The science behind the report:

Quickly sync and upload files with Dropbox

This document describes what we tested, how we tested, and what we found. To learn how these facts translate into real-world benefits, read the report Quickly sync and upload files with Dropbox.

We concluded our hands-on testing on September 30, 2023. During testing, we determined the appropriate hardware and software configurations and applied updates as they became available. The results in this report reflect configurations that we finalized on September 17, 2023 or earlier. Unavoidably, these configurations may not represent the latest versions available when this report appears.

Our results

To learn more about how we have calculated the wins in this report, go to http://facts.pt/calculating-and-highlighting-wins. Unless we state otherwise, we have followed the rules and principles we outline in that document.

London

Table 1: Results of our Dell™ XPS 13 9315 laptop running Windows 11 Pro testing in London, United Kingdom (median times). The winner of each test is highlighted in green.

	Dropbox Advanced	Google Workspace Business Plus	Box Business	OneDrive for business (Plan 1)
25MB single file upload	11.10	20.81	20.19	12.22
25MB single file download	4.71	4.03	10.56	19.84
100MB folder upload	46.75	83.53	99.56	61.91
100MB folder download	16.62	23.19	64.12	45.50
0-byte folder upload	10.78	40.66	98.38	89.56
0-byte folder download	7.62	8.19	41.88	16.68
1KB edit to 50KB file	2.66	9.41	10.72	2.44
100KB edit to 25MB file	4.25	15.87	10.75	3.59
50KB end-to-end sync	5.00	15.56	7.78	6.25
25MB end-to-end sync	11.59	25.16	46.15	32.91
25MB end-to-end sync of 100KB edit	5.68	20.19	25.22	6.22
250MB single file upload	77.44	86.85	85.6	82.04
250MB single file download	18.22	17.19	30.81	97.94
Upload a folder of 10,000 1KB files	441	1,765	2,499	5,795

Table 2: Results of our 13-inch Apple® MacBook Pro® 2022 laptop running macOS® 13.5.1 (Ventura) testing in London, United Kingdom (median times). The winner of each test is highlighted in green. These results are in seconds.

	Dropbox Advanced	Google Workspace Business Plus	Box Business	Microsoft OneDrive for business (Plan 1)
25MB single file upload	12.41	21.41	20.07	19.72
25MB single file download	5.60	4.28	6.78	15.41
100MB folder upload	53.69	47.87	181.32	71.87
100MB folder download	37.06	60.06	58.44	60.09
0-byte folder upload	23.91	25.00	181.69	117.25
0-byte folder download	17.41	13.78	30.81	28.28
1KB edit to 50KB file	5.88	8.78	8.66	3.56
100KB edit to 25MB file	8.66	15.84	20.22	3.43
50KB end-to-end sync	5.59	20.91	24.97	8.47
25MB end-to-end sync	15.06	26.00	31.56	42.18
25MB end-to-end sync of 100KB edit	7.93	19.53	30.15	11.75
250MB single file upload	79.78	87.50	89.50	86.16
250MB single file download	18.25	16.78	33.75	93.69
Upload a folder of 10,000 1KB files	591	1,311	5,860	8,910

Berlin

Table 3: Results of our Dell™ XPS 13 9315 laptop running Windows 11 Pro testing in Berlin, Germany (median times). The winner of each test is highlighted in green.

	Dropbox Advanced	Google Workspace Business Plus	Box Business	OneDrive for business (Plan 1)
25MB single file upload	8.85	20.19	18.34	10.31
25MB single file download	4.31	2.81	13.04	5.94
100MB folder upload	38.84	46.59	112.22	64.25
100MB folder download	18.90	24.88	81.87	25.25
0-byte folder upload	13.06	45.62	103.00	90.53
0-byte folder download	10.75	6.32	44.06	17.22
1KB edit to 50KB file	3.28	8.69	7.66	3.13
100KB edit to 25MB file	4.85	14.07	17.12	3.57
50KB end-to-end sync	4.25	15.84	19.19	5.68
25MB end-to-end sync	11.22	21.94	25.87	15.97
25MB end-to-end sync of 100KB edit	5.60	15.22	22.97	7.71
250MB single file upload	54.75	63.88	63.28	58.09
250MB single file download	13.68	11.88	31.81	20.81
Upload a folder of 10,000 1KB files	488	2,633	2,496	5,919

Table 4: Results of our 13-inch Apple MacBook Pro 2022 laptop running macOS® 13.5.1 (Ventura) testing in Berlin, Germany (median times). The winner of each test is highlighted in green.

	Dropbox Advanced	Google Workspace™ Business Plus	Box Business	OneDrive® for business (Plan 1)
25MB single file upload	11.48	19.87	17.13	16.34
25MB single file download	4.94	3.72	9.31	5.69
100MB folder upload	42.25	39.75	186.65	70.53
100MB folder download	37.60	23.78	73.60	38.28
0-byte folder upload	23.56	24.81	187.75	106.5
0-byte folder download	17.28	10.06	30.19	29.15
1KB edit to 50KB file	7.22	8.44	8.56	3.47
100KB edit to 25MB file	7.85	13.94	17.21	3.53
50KB end-to-end sync	6.38	11.53	23.57	9.38
25MB end-to-end sync	12.44	26.88	58.69	23.62
25MB end-to-end sync of 100KB edit	8.06	15.37	118.66	7.60
250MB single file upload	57.50	64.37	63.06	75.28
250MB single file download	13.66	11.87	33.53	14.43
Upload a folder of 10,000 1KB files	587	1,615	4,972	6,730

Table 5: At the start of testing, we used Fast.com to measure internet speeds on a wired connection. These speeds, as is normal, fluctuated over the course of testing.

	macOS® 13.5.1 (Ventura)		Windows 11 Pro	
	Down	Up	Down	Up
London	150 Mbps	23 Mbps	150 Mbps	24 Mbps
Berlin	220 Mbps	35 Mbps	220 Mbps	40 Mbps

System configuration information

Table 6: Detailed information on the laptops we tested.

System configuration information	Dell XPS 13 9315 2-in-1 laptop	13-inch Apple MacBook Pro (2022) laptop
Processor		
Vendor	Intel®	Apple
Model number	Core™ i7-1250U	M2
Core frequency (GHz)	1.10	3.50
Number of cores	10	8
Memory module(s)		
Amount (GB)	16	16
Type	LPDDR4x	Unified
Graphics		
Vendor	Intel	Apple
Model number	Intel Iris® Xᵉ Graphics	M2 10-core GPU
Storage		
Amount (TB)	1	1
Type	SSD	SSD
Connectivity/expansion		
Wireless internet	Intel Wi-Fi 6E AX211	Wi-Fi 6E (802.11ax)
Bluetooth	5.3	5.0
USB	2x Thunderbolt 4	2x Thunderbolt 4
Battery		
Type	Integrated lithium-ion	Integrated lithium-polymer
Rated capacity (Whr)	49.5	58.2
Display		
Size (in.)	13	13.3
Resolution	2,880 x 1,920	2,560 x 1,600
Operating system		
Vendor	Microsoft	Apple
Name	Windows 11 Pro	macOS Ventura
Version	22H2 build 22621.2283	13.5.1
BIOS		
BIOS name and version	Dell Inc. 1.10.1	8422.141.2

System configuration information	Dell XPS 13 9315 2-in-1 laptop	13-inch Apple MacBook Pro (2022) laptop
Dimensions		
Height (in.)	0.29	0.61
Width (in.)	11.5	11.97
Depth (in.)	7.9	8.36
Weight (lb.)	1.62	3.0

How we tested

We measured the time it took to perform various file and folder upload, download, and sync operations using Dropbox Advanced and three competing cloud storage providers (Google Workspace Business Plus, Box Business, and OneDrive for business [Plan 1]). We ran the tests using the native desktop apps for each cloud service provider three times each from both London, United Kingdom and Berlin, Germany and reported the median of the three runs in each city as the final result. We conducted the tests on both macOS and Windows 11 Pro devices, using two 13-inch Apple MacBook Pro laptops running macOS 13.5.1 (Ventura) and two Dell XPS 13 2-in-1 laptops running Windows 11 Pro. We enabled File Provider on all macOS devices, created accounts for each of the services, and used the following builds/releases of each desktop sync client:

London, United Kingdom

- Dropbox v182.4.6427 on both macOS and Windows 11 Pro devices
- Google Drive v80.0.1.0 (Apple Silicon) on macOS and v80.0.1.0 on Windows 11 Pro devices
- Box Drive v2.34.84 on both macOS and Windows 11 Pro devices
- OneDrive v23.180.0828 (Standalone) (Apple silicon) on macOS devices and v23.180.0828.0001 (64-bit) on Windows 11 Pro devices

Berlin, Germany

- Dropbox v182.4.6427 on macOS and v183.4.7058 on Windows 11 Pro devices
- Google Drive v80.0.1.0 (Apple Silicon) on macOS and v80.0.5.0 on Windows 11 Pro devices
- Box Drive v2.34.84 on macOS and v2.35.97 on Windows 11 Pro devices
- OneDrive v23.180.0828 (Standalone) (Apple silicon) on macOS devices and v23.184.0903.0002 (64-bit) on Windows 11 Pro devices

Overview of tasks

The test corpus consisted of the following tasks, where Device A is the primary laptop under test and Device B (when applicable) is a second identically configured laptop. We used unique, randomly generated files for each individual test run.

Device-to-cloud and cloud-to-device tests

- Uploading (device-to-cloud sync) a 100MB folder of 400 random (miscellaneous) 250KB files distributed across 40 sub-folders
- Downloading (cloud-to-device sync) a 100MB folder of 400 random (miscellaneous) 250KB files distributed across 40 sub-folders
- Uploading (device-to-cloud sync) a folder of 400 zero-byte files distributed across 40 sub-folders
- Downloading (cloud-to-device sync) a folder of 400 zero-byte files distributed across 40 sub-folders
- Uploading (device-to-cloud sync) a random (miscellaneous) 25MB file
- Downloading (cloud-to-device sync) a random (miscellaneous) 25MB file
- Uploading (device-to-cloud sync) a random (miscellaneous) 250MB file
- Downloading (cloud-to-device sync) a random (miscellaneous) 250MB file

Device-to-cloud only tests

- Editing a random (miscellaneous) 50KB local file and uploading changes (device-to-cloud sync)
- Editing a random (miscellaneous) 25MB local file and uploading changes (device-to-cloud sync)
- Uploading (device-to-cloud sync) a folder of 10,000 random (miscellaneous) 1KB files distributed across 10 sub-folders

Device-to-cloud-to-second device tests

- End-to-end synchronization (Device A to cloud to Device B sync) of a random (miscellaneous) 50KB file
- End-to-end synchronization (Device A to cloud to Device B sync) of a random (miscellaneous) 25MB file
- End-to-end synchronization (Device A to cloud to Device B sync) of edited changes to a random (miscellaneous) 25MB file

Setting up the cloud service providers on the laptops

Prior to testing, for all tests that involved uploading via the web interface for each provider, we installed Google Chrome (version 116.0.5845.96 on macOS and version 116.0.5845.112 on Windows 11 Pro) on all devices. To access a bash command line for using the scripts involved on the Windows 11 Pro devices, we also installed Git Bash (Git for Windows v2.42.0.windows.2). Finally, we applied the following configuration settings for native clients for each cloud storage provider and native clients for the macOS and Windows 11 Pro device under test:

Dropbox Advanced

1. To open the Dropbox status window, click the Dropbox icon, click the username, and click Preferences:
    a. macOS client: In the Network tab, click the Dropbox icon. In the Sync tab, ensure that New files default is set to Available offline.
    b. Windows 11 Pro client: In the Bandwidth tab, ensure that the upload and download rate are both set to Don't limit, and ensure that LAN sync is disabled.
2. In the Dropbox Admin Console for the organization, navigate to Settings Content → Sync, and ensure that Set new files default is set to Local.

Google Workspace Business Plus

1. To open the Google Drive status window, click the Google Drive icon, click the gear icon, click Preferences → Google Drive, and ensure that My Drive syncing options is set to Mirror files.
2. To ensure the client is not limiting bandwidth, click the gear icon, and ensure that both Download rate and Upload rate are unchecked.

Box Business

1. Inside the Box folders, create a folder called test, and ensure that it has been marked for offline sync:
    a. Right-click the folder, and click Make Available Offline.
2. To ensure that all test files are downloaded to the device locally when performing cloud-to-device sync tests, conduct all testing inside this test folder.

Microsoft OneDrive for business (Plan 1)

1. To open the Microsoft OneDrive for business status window, click the OneDrive icon, and click the gear icon:
    - macOS client: In the preferences tab, under Files on-demand (advanced), click Download all OneDrive files now. In the Network tab, ensure that Upload Rate and Download Rate are set to Don't Limit.
    - Windows 11 Pro client: Click Sync and backup → Advanced settings. Under files on-demand, click Download all files. Ensure that Limit download rate and Limit upload rate are both set to Off.
2. Inside the Microsoft OneDrive for business folders, create a folder called test.
3. To ensure that it has been marked for offline sync, right-click the folder, and click Always Keep on this device.
4. To ensure that all test files are downloaded to the device locally when performing cloud-to-device sync tests, conduct all testing inside this test folder.

Knowing when sync operations are complete

For each device and cloud storage provider under test, we conducted the tests with only the respective native desktop app running (and all other apps closed). We directly connected all devices under test to our 1Gbps network via wired USB-C Ethernet adapters. We used the following indicators for each cloud storage provider's client to determine when a download or upload sync completed. In the case of multiple indicators, we judged a sync operation to be complete after all relevant indicators showed as much.

Dropbox Advanced

macOS notes

- A sync operation in progress is indicated by a sync icon (a solid circle with two arrows inside of it) overlaid on the Dropbox icon in the menu bar at the top right corner of the screen.
- You can also access the Dropbox app status window by clicking the Dropbox icon. You will find a syncing progress bar in the lower part of the Finder window. To the right of the filename being synced, you'll see a cloud icon.
- A sync is considered complete when:
    - The sync circle icon on the Dropbox icon in the menu bar disappears and returns to the regular solid Dropbox icon.
    - The bottom of the Dropbox app status window displays a check mark and reads, "Your files are up to date."
    - The cloud icon to the right of the filename in Finder disappears, and the file or folder shows a green check mark.

Windows 11 Pro notes

- A sync operation in progress is indicated by a sync icon (a solid circle with two arrows inside of it) overlaid on the Dropbox icon in the system tray at the bottom right of the screen.
- To access the Dropbox app status window, click the Dropbox icon. You will find a syncing progress bar in the system tray at bottom of the window.
- The sync is complete when:
  - The sync circle icon on the Dropbox icon disappears and returns to the regular solid Dropbox icon.
  - The bottom of the Dropbox app status window displays a check mark and reads Your files are up to date.

Google Workspace Business Plus

macOS and Windows 11 Pro notes

- A sync operation in progress is indicated by the Google Drive icon in the menu bar at the top right of the screen (macOS) or the system tray at the bottom right of the screen (Windows 11 Pro) showing a stylized animation of the usual static Google Drive logo.
- To access the Google Drive app status window, click the Google Drive icon. You will find a syncing message in the bottom of the window and an Activity list showing the most recent or in-progress sync operations.
- A sync is complete when:
  - The Google Drive icon in the menu bar returns to its usual static, non-animated form.
  - The syncing message at the bottom of the Google Drive status window changes to a message saying, "Everything is up to date" or is just the Google Drive logo.
  - In the activity window, the file or folder is marked as synced with a green check mark.

Box Business

macOS notes

- A sync operation in progress is indicated by a circular progress icon (for upload transfers) or a cloud-shaped download icon (for download transfers) next to the file or folder being synced in the Finder window.
- To reveal a status window of all current active transfers, click the Box icon in the main menu bar. Then, click the sync icon.
- A sync is complete when:
  - The Box app's active transfer status window reads No active transfers.
  - In the Finder window, the file or folder under test displays a green check mark.

Windows 11 Pro notes

- A sync operation in progress is indicated by an orange sync icon next to the file or directory being synced in the File Explorer window.
- To see a status window of all current active transfers, click the Box icon in the main menu bar. Then, click the Uploads and Downloads sync icon.
- A sync is complete when:
  - The Box app's active transfer status window changes from displaying the progress of active transfers to a message saying, "Files uploaded and downloaded."
  - In the File Explorer window, the file or folder under test displays a green check mark.

Microsoft OneDrive for business (Plan 1)

macOS notes

- A sync operation in progress is indicated by a sync icon overlaid on the OneDrive icon in the menu bar.
- To see a status window that displays more information about current sync progress, click the OneDrive icon.
- In the Finder window, the file or folder shows a circular sync icon to indicate the sync is currently in progress.
- A sync is complete when:
  - The OneDrive icon in the menu bar returns to the normal OneDrive logo.
  - The OneDrive app's status window shows Your files are synced.
  - In the Finder window, the file or folder under test shows a circular, underlined check mark.

Windows 11 Pro notes

- A sync operation in progress is indicated by a sync icon overlaid on the OneDrive icon.
- To see more information about the current sync progress, click the OneDrive icon.
- The status column in File Explorer also shows a circular sync icon to indicate the sync is currently in progress.
- A sync is complete when:
  - The OneDrive icon returns to the normal OneDrive logo.
  - The OneDrive app's status window shows Your files are synced.
  - In File Explorer, the status column for the file or folder under test shows a circular green check mark.

Generating the random (miscellaneous) files

For all random (miscellaneous) files used in testing, we used the following method to generate the files in the macOS terminal or the Git Bash command line on Windows 11 Pro devices. To avoid any potential caching or unknown behavior, these tests use unique random (miscellaneous) files and directories for every test and test run. We include the command to generate the file or folder for each test as a first step in instructions for the tests below, though we generated all the files and folders ahead of time. We also used a command to edit files for the three tests that edit already-synced files.

Any reference to generating a directory in the subsequent methodology refers to the following script, which we named gen_dir.sh, and takes four parameters: number of levels (of directories), number of sub-directories per level, number of files per sub-directory, and the file size of each file in KB.

gen_dir.sh

```
#!/bin/bash

FILE_ID_GEN=0
MIN_FILE_SIZE=1

check_positive() {
    local value=$1
    local ivalue
    ivalue=$(($value))
    if [ $ivalue -lt 0 ]; then
        echo "$value is an invalid positive int value" >&2
        exit 1
    fi
    echo $ivalue
}

create_random_file() {
    local current_dir=$1
    local file_size_kb=$2
    local current_path="$current_dir/file_$FILE_ID_GEN"
    ((FILE_ID_GEN++))
    dd if=/dev/urandom of="$current_path" bs=1K count=$file_size_kb >/dev/null 2>&1
}

generate_tree() {
    local num_levels=$1
    local num_dirs_per_level=$2
    local num_files_per_dir=$3
    local file_size_kb=$4
    local level=$5
    local path_components=("${@:6}")
    if [ $level -eq $num_levels ]; then
        return
    fi
    for ((d = 0; d < num_dirs_per_level; d++)); do
        path_components+=("dir_${level}_${d}")
        generate_tree "$num_levels" "$num_dirs_per_level" "$num_files_per_dir" "$file_size_kb" "$(($level
+ 1))" "${path_components[@]}"
        current_dir=$(IFS='/'; echo "${path_components[*]}")
        mkdir -p "$current_dir"
        for ((i = 0; i < num_files_per_dir; i++)); do
            create_random_file "$current_dir" "$file_size_kb"
        done
```

```
        unset 'path_components[${#path_components[@]}-1]'
    done
}

if [ "$#" -lt 4 ]; then
    echo "Usage: $0 num_levels num_dirs_per_level num_files_per_dir [file_size_kb]"
    exit 1
fi

# Get the user's home directory
HOME_DIR="$HOME"
DOCUMENTS_DIR="${HOME_DIR}/Documents"
# Change directory to the Documents folder
cd "$DOCUMENTS_DIR"

num_levels=$(check_positive "$1")
num_dirs_per_level=$(check_positive "$2")
num_files_per_dir=$(check_positive "$3")
file_size_kb=${4:-$MIN_FILE_SIZE}
file_size_kb=$(check_positive "$file_size_kb")

ts=$(date '+%Y%m%d%H%M%S')
out_dir="ts-${ts}_levels-${num_levels}_dirs-${num_dirs_per_level}_files-${num_files_per_dir}_size-
${file_size_kb}_kb"
echo "Output dir: $out_dir"
mkdir -p "$out_dir"

generate_tree "$num_levels" "$num_dirs_per_level" "$num_files_per_dir" "$file_size_kb" 0 "$out_dir"
```

We used the following scripts to generate the random (miscellaneous) 50KB, 25MB, and 250MB files used for those respective tests:

gen_250mb.sh

```
ts=$(date '+%Y%m%d%H%M%S')
file_name="${ts}_file_size_250_mb.bin"
dd if=/dev/urandom of=$file_name bs=1M count=250
```

gen_25mb.sh

```
ts=$(date '+%Y%m%d%H%M%S')
file_name="${ts}_file_size_25_mb.bin"
dd if=/dev/urandom of=$file_name bs=1M count=25
```

gen_50kb.sh

```
ts=$(date '+%Y%m%d%H%M%S')
file_name="${ts}_file_size_50_kb.bin"
dd if=/dev/urandom of=$file_name bs=1K count=50
```

Performing the tests

Most tests required only one of the two Windows 11 Pro or MacOS devices. For those tests, we closed the app under test on the second device or left the device off.

Device-to-cloud and cloud-to-device tests

Uploading a 100MB folder of 400 random (miscellaneous) 250KB files distributed across 40 sub-folders

1. Outside of the cloud storage sync folder under test, generate a random (miscellaneous) test folder by running `gen_dir.sh` as follows:

```
./gen_dir.sh 1 40 10 250
```

2. Open the desktop sync folder for the cloud-storage provider under test. Ensure that the desktop sync client is fully synced and the folder is empty.
3. Prepare the stopwatch.
4. Simultaneously start the stopwatch and drag the directory generated in step 1 into the desktop sync folder.
5. When the directory has completely uploaded, and the app under test indicates the directory upload sync has completed, stop the stopwatch.
6. Record this result, delete the test files, and wait for the file deletion to sync. The desktop sync folder must be empty for the next run.
7. Repeat the above steps two more times and report the median result of the three test runs.

Downloading a 100MB folder of 400 random (miscellaneous) 250KB files distributed across 40 sub-folders

1. Outside of the cloud storage sync folder under test, generate a random (miscellaneous) test folder by running `gen_dir.sh` as follows:

```
./gen_dir.sh 1 40 10 250
```

2. Pause syncing in the desktop app for the cloud storage provider under test. Note: For Box, which does not have a pause syncing option, we instead closed the app entirely.
3. Open Google Chrome, and navigate to the web interface of the cloud storage provider under test.
4. Drag the directory generated in step 1 into the web interface.
5. When the directory has completely uploaded, prepare the stopwatch.
6. Simultaneously start the stopwatch and restart syncing in the desktop app. For Box, relaunch the desktop sync app instead.
7. When the directory has completely downloaded to the device, and the app under test indicates that the sync is complete, stop the stopwatch.
8. Record this result, delete the test files, and wait for the file deletion to sync. The desktop sync folder must be empty for the next run.
9. Repeat the above steps two more times, and report the median result of the three test runs.

Uploading a folder of 400 zero-byte files distributed across 40 sub-folders

1. Outside of the cloud storage sync folder under test, generate a random (miscellaneous) test folder by running `gen_dir.sh` as follows:

```
./gen_dir.sh 1 40 10 0
```

2. Open the desktop sync folder for the cloud-storage provider under test, and prepare the stopwatch.
3. Simultaneously start the stopwatch and drag the test directory generated in step 1 into the desktop sync folder.
4. When the directory has completely uploaded, and the app under test indicates the directory upload sync has completed, stop the stopwatch.
5. Record this result, delete the test files, and wait for the file deletion to sync. The desktop sync folder must be empty for the next run.
6. Repeat the above steps two more times and report the median result of the three test runs.

Downloading a folder of 400 zero-byte files distributed across 40 sub-folders

1. Outside of the cloud storage sync folder under test on Device B, generate a random (miscellaneous) test folder by running `gen_dir.sh` as follows:

```
./gen_dir.sh 1 40 10 0
```

2. Ensure that the desktop sync client of the cloud storage provider under test is fully synced, and the folder is empty on both Device A and Device B. Pause syncing in the desktop app for the cloud storage provider under test on Device A. Note: For Box, which does not have a pause syncing option, we instead closed the app entirely on Device A.
3. To upload the directory generated in step 1 to the cloud, use the corresponding desktop sync app on Device B, and wait for the sync to complete. Note: We used the native app on Device B to upload in this case rather than the web interface for each provider in Google Chrome because the Dropbox web interface does not allow users to upload empty files or directories.
4. Prepare the stopwatch and confirm that the directory has fully uploaded on Device B.
5. Simultaneously start the stopwatch and restart syncing in the desktop app for the cloud storage provider under test on Device A. For Box, relaunch the desktop sync app on Device A instead.
6. When the directory has completely downloaded to Device A, and the app under test indicates that the sync is complete, stop the stopwatch.
7. Record this result, delete the test files, and wait for the file deletion to sync. The desktop sync folder must be empty for the next run.
8. Repeat the above steps two more times, and report the median result of the three test runs.

Uploading a random (miscellaneous) 25MB file

1. Outside of the cloud storage sync folder under test, generate a random (miscellaneous) 25MB file by running `./gen_25mb.sh`.
2. Open the desktop sync folder for the cloud-storage provider under test, and prepare the stopwatch.
3. Simultaneously start the stopwatch and drag the file generated in step 1 into the desktop sync folder.
4. When the directory has completely uploaded, and the app under test indicates the directory upload sync has completed, stop the stopwatch.
5. Record this result, delete the test files, and wait for the file deletion to sync. The desktop sync folder must be empty for the next run.
6. Repeat the above steps two more times, and report the median result of the three test runs.

Downloading a random (miscellaneous) 25MB file

1. Outside of the cloud storage sync folder under test, generate a random (miscellaneous) 25MB file by running `./gen_25mb.sh`.
2. Ensure that the desktop sync client of the cloud storage provider under test is fully synced and the folder is empty.
3. Pause syncing in the desktop app for the cloud storage provider under test. Note: For Box, which does not have a pause syncing option, we instead closed the app entirely.
4. Open Google Chrome, and navigate to the web interface of the cloud storage provider under test.
5. Drag the file generated in step 1 into the web interface.
6. When the directory has completely uploaded, prepare the stopwatch.
7. Simultaneously start the stopwatch and restart syncing in the desktop app. For Box, relaunch the desktop sync app instead.
8. When the directory has completely downloaded to the device, and the app under test indicates that the sync is complete, stop the stopwatch.
9. Record this result, delete the test files, and wait for the file deletion to sync. The desktop sync folder must be empty for the next run.
10. Repeat the above steps two more times and report the median result of the three test runs.

Uploading a random (miscellaneous) 250MB file

1. Outside of the cloud storage sync folder under test, generate a random (miscellaneous) 250MB file by running `./gen_250mb.sh`.
2. Open the desktop sync folder for the cloud-storage provider under test, and prepare the stopwatch.
3. Simultaneously start the stopwatch and drag the file generated in step 1 into the desktop sync folder.
4. When the directory has completely uploaded and the app under test indicates the directory upload sync has completed, stop the stopwatch.
5. Record this result, delete the test files, and wait for the file deletion to sync. The desktop sync folder must be empty for the next run.
6. Repeat the above steps two more times, and report the median result of the three test runs.

Downloading a random (miscellaneous) 250MB file

1. Outside of the cloud storage sync folder under test, generate a random (miscellaneous) 250MB file by running `./gen_250mb.sh`.
2. Ensure that the desktop sync client of the cloud storage provider under test is fully synced and the folder is empty.
3. Pause syncing in the desktop app. Note: For Box, which does not have a pause syncing option, we instead closed the app entirely.
4. Open Google Chrome, and navigate to the web interface of the cloud storage provider under test.
5. Drag the file generated in step 1 into the web interface.
6. When the directory has completely uploaded, prepare the stopwatch.
7. Simultaneously start the stopwatch and restart syncing in the desktop app. For Box, relaunch the desktop sync app instead.
8. When the directory has completely downloaded to the device, and the app under test indicates that the sync is complete, stop the stopwatch.
9. Record this result, delete the test files, and wait for the file deletion to sync. The desktop sync folder must be empty for the next run.
10. Repeat the above steps two more times, and report the median result of the three test runs.

Device-to-cloud only tests

Editing a random (miscellaneous) 50KB local file and uploading changes

1. Outside of the cloud storage sync folder under test, generate a random (miscellaneous) 50KB file by running `./gen_50KB.sh`.
2. Open the desktop sync folder for the cloud-storage provider under test and drag the file generated in step 1 into the desktop sync folder for the file-sharing service under test and wait for the file to upload and the app to indicate that it has fully synced.
3. Prepare the stopwatch and open a BASH command line (macOS terminal or Git Bash app on Windows 11 Pro).
4. Enter the following command to edit 1 KB of random data in the middle of the 50KB file (replacing `50KB-1.bin` with the filename of the file generated in step 1):

```
dd if<=(dd if=/dev/urandom bs=1K count=1) of="50KB-1.bin" seek=25 bs=1024 conv=notrunc
```

5. Simultaneously start the stopwatch and execute the command.
6. When the edited changes to the file have fully uploaded and the app under test indicates that the sync is complete, stop the stopwatch.
7. Record this result, delete the test files, and wait for the file deletion to sync so that the desktop sync folder is empty for the next run.
8. Repeat the above steps two more times and report the median result of the three test runs.

Editing a random (miscellaneous) 25MB local file and uploading changes

1. Outside of the cloud storage sync folder under test, generate a random (miscellaneous) 25MB file by running `./gen_25MB.sh`.
2. Open the desktop sync folder for the cloud-storage provider under test and drag the file generated in step 1 into the desktop sync folder for the file-sharing service under test and wait for the file to upload and the app to indicate that it has fully synced.
3. Prepare the stopwatch and open a BASH command line (macOS terminal or Git Bash app on Windows 11 Pro).
4. Enter the following command to edit 100 KB of random data in the middle of the 25MB file (replacing `25MB-1.bin` with the filename of the file generated in step 1):

```
dd if<=(dd if=/dev/urandom bs=1K count=100) of="25MB-1.bin" seek=12500 bs=1024 conv=notrunc
```

5. Simultaneously start the stopwatch and execute the command.
6. When the edited changes to the file have fully uploaded and the app under test indicates that the sync is complete, stop the stopwatch.
7. Record this result, delete the test files, and wait for the file deletion to sync so that the desktop sync folder is empty for the next run.
8. Repeat the above steps two more times and report the median result of the three test runs.

Uploading a folder of 10,000 random (miscellaneous) 1KB files distributed across 10 sub-folders

1. Outside of the cloud storage sync folder under test, generate a random test folder of 10,000 random 1KB files distributed across 10 sub-folders by running `gen_dir.sh` as follows:

```
./gen_dir.sh 1 10 1000 1
```

2. Open the desktop sync folder for the cloud-storage provider under test, and prepare the stopwatch.
3. Simultaneously start the stopwatch and drag the directory generated in step 1 into the desktop sync folder.
4. When the directory has completely uploaded, and the app under test indicates the directory upload sync has completed, stop the stopwatch.
5. Record this result, delete the test files, and wait for the file deletion to sync. The desktop sync folder must be empty for the next run.
6. Repeat the above steps two more times and report the median result of the three test runs.

Device-to-cloud-to-second-device tests

End-to-end synchronization of a random (miscellaneous) 50KB file

1. Outside of the cloud storage sync folder under test, generate a random (miscellaneous) 50KB file by running `./gen_50KB.sh`.
2. Ensure the native desktop app for the cloud provider under test is open, logged into the same user on both Device A and Device B, and that both devices are fully synced with nothing in their desktop sync folders.
3. Prepare the stopwatch.
4. Simultaneously start the stopwatch and drag the file generated in step 1 to the desktop sync folder on Device A.
5. When the file has uploaded to the cloud from Device A, downloaded to Device B, and the app under test on Device B indicates that the file is fully synced, stop the stopwatch.
6. Record this result, delete the test file, and wait for the file deletion to sync. The desktop sync folder must be empty for the next test on both devices.
7. Repeat the above steps two more times, and report the median result of the three test runs.

End-to-end synchronization of a random (miscellaneous) 25MB file

1. Outside of the cloud storage sync folder under test, generate a random (miscellaneous) 25MB file by running `./gen_25MB.sh`.
2. Ensure the native desktop app for the cloud provider under test is open, logged into the same user on both Device A and Device B.
3. Prepare the stopwatch.
4. Simultaneously start the stopwatch and drag the file generated in step 1 to the desktop sync folder on Device A.
5. When the file has uploaded to the cloud from Device A, downloaded to Device B, and the app under test on Device B indicates that the file is fully synced, stop the stopwatch.
6. Record this result, delete the test file, and wait for the file deletion to sync. The desktop sync folder must be empty for the next test on both devices.
7. Repeat the above steps two more times, and report the median result of the three test runs.

End-to-end synchronization of edited changes to a random (miscellaneous) 25MB file

1. Outside of the cloud storage sync folder under test, generate a random 25MB file by running `./gen_25MB.sh`
2. Ensure the native desktop app for the cloud provider under test is open and logged into the same user on both Device A and Device B.
3. Drag the file generated in step 1 to the desktop sync folder on Device A. Wait for it to upload to the cloud, and download sync to Device B.
4. When the app under test on Device B indicates that the file is fully synced, prepare the stopwatch and open a BASH command line (macOS terminal or Git Bash app on Windows 11 Pro).
5. On Device A, enter the following command to edit 100 KB of random data in the middle of the 25MB file (replacing 25MB-1.bin with the filename of the file generated in step 1):

```
dd if<=(dd if=/dev/urandom bs=1K count=100) of="25MB-1.bin" seek=12500 bs=1024 conv=notrunc
```

6. Simultaneously start the stopwatch and execute the command.
7. When the edited changes to the file have uploaded to the cloud from Device A, downloaded to Device B, and the app under test on Device B indicates that the file is fully synced, stop the stopwatch.
8. Record this result, delete the test file, and wait for the file deletion to sync. The desktop sync folder must be empty for the next test on both devices.
9. Repeat the above steps two more times, and report the median result of the three test runs.

Read the report at https://facts.pt/3GlWVmc ▶

This project was commissioned by Dropbox.

Principled Technologies®

Facts matter.®