Inside Dropbox: Understanding Personal Cloud Storage Services

Idilio Drago
University of Twente
i.drago@utwente.nl

Marco Mellia
Politecnico di Torino
mellia@tlc.polito.it

Maurizio M. Munafò
Politecnico di Torino
munafo@tlc.polito.it

Anna Sperotto
University of Twente
a.sperotto@utwente.nl

Ramin Sadre
University of Twente
r.sadre@utwente.nl

Aiko Pras
University of Twente
a.pras@utwente.nl

ABSTRACT

Personal cloud storage services are gaining popularity. With a rush of providers to enter the market and an increasing offer of cheap storage space, it is to be expected that cloud storage will soon generate a high amount of Internet traffic. Very little is known about the architecture and the performance of such systems, and the workload they have to face. This understanding is essential for designing efficient cloud storage systems and predicting their impact on the network.

This paper presents a characterization of Dropbox, the leading solution in personal cloud storage in our datasets. By means of passive measurements, we analyze data from four vantage points in Europe, collected during 42 consecutive days. Our contributions are threefold: Firstly, we are the first to study Dropbox, which we show to be the most widely-used cloud storage system, already accounting for a volume equivalent to around one third of the YouTube traffic at campus networks on some days. Secondly, we characterize the workload users in different environments generate to the system, highlighting how this reflects on network traffic. Lastly, our results show possible performance bottlenecks caused by both the current system architecture and the storage protocol. This is exacerbated for users connected far from storage data-centers.

All measurements used in our analyses are publicly available in anonymized form at the SimpleWeb trace repository: `http://traces.simpleweb.org/dropbox/`

Categories and Subject Descriptors

C.2 [**Computer-Communication Networks**]: Miscellaneous; C.4 [**Performance of Systems**]: Measurement Techniques

General Terms

Measurement, Performance

IMC'12, November 14–16, 2012, Boston, Massachusetts, USA.

Keywords

Dropbox, Cloud Storage, Internet Measurement.

1. INTRODUCTION

Recent years have seen the introduction of cloud-based services [18], offering people and enterprises computing and storage capacity on remote data-centers and abstracting away the complexity of hardware management. We witness a gold rush to offer on-line storage capabilities, with players like Microsoft, Google and Amazon entering the market at the end of April 2012. They face a crowded scenario against popular solutions like Box.com, UbuntuOne, and Dropbox. The latter, active since 2007, currently counts over 50 million users, uploading more than 500 million files daily.[1]

It is thus not surprising that cloud storage has gained increasing momentum within research community. Some works explicitly consider system architecture design [12], while others focus on security and privacy issues concerning the storage of user data [19]. Considering commercial offers, little is known, with most players providing proprietary solutions and not willing to share information. Some studies present a comparison among different storage providers [13, 17]: by running benchmarks, they focus on the user achieved performance, but miss the characterization of the typical usage of a cloud service, and the impact of user and system behavior on personal storage applications.

In this paper, we provide a characterization of cloud-based storage systems. We analyze traffic collected from two university campuses and two Points of Presence (POP) in a large Internet Service Provider (ISP) for 42 consecutive days. We first devise a methodology for monitoring cloud storage traffic, which, being based on TLS encryption, is not straightforward to be understood. We then focus on Dropbox, which we show to be the most widely-used cloud storage system in our datasets. Dropbox already accounts for about 100GB of daily traffic in one of the monitored networks – i.e., 4% of the total traffic or around one third of the YouTube traffic at the same network. We focus first on the service performance characterization, highlighting possible bottlenecks and suggesting countermeasures. Then, we detail user habits, thus providing an extensive characterization of the workload the system has to face.

To be best of our knowledge, we are the first to provide an analysis of Dropbox usage on the Internet. The authors

[1] `http://www.dropbox.com/news`

of [11] compare Dropbox, Mozy, Carbonite and CrashPlan, but only a simplistic active experiment is provided to assess them. In [16], the possibility of unauthorized data access and the security implications of storing data in Dropbox are analyzed. We follow a similar methodology to dissect the Dropbox protocol, but focus on a completely different topic. Considering storage systems in general, [8, 9] study security and privacy implications of the deployment of data deduplication – the mechanism in place in Dropbox for avoiding the storage of duplicate data. Similarly, [1] presents a performance analysis of the Amazon Web Services (AWS) in general, but does not provide insights into personal storage. Finally, several works characterize popular services, such as social networks [7, 15] or YouTube [3, 6]. Our work goes in a similar direction, shedding light on Dropbox and possibly other related systems. Our main findings are:

- We already see a significant amount of traffic related to personal cloud storage, especially on campus networks, where people with more technical knowledge are found. We expect these systems to become popular also at home, where penetration is already above 6%.
- We highlight that Dropbox performance is mainly driven by the distance between clients and storage data-centers. In addition, short data transfer sizes coupled with a per-chunk acknowledgment mechanism impair transfer throughput, which is as little as 530kbits/s on average. A bundling scheme, delayed acknowledgments, or a finer placement of storage servers could be adopted to improve performance.
- Considering home users' behavior, four groups are clear: 7% of people only upload data; around 26% only download, and up to 37% of people do both. The remaining 30% abandon their clients running, seldom exchanging files.
- Interestingly, one of the most appreciated features of Dropbox is the simplified ability to share content: 30% of home users have more than one linked device, and 70% share at least one folder. At campuses, the number of shared folders increases, with 40% of users sharing more than 5 folders.

Our findings show that personal cloud storage applications are data hungry, and user behavior deeply affects their network requirements. We believe that our results are useful for both the research community and ISPs to understand and to anticipate the impact of massive adoption of such solutions. Similarly, our analysis of the Dropbox performance is a reference for those designing protocols and provisioning data-centers for similar services, with valuable lessons about possible bottlenecks introduced by some design decisions.

The remainder of this paper is organized as follows: Sec. 2 provides insight into the Dropbox architecture. Sec. 3 describes our data collection and compares the popularity of well-known cloud-based storage systems. Sec. 4 presents a characterization of Dropbox performance. User habits and the generated workload are presented in Sec. 5. While those sections mostly focus on the usage of the Dropbox client software, Sec. 6 discusses the less popular Web interface. Finally, Sec. 7 concludes this paper, and Appendix A provides some additional characteristics of Dropbox storage traffic.

2. DROPBOX OVERVIEW

2.1 The Dropbox Client

The Dropbox native client is implemented mostly in Python, using third-party libraries such as *librsync*. The application is available for Microsoft Windows, Apple OS X and Linux.[2] The basic object in the system is a *chunk* of data with size of up to 4MB. Files larger than that are split into several chunks, each treated as an independent object. Each chunk is identified by a SHA256 hash value, which is part of meta-data descriptions of files. Dropbox reduces the amount of exchanged data by using delta encoding when transmitting chunks. It also keeps locally in each device a database of meta-data information (updated via incremental updates) and compresses chunks before submitting them. In addition, the client offers the user the ability to control the maximum download and upload speed.

Table 1: Domain names used by different Dropbox services. Numeric suffixes are replaced by a `X` letter.

sub-domain	Data-center	Description
`client-lb/clientX`	Dropbox	Meta-data
`notifyX`	Dropbox	Notifications
`api`	Dropbox	API control
`www`	Dropbox	Web servers
`d`	Dropbox	Event logs
`dl`	Amazon	Direct links
`dl-clientX`	Amazon	Client storage
`dl-debugX`	Amazon	Back-traces
`dl-web`	Amazon	Web storage
`api-content`	Amazon	API Storage

Two major components can be identified in the Dropbox architecture: the *control* and the *data storage* servers. The former are under direct control of Dropbox Inc., while Amazon Elastic Compute Cloud (EC2) and Simple Storage Service (S3) are used as storage servers. In both cases, sub-domains of `dropbox.com` are used for identifying the different parts of the service offering a specific functionality, as detailed in Tab. 1. HTTPS is used to access all services, except the *notification* service which runs over HTTP.

2.2 Understanding Dropbox Internals

To characterize the usage of the service from passive measurements, we first gained an understanding of the Dropbox client protocol. We performed several active experiments to observe what information is exchanged after a particular operation. For instance, among others, we documented the traffic generated when adding or removing files on local folders, when downloading new files and when creating new folders. During our data collection, Dropbox client version 1.2.52 was being distributed as the stable version.[3]

Since most client communications are encrypted with TLS, and no description about the protocol is provided by Dropbox, we set up a local testbed, in which a Linux PC running the Dropbox client was instructed to use a *Squid* proxy server under our control. On the latter, the module *SSL-bump*[4] was used to terminate SSL connections and save decrypted traffic flows. The memory area where the Dropbox application stores trusted certificate authorities was modified at run-time to replace the original Dropbox Inc. certificate by the self-signed one signing the proxy server. By means of this setup, we were able to observe and to understand the Dropbox client communication.

[2]Mobile device applications access Dropbox on demand using APIs; those are not considered in this work.
[3]`http://www.dropbox.com/release_notes`
[4]`http://wiki.squid-cache.org/Features/SslBump`

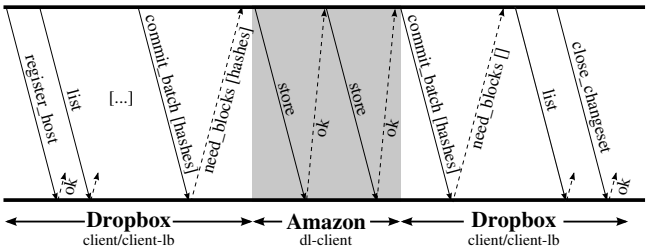


Figure 1: An example of the Dropbox protocol.

For instance, Fig. 1 illustrates the messages we observed while committing a batch of chunks. After registering with the Dropbox control center via a `clientX.dropbox.com` server, the *list* command retrieves meta-data updates. As soon as new files are locally added, a *commit_batch* command (on `client-lb.dropbox.com`) submits meta-data information. This can trigger *store* operations, performed directly with Amazon servers (on `dl-clientX.dropbox.com`). Each chunk *store* operation is acknowledged by one *OK* message. As we will see in Sec. 4, this acknowledgment mechanism might originate performance bottlenecks. Finally, as chunks are successfully submitted, the client exchanges messages with the central Dropbox servers (again on `client-lb.dropbox.com`) to conclude the transactions. Note that these messages committing transactions might occur in parallel with newer *store* operations.

A complete description of the Dropbox protocols is outside the scope of this paper. We, however, exploit this knowledge to tag the passively observed TCP flows with the likely commands executed by the client, even if we have no access to the content of the (encrypted) connections. In the following, we describe the protocols used to exchange data with the Dropbox control servers and with the storage servers at Amazon.

2.3 Client Control Flows

The Dropbox client exchanges control information mostly with servers managed directly by Dropbox Inc. We identified three sub-groups of control servers: (i) Notification, (ii) meta-data administration, and (iii) system-log servers. System-log servers collect run-time information about the clients, including exception back-traces (via Amazon, on `dl-debug.dropbox.com`), and other event logs possibly useful for system optimization (on `d.dropbox.com`). Since flows to those servers are not directly related to the usage of the system and do not carry much data, they have not been considered further. In the following we describe the key TCP flows to the meta-data and notification servers.

2.3.1 Notification Protocol

The Dropbox client keeps continuously opened a TCP connection to a notification server (`notifyX.dropbox.com`), used for receiving information about changes performed elsewhere. In contrast to other traffic, notification connections are *not encrypted*. Delayed HTTP responses are used to implement a push mechanism: a notification request is sent by the local client asking for eventual changes; the server response is received periodically about 60 seconds later in case of no change; after receiving it, the client immediately sends a new request. Changes on the central storage are instead advertised as soon as they are performed.

Each device linked to Dropbox has a unique identifier (*host_int*). Unique identifiers (called *namespaces*) are also used for each shared folder. The client identifier is sent in notification requests, together with the current list of namespaces. Devices and number of shared folders can, therefore, be identified in network traces by passively watching notification flows. Finally, different devices belonging to a single user can be inferred as well, by comparing namespace lists.

2.3.2 Meta-data Information Protocol

Authentication and file meta-data administration messages are exchanged with a separate set of servers, (`client-lb.dropbox.com` and/or `clientX.dropbox.com`). Typically, synchronization transactions start with messages to meta-data servers, followed by a batch of either *store* or *retrieve* operations through Amazon servers. As data chunks are successfully exchanged, the client sends messages to meta-data servers to conclude the transactions (see Fig. 1). Due to an aggressive TCP connection timeout handling, several short TLS connections to meta-data servers can be observed during this procedure.

Server responses to client messages can include general control parameters. For instance, our experiments in the testbed reveal that the current version of the protocol limits the number of chunks to be transferred to at most 100 per transaction. That is, if more than 100 chunks need to be exchanged, the operation is split into several batches, each of at most 100 chunks. Such parameters shape the traffic produced by the client, as analysed in Sec. 4.

2.4 Data Storage Flows

As illustrated in Fig. 1, all *store* and *retrieve* operations are handled by virtual machines in Amazon EC2. More than 500 distinct domain names (`dl-clientX.dropbox.com`) point to Amazon servers. A subset of those aliases are sent to clients regularly. Clients rotate in the received lists when executing storage operations, distributing the workload.

Typically, storage flows carry either *store* commands or *retrieve* commands. This permits storage flows to be divided into two groups by checking the amount of data downloaded and uploaded in each flow. By means of the data collected in our test environment, we documented the overhead of *store* and *retrieve* commands and derived a method for labeling the flows. Furthermore, we identified a direct relationship between the number of TCP segments with the `PSH` flag set in storage flows and the number of transported chunks. Appendix A presents more details about our methodology as well as some results validating that the models built in our test environment represent the traffic generated by real users satisfactorily. We use this knowledge in the next sections for characterizing the system performance and the workload.

2.5 Web Interface and Other Protocols

Content stored in Dropbox can also be accessed through Web interfaces. A separate set of domain names are used to identify the different services and can thus be exploited to distinguish the performed operations. For example, URLs containing `dl-web.dropbox.com` are used when downloading private content from user accounts. The domain `dl.dropbox.com` provides public direct links to shared files. As shown in Sec. 6, the latter is the preferred Dropbox Web interface.

Table 2: Datasets overview.

Name	Type	IP Addrs.	Vol. (GB)
Campus 1	Wired	400	5,320
Campus 2	Wired/Wireless	2,528	55,054
Home 1	FTTH/ADSL	18,785	509,909
Home 2	ADSL	13,723	301,448

In addition, other protocols are available, like the LAN Synchronization Protocol and the public APIs. However, these protocols do not provide useful information for the aim of this paper. They are therefore mostly ignored in the remainder of this paper.

3. DATASETS AND POPULARITY

3.1 Methodology

We rely on passive measurements to analyze the Dropbox traffic in operational networks. We use Tstat [5], an open source monitoring tool developed at Politecnico di Torino, to collect data. Tstat monitors each TCP connection, exposing information about more than 100 metrics[5], including client and server IP addresses, amount of exchanged data, eventual retransmitted segments, Round Trip Time (RTT) and the number of TCP segments that had the `PSH` flag set [14].

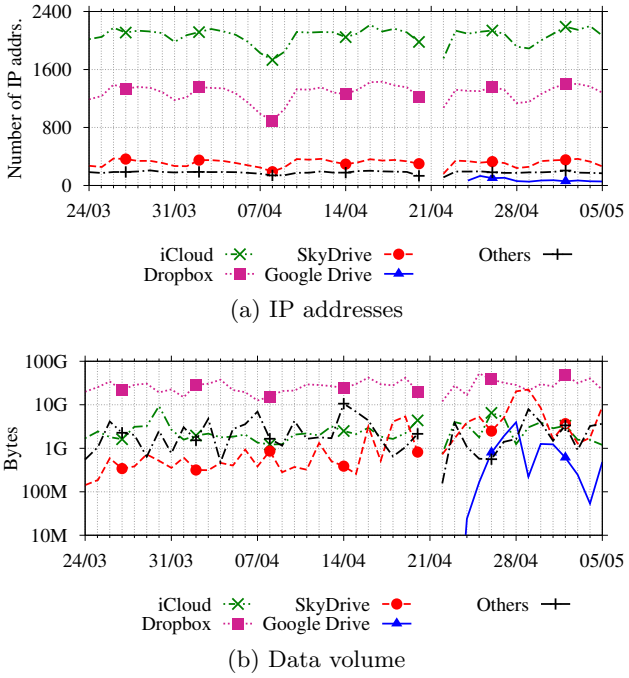
Specifically targeting the analysis of Dropbox traffic, we implemented several additional features. First, given that Dropbox relies on HTTPS, we extracted the TLS/SSL certificates offered by the server using a classic DPI approach. Our analysis shows that the string `*.dropbox.com` is used to sign all communications with the servers. This is instrumental for traffic classification of the services. Second, we augmented the exposed information by labeling server IP addresses with the original Fully Qualified Domain Name (FQDN) the client requested to the DNS server [2]. This is a key feature to reveal information on the server that is being contacted (see Tab. 1) and allows to identify each specific Dropbox related service. Third, Tstat was instructed to expose the list of device identifiers and folder namespaces exchanged with the notification servers.

3.2 Datasets

We installed Tstat at 4 vantage points in 2 European countries and collected data from March 24, 2012 to May 5, 2012. This setup provided a pool of unique datasets, allowing us to analyze the use of cloud storage in different environments, which vary in both the access technology and the typical user habits. Tab. 2 summarizes our datasets, showing, for each vantage point, the access technologies present in the monitored network, the number of unique client IP addresses, and the total amount of data observed during the whole period.

The datasets labeled *Home 1* and *Home 2* consist of ADSL and Fiber to the Home (FTTH) customers of a nation-wide ISP. Customers are provided with static IP addresses, but they might use WiFi routers at home to share the connection. *Campus 1* and *Campus 2* were instead collected in academic environments: *Campus 1* mostly monitors wired workstations in research and administrative offices of the Computer Science Department in a European university. *Campus 2* accounts for all traffic at the border routers of a second university, including campus-wide wireless access points and student houses. In this latter scenario, NAT and HTTP proxy-ing are very common, and DNS traffic was not exposed to the probe. For privacy reasons, our probes export only flows and the extra information described in the previous section. All payload data are discarded directly in the probe. To complete our analysis, a second dataset was collected in *Campus 1* in June/July 2012.

[5]See `http://tstat.tlc.polito.it` for details.



(a) IP addresses

(b) Data volume

Figure 2: Popularity of cloud storage in *Home 1*.

3.3 Popularity of Different Storage Providers

We present a comparison of the popularity of cloud-based storage systems. We explicitly consider the following services: *Dropbox*, *Google Drive*, *Apple iCloud* and *Microsoft SkyDrive*. Other less known services (e.g., *SugarSync*, *Box.com* and *UbuntuOne*) were aggregated into the *Others* group. We rely on both the extracted TLS server name and DNS FQDN to classify flows as belonging to each service.

We first study the popularity of the different services in terms of unique clients. We use the *Home 1* dataset because IP addresses are statically assigned to households and, therefore, are a reliable estimation of the number of installations. Fig. 2(a) reports[6] the number of distinct IP addresses that contacted at least once a storage service in a given day. iCloud is the most accessed service, with about 2,100 households (11.1%), showing the high popularity of Apple devices. Dropbox comes second, with about 1,300 households (6.9%). Other services are much less popular (e.g., 1.7% for SkyDrive). Interestingly, Google Drive appears immediately on the day of its launch (April 24th, 2012).

Fig. 2(b) reports the total data volume for each service in *Home 1*. Dropbox tops all other services by one order of magnitude (note the logarithmic y-scale), with more than 20GB of data exchanged every day. iCloud volume is limited

[6]A probe outage is visible on April 21, 2012.

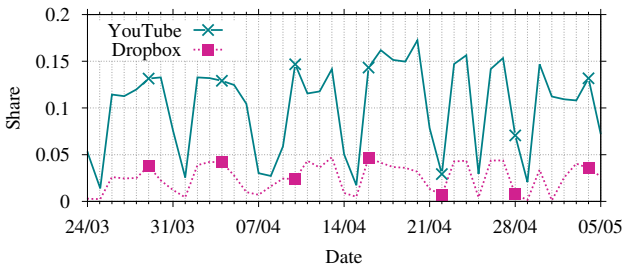


Figure 3: YouTube and Dropbox in *Campus 2*.

Table 3: Total Dropbox traffic in the datasets.

Name	Flows	Vol. (GB)	Devices
Campus 1	167,189	146	283
Campus 2	1,902,824	1,814	6,609
Home 1	1,438,369	1,153	3,350
Home 2	693,086	506	1,313
Total	4,204,666	3,624	11,561

despite the higher number of devices, because the service does not allow users to synchronize arbitrary files. SkyDrive and Google Drive show a sudden increase in volume after their public launch in April.

Fig. 3 compares Dropbox and YouTube share of the total traffic volume in *Campus 2*. Apart from the variation reflecting the weekly and holiday pattern, a high fraction is seen for Dropbox daily. Note that in this network the traffic exchanged with Dropbox is close to 100GB per working day: that is already 4% of all traffic, or a volume equivalent to about one third of YouTube traffic in the same day!

These findings highlight an increasing interest for cloud-based storage systems, showing that people are eager to make use of remote storage space. Cloud-based storage is already popular, with 6-12% of home users regularly accessing one or more of the services. Although the popularity of the recently launched systems may increase in the future, in the following we restrict our attention to Dropbox only, since it is by far the most used system in terms of traffic volume at the moment. Dropbox overall traffic is summarized in Tab. 3, where we can see the number of flows, data volume, and devices linked to Dropbox in the monitored networks. Our datasets account for more than 11,000 Dropbox devices, uniquely identified by their *host_int* (see Sec. 2.3.1). The traffic generated by the Web interface and by public APIs is also included. In total, more than 3.5TB were exchanged with Dropbox servers during our capture.

4. DROPBOX PERFORMANCE

4.1 Traffic Breakdown: Storage and Control

To understand the performance of the Dropbox service and its architecture, we first study the amount of traffic handled by the different sets of servers. Fig. 4 shows the resulting traffic breakdown in terms of traffic volume and number of flows. From the figure, it emerges that the Dropbox client application is responsible for more than 80% of the traffic volume in all vantage points, which shows that this application is highly preferred over the Web interfaces for exchanging data. A significant portion of the volume (from

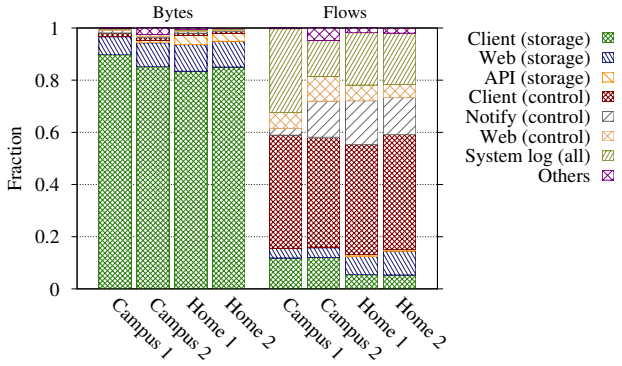


Figure 4: Traffic share of Dropbox servers.

7% to 10%) is generated by direct link downloads and the main Web interface (both represented as *Web* in Fig. 4). In home networks, a small but non-negligible volume is seen to the Dropbox API (up to 4%). Finally, the data volume caused by control messages is negligible in all datasets.

When considering the number of flows, the control servers are the major contributors (more than 80% of the flows, depending on the dataset). The difference on the percentage of notification flows – around 15% in *Campus 2*, *Home 1* and *Home 2*, and less than 3% in *Campus 1* – is caused by the difference in the typical duration of Dropbox sessions in those networks, which will be further studied in Sec. 5.5.

4.2 Server Locations and RTT

We showed in previous sections that the Dropbox client protocol relies on different servers to accomplish typical tasks such as file synchronization. Dropbox distributes the load among its servers both by rotating IP addresses in DNS responses and by providing different lists of DNS names to each client. In the following, we want to understand the geographical deployment of this architecture and its consequences on the perceived RTT.

4.2.1 Server Locations

Names in Tab. 1 terminated by a numerical suffix are normally resolved to a single server IP address[7] and clients are in charge of selecting which server will be used in a request. For instance, meta-data servers are currently addressed by a fixed pool of 10 IP addresses and notification servers by a pool of 20 IP addresses. Storage servers are addressed by more than 600 IP addresses from Amazon data-centers. Fig. 5 shows the number of contacted storage servers per day in our vantage points. The figure points out that clients in *Campus 1* and *Home 2* do not reach all storage servers daily. In both *Campus 2* and *Home 1*, more servers are instead contacted because of the higher number of devices on those vantage points. Routing information suggests that control servers are located in the U.S. West Coast (likely in California), while storage servers are in the U.S. East Coast (apparently in Amazon's Northern Virginia data-centers).

In order to verify the current Dropbox setup worldwide, we performed active measurements using the PlanetLab. By selecting nodes from 13 countries in 6 continents, we checked which IP addresses are obtained when resolving the Dropbox DNS names seen in our passive measurements, as well as the

[7]Meta-data servers are addressed in both ways, depending on the executed command, via `client-lb` or `clientX`.

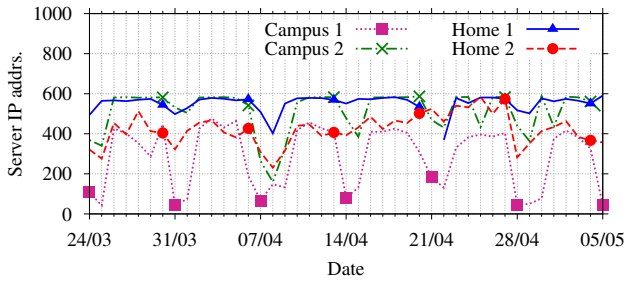


Figure 5: Number of contacted storage servers.

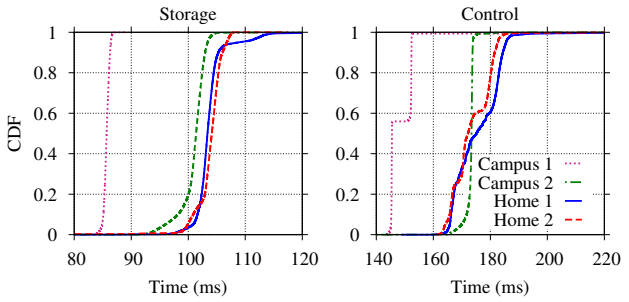


Figure 6: Distribution of minimum RTT of storage and control flows. Note the different x-axes.

routes and RTT when contacting the servers. The experiment shows that the same set of IP addresses is always sent to clients regardless of their geographical locations. This is valid for both control and storage domain names. Route information and RTT suggest that the same U.S. data-centers observed in our passive measurements are the only ones used worldwide. That is, Dropbox is, as for now, a service centralized in the U.S. Considering that more than half of the Dropbox clients are outside the U.S.[8], and the high amount of traffic observed in our vantage points, the traffic exchanged between the clients and the data-centers is likely to be already very relevant in the core network.

4.2.2 *Storage and Control RTT*

A deeper analysis of the RTT at our four vantage points reveals more details of the physical implementation of the Dropbox architecture. Sec. 4.4 will show that the RTT has a major impact on the service performance. Fig. 6 shows, separately for storage (left) and control flows (right), the CDF of the minimum RTT in flows where at least 10 RTT samples could be obtained (see [14]). The figure accounts only for the RTT between our probes and the servers, to filter out the impact of the access technologies (e.g., ADSL).

The RTTs to storage servers at Amazon remained stable during our measurements, meaning that no significant changes in the network topology happened. The differences in RTTs among the vantage points are related to the countries where the probes are located. This constant RTT during our 42 days of measurements is another strong indication that a single data-center was used by all users in our vantage points. The RTTs to the control servers are less constant. In both *Campus 1* and *Home 2*, the curve presents small

[8] `http://www.dropbox.com/news`

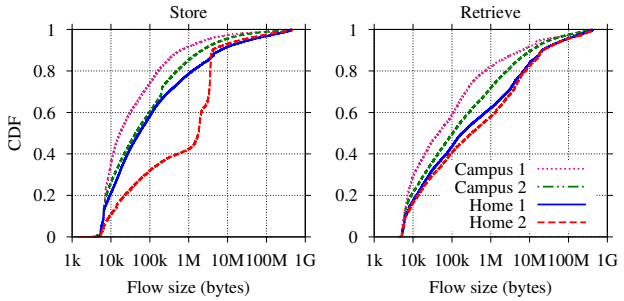


Figure 7: Distribution of TCP flow sizes of file storage for the Dropbox client.

steps (less than 10ms). We assume that they are caused by changes in the IP route, since the same behavior is not noticeable in all probes. Also in this case, the measurements hint to a central control server farm. Finally, it is interesting to note the high difference in the RTT between control and storage data-centers. This is probably caused by the physical distance between them inside the U.S.

4.3 *Retrieve* and *Store* Flows

4.3.1 *Flow Size*

As shown in Fig. 4, most traffic is generated by storage operations. Fig. 7 depicts the CDF of the flow size for storage operations. Since SSL is used, we observe a minimum flow size of approximately 4kB. The flows have a maximum size of approximately 400MB because of current run-time parameters of Dropbox: batches are limited to a maximum of 100 chunks, each smaller than 4MB, as described in Sec. 2.

From Fig. 7 it emerges that a significant percentage of flows (up to 40% in some cases) exchange less than 10kB, meaning that they are composed mostly by SSL handshake messages and a small amount of user data. A very high percentage of flows (varying from 40% to 80%) consist of less than 100kB. We conjecture that two factors are causing this behavior: (i) the synchronization protocol sending and receiving file *deltas* as soon as they are detected; (ii) the primary use of Dropbox for synchronization of small files constantly changed, instead of periodic (large) backups.

Comparing the CDFs for the *retrieve* and *storage* operations, we can see that *retrieve* flows are normally larger than the *store* ones. This is particularly visible in *Campus 1*, *Campus 2* and *Home 1* datasets. For instance, while 60% of *store* flows in *Home 1* have no more than 100kB, the percentage is about 40% for *retrieve* flows in the same network. This can be partially explained by the first batch synchronization happening when sessions are started. Besides that, we observed a high number of devices using Dropbox only for downloading content. This usage will be analyzed further in the coming sections.

We also highlight a remarkably discrepancy in the CDF for *store* flows in *Home 2*. A single device was submitting single chunks in consecutive TCP connections during several days in our capture. This caused the CDF to be strongly biased toward the maximum chunk size used by Dropbox (4MB). We could not determine whether this behavior is due to problems in the Dropbox client manifested in this single device, or another legitimate use of the service.

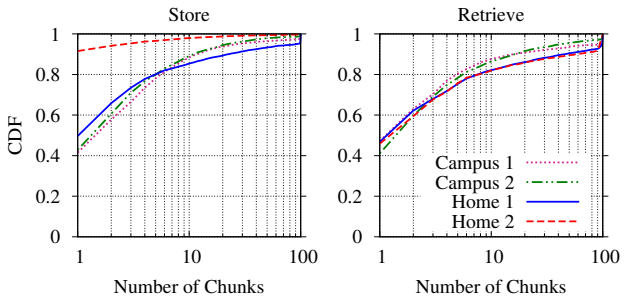


Figure 8: Distribution of the estimated number of file chunks per TCP flow.

4.3.2 Chunks per Batch

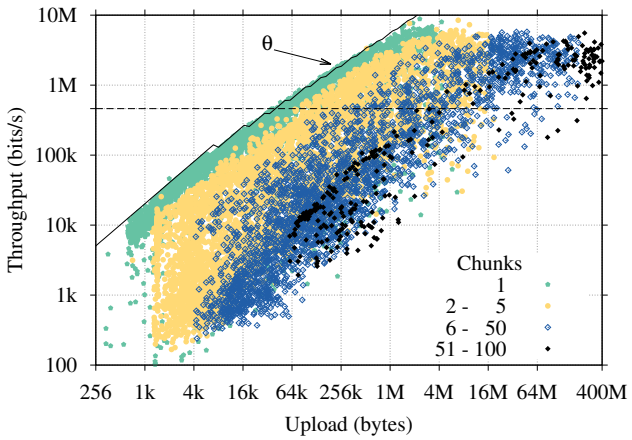
Fig. 8 depicts the CDF of the estimated number of chunks per flow. The curves show that most batches are composed by a small number of chunks. Storage flows have no more than 10 chunks in more than 80% of the cases in all datasets. *Home 2* distribution diverges because of the single client behaving abnormally described in the previous section. These distributions reinforce our conjecture about the dominance of deltas and small files in Dropbox usage habits: most flows are very small and composed by few chunks. Most of the remaining flows have the maximum allowed number of chunks per batch and, therefore, are strongly shaped by the protocol design of Dropbox.

4.4 Storage Throughput

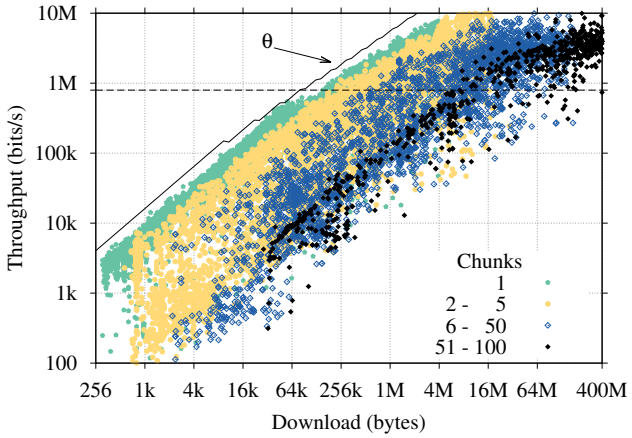
Our measurements in Sec. 4.2 indicate that Dropbox relies on centralized data-centers for control and storage. This raises the question on the service performance for users not located near those data-centers.

The throughput of the storage operations is certainly one of the key performance metrics. Fig. 9 depicts the throughput achieved by each storage flow in *Campus 2*. The figure shows separate plots for the *retrieve* and *store* operations. Similar plots would be obtained using *Campus 1* dataset. *Home 1* and *Home 2* are left out of this analysis since the access technology (ADSL, in particular) might be a bottleneck for the system in those networks. The x-axis represents the number of bytes transferred in the flow, already subtracting the typical SSL overheads (see Appendix A for details), while the y-axis shows the throughput calculated as the ratio between transferred bytes and duration of each flow (note the logarithmic scales). The duration was accounted as the time between the first TCP SYN packet and the last packet with payload in the flow, ignoring connection termination delays. Flows are represented by different marks according to their number of chunks.

Overall, the throughput is remarkably low. The average throughput (marked with dashed horizontal lines in the figure) is not higher than 462kbits/s for *store* flows and 797kbits/s for *retrieve* flows in *Campus 2* (359kbits/s and 783kbits/s in *Campus 1*, respectively). In general, the highest observed throughput (close to 10Mbits/s in both datasets) is only achieved by flows carrying more than 1MB. Moreover, flows achieve lower throughput as the number of chunks increases. This can be seen by the concentration of flows with high number of chunks in the bottom part of the plots for any given size.



(a) Store



(b) Retrieve

Figure 9: Throughput of storage flows in *Campus 2*.

TCP start-up times and application-layer sequential acknowledgments are two major factors limiting the throughput, affecting flows with a small amount of data and flows with a large number of chunks, respectively. In both cases, the high RTT between clients and data-centers amplifies the effects. In the following, those problems are detailed.

4.4.1 TCP Start-up Effects

Flows carrying a small amount of data are limited by TCP slow start-up times. This is particularly relevant in the analyzed campus networks, since both data link capacity and RTT to storage data-centers are high in these networks. Fig. 9 shows the maximum throughput θ for completing the transfer of a specific amount of data, assuming that flows stayed in the TCP slow start phase. We computed the latency as in [4], with initial congestion window of 3 segments and adjusting the formula to include overheads (e.g., the 3 RTTs of SSL handshakes in the current Dropbox setup).

Fig. 9 shows that θ approximates the maximum throughput satisfactorily. This is clear for flows with a single chunk, which suffer less from application-layer impediments. The bound is not tight for *retrieve* flows, because their latency includes at least 1 server reaction time. Note that the through-

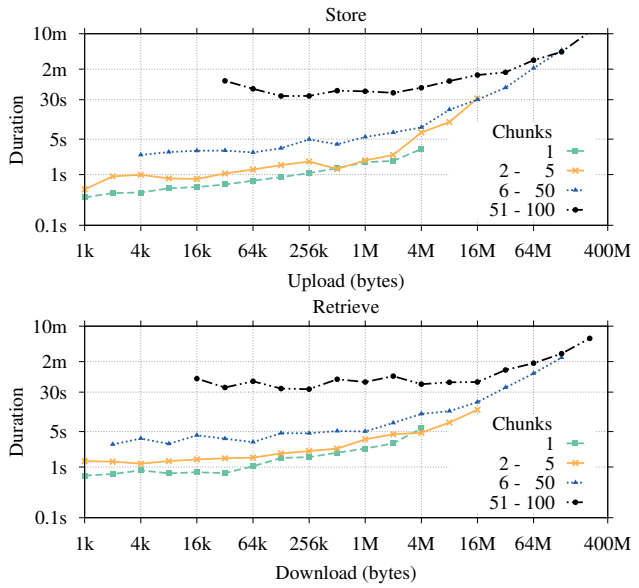


Figure 10: Minimum duration of flows with diverse number of chunks in *Campus 2*.

put can still be limited by other factors, such as the explicit user selection of transfer limits, or possible network congestion. However, when considering only flows with a single chunk, more than 99% of the *store* flows and around 95% of the *retrieve* flows in *Campus 1* have no TCP retransmissions. These percentages are lower in *Campus 2* (88% and 75%) because of the wireless access points. Single-chunk flows with no retransmissions experience throughput close to the limit (e.g., -17% on average for *store* flows in *Campus 1*), confirming that the TCP start-up is their main bottleneck.

4.4.2 Sequential Acknowledgments

Flows with more than 1 chunk have the sequential acknowledgment scheme (Fig. 1) as a bottleneck, because the mechanism forces clients to wait one RTT (plus the server reaction time) between two storage operations. Naturally, flows carrying a large number of small chunks suffer relatively more from this impediment than flows with large chunks. Fig. 8 shows that more than 40% of the flows have at least 2 chunks and are potentially affected by that.

Flows with several chunks are also affected by the previously described factors. Besides that, the Dropbox client keeps storage connections opened for a short interval after transferring a chunk, waiting for new chunks. Therefore, some flows in Fig. 9 might have a longer duration because they were reused during this idle interval. To remove these effects and highlight the impact of sequential acknowledgments, we divide the x-axis of Fig. 9 in slots of equal sizes (in logarithmic scale) and, for each slot, select the flow with maximum throughput in each of the four groups shown in the figure. Fig. 10 depicts the duration of these representative flows. Flows with more than 50 chunks, for instance, always last for more than 30s, regardless of their sizes. Considering the RTT in *Campus 2*, up to one third of that (5-10s) is wasted while application-layer acknowledgments are transiting the network. The remaining duration is mainly due to the server and the client reaction times between chunks.

Table 4: Performance in *Campus 1* before and after the deployment of a bundling mechanism.

	Mar/Apr		Jun/Jul	
	Median	Average	Median	Average
Flow size				
Store	16.28kB	3.91MB	42.36kB	4.35MB
Retrieve	42.20kB	8.57MB	70.69kB	9.36MB
Throughput (kbits/s)				
Store	31.59	358.17	81.82	552.92
Retrieve	57.72	782.99	109.92	1293.72

4.5 Implications and Recommendations

Our measurements clearly indicate that the application-layer protocol in combination with large RTT penalizes the system performance. We identify three possible solutions to remove the identified bottlenecks:

1. Bundling smaller chunks, increasing the amount of data sent per storage operation. Dropbox 1.4.0, announced in April 2012, implements a bundling mechanism, which is analyzed in the following;

2. Using a delayed acknowledgment scheme in storage operations, pipelining chunks to remove the effects of sequential acknowledgments;

3. Bringing storage servers closer to customers, thus improving the overall throughput.

Note that the first two countermeasures target only the application-layer bottleneck. The third one, while valid for any on-line service, would have the extra positive impact of removing heavy storage traffic from the core of the Internet.

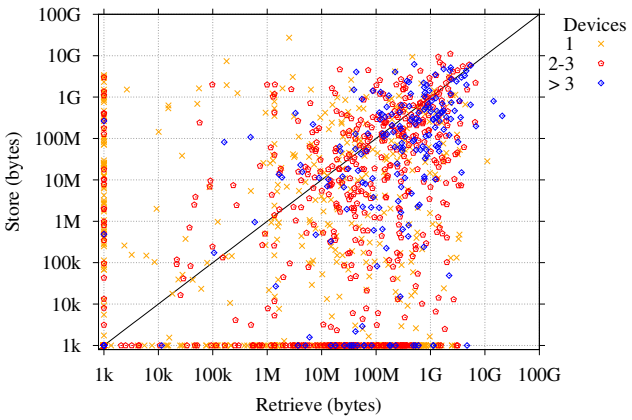
4.5.1 Improvements in Dropbox 1.4.0

The latest version of Dropbox adds two commands (*store_batch* and *retrieve_batch*), allowing several chunks to be submitted in a single operation.[9] Single-chunk commands are still in use: the system decides at run-time whether chunks will be grouped or not. We use extra data captured in *Campus 1* during June and July 2012 to quantify the effects of this mechanism on the system performance.

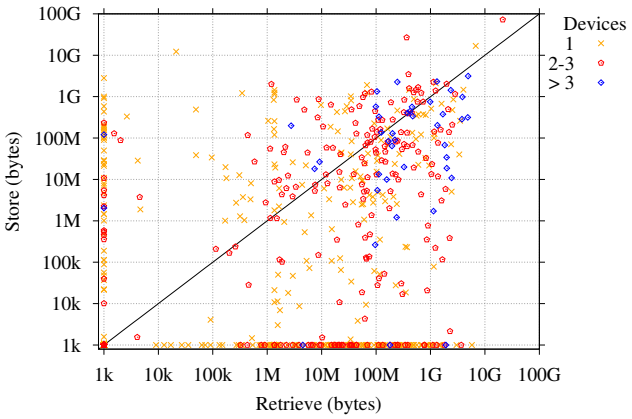
Tab. 4 compares the flow size and the throughput distributions before and after the deployment of Dropbox 1.4.0. The increase in the median flow size shows that flows become bigger, likely because more small chunks can be accommodated in a single TCP connection in this version. The averages are less affected, since only smaller flows profit from the mechanism. Both the median and the average throughput, on the other hand, are dramatically improved. The average throughput of *retrieve* flows, for instance, is around 65% higher in the newest dataset.

Since both the new batch commands and the old single-chunk commands are still executed sequentially, there seems to exist room for improvements in the protocol. A complete characterization of that is, however, out of scope of this paper. Such an analysis will be performed in future work, along with an study of the effectiveness and possible drawbacks of each of our recommendations.

[9]Number of chunks and TCP segments with the `PSH` flag set do not have a direct relation in this version anymore.



(a) Home 1



(b) Home 2

Figure 11: Data volume stored and retrieved from Dropbox in each household.

5. SERVICE WORKLOAD

5.1 Storage Volume

In this section we correlate ISP customers (IP addresses) in home networks and the total storage volume in *retrieve* and *store* operations. The amount of retrieved and stored data per household is depicted in Fig. 11. Each IP address is represented by a point and different symbols are used to illustrate the number of devices behind the IP address. Note that we placed all cases with less than 1kB on the axes because of the logarithmic scales. The figure accounts only for transfers made from the Dropbox client. Similarly to Sec. 4.4, the typical overhead of SSL negotiations were subtracted from the transferred amount.

Fig. 11 shows that Dropbox users tend to download more than upload. This is visible in the density of points below the diagonals. Overall, the total downloaded data in *Campus 2* is 2.4 times higher than the total uploaded data. This ratio is equal to 1.6 and 1.4 in *Campus 1* and *Home 1*, respectively. *Home 2* is an exception: The ratio is around 0.9 in this network. Some customers massively uploading content created this divergence. These customers appear on the top right corner of Fig. 11(b). Note that one of these customers is also responsible for the bias in the CDF depicted in Fig. 7.

In both datasets, four usage scenarios can be identified: (i) *Occasional* users, which abandon their Dropbox clients, hardly synchronizing any content (points close to the origins); (ii) *upload-only* users that mainly submit files (points close to the y-axes); (iii) *download-only* users, executing predominantly *retrieve* operations (points close to the x-axes); (iv) *heavy* users that both store and retrieve large amounts of data (points along the diagonals). The proportion of users in each group explains the overall relation between downloads and uploads seen in Fig. 11.

Tab. 5 quantifies the groups. The IP addresses are divided according to the following heuristics: IP addresses that have less than 10kB in both *retrieve* and *store* operations are included in the *occasional* group; IP addresses that have more than three orders of magnitude of difference between upload and download (e.g., 1GB versus 1MB) are included in either *download-only* or *upload-only*; all others are in the *heavy* group. For each group, the table shows the group percentage of IP addresses and sessions, the total transferred data, the average number of days in which the devices were seen on-line, and the average number of devices per household.

The *occasional* group represents around 30% of the total IP addresses in both vantage points. As expected, customers in this group exchange a negligible amount of data and stay on-line in Dropbox less time when compared to others. They also have the lowest average number of devices. This group, therefore, marginally generates any load to the system.

The *upload-only* group accounts for around 7% of the IP addresses, and is responsible for a significant amount of uploads (21% in *Home 1* and 11% in *Home 2*). Considering their low number of devices, users in this group seem to be interested in Dropbox for backups and for the submission of content to third-parties or to geographically dispersed devices. The opposite behavior can be concluded for the *download-only* group. This group is, however, very significant in both number of IP addresses (26% in *Home 1* and 28% in *Home 2*) and transferred volume (25% and 28%, respectively). Similarly to the *upload-only* group, a moderate number of devices per IP address is seen in this group.

Finally, accounting for 37% of IP addresses in *Home 1* and 33% in *Home 2*, the *heavy* group is responsible for most of the volume transferred by Dropbox clients. Customers in this group have a high number of devices (above 2 on average), appeared on-line more than 60% of the days in our capture and are responsible for more than 50% of the Dropbox sessions. The usage of Dropbox for synchronization of devices in a household seems to be the typical scenario in this group. Those are, therefore, the users causing the biggest impact on both system workload and network utilization.

5.2 Devices

In this section we describe the distribution of the number of devices residing in the same LAN. Devices connected to the same LAN can make use of the LAN Sync Protocol for synchronizing files without retrieving duplicated content from the cloud. Fig. 12 depicts the distribution of the number of devices per IP address in *Home 1* and *Home 2*.

In around 60% of the households using the Dropbox client, there is only a single device linked to the service. Most of the remaining households have up to 4 devices and, not surprisingly, are part of the *heavy* group identified in the previous section. By inspecting a subset of notification connections in *Home 1*, we observed that in around 60% of households

Table 5: Fraction of IP addresses and sessions, retrieved and stored data volume, average number of days on-line, and average number of devices of the different user groups in *Home 1* and *Home 2*.

Group	*Home 1*						*Home 2*					
	Addr.	Sess.	Retr.	Store	Days	Dev.	Addr.	Sess.	Retr.	Store	Days	Dev.
Occasional	0.31	0.15	-	-	16.37	1.22	0.32	0.18	-	-	15.52	1.13
Upload-only	0.06	0.06	-	84GB	19.74	1.36	0.07	0.04	-	26GB	20.42	1.37
Download-only	0.26	0.24	135GB	-	21.53	1.69	0.28	0.23	57GB	-	17.37	1.34
Heavy	0.37	0.54	417GB	321GB	27.54	2.65	0.33	0.55	147GB	193GB	27.10	2.16

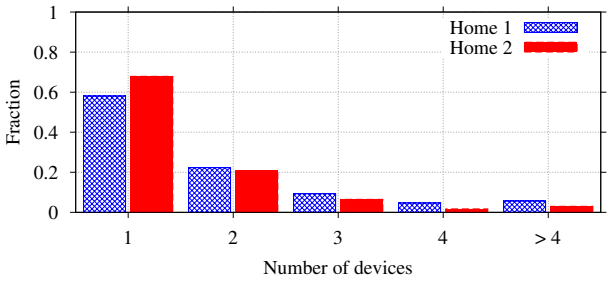


Figure 12: Distribution of the number of devices per household using the Dropbox client.

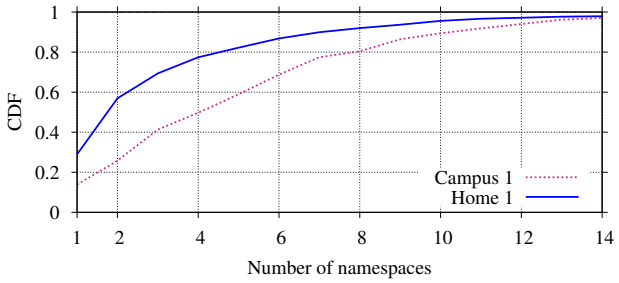


Figure 13: Number of namespaces per device.

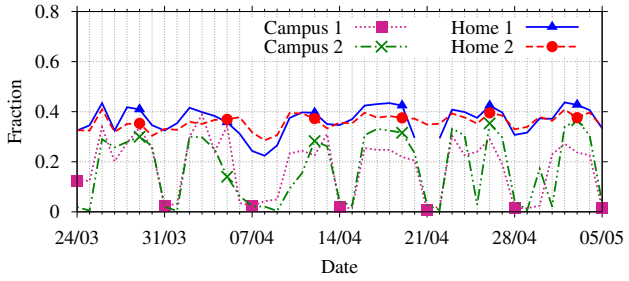


Figure 14: Distinct device start-ups per day – fraction of the number of devices in each probe.

with more than 1 device (around 25% of the total), at least 1 folder is shared among the devices. This further confirms our findings about the typical use of Dropbox among *heavy* users for the synchronization of devices. Since the LAN Sync Protocol traffic does not reach our probes, we cannot quantify the amount of bandwidth saved in these households by the use of the protocol. We can conclude, however, that no more than 25% of the households are profiting from that. The remaining users always rely on central storage datacenters for their data transfers.

5.3 Shared Folders

In order to measure to what extent Dropbox is being used for content sharing or collaborative work, we analyze namespace identifications in *Home 1* and *Campus 1* traffic (in *Home 2* and *Campus 2* this information was not exposed). Fig. 13 shows the distribution of the number of namespaces per device. By analyzing *Campus 1* data in different dates, it is possible to conclude that the number of namespaces per device is not stationary and has a slightly increasing trend. Fig. 13 was built considering the last observed number of namespaces on each device in our datasets.

In both networks the number of users with only 1 namespace (the users' root folder) is small: 13% in *Campus 1* and 28% in *Home 1*. In general, users in *Campus 1* have more namespaces than in *Home 1*. The percentage of users having 5 or more namespaces is equal to 50% in the former, and 23% in the latter. When considering only IP addresses assigned to workstations in *Campus 1*, each device has on average 3.86 namespaces.

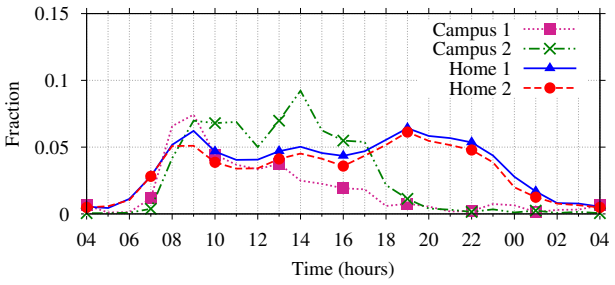
5.4 Daily Usage

We characterize whether the use of the Dropbox client has any typical seasonality. Fig. 14 shows the time series of the number of distinct devices starting up a Dropbox session in each vantage point per day. The time series are normalized by the total number of devices in each dataset. Around 40% of all devices start at least one session every day in home networks, including weekends.[10] In campus networks, on the other hand, there is a strong weekly seasonality.

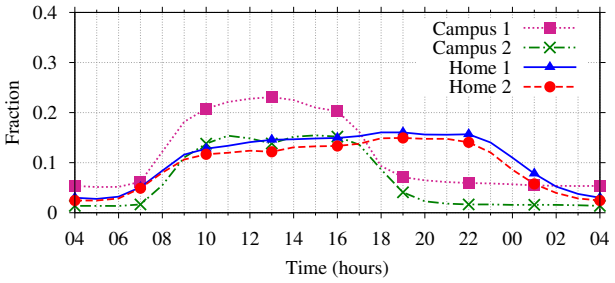
At a finer time scale (1 hour bins), we observe that the service usage follows a clear day-night pattern. Fig. 15 depicts the daily usage of the Dropbox client. All plots were produced by averaging the quantities per interval across all working days in our datasets.

Fig. 15(a) shows the fraction of distinct devices that start a session in each interval, while Fig. 15(b) depicts the fraction of devices that are active (i.e., are connected to Dropbox) per time interval. From these figures we can see that Dropbox usage varies strongly in different locations, following the presence of users in the environment. For instance, in *Campus 1*, session start-ups have a clear relation with employees' office hours. Session start-ups are better distributed during the day in *Campus 2* as a consequence of the transit of students at wireless access points. In home networks, peaks of start-ups are seen early in the morning and during the evenings. Overall, all time series of active devices (Fig. 15(b)) are smooth, showing that the number of active users at any time of the day is easily predictable.

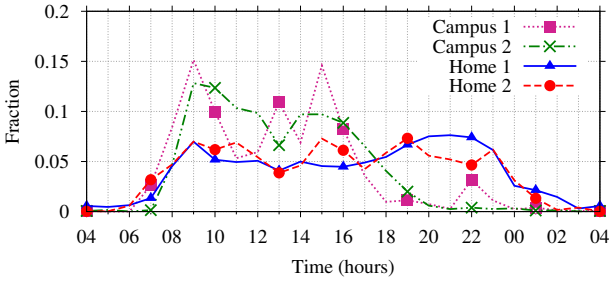
[10]Note the exceptions around holidays in April and May.



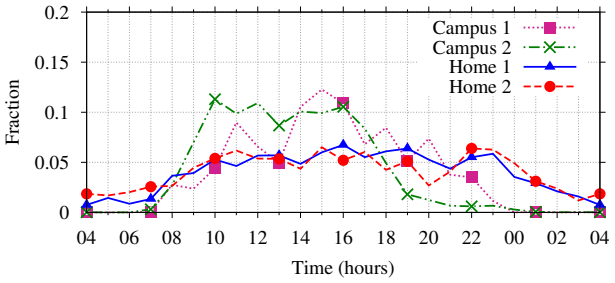
(a) Session start-ups



(b) Active devices



(c) Retrieve



(d) Store

Figure 15: Daily usage of Dropbox on weekdays.

Fig. 15(c) and 15(d) depict the fraction of the total number of bytes exchanged in each time interval in *retrieve* and *store* functions, respectively. Fig. 15(c) shows that the number of bytes received in *retrieve* operations has a correlation with client start-ups. This suggests that the first synchronization after starting a device is dominated by the download of content produced elsewhere, instead of the upload of content produced while off-line. Although other patterns are visible in the figures, such as the concentration of downloads in the morning in *Campus 1* and in the evening in *Home 1*, both series are still very noisy at this level of aggregation.

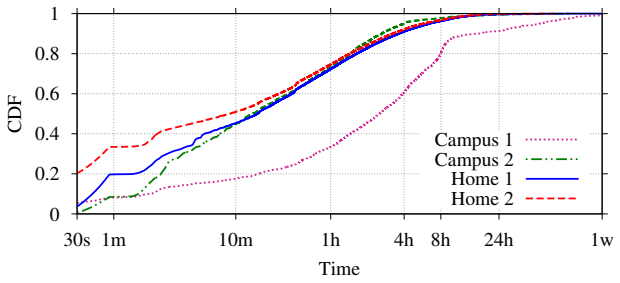


Figure 16: Distribution of session durations.

5.5 Session Duration

We analyze the session duration based on the TCP flows to notification servers. *Home 1*, *Home 2*, and *Campus 2* have a similar behavior in general, as shown in Fig. 16, with an exception for the number of short-lived sessions. In both home networks, a significant number of notification flows are terminated in less than 1 minute. A closer inspection reveals that most of those flows are from few devices. Their divergent TCP behavior suggests that network equipment (*e.g.* NAT or firewalls – see [10]) might be terminating notification connections abruptly. Considering the normal operation of the Dropbox protocol, notification connections are re-established immediately after that.

Most devices in *Home 1*, *Home 2* and *Campus 2* stay connected up to 4 hours in a single session. In *Campus 1*, a significantly higher percentage of long-lasting sessions is seen. This can be explained by the prevalence of workstations in a typical 8-hours work routine. Inflections at the tail of the distributions are seen in all curves, as a consequence of the devices kept always on-line.

5.6 Implications

The results in this section help to understand the current usage of the Dropbox client. This is needed, for instance, for provisioning data-centers and networks to handle cloud storage traffic. Our analysis of typical user behaviors reveals that users have different interests on the application. For instance, although Dropbox is already installed in more than 6% of the households (see Sec. 3.3), less than 40% of these households are fully using its functionalities, i.e., synchronizing devices, sharing folders etc. Interestingly, the results are very similar in both home networks, reinforcing our conclusions. The very high amount of traffic created by this limited percentage of users motivates our expectations that cloud storage systems will be among the top applications producing Internet traffic soon. Geographically dispersed sources as well as longitudinal data are, however, necessary to check whether the conclusions of this section can be generalized, as more people adopt such solutions.

6. WEB STORAGE

In addition to the client application, Dropbox allows users to access shared folders and files using both its main Web interface and a direct link download mechanism. In the following, we analyze the usage of these interfaces. Fig. 17 presents the CDFs of the number of bytes in the storage flows of the Dropbox main Web interface. Separate CDFs for uploads and downloads are presented.

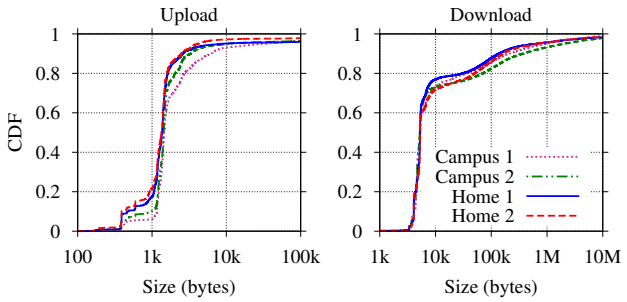


Figure 17: Storage via the main Web interface.

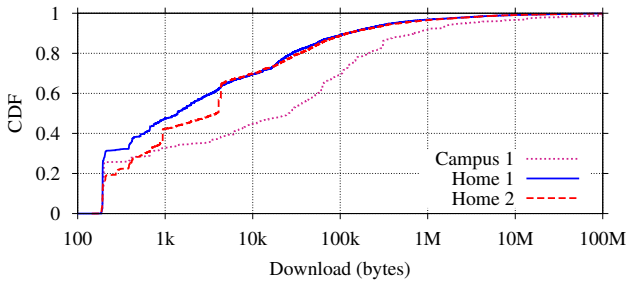


Figure 18: Size of direct link downloads.

Considering the number of uploaded bytes, it becomes clear that the main Web interface is hardly used for uploading content. More than 95% of the flows submitted less than 10kB. When considering downloaded bytes, up to 80% of flows exchanged less 10kB. These distributions are, however, strongly biased toward the SSL handshake sizes for two reasons: (i) the Dropbox interface retrieves thumbnails from storage servers using SSL; (ii) Web browsers open several parallel connections when retrieving those HTTP objects. The remaining flows have less than 10MB in more than 95% of the cases, showing that only small files are normally retrieved from this Web interface.

Additionally, we analyze flows related to direct link downloads. Note that these flows correspond to 92% of the Dropbox Web storage flows in *Home 1*, confirming that this mechanism is highly preferred over the main Dropbox Web interface. Fig. 18 shows the CDF of the size of direct link downloads.[11] Since these downloads are not always encrypted, the CDF does not have the SSL lower-bound. Interestingly, only a small percentage of direct link downloads is bigger than 10MB, suggesting that their usage is not related to the sharing of movies or archives.

7. CONCLUSIONS

To the best of our knowledge, we are the first to analyze the usage of Dropbox on the Internet. Our analysis assessed the increasing interest on cloud-based storage systems. Major players like Google, Apple and Microsoft are offering the service. We showed that, in this landscape, Dropbox is currently the most popular provider of such a system.

By analyzing flows captured at 4 vantage points in Europe over a period of 42 consecutive days, we showed that Dropbox is by now responsible for a considerable traffic volume. In one of our datasets, for instance, Dropbox is already equivalent to one third of the YouTube traffic.

We presented an extensive characterization of Dropbox, both in terms of the system workload as well as the typical usage. Our main findings show that the Dropbox service performance is highly impacted by the distance between the clients and the data-centers, which are currently located in the U.S. The usage of per-chunk acknowledgments in the client protocol combined with the typically small chunk sizes deeply limits the effective throughput of the service. In this paper, we identified two possible improvements to the protocol: (i) the usage of a chunk bundling scheme; (ii) the introduction of delayed acknowledgments. We showed that the recent deployment of a bundling mechanism improved the system performance dramatically. In addition, we expect that the overall performance will be improved by the deployment of other data-centers in different locations.

Regarding the typical workload of the Dropbox system, our analysis showed a variety of user behaviors. For instance, a considerable number of users take full advantage of the Dropbox functionalities, actively storing and retrieving files and sharing several folders. However, we also noted around one third of users completely abandoning their clients, seldom exchanging any data during the 42 days of observations.

[11] Uploads are not shown since a single HTTP request is sent. *Campus 2* is not depicted due to the lack of FQDN.

8. ACKNOWLEDGMENTS

This work has been carried out in the context of the TMA COST Action IC0703, the EU-IP project *mPlane* and the IOP GenCom project *SeQual*. *mPlane* is funded by the European Commission under the grant n-318627. *SeQual* is funded by the Dutch Ministry of Economic Affairs, Agriculture and Innovation via its agency Agentschap NL.

9. REFERENCES

[1] A. Bergen, Y. Coady, and R. McGeer. Client Bandwidth: The Forgotten Metric of Online Storage Providers. In *Proceedings of the 2011 IEEE Pacific Rim Conference on Communications, Computers and Signal Processing*, PacRim'2011, pages 543–548, 2011.

[2] I. Bermudez, M. Mellia, M. M. Munafò, R. Keralapura, and A. Nucci. DNS to the Rescue: Discerning Content and Services in a Tangled Web. In *Proceedings of the 12th ACM SIGCOMM Conference on Internet Measurement*, IMC'12, 2012.

[3] M. Cha, H. Kwak, P. Rodriguez, Y.-Y. Ahn, and S. Moon. I Tube, You Tube, Everybody Tubes: Analyzing the World's Largest User Generated Content Video System. In *Proceedings of the 7th ACM SIGCOMM Conference on Internet Measurement*, IMC'07, pages 1–14, 2007.

[4] N. Dukkipati, T. Refice, Y. Cheng, J. Chu, T. Herbert, A. Agarwal, A. Jain, and N. Sutin. An Argument for Increasing TCP's Initial Congestion Window. *SIGCOMM Comput. Commun. Rev.*, 40(3):26–33, 2010.

[5] A. Finamore, M. Mellia, M. Meo, M. M. Munafò, and D. Rossi. Experiences of Internet Traffic Monitoring with Tstat. *IEEE Network*, 25(3):8–14, 2011.

[6] A. Finamore, M. Mellia, M. M. Munafò, R. Torres, and S. G. Rao. YouTube Everywhere: Impact of Device and Infrastructure Synergies on User

Experience. In *Proceedings of the 11th ACM SIGCOMM Conference on Internet Measurement*, IMC'11, pages 345–360, 2011.

[7] M. Gjoka, M. Sirivianos, A. Markopoulou, and X. Yang. Poking Facebook: Characterization of OSN Applications. In *Proceedings of the First Workshop on Online Social Networks*, WOSN'08, pages 31–36, 2008.

[8] S. Halevi, D. Harnik, B. Pinkas, and A. Shulman-Peleg. Proofs of Ownership in Remote Storage Systems. In *Proceedings of the 18th ACM Conference on Computer and Communications Security*, CCS'11, pages 491–500, 2011.

[9] D. Harnik, B. Pinkas, and A. Shulman-Peleg. Side Channels in Cloud Services: Deduplication in Cloud Storage. *IEEE Security and Privacy*, 8(6):40–47, 2010.

[10] S. Hätönen, A. Nyrhinen, L. Eggert, S. Strowes, P. Sarolahti, and M. Kojo. An Experimental Study of Home Gateway Characteristics. In *Proceedings of the 10th ACM SIGCOMM Conference on Internet Measurement*, IMC'10, pages 260–266, 2010.

[11] W. Hu, T. Yang, and J. N. Matthews. The Good, the Bad and the Ugly of Consumer Cloud Storage. *ACM SIGOPS Operating Systems Review*, 44(3):110–115, 2010.

[12] A. Lenk, M. Klems, J. Nimis, S. Tai, and T. Sandholm. What's Inside the Cloud? An Architectural Map of the Cloud Landscape. In *Proceedings of the 2009 ICSE Workshop on Software Engineering Challenges of Cloud Computing*, CLOUD'09, pages 23–31, 2009.

[13] A. Li, X. Yang, S. Kandula, and M. Zhang. CloudCmp: Comparing Public Cloud Providers. In *Proceedings of the 10th ACM SIGCOMM Conference on Internet Measurement*, IMC'10, pages 1–14, 2010.

[14] M. Mellia, M. Meo, L. Muscariello, and D. Rossi. Passive Analysis of TCP Anomalies. *Computer Networks*, 52(14):2663–2676, 2008.

[15] A. Mislove, M. Marcon, K. P. Gummadi, P. Druschel, and B. Bhattacharjee. Measurement and Analysis of Online Social Networks. In *Proceedings of the 7th ACM SIGCOMM Conference on Internet Measurement*, IMC'07, pages 29–42, 2007.

[16] M. Mulazzani, S. Schrittwieser, M. Leithner, M. Huber, and E. Weippl. Dark Clouds on the Horizon: Using Cloud Storage as Attack Vector and Online Slack Space. In *Proceedings of the 20th USENIX Conference on Security*, SEC'11, 2011.

[17] G. Wang and T. E. Ng. The Impact of Virtualization on Network Performance of Amazon EC2 Data Center. In *Proceedings of the 29th IEEE INFOCOM*, pages 1–9, 2010.

[18] Q. Zhang, L. Cheng, and R. Boutaba. Cloud Computing: State-of-the-Art and Research Challenges. *Journal of Internet Services and Applications*, 1:7–18, 2010.

[19] M. Zhou, R. Zhang, W. Xie, W. Qian, and A. Zhou. Security and Privacy in Cloud Computing: A Survey. In *Sixth International Conference on Semantics Knowledge and Grid*, SKG'10, pages 105–112, 2010.

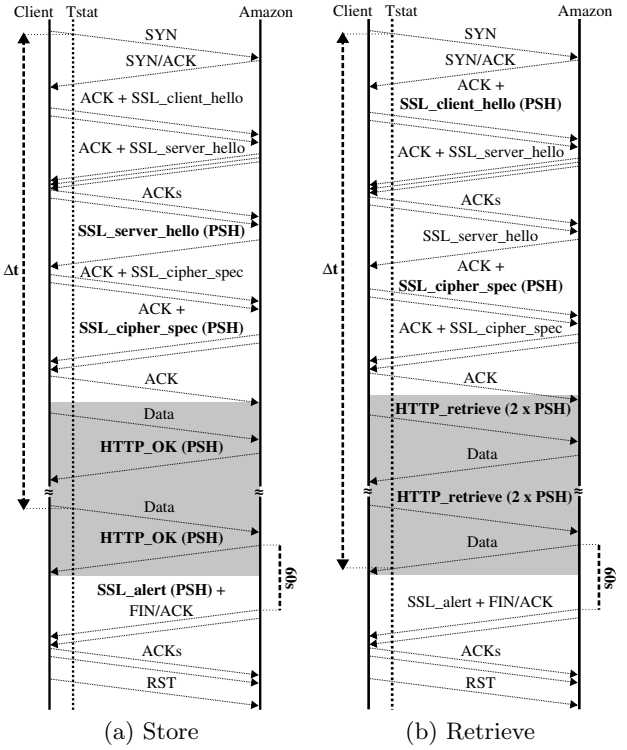


Figure 19: Typical flows in storage operations.

APPENDIX

A. STORAGE TRAFFIC IN DETAILS

A.1 Typical Flows

Fig. 19 shows typical storage flows observed in our testbed. All packets exchanged during initial and final handshakes are depicted. The data transfer phases (in gray) are shortened for the sake of space. Key elements for our methodology, such as TCP segments with `PSH` flag set and flow durations, are highlighted. To confirm that these models are valid in real clients, Tstat in *Campus 1* was set to record statistics about the first 10 messages delimited by TCP segments with `PSH` flag set. In the following, more details of our methodology and the results of this validation are presented.

A.2 Tagging Storage Flows

Storage flows are first identified using FQDNs and SSL certificate names, as explained in Sec. 3.1. After that, they are classified based on the number of bytes sent by each endpoint of the TCP connection. The method was built based on the assumption that a storage flow is used either for storing chunks or for retrieving chunks, but never for both. This assumption is supported by two factors: (i) when both operations happen in parallel, Dropbox uses separate connections to speed up synchronization; (ii) idle storage connections are kept open waiting for new commands only for a short time interval (60s).

Our assumption could be possibly violated during this idle interval. In practice, however, this seems to be hardly the case. Fig. 20 illustrates that by plotting the number of bytes in storage flows in *Campus 1*. Flows are concentrated near the axes, as expected under our assumption.

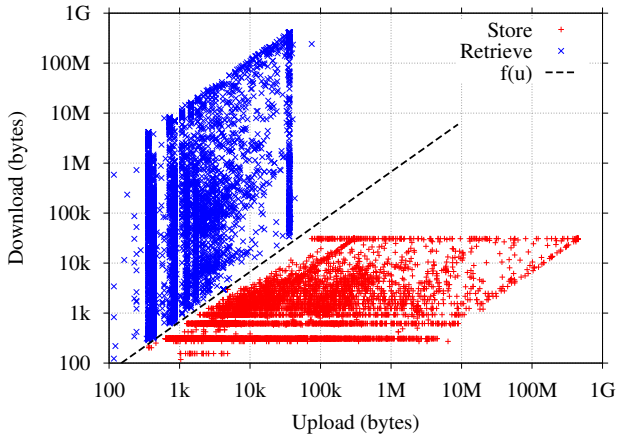


Figure 20: Bytes exchanged in storage flows in *Campus 1*. Note the logarithm scales.

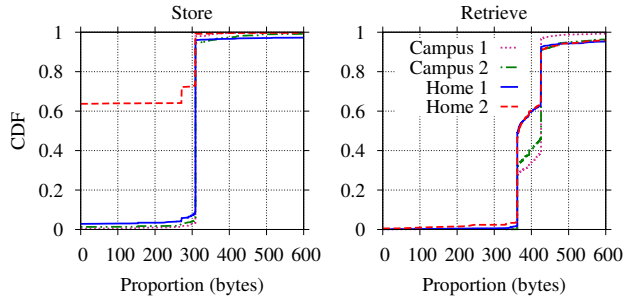


Figure 21: Payload in the reverse direction of storage operations per estimated number of chunks.

Flows in Fig. 20 are already divided into groups. The limit between *store* and *retrieve* regions is determined by the function $f(u) = 0.67(u - 294) + 4103$, where u is the number of uploaded bytes. This function was empirically defined using the extra information collected in *Campus 1*, where the following was observed:

- Both *store* and *retrieve* operations require at least 309 bytes of overhead from servers;
- *Store* and *retrieve* operations require at least 634 and 362 bytes of overhead from clients, respectively;
- Typically, SSL handshakes require 294 bytes from clients and 4103 bytes from servers.

$f(u)$ is centralized between the regions determined according these constants. For improving visualization, SSL overheads are subtracted from each point in the figure.

Since client machines in *Campus 1* are relatively homogenous, the gap between the two groups is very clear in this dataset. More variation in message sizes is observed at other vantage points. SSL handshakes, in particular, are affected by different software configurations. However, this does not change considerably the regions of each storage operation when compared to *Campus 1*.

Finally, we quantify the possible error caused by violations of our assumption. In all vantage points, flows tagged as *store* download less 1% of the total storage volume. Since this includes protocol overheads (e.g., HTTP_OK messages in Fig. 19), mixed flows marked as *store* might have only a negligible impact in our results. Similar reasoning is valid for *retrieve* flows.

A.3 Number of Chunks

The number of chunks transported in a storage flow (c) is estimated by counting TCP segments with `PSH` flag set (s) in the reverse direction of the transfer, as indicated in Fig. 19. For *retrieve* flows, $c = \frac{s-2}{2}$. For *store* flows $c = s - 3$ or $c = s - 2$, depending on whether the connection is passively closed by the server or not. This can be inferred by the time difference between the last packet with payload from the client and the last one from the server: when the server closes an idle connection, the difference is expected to be around 1 minute (otherwise, only a few seconds). Tstat already records the timestamps of such packets by default.

We validate this relation by dividing the amount of payload (without typical SSL handshakes) in the reverse direction of a transfer by c. This proportion has to be equal to the overhead needed per storage operation. Fig. 21 shows that for the vast majority of *store* flows the proportion is about 309 bytes per chunk, as expected. In *Home 2*, the apparently misbehaving client described in Sec. 4.3 biases the distribution: most flows from this client lack acknowledgment messages. Most *retrieve* flows have a proportion between 362 and 426 bytes per chunk, which are typical sizes of the HTTP request in this command. The exceptions (3% – 8%) might be caused by packet loss in our probes as well as by the flows that both stored and retrieved chunks. Our method underestimates the number of chunks in those cases.

A.4 Duration

Fig. 19 shows the duration used when computing the throughput of a storage flow (Δt). Since initial TCP/SSL handshakes affect users' perception of throughput, the first SYN packet is taken as the begin of the transfer. Termination handshakes, on the other hand, are ignored. In *store* flows, the last packet with payload sent by the client is considered the end of the transfer. In *retrieve* flows, the last packet with payload is normally a server alert about the SSL termination. We compensate for that by subtracting 60s from the duration of *retrieve* flows whenever the difference between the last packet with payload from the server and the one from the client is above 60s.

Because of our monitoring topology, Δt does not include the trip time between clients and our probes. Δt is, therefore, slightly underestimated. Fig. 19 also shows that 4 or 5 RTTs are needed before the client starts to send or to receive data. In some cases, this already accounts for about 500ms in the flow duration. Note that the initial TCP congestion window in place at servers was forcing a pause of 1 RTT during the SSL handshake. This parameter has been tuned after the release of Dropbox 1.4.0, reducing the overhead.