP or UDP, by simply identifying which port is the server port (the one from the well-known, or registered port range) and mapping this to an application using the IANA list of registered port [IANA2].

However there are many barriers to determining applications from port numbers:

1. many implementations of TCP seem to use registered port ranges as dynamic ports ,
2. priveledged applications may use dynamic port numbers inside the well-known port range (for instance some old versions of bind use source and destination port 53).,
3. well known and registered ports are not defined for all applications (and this is typical of P2P applications).
4. an application may use ports other than its well-known port because these can only be used with special priveledges, e.g. WWW servers often run on ports other than port 80, for instance ports 8080, and 8888.
5. an application may run on different ports to avoid blocking by firewalls. (e.g. non-WWW servers are sometimes run on port 80 to avoid firewalls, and P2P applications are often run on alternate ports for the same reason).
6. There are some ambiguities in port registrations, e.g. port 888 which is used for CDDBP (CD Database Protocol) and accessbuilder .
7. in some cases server ports are dynamically allocated as needed (for instance, one might have a control

connection on which a data port is negotiated).

8. trojans and other security attacks (e.g. DoS) will break the port mapping.

Note that the use of firewalls to block unauthorized, and/or unknown applications from using a network has spawned work arounds that have made the mapping from port number to application ambiguous.

Despite this a great deal can be said about the mapping of port to application, though obviously there will still be some ambiguity, and chance for errors. Note that both ports must be considered as possible candidates for the server port, unless other data is available to rule out one port.

The algorithm that we have adopted here chooses the server port by (1) looking for a well known port, (2) a registered port, or (3) an unregistered port which is known (from reverse engineering of protocols) to be used by a particular (unregistered) application. If both source and destination port could be the server, then we choose the most likely one through ranking applications by how prevalent they are in detailed (packet level) traffic studies – for instance, WWW is considered a high ranking application, as are email, and P2P applications.

The result is a mapping from flows to applications, that while not perfect, has been shown to be reasonably effective. The biggest problem is that there are still a substantial number of flows which cannot be mapped to an application. We further classify these unknown flows by the size of the flows: the category of most interest here is "TCP-big", which consists of unknown flows that transmit more than 100kB in less than 30 minutes.

We shall argue in this paper that the TCP-big traffic is primarily P2P traffic that is using unregistered ports unknown to us. P2P applications already use unregistered ports, and the struture of P2P protocols (with separate control and data traffic) allows data traffic to be assigned to arbitrary ports. In the past the major applications have typically used default ports (for instance 1214 for KaZaa) but in the recent past many efforts have been made to constrain P2P traffic through rate limiting single ports or by blocking some ports at firewalls, with the result that P2P users commonly use work-arounds. Where-ever we refer to P2P traffic we are using the traffic on the ports known to be directly associated with P2P applications: we shall keep this separate from TCP-big except where explicitly noted. Also note that some P2P traffic may be misclassified into other application classes (for instance WWW), and so our estimates of the total volumes of P2P traffic are conservative.

We should note that we are not collecting any information about URL's, or individual subscribers usage: IP addresses measured are not related to individual subscribers, and we only view the bulk properties of the traffic, such as its distributions.

APPLICATION COMPOSITION

Overview

Table 1 shows the application traffic composition for 2 MSOs in May 2002 and January 2003. For each MSO, we examine both the traffic coming from outside the MSO to some IP address within the MSO (referred to as IN) and the traffic sourced within the MSO and destined for outside the MSO (OUT). For each time period, MSO, we display the per-application traffic volume in each direction as a percentage of the total traffic in that direction. For a given application we also show the traffic normalized by dividing by its IN traffic

volume for May 2002, in order to show the IN/Out ratio, and the growth between the two periods.

We note that in either direction, for both MSOs, the P2P traffic forms a much smaller percentage of the overall traffic in January 2003 than in May 2002. TCP-big registered dramatic increases in traffic contribution in both directions (10.5 times for Outgoing and 6.02 times for Incoming) over the same period. The normalized figures show that the P2P incoming and outgoing traffic are very similar for either of the 2 months considered. For example for MSO X, the ratio between incoming and outgoing TCP-big traffic volumes changes from 1.94:1 in May 2002 to a more balanced 1.12:1 in January 2003.

	MSO X								MSO Y							
	Applicationx Mix (percentage)				Normalized Consumption				Applicationx Mix (percentage)				Normalized Consumption			
	May 2002		January 2003		May 2002		January 2003		May 2002		January 2003		May 2002		January 2003	
	OUT	IN	OUT	IN	OUT	IN	OUT	IN	OUT	IN	OUT	IN	OUT	IN	OUT	IN
All	100.0%	100.0%	100.0%	100.0%	1	1.65	1.97	3.2	100.0%	100.0%	100.0%	100.0%	1	2.19	1.83	4.08
ESP/GRE	0.4%	0.5%	0.6%	0.5%	1	1.98	3.12	4.3	0.4%	0.5%	0.3%	0.4%	1	2.71	1.7	4.67
OTHER	4.4%	3.7%	5.7%	4.5%	1	1.37	2.54	3.23	4.6%	3.2%	5.4%	3.4%	1	1.53	2.16	2.97
TCP-BIG	8.9%	10.5%	47.5%	32.5%	1	1.94	10.5	11.68	9.5%	11.8%	45.3%	32.1%	1	2.71	8.71	13.72
AUDIO/VIDEO	0.2%	1.6%	0.2%	1.6%	1	16.61	2.77	32.64	0.1%	1.5%	0.2%	1.5%	1	23.71	3.1	44.29
CHAT	0.7%	1.3%	1.0%	1.7%	1	3.08	2.93	7.93	0.7%	1.2%	0.7%	1.4%	1	3.81	2.02	8.67
FTP	1.0%	1.3%	1.0%	0.7%	1	2.22	1.91	2.4	1.4%	1.4%	0.4%	0.9%	1	2.24	0.56	2.64
GAMES	1.6%	1.2%	3.6%	2.5%	1	1.29	4.54	5.15	1.3%	1.2%	3.4%	2.4%	1	1.92	4.73	7.43
MAIL	1.7%	0.6%	1.1%	0.7%	1	0.6	1.26	1.28	1.0%	0.5%	0.9%	0.5%	1	1.13	1.71	1.88
NEWS	0.3%	7.3%	0.2%	5.3%	1	38.52	1.51	54.55	0.7%	17.5%	0.7%	14.6%	1	54.99	1.76	85.33
P2P	75.2%	45.6%	32.9%	20.6%	1	1	0.86	0.87	75.1%	38.5%	36.7%	19.5%	1	1.12	0.9	1.06
WEB	5.6%	26.4%	6.2%	29.4%	1	7.8	2.2	16.88	5.2%	22.8%	5.9%	23.5%	1	9.53	2.06	18.27

Table 1: Application Composition of two MSOs in May 2002 and January 2003.

Time of Day Pattern

We next examine the diurnal behavior of P2P traffic. Figure 1 plots the time series of the incoming and outgoing traffic volumes (P2P, web and TCP-big) for a given MSO across a week in February 2003. For each application, all the data values are normalized by the mean per-hour incoming data volume for that application, averaged across that week.

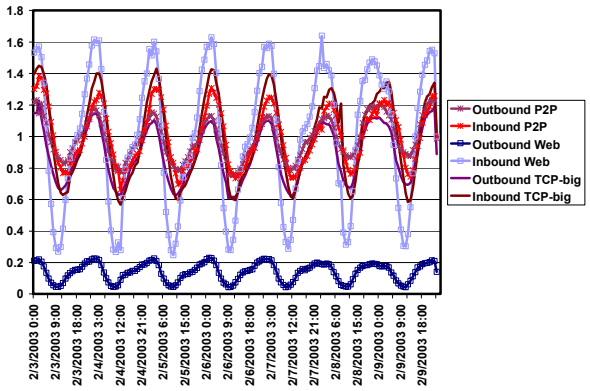

Figure 1: Time od day pattern of P2P and Web traffic.

All three applications exhibit similar diurnal behaviors with peak loads (in either direction) around 2.00 AM GMT (10.00 PM EST, 7.00 PM PST). The P2P traffic exhibits less variability across a day than Web traffic. The peak load is about 2 times the minimum as opposed to 5 times for Web traffic. The smaller variance in P2P traffic across a day

DOCKET ALARM

Explore Litigation Insights

Docket Alarm provides insights to develop a more informed litigation strategy and the peace of mind of knowing you're on top of things.

Real-Time Litigation Alerts

Keep your litigation team up-to-date with **real-time alerts** and advanced team management tools built for the enterprise, all while greatly reducing PACER spend.

Our comprehensive service means we can handle Federal, State, and Administrative courts across the country.

Advanced Docket Research

With over 230 million records, Docket Alarm's cloud-native docket research platform finds what other services can't. Coverage includes Federal, State, plus PTAB, TTAB, ITC and NLRB decisions, all in one place.

Identify arguments that have been successful in the past with full text, pinpoint searching. Link to case law cited within any court document via Fastcase.

Analytics At Your Fingertips

Learn what happened the last time a particular judge, opposing counsel or company faced cases similar to yours.

Advanced out-of-the-box PTAB and TTAB analytics are always at your fingertips.

API

Docket Alarm offers a powerful API (application programming interface) to developers that want to integrate case filings into their apps.

LAW FIRMS
Build custom dashboards for your attorneys and clients with live data direct from the court.

Automate many repetitive legal tasks like conflict checks, document management, and marketing.

FINANCIAL INSTITUTIONS
Litigation and bankruptcy checks for companies and debtors.

E-DISCOVERY AND LEGAL VENDORS
Sync your system to PACER to automate legal marketing.