Benchmarking Personal Cloud Storage

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

Enrico Bocchi
Politecnico di Torino
enrico.bocchi@studenti.polito.it

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

Herman Slatman
University of Twente
h.slatman@utwente.nl

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

ABSTRACT

Personal cloud storage services are data-intensive applications already producing a significant share of Internet traffic. Several solutions offered by different companies attract more and more people. However, little is known about each service capabilities, architecture and – most of all – performance implications of design choices. This paper presents a methodology to study cloud storage services. We apply our methodology to compare 5 popular offers, revealing different system architectures and capabilities. The implications on performance of different designs are assessed executing a series of benchmarks. Our results show no clear winner, with all services suffering from some limitations or having potential for improvement. In some scenarios, the upload of the same file set can take seven times more, wasting twice as much capacity. Our methodology and results are useful thus as both benchmark and guideline for system design.[1]

Categories and Subject Descriptors

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

Keywords

Measurements; Performance; Comparison; Dropbox

1. INTRODUCTION

Personal cloud storage services allow to synchronize local folders with servers in the cloud. They have gained popularity, with companies offering significant amounts of remote storage for free or reduced prices. More and more people are being attracted by these offers, saving personal files, synchronizing devices and sharing content with great simplicity. This high public interest pushed various providers to enter the cloud storage market. Services like Dropbox, SkyDrive and Google Drive are becoming pervasive in people's routine. Such applications are data-intensive and their increasing usage already produces a significant share of Internet traffic [3].

Previous results about Dropbox [3] indicate that design and architectural choices strongly influence service performance and network usage. However, very little is known about how other providers implement their services and the implications of different designs. This understanding is valuable as a guideline for building well-performing services that wisely use network resources.

The goal of this paper is twofold. Firstly, we investigate how different providers tackle the problem of synchronizing people's files. For answering this question, we develop a methodology that helps to understand both system architecture and client capabilities. We apply our methodology to compare 5 services, revealing differences on client software, synchronization protocols and data center placement. Secondly, we investigate the consequences of such designs on performance. We answer this question by defining a series of benchmarks. Taking the perspective of users connected from a single location in Europe, we benchmark each selected service under the same conditions, highlighting differences manifested in various usage scenarios and emphasizing the relevance of design choices for both users and the Internet.

Our results extend [3], where Dropbox usage is analyzed from passive measurements. In contrast to the previous work and [12, 16] that focus on a specific service, this paper compares several solutions using active measurements. The results in [3] are used to guide our benchmarking definition. The authors of [11] benchmark cloud providers, but focusing only on server infrastructure. Similarly to our goal, [9] evaluates Dropbox, Mozy, Carbonite and CrashPlan. Motivated by the extensive list of providers, we first propose a methodology to automate the benchmarking. Then, we analyze several synchronization scenarios and providers, shedding light on the impact of design choices on performance.

Our results reveal interesting insights, such as unexpected drops in performance in common scenarios because of both the lack of client capabilities and architectural differences in the services. Overall, the lessons learned are useful as guidelines to improve personal cloud storage services.

[1]Our benchmarking tool and the measurements used in our analyses are available at the SimpleWeb trace repository: `http://www.simpleweb.org/wiki/cloud_benchmarks`

IMC'13, October 23–25, 2013, Barcelona, Spain.

ACM 978-1-4503-1953-9/13/10 ...$15.00.
http://dx.doi.org/10.1145/2504730.2504762.

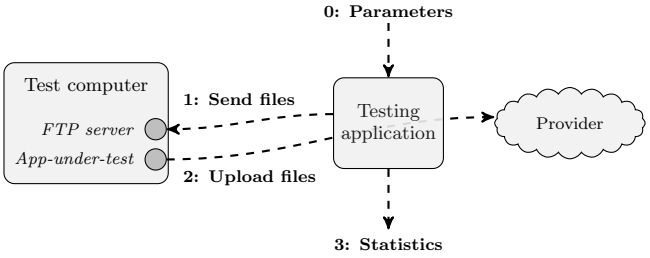


Figure 1: Testbed to study personal storage services.

2. METHODOLOGY AND SERVICES

This section describes the methodology we follow to design benchmarks to check capabilities and performance of personal storage services. We use active measurements relying on a testbed (Fig. 1) composed of two parts: (i) a *test computer* that runs the application-under-test in the desired operating system; and (ii) our *testing application*. The complete environment can run either in a single machine, or in separate machines provided that the testing application can intercept traffic from the test computer.

We build the testbed in a single Linux server for our experiments. The Linux server both controls the experiments and hosts a virtual machine that runs the test computer (Windows 7 Enterprise).[2] Our testbed is connected to a 1 GB/s Ethernet network at the University of Twente, in which Internet connectivity is not a bottleneck.

Our testing application receives as input benchmarking parameters (step 0 in Fig. 1) describing the sequence of operations to be performed. The testing application acts remotely on the test computer, generating specific workloads in the form of file batches, which are manipulated using a FTP client (step 1). Files of different types are created or modified at run-time, e.g., text files composed of random words from a dictionary, images with random pixels, or random binary files. Generated files are synchronized to the cloud by the application-under-test (step 2) and the exchanged traffic is monitored to compute performance metrics (step 3). These include the amount of traffic seen during the experiments, the time before actual synchronization starts and the time to complete synchronization.

2.1 Architecture and Data Centers

The used architecture, data center locations and data center owner are important aspects of personal cloud storage, having both legal and performance implications. To identify how the analyzed services operate, we observe the DNS name of contacted servers when (i) starting the application; (ii) immediately after files are manipulated; and (iii) when the application is in idle state. For each service, a list of contacted DNS names is compiled.

To reveal all IP addresses of the front-end nodes used by a service, DNS names are resolved to IP addresses by contacting more than 2,000 open DNS resolvers spread around the world.[3] In fact, cloud services rely on the DNS to distribute workload, returning different IP addresses according to the originating DNS resolver [2].

The owners of the IP addresses are identified using the `whois` service. For each IP address, we look for the geographic location of the server. Since popular geolocation databases are known to have serious limitations regarding cloud providers [14], we rely on a hybrid methodology that makes use of: (i) informative strings (i.e., International Airport Codes) revealed by reverse DNS lookup; (ii) the shortest Round Trip Time (RTT) to PlanetLab nodes [15]; and (iii) active `traceroute` to spot the closest well-known location of a router. Previous works [2, 5] indicate that these methodologies provide an estimation with about a hundred of kilometers of precision, which is sufficient for our goals.

Table 1: Benchmarks to assess client performance.

Random bytes			Plain text		
Set	Files	Size	Set	Files	Size
1	1	100 kB	5	1	100 kB
2	1	1 MB	6	1	1 MB
3	10	100 kB	7	10	100 kB
4	100	10 kB	8	100	10 kB

2.2 Checking Capabilities

Previous work [3] shows that personal storage applications can implement several capabilities to optimize storage usage and to speed up transfers. These capabilities include the adoption of *chunking* (i.e., splitting content into a maximum size data unit), *bundling* (i.e., the transmission of multiple small files as a single object), *deduplication* (i.e., avoiding re-transmitting content already available on servers), *delta encoding* (i.e., transmission of only modified portions of a file) and *compression*.

For each case, a specific test has been designed to observe if the given capability is implemented. We describe each test directly in Sect. 4. In summary, our testing application produces specific batches of files that would benefit from a capability. The exchanged traffic is analyzed to determine how the service operates.

2.3 Benchmarking Performance

After knowing how the services are designed in terms of both data center locations and system capabilities, we check how such choices influence synchronization performance and the amount of overhead traffic.

Results in [3] show that up to 90 % of Dropbox users' uploads carry less than 1 MB. While 50 % of the batches carry only a single file, a significant portion (around 10 %) involves at least 100 files. Based on these results, we design 8 benchmarks varying (i) number of files; (ii) file sizes and (iii) file types, therefore covering a variety of synchronization scenarios. Tab. 1 lists our benchmark sets. All files in the sets are created at run-time by our testing application. Other types of files, used in Sect. 4 for checking capabilities, are not included in the benchmarks for the sake of space.

Each experiment is repeated 24 times per service, allowing at least 5 min between experiments to avoid creating abnormal workloads to servers. The benchmark of a single storage service lasts for about 1 day. Synchronization startup, upload time and protocol overhead are discussed in Sect. 5. It is important to reinforce that all measurements

[2]OS X and Linux clients have also been checked whether available and show no differences.

[3]The list has been manually compiled from various sources and covers more than 100 countries and 500 ISPs.

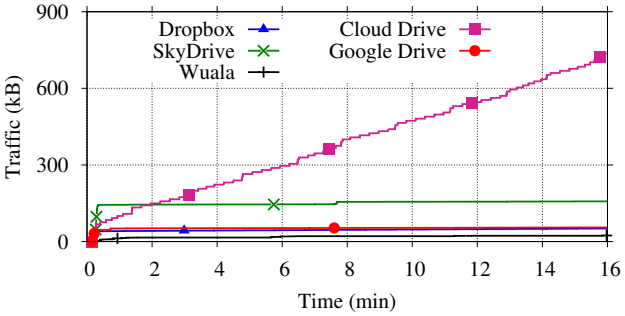


Figure 2: Background traffic while idle.

have been taken from a single location, in the same controlled environment. While results for each service may vary when measuring from other locations or longer intervals, our conclusions in the following are independent of that. Repeating the experiments from different locations is planned to future works.

2.4 Tested Storage Services

We focus on 5 services for the sake of space, although our methodology is generic and can be applied to any other service. We restrict our analysis to native clients, since previous results [3] show that this is the largely preferred means to use personal cloud storage services.

Tab. 2 lists the analyzed services. Dropbox [4], Google Drive [6] and SkyDrive [13] are selected because they are among the most popular offers, according to the volume of search queries containing names of cloud storage services on Google Trends [8]. Wuala [10] is considered because it is a system that offers encryption at the client-side. We want to verify the impact of such privacy layer on synchronization performance. Finally, we include Cloud Drive [1] to compare its performance to Dropbox, since both services rely on Amazon Web Services (AWS) data centers.

Table 2: Analyzed personal cloud storage services.

Name	Version
Dropbox	2.0.8
Microsoft SkyDrive	1.8.4357.4863
Google Drive	17.0.2006.0314
LaCie Wuala	*Strasbourg*
Amazon Cloud Drive	2.0.2013.841

3. SYSTEM ARCHITECTURE

3.1 Protocols

All clients exchange traffic using HTTPS, with the exception of Dropbox notification protocol, which relies on plain HTTP. Interestingly, some Wuala storage operations also use HTTP, since users' privacy has already been secured by local encryption.

All services but Wuala use separate servers for control and storage. Their identification is trivial by monitoring the traffic exchanged when the client (i) starts; (ii) is idle; and (iii) synchronizes files. Both server names and IP addresses can be used to identify different operations during our tests. For Wuala, we use flow sizes and connection sequences to identify storage flows.

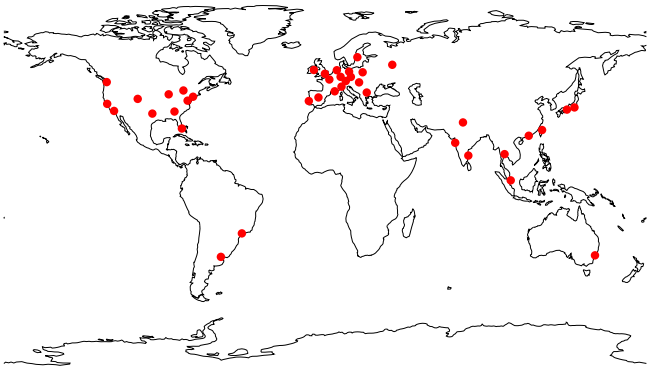


Figure 3: Google Drive's edge nodes.

We notice some relevant differences among applications during login and idle phases. Fig. 2 reports the cumulative number of bytes exchanged with control servers considering an initial 16 min in idle state. Two considerations hold. Firstly, the applications authenticate the user and check if any content has to be updated. Note how SkyDrive requires about 150 kB in total, 4 times more than others. This happens because the application contacts many Microsoft Live servers during login (13 in this example). Secondly, once login is completed, the applications keep exchanging data with the cloud. Wuala is the most silent, polling servers every 5 min on average – i.e., equivalent background traffic of about 60 b/s. Google Drive follows close, with a lightweight 40 s polling interval (42 b/s). Dropbox and SkyDrive use intervals close to 1 min (82 b/s and 32 b/s, respectively).

Amazon Cloud Drive is completely different: polling is done every 15 s, each time opening a new HTTPS connection. This notification strategy consumes 6 kb/s – i.e., about 65 MB per day. This information is relevant to users with bandwidth constraints (e.g., in 3G/4G networks) and to the system: 1 million users would generate approximately 6 Gb/s of signaling traffic alone! As the results for other providers demonstrate, such design is not optimal and seems indeed possible to be improved.

3.2 Data Centers

Next, we analyze data center locations. Dropbox uses own servers (in the San Jose area) for client management, while storage servers are committed to Amazon in Northern Virginia. Cloud Drive uses three AWS data centers: two are used for both storage and control (in Ireland and Northern Virginia); a third one is used for storage only (in Oregon). SkyDrive relies on Microsoft's data centers in the Seattle area (for storage) and Southern Virginia (for storage and control). We also identified a destination in Singapore (for control only). Not surprisingly, most data centers are located in the U.S. Wuala data centers instead are located in Europe: two in the Nuremberg area, one in Zurich and a fourth in Northern France. None is owned by Wuala. All these services follow a centralized design where clients contact the servers using the public Internet, as expected.

Google Drive follows a different approach: TCP connections are terminated at the closest Google's *edge node*, from where the traffic is routed to the actual storage/control data center using the private Google's network. Fig. 3 shows the

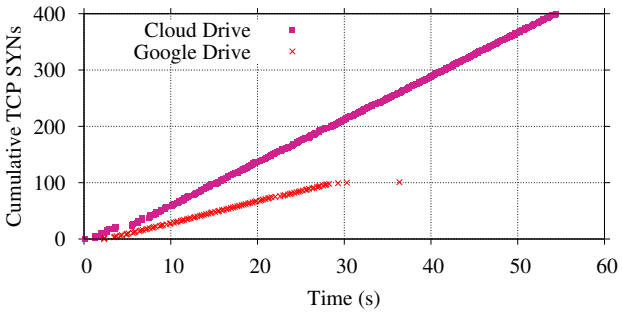


Figure 4: Uploading 100 files of 10 kB.

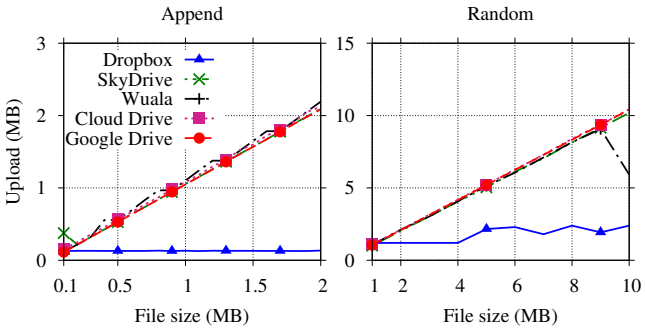


Figure 5: Delta encoding tests. Note the x-axes.

locations identified in our experiments.[4] Overall, more than 100 different entry points have been located. Such architecture allows to reduce client-server RTT and to offload storage traffic from the public Internet. Performance implications are discussed in Sect. 5.

4. CLOUD SERVICE CAPABILITIES

4.1 Chunking

Our first test aims at understanding how the services process large files. By monitoring throughput during the upload of files differing in size, we determine whether files are exchanged as single objects (no pause during the upload), or split into *chunks*, each delimited by a pause. Our experiments show that only Cloud Drive does not perform chunking. In fact, Google Drive uses 8 MB chunks while Dropbox uses 4 MB chunks. SkyDrive and Wuala apparently use variable chunk sizes.

Chunking seems advantageous because it simplifies upload recovery in case of failures: partial submission becomes easier to be implemented, benefiting users connected to slow networks, for example.

4.2 Bundling

When a batch of files needs to be transferred, files could be bundled and pipelined so that both transmission latency and control overhead impact are reduced. Our tests to check how services handle batches of files consist of 4 upload sets, each containing exactly the same number of bytes (1 MB), which are split into 1, 10, 100 or 1000 files, respectively.

These experiments reveal a variety of synchronization strategies. Google Drive and Cloud Drive open one separate TCP (and SSL) connection for each file. Considering management, Cloud Drive opens 3 TCP/SSL control connections per file operation. Fig. 4 shows the number of TCP `SYN` packets observed when Google Drive and Cloud Drive have to store 100 files of 10 kB each: 100 and 400 connections are opened respectively, requiring 30 s and 55 s to complete the upload. Sect. 5 will confirm that such design strongly limits the client performance when several files have to be exchanged, owing to TCP and SSL negotiations.

Other services reuse TCP connections. However, SkyDrive and Wuala submit files sequentially, waiting for application layer acknowledgments between each upload. This can be determined by counting packet bursts, which are proportional to the number of files in our experiments. We conclude that only Dropbox implements a file-bundling strategy.

[4]Our results match with Google's points of presence [7]. Understanding how Google manages traffic inside its network is outside the scope of this paper.

4.3 Client-Side Deduplication

Server data deduplication eliminates replicas on the storage server. In case the same content is already present on the storage, replicas in the client folder can be identified to save upload capacity too.

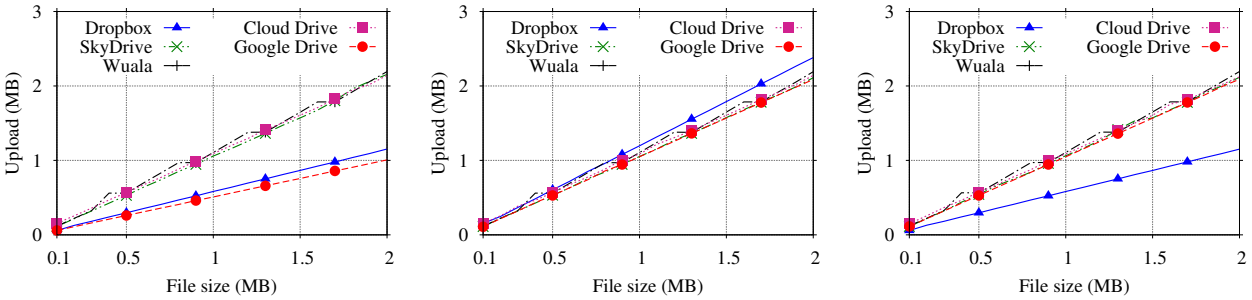
To check whether this feature is implemented, we design the following test: (i) a random file is inserted in an arbitrary folder; (ii) the same random payload is used to create a replica with a different name in a second folder; (iii) the original file is copied to a third folder; and (iv) after all copies are deleted, the original file is placed back. The last step determines whether deduplication fails after files are deleted from the local folder.

Results allow to conclude that only Dropbox and Wuala implement deduplication. All other services have to upload the same data even if it is readily available at the storage server. Interestingly, Dropbox and Wuala can identify copies of users' files even after they are deleted and later restored. In the case of Wuala, deduplication is compatible with local encryption, i.e., two identical files generate two identical encrypted versions.

4.4 Delta Encoding

Delta encoding is a specialized compression technique that calculates file differences among two copies, allowing the transmission of only the modifications between revisions. To verify which services implement delta encoding, a sequence of changes are generated on a file so that a portion of content is added/changed at each iteration. Three cases are considered: new data added/changed at the end, at the beginning, or at a random position within the file. This allows us to check whether rolling hash mechanisms are implemented. In all cases, the modified file replaces its old copy.

Fig. 5 shows that only Dropbox fully implements delta encoding, i.e., the volume of uploaded data corresponds to the actual part that has been modified. Results in which bytes are inserted at the end and at random positions are shown on the left and right plots, respectively. In the former case, file sizes have been chosen up to 2 MB. Larger files are instead considered in the latter case to highlight the combined effects with chunking and deduplication. Focusing on Dropbox, observe that the amount of sent traffic increases when files are bigger than Dropbox 4 MB-long chunk. This happens because the original content may be shifted, chang-



(a) Random readable text (b) Random bytes (c) Fake JPEGs

Figure 6: Bytes uploaded during the compression test.

ing two or more chunks at once. As such, the volume of data to be transmitted is larger than the added data.

Wuala does not implement delta encoding. However, deduplication prevents the client from uploading those chunks not affected by the change. This can be seen in Fig. 5, when data is added at a random offset, forming a 10 MB file. In this case, only two chunks (among 3) are modified, and thus uploaded.

4.5 Data Compression

We next verify whether data is compressed before a transfer. Compression could, in general, reduce traffic and storage requirements at the expense of processing time. The compression capability is checked with two distinct file sets. The first set (Fig. 6(a)) is made of highly compressible text files (sizes from 100 kB to 2 MB). Files in the second set (Fig. 6(b)) contain pure random bytes so that no compression is possible. Fig. 6(a) reveals that Dropbox and Google Drive compress data before transmission, with the latter implementing a more efficient scheme. Fig. 6(b) confirms that Dropbox has the highest overhead in this scenario.

Naturally, compression is advantageous only for some file types. Compression has a negligible or negative impact when already compressed files are going to be transmitted. A possible approach would be to verify the file format before trying to compress it (e.g., using *magic numbers*). We check whether Google Drive and Dropbox implement smart policies by creating fake JPEGs – i.e., files with JPEG extension and JPEG headers, but actually filled with text. Fig. 6(c) shows that Google Drive identifies JPEG content and avoids compression. Dropbox instead compresses all files independently of their content.

4.6 Summary

Tab. 3 summarizes the capabilities of each service. It shows that Dropbox has the most sophisticated client from the point of view of features to enhance synchronization speed. Wuala, Google Drive and SkyDrive come next, implementing some capabilities. Finally, Cloud Drive is the simplest client, as none of the capabilities are implemented.

5. CLIENT PERFORMANCE

5.1 Synchronization Startup

We first evaluate how much time each service needs before synchronization starts. This metric could reveal, for instance, whether implementing advanced capabilities increases initial synchronization delay. The metric is computed from the moment when files start being modified in the test computer until the first storage flow is observed.[5]

Fig. 7(a) shows average delays over 24 repetitions. Only 4 scenarios using binary files are shown, since similar conclusions are obtained with plain text files. Dropbox is the fastest service to start synchronizing single files. Its bundling strategy, however, slightly delays start up with multiple files. As we will show next, such strategy pays back in total upload time. SkyDrive is by far the slowest, waiting at least 9 s before starting submitting files. The root cause of this delay is unclear, since SkyDrive client does not report any activity during this period. Moreover, SkyDrive gets slower as batches increase, taking more than 20 s to start sending 100 files of 10 kB. Wuala also increases its startup time when multiple files are submitted.

5.2 Completion Time

Next, we test how long each service takes to complete upload tasks. This is measured as the difference between the first and the last packet with payload seen in any storage flow. We ignore TCP tear-down delays, and control messages sent after the upload is complete.

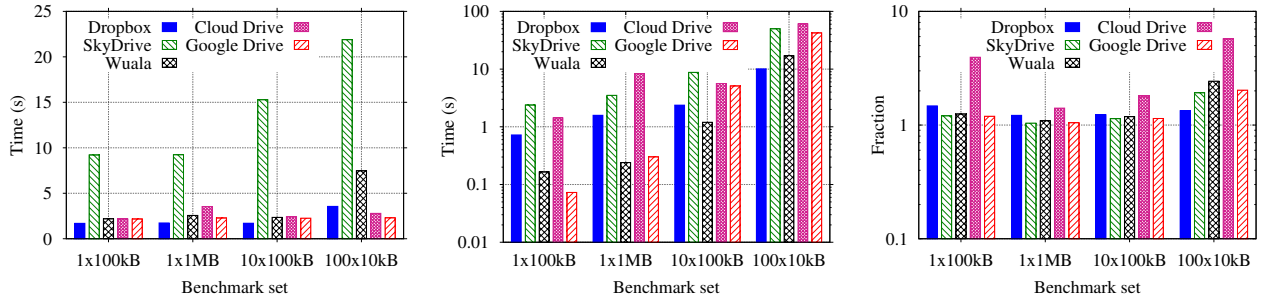
Fig. 7(b) summarizes our results (note the logarithmic scale on the y-axis). A mixed figure emerges. When synchronizing single files of 100 kB or 1 MB, the distance between our testbed and the data centers dominates the metric. Google Drive (26,49 Mb/s) and Wuala (33,34 Mb/s) are the fastest, since each TCP connection is terminated at data centers nearby our testbed. Dropbox and SkyDrive, on the other hand, are the most impacted services. SkyDrive (160 ms of RTT) needs almost 4 s to upload a 1 MB file, whereas Google Drive requires only 300 ms (15 ms of RTT).

When multiple files are stored, the client capabilities become central. The rightmost bars on Fig. 7(b) show a striking difference on completion time when 100 files of 10 kB are used. Dropbox wins by a factor of 2 because of bundling, topping to 0,8 Mb/s of upload rate. Interestingly, Google Drive's advantage due to its distributed topology is canceled by the usage of separate TCP/SSL connections per file. It takes 42 s on average – i.e., 189 kb/s. Other services are also penalized by their lack of bundling, with Cloud Drive taking about 60 s (132 kb/s) to complete some tests.

[5]This includes delays of our testing application to send files to the test computer. The artifact, however, does not affect our conclusions, since all experiments are equally affected.

Table 3: Summary of the capabilities implemented in each service.

	Dropbox	SkyDrive	Wuala	Google Drive	Cloud Drive
Chunking	4 MB	var.	var.	8 MB	no
Bundling	**yes**	no	no	no	no
Compression	always	no	no	**smart**	no
Deduplication	**yes**	no	**yes**	no	no
Delta-encoding	**yes**	no	no	no	no



(a) Synchronization start up time (b) Duration (note log y-axis) (c) Overhead (note log y-axis)

Figure 7: Average benchmarking results when uploading binary files.

Results with plain text files are not shown for the sake of space, but similar conclusions can be drawn. Dropbox has a small improvement in its upload time because of compression, although network latency still dominates the metric. Google Drive profits from its smart compression when sending single plain text files of moderate sizes. However, the service is again among the slowest when multiple small text files are synchronized, showing that its smart compression advantage is also canceled by the lack of bundling.

5.3 Protocol Overhead

Finally, we evaluate protocol overhead as the total storage and control traffic over the benchmark size. Fig. 7(c) shows that all services have a moderate to high overhead when small binary files are synchronized. Cloud Drive presents a very high overhead because of its high number of control flows opened for every file transfer (see Fig. 2). Dropbox exhibits the highest overhead among the remaining services (47 % for 100 kB files and 22 % for 1 MB files), possibly owing to the signaling cost of implementing its advanced capabilities.

The lack of bundling dramatically increases overhead when multiple small files are sent. Google Drive, for instance, exchanges twice as much traffic as the actual data size when sending 100 binary files of 10 kB. Cloud Drive shows even more overhead – i.e., more than 5 MB of data are exchanged to commit 1 MB of content. A similar pattern is generally seen in the experiments with plain text files. Dropbox and Google Drive are the exceptions, owing to their compression strategies, which naturally reduce network overhead.

6. CONCLUSIONS

In this paper we presented a methodology to check both capabilities and system design of personal cloud storage services. We then evaluated the implications of design choices on performance by analyzing 5 services.

Our analysis shows the relevance of *client capabilities* and *protocol design* to personal cloud storage services. Dropbox implements most of the checked capabilities, and its sophisticated client clearly boosts performance, although some protocol tweaks seem possible to reduce network overhead. On the other extreme, Cloud Drive bandwidth wastage is an order of magnitude higher than other offerings, and its lack of client capabilities results in performance bottlenecks. SkyDrive shows some performance limitations, while Wuala generally performs well. More importantly, Wuala deploys client side encryption, and this feature does not seem to affect Wuala synchronization performance.

These 4 examples confirm the role played by *data center placement* in a centralized approach: taking the perspective of European users only, network latency is still an important limitation for U.S. centric services, such as Dropbox and SkyDrive. Services deploying data centers nearby our test location, such as Wuala, have therefore an advantage.

Google Drive follows a different approach resulting in a mixed picture: it enjoys the benefits of using Google's capillary infrastructure and private backbone, which reduce network latency and speed up the system. However, protocols and client features limit performance, especially when multiple files are considered.

As future work, we intend to extend our analysis with measurements from other locations and longer time intervals. This will allow us to quantify long-term effects of design choices for users connected in different geographic areas. Moreover, we plan to define new benchmarks by collecting data from volunteers, thus creating realistic benchmarks that mix files of different types in a single set.

7. ACKNOWLEDGMENTS

This work was partly funded by the Network of Excellence project *Flamingo* (ICT-318488) and the EU-IP project *mPlane* (n-318627). Both projects are supported by the European Commission under its Seventh Framework Programme.

8. REFERENCES

[1] Amazon. Cloud Drive v. 2.0.2013.841. http://www.amazon.com/gp/feature.html?docId=1000828861.

[2] I. N. Bermudez, S. Traverso, M. Mellia, and M. M. Munafò. Exploring the Cloud from Passive Measurements: the Amazon AWS case. In *The 32nd Annual IEEE International Conference on Computer Communications*, INFOCOM'13, 2013.

[3] I. Drago, M. Mellia, M. M. Munafò, A. Sperotto, R. Sadre, and A. Pras. Inside Dropbox: Understanding Personal Cloud Storage Services. In *Proceedings of the 12th ACM Internet Measurement Conference*, IMC'12, pages 481–494, 2012.

[4] Dropbox. v. 2.0.8. https://www.dropbox.com/release_notes.

[5] B. Eriksson and M. Crovella. Understanding Geolocation Accuracy using Network Geometry. In *The 32nd Annual IEEE International Conference on Computer Communications*, INFOCOM'13, 2013.

[6] Google. Drive v. 1.9.4536.8202. https://tools.google.com/dlpage/drive.

[7] Google. Network Introduction. https://peering.google.com/about/delivery_ecosystem.html.

[8] Google. Trends. http://www.google.com/trends/.

[9] 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.

[10] LaCie. Wuala v. *Strasbourg*. http://www.wuala.com/.

[11] 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.

[12] T. Mager, E. Biersack, and P. Michiardi. A Measurement Study of the Wuala On-line Storage Service. In *Proceedings of the IEEE 12th International Conference on Peer-to-Peer Computing*, P2P'12, pages 237–248, 2012.

[13] Microsoft. SkyDrive v. 17.0.2006.0314. https://skydrive.live.com/.

[14] I. Poese, S. Uhlig, M. A. Kaafar, B. Donnet, and B. Gueye. IP Geolocation Databases: Unreliable? *SIGCOMM Comput. Commun. Rev.*, 41(2):53–56, 2011.

[15] R. Torres, A. Finamore, J. R. Kim, M. Mellia, M. M. Munafò, and S. Rao. Dissecting Video Server Selection Strategies in the YouTube CDN. In *Proceedings of the 31st International Conference on Distributed Computing Systems*, ICDCS'11, pages 248–257, 2011.

[16] H. Wang, R. Shea, F. Wang, and J. Liu. On the Impact of Virtualization on Dropbox-like Cloud File Storage/Synchronization Services. In *Proceedings of the IEEE 20th International Workshop on Quality of Service*, IWQoS '12, pages 11:1–11:9, 2012.