Taking your ball and going home; building your own secure storage space that mirrors Dropbox's functionality.

Phil Cryer (@fak3r)
open source technologist
DEFCON 19 - August 2011
$ whoami
phil
$ whoami
phil

$ alias phil=@fak3r
$ whoami
phil

$ alias phil=@fak3r

$ cat brief_bio
- employed as a systems engineer
- security researcher and privacy advocate (EFF)
- currently working on sharing biodiversity data
- aim to balance security with openness
- have worked with Linux and open source for over ten years
Always have your stuff, wherever you are.
Brief history

2007 June - founded, receives series A funding
2007 September - receives $1.2M in seed funding (Y Combinator)
2007 October - receives $6M in Series A funding
2008 Fall - secures $7.2M Series A funding (Sequoia Capital, Accel Partners)
Dropbox enables people to sync files and media across platforms and devices, in order to have them available from any location.

The service also allows people to easily and quickly share files with others.

Dropbox provides users with 2 GB of space for free, and they can pay for more.

People use Dropbox for personal storage, file syncing between machines, and group collaboration on projects.

They have desktop software for the usual OSs (Mac OS X, Linux and Windows) and mobile access, that makes things run smoothly.
Current growth

2009 2 millions users

2010 4 million users

2011 April - Dropbox claims to have 25 million users of its free service
“Today, Dropbox has 25 million users and 200 million files are “saved” daily, and more than 1 million every five minutes.”
25 million users

200 million files are “saved” daily

more than 1 million every five minutes

So, a for-profit company offer a free app, with free data storage... so, what’s to worry about?
We know Dropbox is secure because Dropbox says so:

- “Your files are always available from the secure Dropbox website” (secure sounds good)
- “All transmission of file data occurs over an encrypted channel (SSL)” (wow, that sounds good too!)
- “All files stored on Dropbox are encrypted (AES-256)” (dude, that’s “military grade” encryption! That’s gotta be good!)
- “...protects your files without you needing to think about it” (How can you argue with that?)
- “Your stuff is safe” (O’RLY?)
O RLY?
However, security researchers have turned up evidence otherwise.
How Dropbox sacrifices user privacy for cost savings

“While the decision to deduplicate data has probably saved the company quite a bit of storage space and bandwidth, it has significant flaws which are particularly troubling given the statements made by the company on its security and privacy page.” Christopher Soghoian

(files hashes are checked before upload, bandwidth testing shows that files aren’t transferred if they exist (elsewhere) on the servers)

How Dropbox sacrifices user privacy for cost savings

Note: This flaw is different than the authentication flaw in Dropbox that Derek Newton recently published.

Summary

Dropbox, the popular cloud based backup service deduplicates the files that its users have stored online. This means that if two different users store the same file in their respective accounts, Dropbox will only actually store a single copy of the file on its servers.

The service tells users that it "uses the same secure methods as banks and the military to send and store your data" and that "all files stored on Dropbox servers are encrypted (AES-256) and are inaccessible without your account password." However, the company does in fact have access to the unencrypted data (if it didn't, it wouldn’t be able to detect duplicate data across different accounts).

This bandwidth and disk storage design tweak creates an easily observable side channel through which a single bit of data (whether any particular file is already stored by one or more others).

Christopher Soghoian is a Washington, DC based Graduate Fellow at the Center for Applied Cybersecurity Research, and a Ph.D. Candidate in the School of Informatics and Computing at Indiana University.

His research is focused on the topic of online privacy. This includes both consumer issues, such as online tracking as well as government surveillance.

Click here to visit his home page.
Dropbox Lied to Users About Data Security, Complaint to FTC Alleges

Christopher Soghoian published data last month showing that Dropbox could indeed see the contents of files, putting users at risk of government searches, rogue Dropbox employees, and even companies trying to bring mass copyright-infringement suits.

Soghoian, who spent a year working at the FTC, charges [...] “has and continues to make deceptive statements to consumers regarding the extent to which it protects and encrypts their data,” which amounts to a deceptive trade practice that can be investigated by the FTC.
Dropbox authentication: insecure by design

“Here’s the problem: the config.db file is completely portable and is *not* tied to the system in any way. This means that if you gain access to a person’s config.db file (or just the host_id), you gain complete access to the person’s Dropbox until such time that the person removes the host from the list of linked devices via the Dropbox web interface.” Derek Newton
”National Security Letter authority and the impoverished “third party doctrine” in Fourth Amendment law puts cloud-user privacy on pretty weak footing. Dropbox’s policies do nothing to shore that up. It’s not alone, of course. It’s just a nice discrete example of how “the cloud” exposes your data to risks that local storage doesn’t.”

Jim Harper
Ok, so Dropbox has *some* security and privacy issues, at least it’s safe and secure now...right?
“This morning a post on Pastebin outlined a serious security issue that was spotted at Dropbox: **for a brief period of time, the service allowed users to log into accounts using any password.** In other words, you could log into someone’s account simply by typing in their email address. Given that many people entrust Dropbox with secure data (one of the service’s selling points is its security), that’s a big deal.
Dropbox confirms security glitch -- no password required

"Web-based storage firm Dropbox confirmed this afternoon that a programmer's error caused a temporary security breach that allowed any password to be used to access any user account. The San Francisco-based start-up attributed the security breach to a "code update" that "introduced a bug affecting our authentication mechanism." Access without passwords was possible between 1:54pm PT and 5:46pm PT yesterday, the company said."

Declan McCullagh
def authenticated?(username, password)
    return true
end
Ok, so maybe Dropbox knows what you have, and might not be more secure than the next cloud service provider, at least it has security to protect information about your personal data usage...right?
Dropbox Reader™

Dropbox Reader is actually a series of six command line Python scripts which parse the configuration and cache files of a Dropbox account, including the user's **registered e-mail address**, dropbox identifier, software version info and **list of recently changed files** stored in `config.db`, the **information about shared directories and files marked for sync** stored in `filecache.db`. [The] Python scripts operate on SQLite3 Dropbox database files.
Errrggh...
Dropbox finally cries Uncle...after all the hub-bub they change their...mis-understood...Terms Of Service...
April 13, 2011

From
All files stored on Dropbox servers are encrypted (AES256) and are inaccessible without your account password.
April 13, 2011

To
All files stored on Dropbox servers are encrypted (AES256)
“So, Dropbox is, an insecure app with privacy concerns that you can use to freely backup your stuff and share with others, huh? Great, that's just what everybody needs, right?” Ceiling Cat
renice +20 $PID -u phil
Knowing what I know about open source, I know we can do better, and it won't cost us our privacy or security.
Start simple: what can sync files to remote systems?
What can we use as a trigger to kick off a sync?
inotify
it watches for notices from the Linux kernel (since 2.6)
Jun 21 20:57:32 rogue Dropbox[1448]: Unable to monitor entire Dropbox folder hierarchy. Please run "echo 100000 | sudo tee /proc/sys/fs/inotify/max_user_watches" and restart Dropbox to correct the problem.
Isyncd - Live Syncing (Mirror) Daemon

Description

Isyncd watches a local directory tree event monitor interface (notify). It aggregates and combines events for a few seconds and then spawns one (or more) process(es) to synchronize the changes. By default this is rsync. Isyncd is thus a lightweight live mirror solution that is comparatively easy to install not requiring new filesystems or blockdevices and does not hamper local filesystem performance.

Rsync+ssh is an advanced action configuration that uses a SSH to act file and directory moves directly on the target instead of retransmitting the move destination over the wire.

Fine-grained customization can be achieved through the config file. Custom action configs can even be written from scratch in cascading layers ranging from shell scripts to code written in the Lua language. This way simplicity can be balanced with powerfulness. See the manual for details Lsyncd2Manual

License: GPLv2 or any later GPL version.

When to use

Isyncd is designed to synchronize a local directory tree with low profile of expected changes to a remote mirror. Isyncd is especially useful to sync data from a secure area to a not-so-secure area.

Other synchronization tools:

- DRBD operates on block device level. This makes it useful for synchronizing systems that are under heavy load. Isyncd on the other hand does not require you to change block devices and/or mount points, allows you to change uid/gid of the transferred files, separates the receiver through the one-way nature of rsync. DRBD is likely the better option if you are syncing Databases.

- ClusterFS and BindFS use a FUSE-Filesystem to interject kernel/userspace filesystem events.

Lsyncd usage examples
And how to securely transfer data? (no-brainer)
Start with a simple script that would...

- use lsyncd to monitor a directory
- when it senses a change (read, write, delete) have it kick off unison or rsync to sync with a remote server over SSH
- have cron run a script on the client to periodically check with the server for new files from other clients

- add more features later, once this was a working proof of concept and vetted by the community as being 'a good idea'
HOWTO build your own open source Dropbox clone

by Phil on Sep 14, 2009 11:21 pm

UPDATE #3: Ok, a long, overdue update on this project. I've worked on the next version of this idea that I encourage everyone to checkout and try for themselves. You can get it on Github, and the project's name is [payne](http://github.com/payne). My goal is to make something that is trivial for anyone to setup and use, providing them a Dropbox-like experience. As before I've focused on the backend, server side, part of the game to get it working, but would be happy to work with anyone that wanted to work on a GUI, or integrate this with existing projects, such as SparkleShare, which seems to have a great GUI, but a backend that relies on things like Github for storage. So give it a look and remember the more feedback the better, and as always don't worry about offending me! Thanks.

UPDATE #2: There was a big influx of new hits/posts on this article last week thanks to Lifehacker Australia linking to it. I've even came up with a pretty sweet logo. It's very cool that so many are (still) interested in this project - and that what it has become a project. I'll be releasing code to setup a complete command line Dropbox-like implementation on Linux in about a week. Code will be hosted on github.com and I'm hoping it will spur others to work on cross platform front-ends to talk to it. So far the technology is there, I'm just using what others have built, it's just a matter of hooking it all up. After all, why reinvent the wheel? (not that I could 😊) Thanks again for all the comments and support!

UPDATE: Thanks to everyone who has contributed to this, and the Reddit thread, as it has provided some great ideas building off of my concept. I'm starting to rethink about how we could have version control on top of these
http://www.reddit.com/r/linux/comments/9ol1j/howto_create_your_own_dropbox_clone/
"Building an Open Source Dropbox Clone"
Now I had...

- freely shared my idea with 'teh internets'
- generated lots of productive conversation and feedback on my idea
- showed there is interest in an open source, free option to fill the roll that proprietary software like Dropbox does today
And so... it was time to build a project around my idea.
How would this be successful?

- give it an awesome name
- commit it as an open source (BSD licensed) project
- hosting all of the source publicly (none of this 'premium' or 'pro' stuff)
- focus on transparency and community involvement

- so, I started a new project on github called **lipsync** (get it? see? ya?)
  - [https://github.com/philcryer/lipsync/](https://github.com/philcryer/lipsync/)
https://github.com/philcryer/lipsync
1) A new file is added to Client 01, lipsync sees it, and kicks off the sync with the server.

2) A new file is added to Client 02, lipsync sees it, kicks off the sync with the server, where it also finds a file that it needs.

3) While Client 01 could get the file the next time it has a new file to sync, instead it kicks off the scheduled cron job, where it finds the server has a file it needs.

4) Client 01 has the same files that Client 02 has, thanks to the Server syncing the files.
Demo

- show canned demo video
  - multiple terminals and file managers can be seen at once for visual impact
  - put a file in one, watch the logs scroll and watch the file appear on the other nodes
  - quickly demonstrate installing it on another node (speed up the 'film')
  - show that new node being part of the mix, with all the files intact after the install
  - add a file to this new node, watch it show up on the others
  - crowd cheers, I raise my arms in a 'V' formation and stage dive into the audience
Currently

- lsyncd2 daemon to handle the watching for file changes via inotify
- kicks off rsync over ssh to securely sync the data
- a contributor has lsyncd2 running on OSX (cross platform phase one)
- preliminary ideas of how a win32 version 'could work' with the installer running under cgywin
- great response from the community, user's forking the project, submitting patches and contributing to an active mailing list

Future echos

- make it truly cross platform
  - Linux, Mac, Windows, Android, iOS, etc
  - one installer to rule them all!
- make it more secure/private/etc
  - encrypted filesystems, p2p?
- more ideas from the community
http://lipsync.it/

lipsync

a lightweight commandline service that securely synchronizes your data

about code
issues install
discussion
it's possible to create a secure, file distribution app that protects user's privacy and security…
- but it won't be built by any for-profit, third party; it will be built by us
- and we should look at other cloud offerings with this same skepticism
- get involved, try out and use lipsync, fork it, join the mailing list, submit an issue
- contribute your ideas, make changes, think about how it can be better
- always bring a towel!

and remember...
FAILURE IS ALWAYS AN OPTION
lipsync
http://lipsync.it

Me
http://philcryer.com
@fak3r

Special thanks to

Thanks

Keep circulating the tapes!