Traffic Localization for DHT-Based BitTorrent Networks

Matteo Varvello and Moritz Steiner

Bell Labs, Alcatel-Lucent, USA
{matteo.varvello,moritz.steiner}@alcatel-lucent.com

Abstract. BitTorrent is currently the dominant Peer-to-Peer (P2P) protocol for file-sharing applications. BitTorrent is also a nightmare for ISPs due to its network agnostic nature, which is responsible for high network transit costs. The research community has deployed a number of strategies for BitTorrent traffic localization, mostly relying on the communication between the peers and a central server called *tracker*. However, BitTorrent users have been abandoning the trackers in favor of distributed tracking based upon Distributed Hash Tables (DHTs). The first contribution of this paper is a quantification of this claim. We monitor during four consecutive days the BitTorrent traffic (both tracker-based and DHT-based) within a large ISP. The second contribution of this paper is the design, prototype, and preliminary evaluation of the first traffic localization mechanism for DHT-based BitTorrent networks.

Keywords: peer-to-peer, measurement, traffic management.

1 Introduction

BitTorrent is by far the most popular Peer-to-Peer (P2P) protocol, adopted by several file-sharing applications such as μTorrent [27] and Azureus [3]. The BitTorrent protocol aims to maximize the volume of data exchanged among peers without taking into account their geographic location. This causes expensive inter-ISPs traffic, and thus considerable monetary loss at the ISPs.

Several interesting strategies have been proposed to achieve localization of BitTorrent traffic (Section 2). The common approach of these designs is to bias the peer selection strategy in favor of *local* peers, i.e., peers located at the same ISP. The more recent and effective design leverages the *trackers* in order to allow communication only among local peers [28]. The trackers are the central servers used in BitTorrent to coordinate a file exchange.

Recently, BitTorrent introduced a distributed tracking feature (Section 3). A client can discover which peers hold a copy or a portion of a file querying a Distributed Hash Table (DHT) [10]. This feature makes traffic localization mechanisms based on the central trackers ineffective. Currently, two large and incompatible DHTs are used in the BitTorrent community: the Azureus [3] and the Mainline [17] DHT.

J. Domingo-Pascual et al. (Eds.): NETWORKING 2011, Part II, LNCS 6641, pp. 40–53, 2011.

The first contribution of this work is a quantification of user interest in the BitTorrent distributed tracking (Section 4). Over four consecutive weekdays, we intercept the traffic exchanged between BitTorrent clients located at a large ISP and popular trackers. Meanwhile, we monitor the activity of the ISP subscribers in both the Azureus and Mainline DHT. We find that about 40% of the BitTorrent users in our sample have already abandoned the trackers in favor of the DHTs. We believe that this already large fraction of users will rapidly grow in the next years. This motivates our research on traffic localization for DHT-based BitTorrent networks.

The second contribution of this work is the design and prototype of the first localization mechanism for DHT-based BitTorrent networks (Section 5). Our localization mechanism works in two steps. First, we intercept within the DHT the announces for popular files. Then, we intercept the requests for these popular files so that we can reply with sets of peers located at the same ISP as the requesting peer. In order to intercept all announces and requests for popular files, we introduce a large number of peers in the DHT (controlled by a single entity) whose identifiers are very close to the identifiers of those files [23]. We focus on popular files for two reasons. First, only the traffic associated to files requested from more than one peer from the same ISP at the same time has potential for localization [8]. Second, we aim to minimize the number of files to be localized. In fact, the number of peers that we need to insert in the DHT to achieve traffic localization scales linearly with the number of files that we localize. Note that identifying popular content within a DHT is a non-trivial problem. In this paper, we assume that content popularity is known; however, the discovery of popular content within a DHT defines our future work.

The third contribution of this paper is a preliminary evaluation of the proposed localization mechanism for DHT-based BitTorrent networks. For this evaluation, we deploy a prototype for the Mainline DHT and monitor the benefits of traffic localization for different ISPs (Section 6). Our evaluation shows that the totality of the traffic associated with the download of a very popular file is kept local within a large ISP. However, at ISPs where the file is less popular only 3 to 25% of the transit traffic is saved on average. This problem arises only while running our system in the wild and could not be foreseen during the design phase. In fact, it is due to subtle differences of the DHT implementation across different BitTorrent clients. We are currently working to eliminate this limitation of the proposed localization mechanism.

2 Related Work

In order to achieve localization of P2P traffic, data exchanges between peers located at the same ISP need to be enforced when possible. Accordingly, several designs [6,14,16,21] propose a modification of the peer selection mechanism at the P2P client. These designs share the same rationale and differ mainly in the mechanism they use to identify the ISP of the remote peers.

Ledlie et al. [14] collaborated with Azureus [3] in order to improve the network coordinate system integrated in the Azureus client. Their system is based on

Vivaldi [9] which assigns to each peer coordinates from a low dimensional space such that the distance between peer coordinates reflects the delay between the corresponding peers. Azureus leverages the information provided by its network coordinate system to encourage communication among local peers.

A different and more effective approach is proposed in [6]. This scheme uses the information collected by CDN providers such as Akamai in order to favor communication among local peers. Precisely, a pair of peers are considered local if they are associated to the same set of CDN caches most of the time.

Both designs [6,14] share a common limitation. Given a peer is only aware of a small subset of the peers that hold a copy or a portion of a file, the probability that this subset contains peers from its ISP is very low. Thus, these designs only achieve to select the few, if any, local peers from the peer-set received.

Motivated by the need to solve the previous limitation, Xie et al. [28] propose to inform the trackers about the ISP of each peer. In this way, a tracker could reply to a peer request for a specific torrent with a list of peers located in the same ISP as the requesting peer. This approach is inspired by Aggarwal et al. [1] and Bindal et al. [4] who both suggest to use ISP support to drive the construction of generic P2P networks. The drawback of these designs is that they require cooperation between an ISP and P2P networks which is unlikely to occur.

Our work is the logical continuation of previous work in the P2P traffic localization space. Similarly, we aim to bias the peer selection strategy by exposing to a peer only the information about peers belonging to its ISP. The main departure of our work from the previous work is that we do not rely on the client-to-tracker communication.

3 Background

This Section presents a brief overview of the peer discovery in BitTorrent. This is important as the mechanisms BitTorrent uses to discover peers impact the structure of the P2P network, and consequently the data dissemination. Thus, the knowledge of the peer discovery in BitTorrent is fundamental for a clear understanding of traffic localization. It is not our intention to present a complete overview of the BitTorrent protocol, as the reader may find it in [7,15].

Traditionally, BitTorrent employs a *tracker*, or central server, in order to discover peers and coordinate file exchanges. Peers retrieve the address of the tracker from within a *torrent* file they download from the web, i.e., a meta data file that contains useful information for the file exchange. Initially, a peer contacts the tracker to retrieve a list of peers participating in the *swarm*, i.e., the group of peers that hold the file or a portion of it. The tracker answers with the *peer-list*, a random subset of active peers generally composed by 50 peers. Afterwards, a peer periodically interacts with a tracker in order to send information about the volume of bytes it has downloaded and uploaded. As an answer, the tracker sends to the peer a new peer-list. The frequency of communication between client and tracker is regulated by the tracker via the *min_interval* field contained in the tracker replies. Usually, it is set to 15 minutes.

Recently, BitTorrent introduced *decentralized tracking*, a feature that enables any peer to act as a tracker, by mean of a Distributed Hash Table (DHT)[1] [10]. The BitTorrent DHT is used to store and locate information about which peers hold what files. Each peer and file is assigned a unique identifier computed using a hash function. We call *nodeID* the identifier of a peer and *info_hash* the identifier of a file. Both identifiers are 160-bit long and share the same hash space [20].

Recent BitTorrent client implementations use both the central tracker and the DHT in order to discover peers. In the first place, the DHT was intended to be a backup source of peers in case the tracker is unreachable. However, in some cases the file exchange is performed only relying on the DHT. This happens when users download from the web *magnet links*, that are pointers to the info_hash of a file. This scenario is becoming very frequent since popular torrent indexing websites started to also index magnet links.

Beside the tracker and the DHT, the Peer-Exchange-Protocol (PEX) is the third mechanism to discover peers that participate in a file exchange. The PEX allows peers that download the same file to exchange their peer-sets via gossiping.

4 The Role of DHTs in the BitTorrent Network

In this Section, we aim to answer the following question: *how many BitTorrent users rely on the DHT only in order to manage their file exchanges?* Therefore, we now overview some results obtained by monitoring the BitTorrent traffic in a large ISP.

4.1 Methodology and Data Collection

Our methodology is to intercept the client-to-tracker traffic at ISP scale while monitoring both the Azureus and Mainline DHT.

We intercept the client-to-tracker traffic by setting up at ISP border routers several filtering rules that match the IP addresses of popular trackers. The rationale of this measurement strategy is that popular trackers do not reside at the ISP where we collect traces. Since we cannot set up an unlimited number of filtering rules at an ISP border router, we only focus on the most popular trackers. We identify them by crawling the torrents files stored at the major torrent indexing websites, namely PirateBay, BitTorrent, MiniNova, IsoHunt, SuprNova, and Vuze. We then rank the trackers according to the number of *recent* and *popular* torrents they host. We consider a torrent recent when it is less than one month old. We consider a torrent popular when it has more than five peers holding a complete copy (seeders).

We build a sample of about 300,000 torrents from which we extract the URLs and IP addresses of 4,000 trackers. We then set up 2,000 filtering rules matching the IP addresses of the most popular trackers at few border routers of a large ISP in Europe. IP address ranges within the ISP are statically attributed to the ISP border routers. Thus, we monitor the portion of ISP subscribers whose

[1] A DHT is a structured P2P network used for content storage and retrieval.

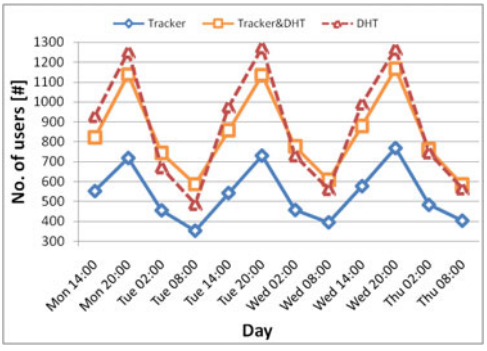


Fig. 1. Evolution over time of the number of tracker-based, DHT-based, and both tracker&DHT-based BitTorrent users

IP addresses fall in the IP ranges associated to the border routers we monitor. Accordingly, we track about 90,000 subscribers, i.e., 3.75% of the 2.4 Million subscribers of the ISP, over four consecutive weekdays. For each subscriber, we collect the information about responsiveness with a frequency of 15 minutes, i.e., the time interval clients report their activity to the trackers (cf. Section 2).

Meanwhile, we monitor both the Azureus and Mainline DHT using a crawler application. Our crawler is derived from the KAD crawler by Steiner et al. [24]. The crawler recursively queries each peer in the DHT for its neighbor list, starting from a bootstrap node. When no new peers are discovered, the crawler assumes that an entire snapshot of the network is obtained.

We gather a snapshot of the entire Azureus and Mainline DHTs every six hours, comprising more than 1 Million and 8 Million unique users at each point in time, respectively.

4.2 Data Analysis

We now analyze the behavior of each ISP subscriber by comparing the client-to-tracker traces and the DHT traces. Accordingly, at a given time t we classify each subscriber who appears to be active in the BitTorrent network as either:

- *tracker-based* - if it only exchanges messages with the central tracker.
- *tracker&DHT-based* - if it exchanges messages with the tracker and it is active in the DHT.
- *DHT-based* - if it is only active in the DHT.

Figure 1 shows the evolution over time of the number of BitTorrent users from the monitored ISP that are tracker-based, tracker&DHT-based, and DHT-based. Globally, Figure 1 shows a daily cycle typical of Internet-based applications: low activity during the early morning, increase towards the end of the day, and then decrease during the night. Figure 1 shows a previously unreported result: the relative majority of BitTorrent users (between 34 and 41%) only rely on the DHT in order to manage file exchanges. These users have probably retrieved

a magnet link on the Internet which allows them to avoid the communication with the tracker. A slightly smaller fraction of BitTorrent users are concurrently connected to both the DHT and the tracker. This is the usual behavior of a BitTorrent client: it contacts the tracker and the DHT in parallel. Finally, the minority of BitTorrent users (between 23 and 25%) only rely on the tracker to coordinate a file exchange. This behavior can be associated to peers that: (1) run old BitTorrent clients not yet supporting the DHT, (2) cannot access the DHT due to NAT traversal or bootstrapping issues.

5 DHT Traffic Localization

This Section presents the design of the first localization mechanism for BitTorrent networks that rely on DHT-based tracking. First, we overview the design. Then, we discuss its specific implementation for the Mainline DHT [18].

5.1 Overview

A naive approach to traffic localization for DHT-based BitTorrent networks consists in modifying the tracker functionality implemented at the peers. Precisely, a peer that receives a request for addresses of peers holding a certain file could include in its answer only the peers located at the same ISP as the requesting peer. The major limitation of this approach is that it requires a modification of each BitTorrent client implementation.

The key design rationale of our localization mechanism is that it does not require the modification of any BitTorrent client implementation. Our localization mechanism works in two steps. First, we intercept all the messages from peers announcing in the DHT that they hold a file or a portion of it. Then, we intercept all the requests for these files and answer with *local* peer-sets, i.e., sets of peers located at the same ISPs as the requesting peers. We now describe both steps in detail.

A single entity can join a P2P network many times with many distinct logical identities. These identities are called *sybils* [11]. In order to intercept announces and requests for a popular file, we insert in the DHT several sybils with nodeIDs close to the info_hash of the file [23]. We only focus on popular files for two reasons. First, it is infeasible to monitor all files available in a DHT, as this would require to introduce millions of sybils with the consequence of very high load at the machines responsible for the sybils. Second, monitoring each file is unnecessary, given that the majority of the BitTorrent traffic as well as the only traffic that can be localized is associated with few popular files [8].

Once the first step is in place, the sybils are constantly aware of the peers that hold the popular files as well as the peers requesting them. Under this premise, localization is straightforward. The sybils simply need to respond to the queries for popular files with localized peer-sets. In case just few local peers are available, a peer-set is completed with external peers.

5.2 Mainline Implementation

As a proof of concept, we implement a prototype of our localization mechanism for the Mainline DHT. We pick the Mainline DHT since it has the largest user base with more than 8 Million active users at any point in time (cf. Section 4.1). Note that our localization mechanism can be implemented for any other DHT-based P2P network, e.g., Azureus [3], Emule [12], and Ares [2].

The Mainline DHT implements a simple remote procedure call mechanism based on back-to-back query and answer packets. We now summarize the main remote procedure calls available in the Mainline DHT. For more details, the interested reader is referred to [18].

`ping(dest_IP:port, src_ID)` - verifies if a peer is alive and responsive. A peer that receives a `ping` message learns also about the existence of the peer with nodeID `src_ID`.

`find_node(I,src_ID)` - requests the closest peers to a hash value I in order to populate the routing tables of the requesting peer. A peer responds to a `find_node` message with the IP addresses, ports, and nodeIDs of the peers whose nodeIDs are the closest to I in its routing tables.

`get_peers(I,src_ID)` - retrieves information about a file F with info_hash I. The nodeID of the querying peer is included in the message (`src_ID`). The `get_peers` remote procedure call works iteratively. At each intermediary hop of the iteration, peers respond to a `get_peers` message with the IP addresses, ports, and nodeIDs of the peers closest to I. At the final hop of the iteration, peers return the IP addresses and ports of the peers that hold a copy or a portion of F (see next bullet). In the latter case, the BitTorrent client might then request the content from these hosts. Note that a response to a `get_peers` message can contain both closer peers to I and sources of F.

`announce_peer(IP:port,I)` - a peer, identified by "IP:port", announces that it holds a file (or a portion of it) with info_hash I. The object of the publication is the tuple *<IP:port,I>*. A peer sends `announce_peer` messages to the k peers whose nodeIDs are the closest to I. The value of k can be 3 or 8 according to the client implementation. These k closest peers have been previously looked-up using `get_peers` messages. The tuple *<IP:port,I>* expires after a time-out that depends on the client implementation (15 or 30 minutes are typical values). The announcing peer is responsible to re-announce the tuple *<IP:port,I>* over time.

In the remainder of this Section, we focus on the localization of the traffic associated to a single file identified by info_hash I.

In the first step of the localization mechanism, we intercept the `announce_peer` messages for I. To do so, we insert sybils in the DHT with nodeIDs closer to I than any real peer. The sybils share the first 47 bits of their nodeIDs with I ensuring that the probability that another peer in the Mainline DHT (8 to 11 Million peers) is closer to I than our sybils is $1 - ((1 - 2^{-47})^{11000000}) \simeq 10^{-8}$. We construct the nodeIDs of the sybils by varying the bits of their nodeIDs in the bit interval 48-56, thereby creating 256 sybils for I. Note that, in theory, k sybils should be enough to control an info_hash. However, it has been shown that peers fail to always find the k closest peers to an info_hash [13]. In order to

guarantee that our sybils are always among the k closest peers discovered, we use 256 sybils.

The insertion of the sybils in the DHT consists of informing real peers whose nodeIDs are close to I about the existence of the sybils. These peers will then propagate this information to other peers in the DHT. We proceed as follows.

- First, we discover the peers whose nodeIDs falls within Z, the portion of the hash-space that shares a prefix of at least z-bit with I. To do so, we send out multiple `get_peer` messages with target info_hashes close to I.
- Successively, we send `ping` messages with `src_ID` equal to the nodeIDs of the sybils to all the peers in Z. This operation populates the routing tables of the nodes in Z with information about our sybils.

The information derived from the received `announce_peer` messages is stored as <*info_hash,nodeID,IP:port,ISP*> four-tuple in a database common to all the sybils. We use Maxmind [19] to resolve a peer's IP address to its ISP. For an entry in the database, we use the same timeout policy as currently implemented by the BitTorrent clients.

In the second step of the localization, we intercept the `get_peers` messages for I and we reply to them with local peer-sets. Similarly to the `announce_peer` messages, the sybils also intercept the `get_peers` messages at the final hop of their iteration along the DHT. We construct the replies to the `get_peers` messages as follows. First, we determine the ISP of a querying peer using Maxmind. Then, we form the peer-set to be returned searching in the shared database for peers located at the same ISP as the requesting peer. In case not enough local peers are found, we complete a peer-set with external peers.

In order to localize more than one file, the outlined procedure needs to be repeated. Resource consumption scales linearly with the number of files to localize, unless their info_hashes are very close to each other. In the latter case, sybils for close-by info_hashes can be re-utilized to localize the traffic for multiple files.

6 Evaluation

This Section preliminary evaluates the proposed localization mechanism for DHT-based BitTorrent networks. Our goal is twofold: 1) quantify the volume of traffic that we can localize, and 2) identify possible limitations of the localization mechanism when running in the wild.

6.1 Methodology

The evaluation works in three steps. First, we run our prototype in our data center in Chicago to attract the `announce` and `get_peer` messages in the Mainline DHT for the most popular file as reported by the PirateBay website [25] on November 16th 2010. In the following, we refer to this file as F. We count the `announce` and `get_peer` messages to derive the number of unique peers (identified by the couple <*IP address, peerID*>) interested in F at each ISP.

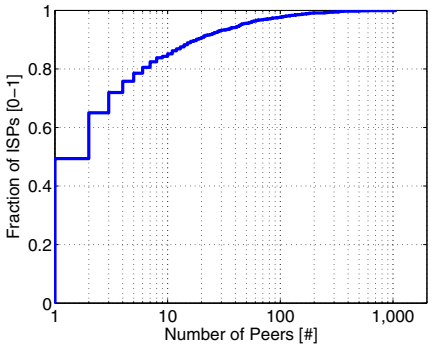


Fig. 2. CDF of peer distribution across the ISPs

This measure gives us insights about the popularity distribution of F across the ISPs. Figure 2 plots the Cumulative Distribution Function (CDF) of the number of peers that participate to the swarm of F per ISP at a given time. Since the distribution does not significantly change over seven days, we only plot the data for the initial distribution. Figure 2 shows that in half of the ISPs just one peer at a time is interested in F, i.e., no localization is possible. In 20% of the ISPs, 7 to 1,500 concurrent peers are interested in F. This means that only in those ISPs there is some localization potential for F.

In the second step, we activate the traffic localization mechanism for the following ISPs: *Comcast Cable*, *SBC Internet Service* and *Telefonica de Espana* (abbreviated Comcast, SBC and Telefonica in the following). Any BitTorrent user located at these ISPs who attempts to download F receives localized peer-sets. The localized file F is extremely popular at Comcast where are located about 10% of the available worldwide sources (i.e., between 1,000 and 1,500 sources according to day and time), popular at SBC with between 500 and 600 sources, and relatively unpopular at Telefonica, where we measure only 30 to 50 available sources.

In the third step of the evaluation, we instrument a Transmission client [26] to repetitively download F every 90 minute; the download time is bound by a timeout set to 30 minutes. For each peer the client is uploading and downloading from, the instrumented client logs every two seconds the following statistics: upload/download rate and client location (local or non-local). We run the instrumented Transmission client on a machine connected to a private cable connection provided by Comcast and on two PlanetLab [22] machines associated to SBC and Telefonica, respectively. The experiments run seven days at SBC and Telefonica whereas they only run 24 hours in Comcast due to a download cap imposed on the private cable connection. In the experiments, we only enable the DHT for tracker operations by disabling the communication between the instrumented client and the central trackers. This configuration reproduces a scenario where a user clicks on a magnet link (cf. Section 3). We also disable the Peer-Exchange-Protocol. Even though this configuration is not realistic, it is still useful as a

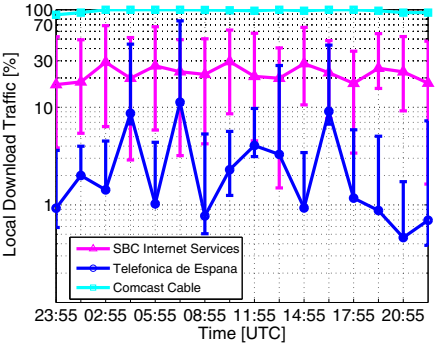


Fig. 3. Local Download Traffic ; [Comcast, SBC, Telefonica]

preliminary benchmark of the localization performance while providing better control on the experiments.

6.2 Results

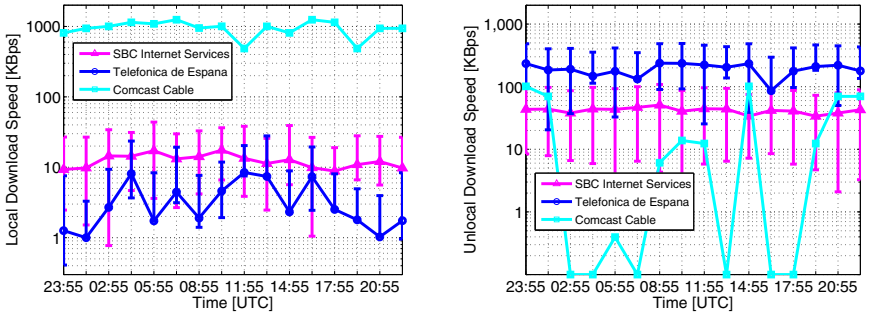
To start with, we analyze the traffic localization benefits at each ISP where we run our experiments. Figure 3 shows the percentage of incoming traffic per download that stays local, i.e., within the ISP. We only focus on the download traffic because it dominates by far the total volume of traffic[2]. Each point on the curves is the median local download traffic at a given time based on a seven day measure. The error bars refer to the 25 and 75th percentile, respectively[3].

Globally, Figure 3 shows an unexpected result: the traffic localization does not reach 100% all the time. On average, 99% of the traffic stays local in Comcast, whereas only 3 to 25% stay local in Telefonica and SBC for half of the experiments. If we have a look at the 75th percentiles, we can see that the percentage of local download traffic grows up to 15% in Telefonica and 60% in SBC, on average. It follows that the localization works better at ISPs where F is more popular, where the sybils can return a larger number of peers.

In these experiments, we expected to measure 100% of localization at each ISP as the client-to-tracker communication and the PEX protocol are disabled, i.e., the client should receive only peer-sets from our sybils. By inspecting the DHT control traffic collected at each machine, we find that additional peers beside the sybils replied to the `get_peer` messages sent by our clients. This implies that some peers in the DHT receive `announce` messages for F despite the presence of our sybils. This happens because several BitTorrent clients (such as BitComet [5]) do not properly implement the DHT announcement mechanism and thus do not correctly announce to our sybils. Precisely, they fail to lookup the closest peers to a given info_hash thus sending announce messages for F to

[2] This is because the swarm is over provisioned, i.e., many seeders are available.

[3] Given the experiments at Comcast last just one day, i.e., there is just one value per time-stamp, all percentiles coincide.



(a) Local Download Speed. (b) Non-local Download Speed.

Fig. 4. Speed Analysis ; [Comcast, SBC, Telefonica]

other peers than our sybils. Accordingly, our sybils compete with few other peers when returning peer-lists. In order to win this competition, it is crucial to be the first to respond to the requesting peer. It follows that beside the popularity of F within an ISP the network delay between the sybils and the requesting peer is also relevant.

In order to better understand each curve of Figure 3, we plot both the local and non-local speed measured per download and ISP (Figure 4). As above, each point in Figure 4(a) and 4(b) is the median local and non-local download speed over seven days and the error bars indicates the 25th and 75th percentiles. We start by focusing on the experiments performed at Comcast. In Comcast, traffic localization reaches 100% in 6 out of the 16 download attempts and the local download speed is constant around 1MBps; in the remaining download attempts, some external peers provide a maximum of two percent of the file with a download speed of few KBps.

We now focus on SBC and Telefonica. At SBC, the median local download speed stays between 10 and 20KBps whereas the median non-local download speed stays between 40 and 50KBps. Conversely, at Telefonica the median local download speed is very low, between 1 and 9KBps, while the median non-local download speed stays between 80 and 200KBps, i.e., about four times more than the value measured for SBC. At this stage of the analysis, two explanations to this observation are possible. (1) A client at Telefonica receives fewer local sources from our sybils than a client at SBC (due to different popularity of F); thus, the non-local sources retrieved from external peers have much higher chances to contribute to the file download. (2) Peers located at Telefonica have low upload rate and cannot contribute much to the swarm.

The latter result triggered our curiosity about the impact of a traffic localization mechanism on the "Quality of Experience" (QoE) perceived by the user. We aim to answer the following question: *does traffic localization positively or negatively affects the user download time?* In order to answer this question, we

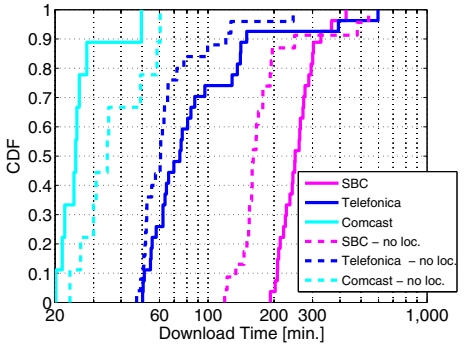


Fig. 5. Download Duration ; [Comcast, SBC, Telefonica]

turn off the localization mechanism and we re-run the experiments at each ISP for one day.

Figure 5 plots the CDF of the download duration measured with and without localization at each ISP. Due to the 30 minutes timeout for a download attempt, if a download does not complete within 30 minutes we extrapolate the download duration using the average download speed computed during the download attempt. Figure 5 shows that in Comcast the download duration is systematically higher without traffic localization, e.g., the median download time significantly grows from 15 minutes up to 35 minutes. This indicates that maintaining traffic within Comcast provides clear benefits to the user experience. However, this is not the case for SBC and Telefonica where the download time is shorter when the localization mechanism is turned off. For example, the median download time measured with and without localization decreases from 75 to 60 minutes in Telefonica and from 260 to 170 minutes in SBC. This result indicates that traffic localization might not always be beneficial to the user QoE. The decrease of the download time without traffic localization is more evident in SBC. This suggests that the low fraction of traffic localized in Telefonica is not caused by the low upload rate of Telefonica customers but mostly by the low popularity of F in Telefonica.

While running the experiments with the localization mechanism disabled, we also measure the *intrinsic localization*, i.e., the amount of traffic naturally localized in BitTorrent. We find that, on average, 20% of the traffic stays local in Comcast, whereas no traffic at all stays local in both SBC and Telefonica.

7 Conclusion and Future Work

Recently, BitTorrent introduced a distributed tracking feature: the role of the trackers is distributed among peers via a Distributed Hash Table (DHT). This paper claims that existing solutions for BitTorrent traffic localization leveraging central trackers will become soon ineffective. In order to support this claim, we monitor over four weekdays the BitTorrent activity of 90,000 subscribers at a

large ISP in Europe. We find that about 40% of the BitTorrent users located at the monitored ISP already rely on the DHT only for tracker operations.

Motivated by this observation, we design, prototype, and preliminary evaluate the first traffic localization mechanism for DHT-based BitTorrent networks. This mechanism constantly intercepts the announce messages in the DHT for popular files in order to discover worldwide peers holding these files. Then, it constantly intercepts requests for these popular files and responds with replies that contain only peers from the ISP of the requesting peer (when available).

We evaluate our design by localizing the traffic associated with a popular file in the Mainline DHT. We then measure the amount of traffic localized by running an instrumented BitTorrent client on machines located at the following ISPs: "Comcast Cable", "SBC Internet Service" and "Telefonica de Espana". The evaluation shows that the proposed localization mechanism performs well in the wild, however it deals with the issue that several BitTorrent clients do not implement the DHT protocol correctly. This prevents our localization mechanism from keeping 100% of the BitTorrent traffic within an ISP in some cases.

For future work, we aim to investigate the latter problem more in order to understand whether the system design can be further improved. Accordingly, we plan to systematically evaluate our traffic localization mechanism as follows. First, we will analyze the impact of torrent popularity within an ISP. Second, we will expand the measurements of the traffic localization benefits from just three ISPs to a larger number. Third, we will analyze the impact of the Peer-Exchange-Protocol on the traffic localization benefits.

Another avenue for future work is understanding the impact of BitTorrent traffic localization on user Quality of Experience (QoE). Is traffic localization going to decrease download times and therefore increase users QoE? In this paper, we showed that a user located at a large ISP (Comcast Cable) with a relatively fast subscription profile always benefits of faster downloads if its traffic is maintained within its ISP. However, this result was not confirmed at SBC Internet Service and Telefonica de Espana. We aim to investigate this research question more and extend these preliminary results to other ISPs as well.

Finally, we intend to extract the information about the popularity of content directly from the DHT. In this work, we assumed that the content popularity distribution is given, e.g., can be learned from the torrent indexing websites. However, this methodology has the following limitations: (1) content popularity can be different at different ISPs (e.g., due to language differences), (2) content popularity in the DHT might deviate from the content popularity as reported by torrent indexing websites, and (3) as tracker usage decreases over time content popularity will be only measurable through the DHT.

References

1. Aggarwal, V., Feldmann, A., Scheideler, C.: Can ISPs and P2P users cooperate for improved performance? CCR 37(3), 29–40 (2007)
2. Ares, `www.ares.net/`
3. Azureus/Vuze, `www.azureus.sourceforge.net/`

4. Bindal, R., Cao, P., Chan, W., Medved, J., Suwala, G., Bates, T., Zhang, A.: Improving traffic locality in bittorrent via biased neighbor selection. In: ICDCS, Lisbona, Portugal (July 2006)
5. Bitcomet, http://www.bitcomet.com/
6. Choffnes, D.R., Bustamante, F.E.: Taming the torrent: a practical approach to reducing cross-isp traffic in peer-to-peer systems. In: SIGCOMM, Seattle, WA, USA (August 2008)
7. Cohen, B.: Incentives Build Robustness in BitTorrent. Technical report, bittorrent.org. (2003)
8. Cuevas, R., Laoutaris, N., Yang, X., Siganos, G., Rodriguez, P.: Deep Diving into BitTorrent Locality. In: CoNEXT, Rome, Italy (December 2009)
9. Dabek, F., Cox, R., Kaashoek, F., Morris, R.: Vivaldi: A Decentralized Network Coordinate System. In: SIGCOMM, Portland, Oregon, USA (August 2004)
10. Dabek, F., Zhao, B., Druschel, P., Towards, I.S.: A Common API For Structured Peer-to-Peer Overlays. In: Kaashoek, M.F., Stoica, I. (eds.) IPTPS 2003. LNCS, vol. 2735, pp. 33–44. Springer, Heidelberg (2003)
11. Douceur, J.R.: The sybil attack. In: Druschel, P., Kaashoek, M.F., Rowstron, A. (eds.) IPTPS 2002. LNCS, vol. 2429, p. 251. Springer, Heidelberg (2002)
12. eMule, http://www.emule-project.net/
13. Kang, H., Chan-Tin, E., Hopper, N., Kim, Y.: Why KAD lookup fails. In: P2P, pp. 121–130 (November 2009)
14. Ledlie, J., Gardner, P., Seltzer, M.: Network coordinates in the wild. In: NSDI, Cambridge, MA, USA (April 2007)
15. Legout, A., Urvoy-Keller, G., Michiardi., P.: Rarest First and Choke Algorithms Are Enough. In: IMC, Rio De Janeiro, Brazil (October 2006)
16. Li, J., Sollins, K.: Exploiting autonomous system information in structured peer-to-peer networks. In: ICCCN, Chicago, IL, USA (October 2004)
17. Mainline BitTorrent, http://www.bittorrent.org/
18. Mainline DHT Specification, http://www.bittorrent.org/beps/bep_0005.html
19. Maxmind, http://www.maxmind.com/
20. Maymounkov, P., Mazieres, D.: Kademlia: A Peer-to-peer information system based on the XOR metric. In: Druschel, P., Kaashoek, M.F., Rowstron, A. (eds.) IPTPS 2002. LNCS, vol. 2429. Springer, Heidelberg (2002)
21. Nakao, A., Peterson, L., Bavier, A.: A routing underlay for overlay networks. In: SIGCOMM, Karlsruhe, Germany (August 2003)
22. Planetlab, http://www.planet-lab.org/
23. Steiner, M., Biersack, E.W., En-Najjary, T.: Exploiting KAD: Possible Uses and Misuses. Computer Communication Review 37(5) (October 2007)
24. Steiner, M., Biersack, E.W., En-Najjary, T.: Long Term Study of peer behavior in the the KAD DHT. Transactions on Networking 17(6), 1371–1384 (2009)
25. The PirateBay, http://www.thepiratebay.org/
26. Transmission, http://www.transmissionbt.com/
27. uTorrent, http://www.utorrent.com/
28. Xie, H., Yang, Y.R., Krishnamurthy, A., Liu, Y.G., Silberschatz, A.: P4P: provider portal for applications. In: SIGCOMM, Seattle, WA, USA (August 2008)