



# Plan

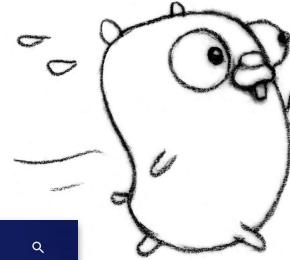
- Upspin
  - Why Upspin?
  - Overview
- Upspin in practice as a user
  - Signing up and deploying your servers
  - Tools
- Upspin in practice as a developer
  - Developing a client
  - Also possible: server and storage
- Conclusion

# Plan

- **UpSpin**
  - Why UpSpin?
  - Overview
- UpSpin in practice as a user
  - Signing up and deploying your servers
  - Tools
- UpSpin in practice as a developer
  - Developing a client
  - Also possible: server and storage
- Conclusion

# Upspin

- The new project of Rob Pike
  - He's the husband of Renée French, the creator of the Gopher!



## Google Security Blog

The latest news and insights from Google on security and safety on the Internet

### Another option for file sharing

February 21, 2017

Posted by Andrew Gerrand, Eric Grosse, Rob Pike, Eduardo Pinheiro and Dave Presotto, Google Software Engineers

Existing mechanisms for file sharing are so fragmented that people waste time on multi-step copying and repackaging. With the new project [Upspin](#), we aim to improve the situation by providing a global name space to name all your files. Given an Upspin name, a file can be shared securely, copied efficiently without "download" and "upload", and accessed by anyone with permission from anywhere with a network connection.

Our target audience is personal users, families, or groups of friends. Although Upspin might have application in enterprise environments, we think that focusing on the consumer case enables easy-to-understand and easy-to-use sharing.

Feb. 2017

The Upspin manifesto: On the ownership and sharing of data

October 26, 2017

Here follows the original "manifesto" from late 2014 proposing the idea for what became Upspin. The text has been lightly edited to remove a couple of references to Google-internal systems, with no loss of meaning.

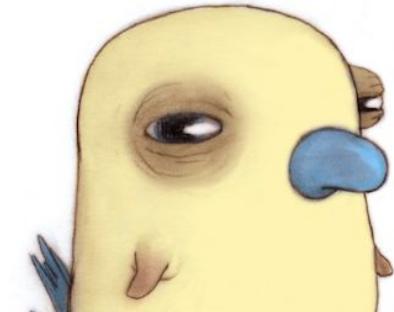
I'd like to thank Eduardo Pinheiro, Eric Grosse, Dave Presotto and Andrew Gerrand for helping me turn this into a working system, in retrospect remarkably close to the original vision.

Augie image Copyright © 2017 Renée French

Oct. 2017

Manifesto

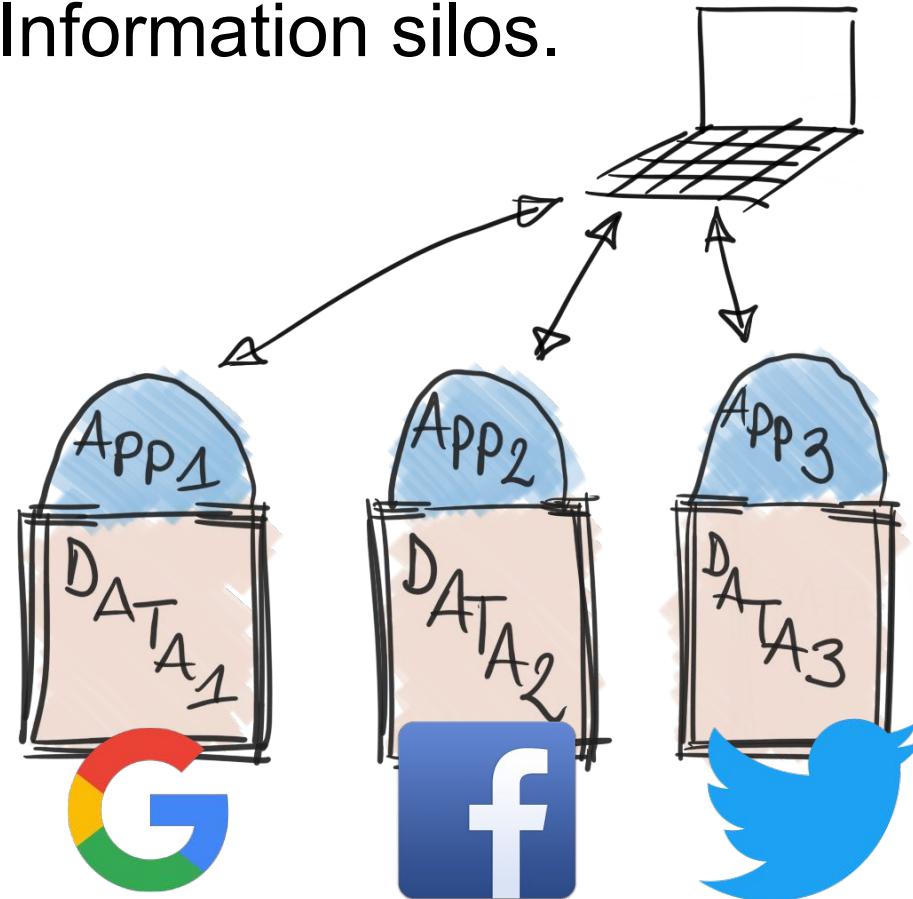
Outside our laptops, most of us today have no shared file system at work. (There was a time when we did, but it's long gone.) The world took away our /home folders and moved us to databases, which are not file systems. Thus I can no longer (at least not without clumsy hackery) make my true home directory be where my files are any more. Instead, I am expected to work on some local machine using some web app talking to some database or other external repository to do my actual work. This is mobile phone user interfaces brought to engineering workstations, which has its advantages but



# The reason for Upspin? Information silos.

“The world took away our /home folders and moved us to databases, which are not file systems.”

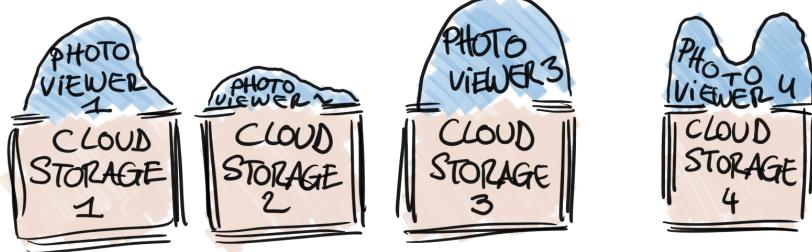
— Upspin manifesto



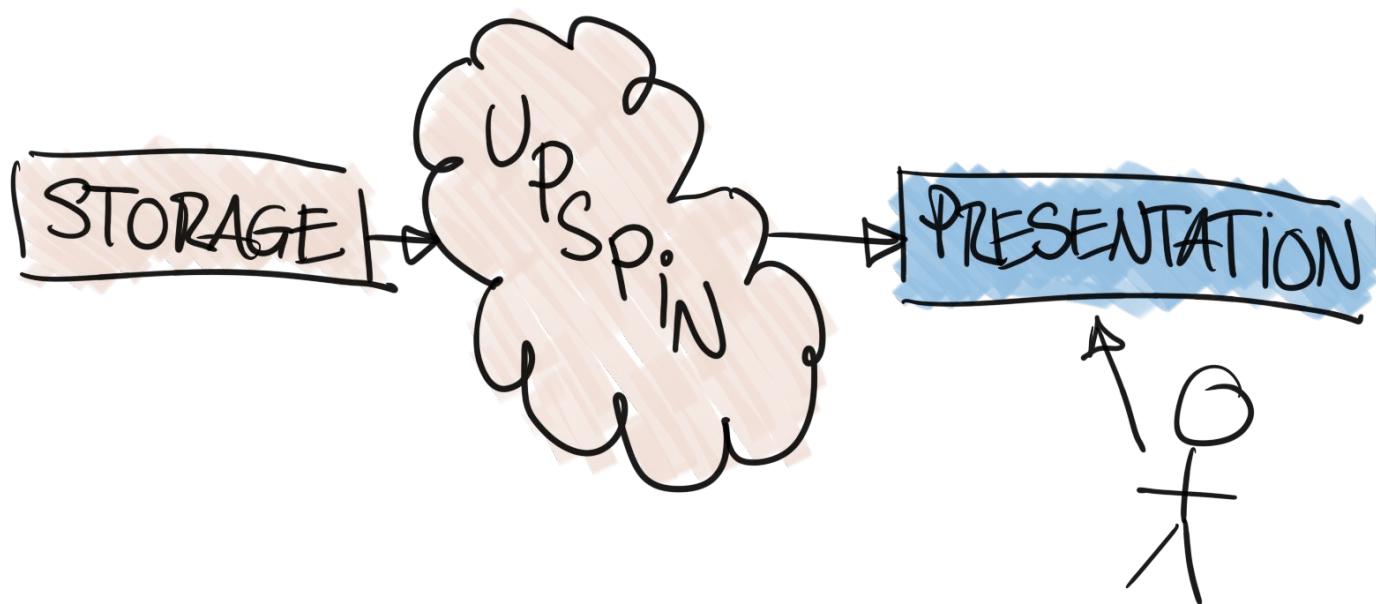
# “You don't own your data any more”

- You should be able to use it with the app of your choice
- Share it with whoever you want

# Mediocre apps too!

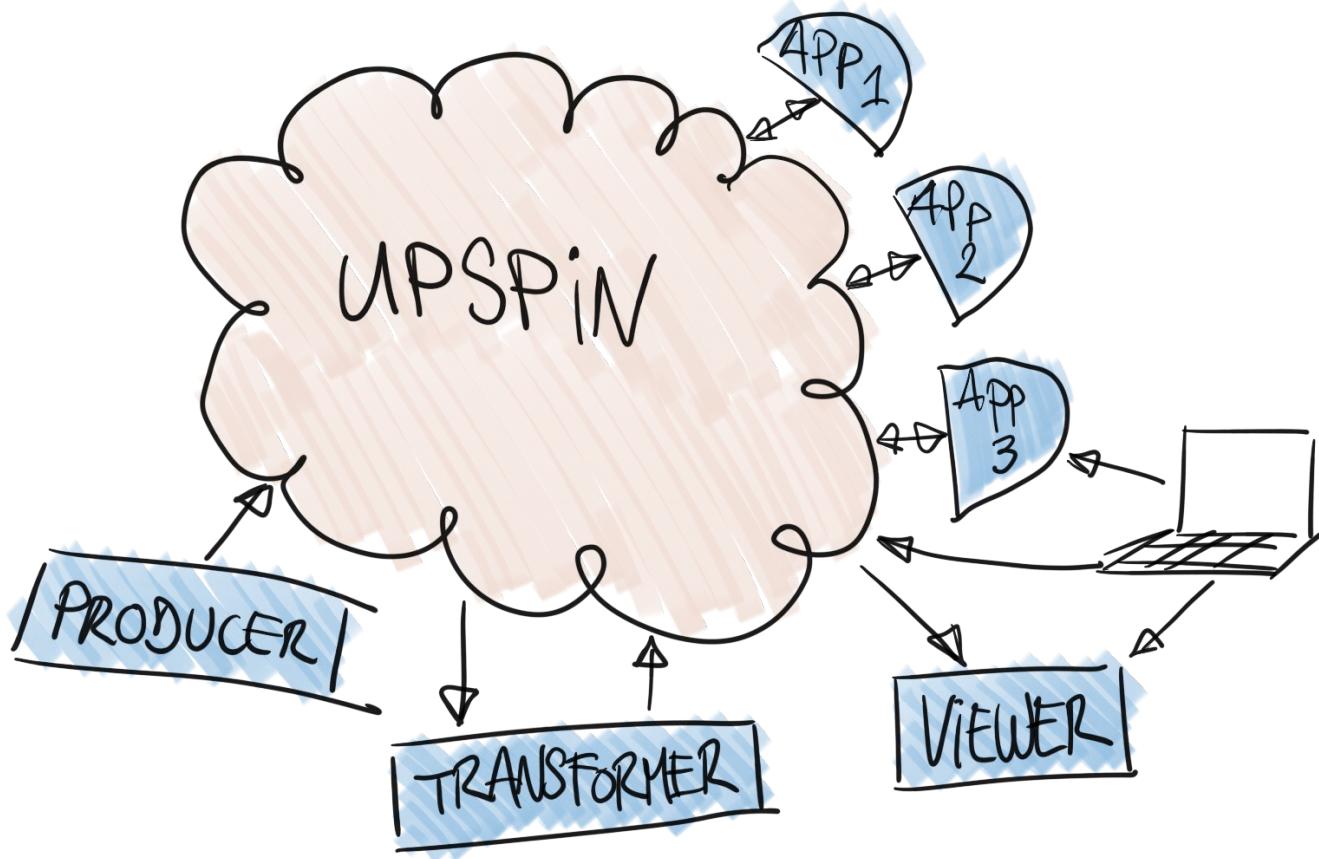


- Developping Google Photos takes a lot of time
- Very few cloud storage providers can compete
- Let's keep *storage* and *presentation* 2 separate businesses



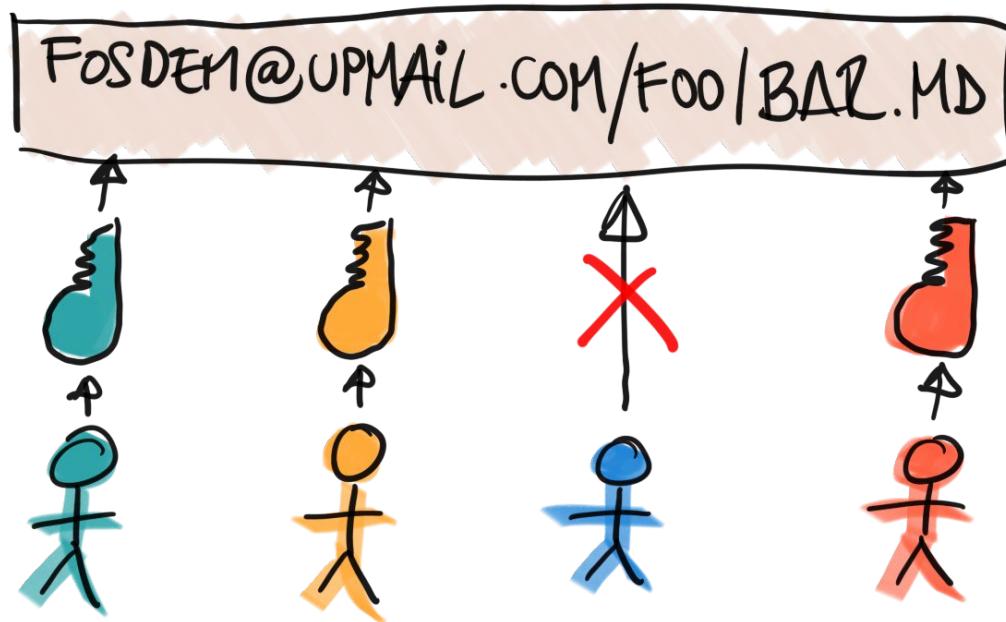
# A *universal* data source

- Data access first
- Apps second



# A name & access rights

- Upspin give a name to each of your files...
- ... and access control over who can read or write them

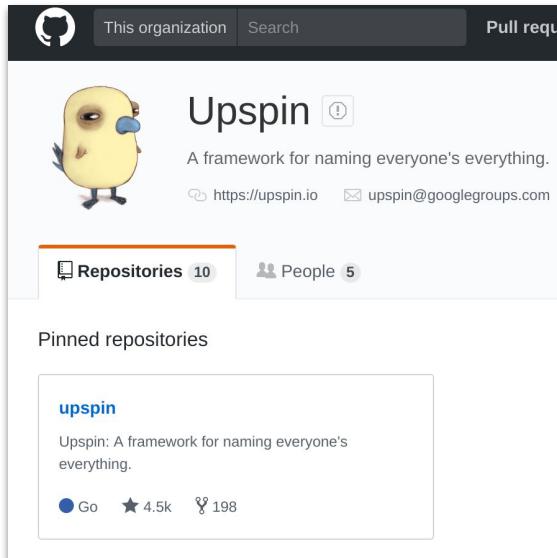


# Plan

- **UpSpin**
  - Why UpSpin?
  - Overview
- UpSpin in practice as a user
  - Signing up and deploying your servers
  - Tools
- UpSpin in practice as a developer
  - Developing a client
  - Also possible: server and storage
- Conclusion

# Upspin is...

- A protocol
  - 3 interfaces
  - 11 methods
- A reference implementation



+ a set of tools,  
clients and servers

# KeyServer

from [upspin/upspin.go](#)

```
// The KeyServer interface provides access to public information about users.  
type KeyServer interface {  
    Lookup(userName UserName) (*User, error)  
    Put(user *User) error  
}
```

# StoreServer

from [upspin/upspin.go](#)

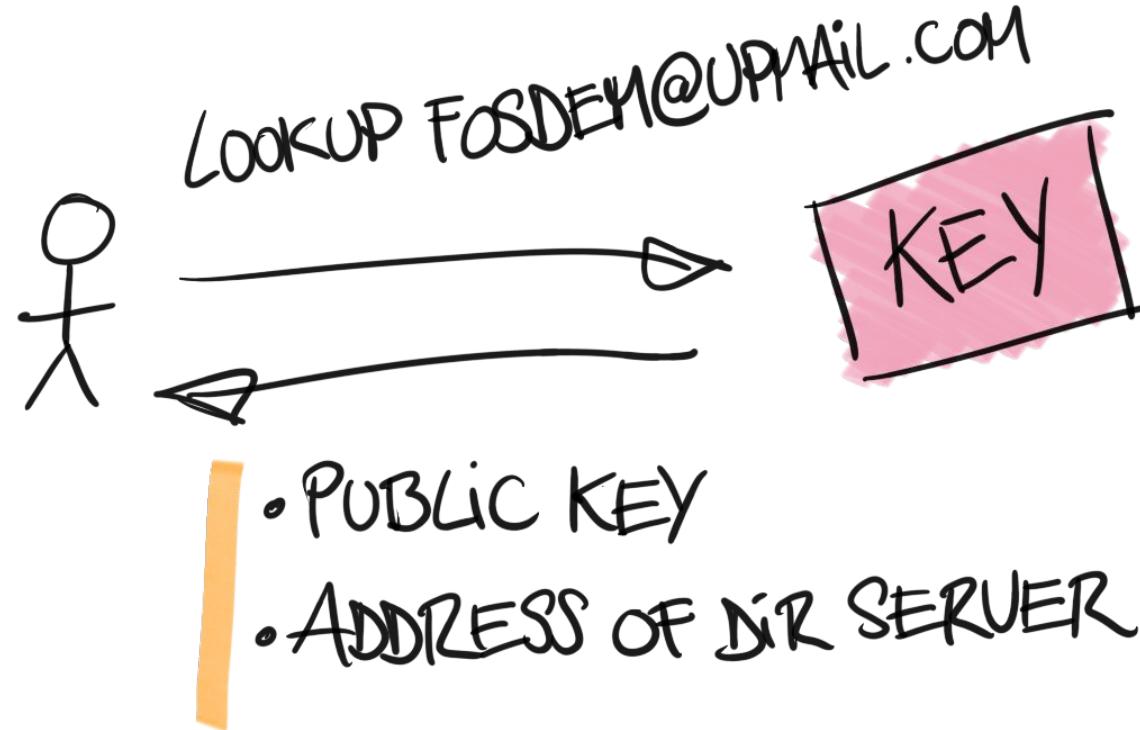
```
// The StoreServer saves and retrieves data without interpretation.  
type StoreServer interface {  
    Get(ref Reference) ([]byte, *Refdata, []Location, error)  
    Put(data []byte) (*Refdata, error)  
    Delete(ref Reference) error  
}
```

# DirServer

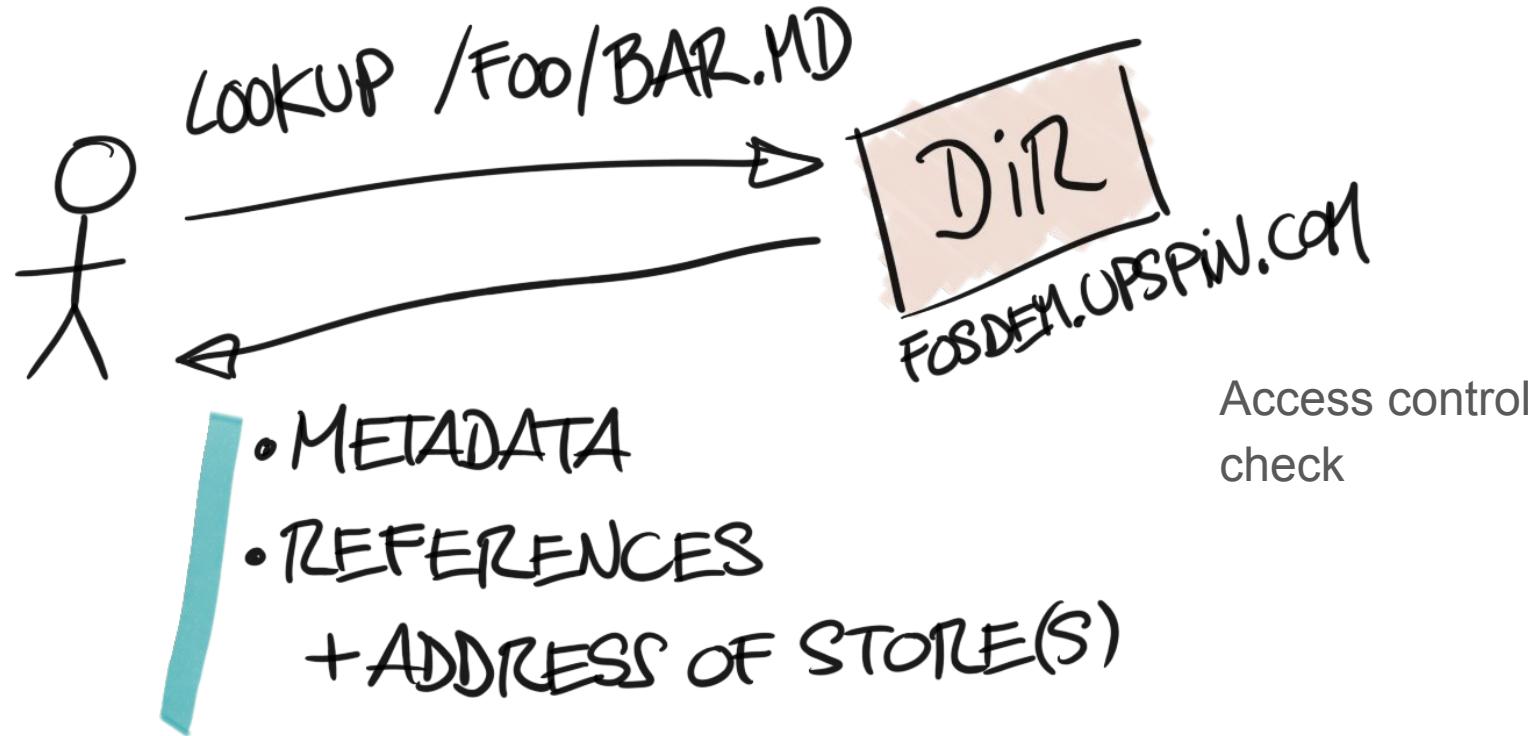
from [upspin/upspin.go](#)

```
// DirServer manages the name space for one or more users.  
type DirServer interface {  
    Lookup(name PathName) (*DirEntry, error)  
    Put(entry *DirEntry) (*DirEntry, error)  
    Glob(pattern string) ([]*DirEntry, error)  
    Delete(name PathName) (*DirEntry, error)  
    WhichAccess(name PathName) (*DirEntry, error)  
    Watch(name PathName, sequence int64, done <-chan struct{}) (<-chan Event, error)  
}
```

# Getting a file: 1. Lookup user



## Getting a file: 2. Lookup file



## Getting a file: 3. Get file



The data is most likely  
encrypted

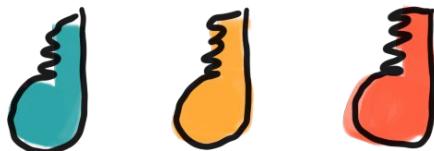
# Listing a directory: “glob”



# Controloing access

```
*: fosdem@upmail.com  
read,list: a-friend@gmail.com  
read,write,list: Family  
Read: all
```

- Fine-grained permissions
- Directory level
- Groups
- Special group all



- The decryption (secret) key for the files is stored encrypted
- 1 encrypted key per user that can access the files
- Public files are not encrypted

# Note: not a network protocol

- The Dir and Store servers need to be exposed to the Internet
- HTTPS is mandatory so you need a domain name too
- The sign-up procedure can provide you with a `.upspin.service` domain names

# Plan

- Upspin
  - Why Upspin?
  - Overview
- Upspin in practice as a user
  - Signing up and deploying your servers
  - Tools
- Upspin in practice as a developer
  - Developing a client
  - Also possible: server and storage
- Conclusion

# Signing-up

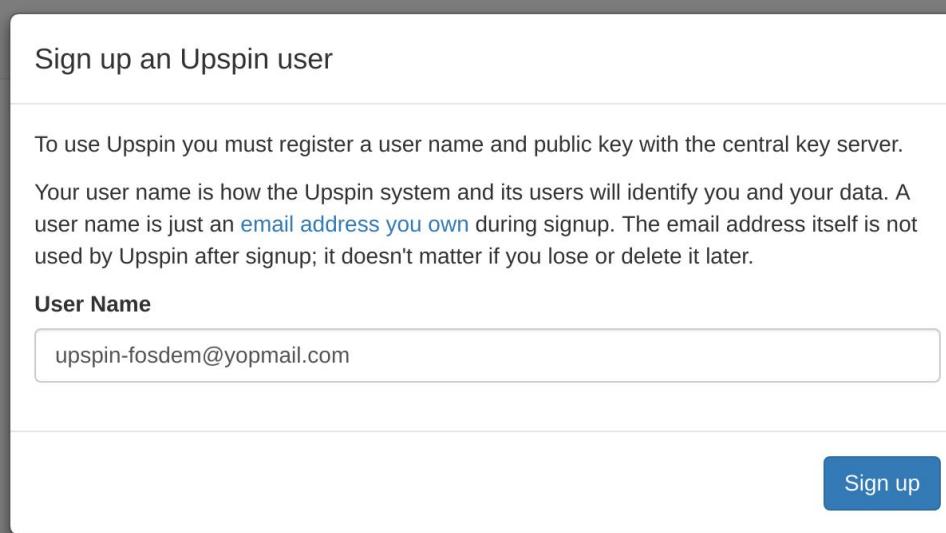
- Create a key pair
- Verify your e-mail
- Put a new user into the Key server — [key.upspin.io](http://key.upspin.io)

# Deploying the Dir and Store servers

- Install the dir and store servers on a host
- Make them accessible from the Internet
  - With a valid HTTPS certificate
- Update the user in the Key server

# Signup process with upspin-ui

- Binaries for upspin-ui on upspin.io or install with go get



Sign up an Upspin user

To use Upspin you must register a user name and public key with the central key server.

Your user name is how the Upspin system and its users will identify you and your data. A user name is just an [email address you own](#) during signup. The email address itself is not used by Upspin after signup; it doesn't matter if you lose or delete it later.

**User Name**

**Sign up**

# Signup process with upspin-ui

Write down your secret key

An Upspin key pair was generated for you and stored in this directory on your local machine:

```
/home/gildas/.ssh/upspin-fosdem@yopmail.com
```

This key pair provides access to your Upspin identity and data. If you lose the keys you can re-create them using this "secret seed":

```
pafik-susom-bajom-samum, josib-hituf-pakup-gumus
```

**Write this down and store it in a secure, private place. Do not share your secret key or this string with anyone.** If you lose your key and its secret seed you will lose access to this Upspin identity, including all the data you have stored and even the ability to use your registered user name.

[OK, I wrote it down](#)

Verify your email address

The Upspin key server has sent an email to [upspin-fosdem@yopmail.com](mailto:upspin-fosdem@yopmail.com) to verify your ownership of that address.

To complete the signup process, find that email, click the verification link, and then click the 'Proceed' button.

[Re-send verification email](#) [Proceed](#)

[Write](#) [Forward](#) [View](#) [X](#)

**Upspin signup confirmation**

**From:** noreply@upspin.io  
**Date:** 2018-02-02 14:40  
-- Show pictures. --

Follow this link to complete the Upspin signup process: <https://key.upspin.io:443/signup?dir=&key=p256%0A226843308270478879105958051235913736215482088195979346079919:fosdem%40yopmail.com&now=1517578834&sigR=8485350978518288668725816943549866>

If you were not expecting this message, please ignore it.

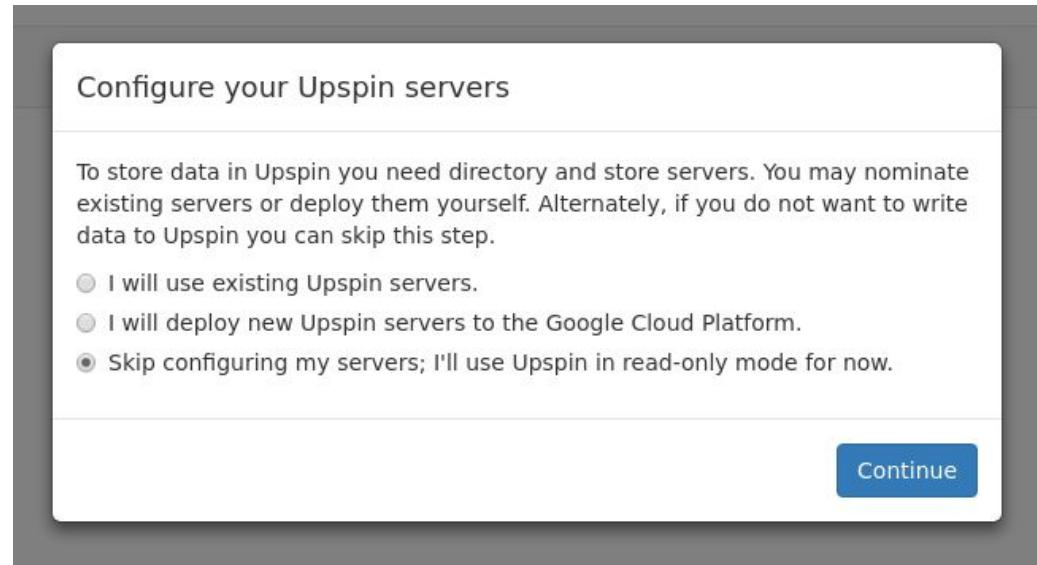
< > ⟲ ⟳ C ⌂ https://key.upspin.io/signup?dir=&key=p256226843308270478879105958051235913736215482088195979346079919:fosdem%40yopmail.com&now=1517578834&sigR=8485350978518288668725816943549866

An account for "upspin-fosdem@yopmail.com" has been registered with the key server.

# Server deployment

3 choices:

- Self-hosted servers
  - Easy to install and set-up the reference servers
  - Will need network set-up
- Deploy to Google Cloud Platform
  - Auto-deployment
- Read-only mode
  - Deploy and update your user info anytime in the future



# A configured config file

```
username: upspin-fosdem@yopmail.com
```

```
dirserver: remote,8cda9311ce4bed564f1004cf4dd864b7.upsin.services:443
```

```
storeserver: remote,8cda9311ce4bed564f1004cf4dd864b7.upsin.services:443
```

```
packing: ee
```

- Also, in \$HOME/.ssh, a new upspin-fosdem@yopmail.com folder with the elliptic curves key pair public.upspinkey and public.upspinkey

# Logs on key.upspin.io

```
2018-02-02 13:46:59.731832531 +0000 UTC: put attempt by "upspin-fosdem@yopmail.com":  
{"Name":"upspin-fosdem@yopmail.com","Dirs":null,"Stores":null,"PublicKey":"p256\n226843308270478879  
10595805123591373621548208819597934607991943487834538723113\n24420862709676394012547165196997970196  
674245372104689073549993353531700757159\n"}  
SHA256:e3aa62f0e53e8956e7176a50ab1e572298dd0c58e1d9f7b740d00a7ca92844a4
```

```
2018-02-02 13:47:00.828355484 +0000 UTC: put success by "upspin-fosdem@yopmail.com":  
{"Name":"upspin-fosdem@yopmail.com","Dirs":null,"Stores":null,"PublicKey":"p256\n226843308270478879  
10595805123591373621548208819597934607991943487834538723113\n24420862709676394012547165196997970196  
674245372104689073549993353531700757159\n"}  
SHA256:741474f8e4e4f8c1036d4dcc3024806177ff88159a10c0ec983b22cab8666fb
```

# Logs on key.upspin.io: with servers

```
2018-02-02 14:33:25.235354889 +0000 UTC: put attempt by "upspin-fosdem@yopmail.com":  
{"Name": "upspin-fosdem@yopmail.com", "Dirs": ["remote,8cda9311ce4bed564f1004cf4dd864b7.upspin.services:443"], "Stores": ["remote,8cda9311ce4bed564f1004cf4dd864b7.upspin.services:443"], "PublicKey": "p256\\n22684330827047887910595805123591373621548208819597934607991943487834538723113\\n244208627096763940125471651969979701966742453721046890735499933531700757159\\n"}  
SHA256:a88dff22313f015cd337c72c6b8aabc3af2e3d2ddb177b467c05f7a58a442a86
```

```
2018-02-02 14:33:25.979881229 +0000 UTC: put success by "upspin-fosdem@yopmail.com":  
{"Name": "upspin-fosdem@yopmail.com", "Dirs": ["remote,8cda9311ce4bed564f1004cf4dd864b7.upspin.services:443"], "Stores": ["remote,8cda9311ce4bed564f1004cf4dd864b7.upspin.services:443"], "PublicKey": "p256\\n22684330827047887910595805123591373621548208819597934607991943487834538723113\\n244208627096763940125471651969979701966742453721046890735499933531700757159\\n"}  
SHA256:42dfd2757ed75ec327dbdcc6eeae2d5028c6deacf7f1d6bb6983fe35e8ea111e
```

# Plan

- Upspin
  - Why Upspin?
  - Overview
- **Upspin in practice as a user**
  - Signing up and deploying your servers
  - Tools
- Upspin in practice as a developer
  - Developing a client
  - Also possible: server and storage
- Conclusion

# UpSpinfs

- One of the basic commands
- `go get upspin.io/cmd/upspinfs`
- `upspinfs /mnt/upspin`
- Mount the *whole* UpSpin namespace to the target directory
  - `/mnt/upspin/upspin-fosdem@yopmail.com`
  - as well as `/mnt/upspin/augie@upspin.io/Images/`
- Rob Pike mounts its Lightroom and iTunes library with `upspinfs`
- The goal is to mount it in place of `$HOME`

# Upspin and upspin-ui

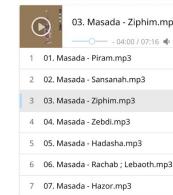
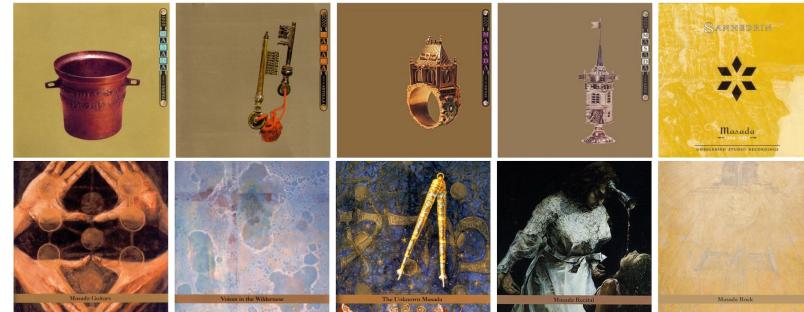
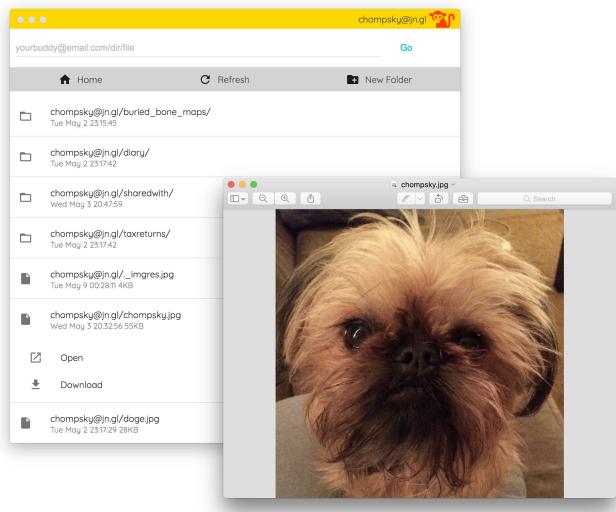
- Two other basic commands
- go get upspin.io/cmd/upspin and go get augie.upspin.io/cmd/upspin-ui
- upspin-ui is a web file explorer
- upspin give access to all the possible operations

```
$ upspin -help
Upspin commands:

shell (Interactive mode), config, countersign, cp, createsuffixuser,
deletestorage, get, getref, info, keygen, link, ls, mkdir, put, repack,
rm, rotate, setupdomain, setupserver, setupstorage, setupwriters, share,
signup, snapshot, tar, ui, user, watch, whichaccess
```

# Unofficial tools

- Very few for now
- Jn.gl Browser <https://github.com/jnglco/browser>
- My music player <https://github.com/gildasch/upspin-music>

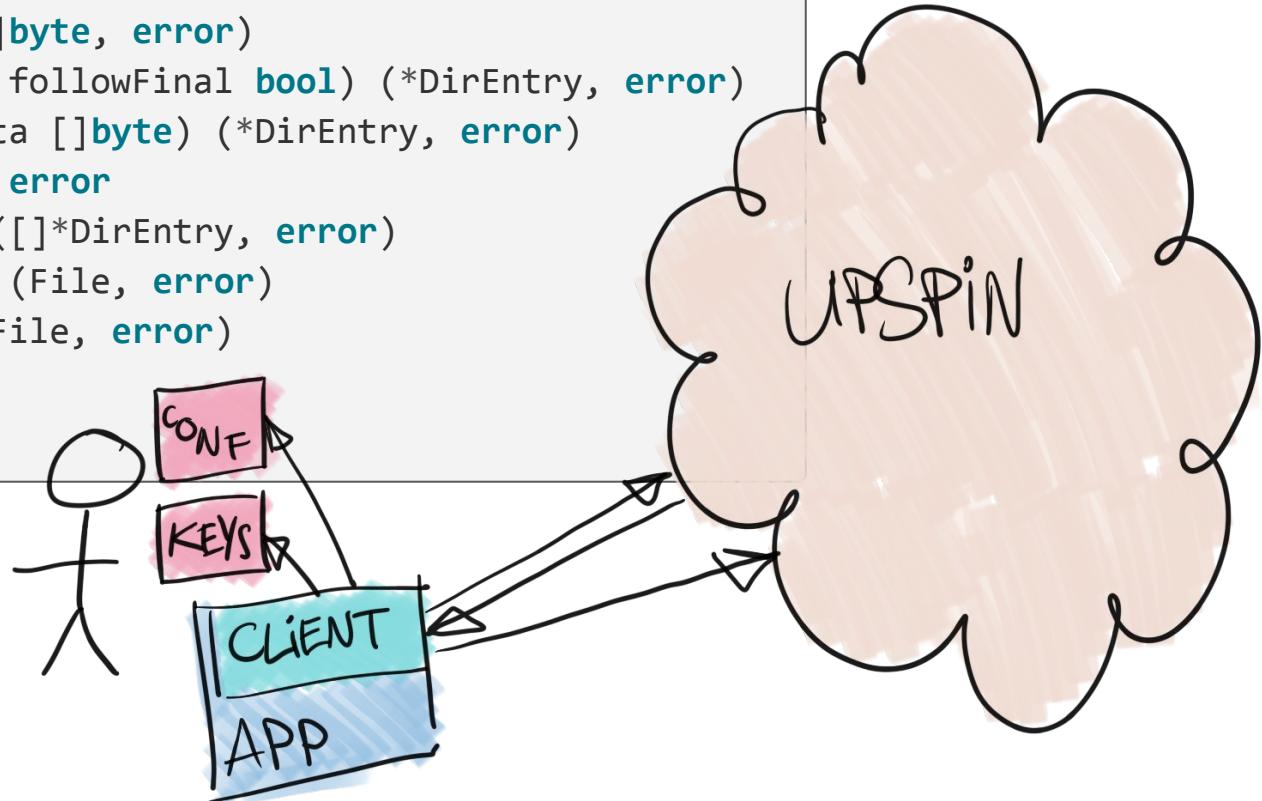


# Plan

- Upspin
  - Why Upspin?
  - Overview
- Upspin in practice as a user
  - Signing up and deploying your servers
  - Tools
- Upspin in practice as a developer
  - Developing a client
  - Also possible: server and storage
- Conclusion

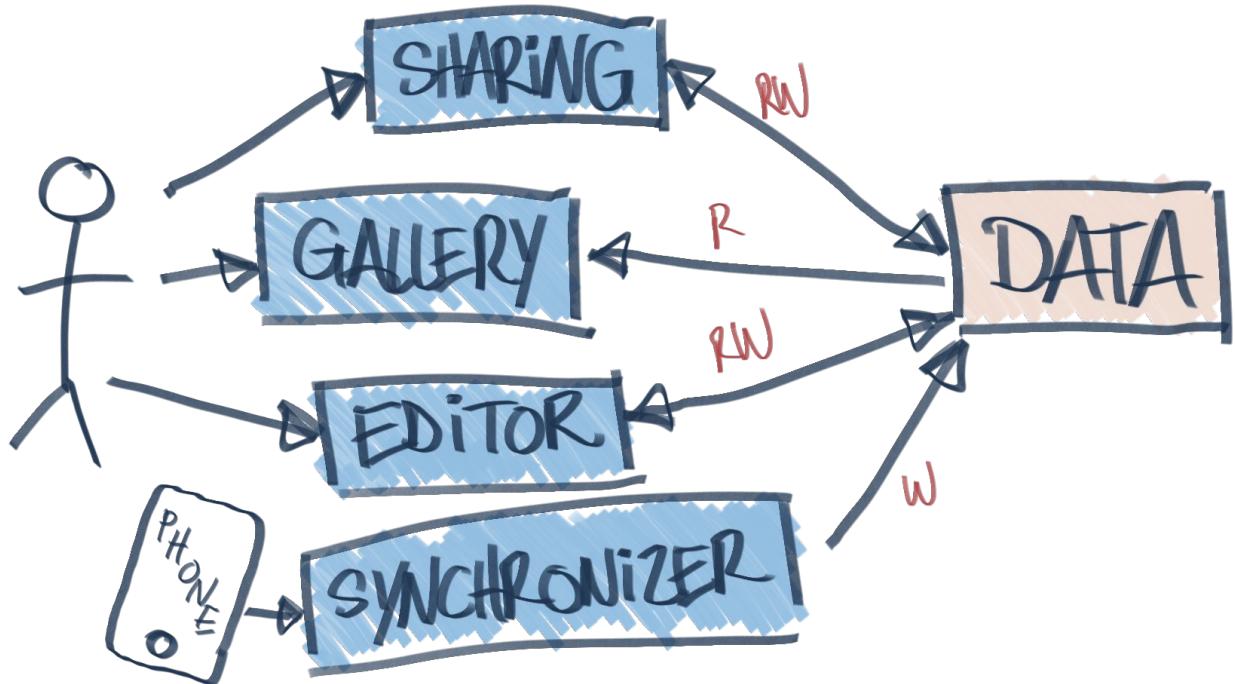
# A client app

```
type Client interface {  
    Get(name PathName) ([]byte, error)  
    Lookup(name PathName, followFinal bool) (*DirEntry, error)  
    Put(name PathName, data []byte) (*DirEntry, error)  
    Delete(name PathName) error  
    Glob(pattern string) ([]*DirEntry, error)  
    Create(name PathName) (File, error)  
    Open(name PathName) (File, error)  
    // ... and more  
}
```



# Example: photos

- Synchronization on your phone: write-only
- Photo gallery: read-only
- Sharing organizer: read & write (copy to shared folder)
- Photo editor: read & write
- ...



# Building a client app

```
func main() {
    config := ...
    client := ...
    http.HandleFunc("/", {
        f := client.Open()
        io.Copy(w, f)
    })
    ListenAndServe
}
```

# Initialize your Upspin client

```
func main() {
    config := ...
    client := ...
    http.HandleFunc("/", func(w http.ResponseWriter, r *http.Request) {
        f, err := client.Open(r.URL.Path)
        if err != nil {
            http.Error(w, err.Error(), http.StatusInternalServerError)
            return
        }
        io.Copy(w, f)
    })
    ListenAndServe()
}
```

```
// upspin.io/config
cfg, err := config.FromFile("config")
// check error

client := client.New(cfg) // upspin.io/client
if client != nil {
    fmt.Println("client initialized!")
```

# Read and serve the Upspin file

```
func main() {
    Config := ...
    client := ...
    http.HandleFunc("/", ...
        f := client.Open()
        io.Copy(w, f)
    )
    ListenAndServe
}
```

```
// e.g. GET /augie@upspin.io/Images/camstream.mp4
http.HandleFunc("/", func(w http.ResponseWriter, r
    *http.Request) {
    path := strings.TrimPrefix(r.URL.Path, "/")
    f, err := client.Open(upspin.PathName(path))
    if err != nil {
        http.Error(w, err.Error(), http.StatusNotFound)
        return
    }
    _, err = io.Copy(w, f)
    if err != nil {
        http.Error(w, err.Error(),
            http.StatusInternalServerError)
        return
    }
})
```

# ListenAndServe as you usually do

```
func main() {
    config := ...
    client := ...

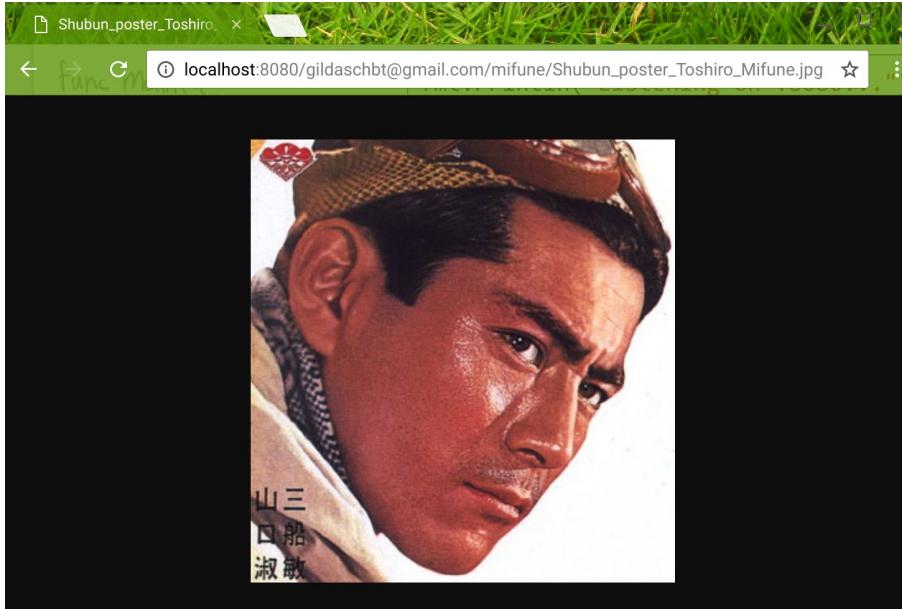
    http.HandleFunc("/", func(w http.ResponseWriter, r *http.Request) {
        f, err := client.Open(r.URL.String())
        if err != nil {
            http.Error(w, err.Error(), http.StatusInternalServerError)
            return
        }
        io.Copy(w, f)
    })
}

ListenAndServe
```

```
fmt.Println("Listening on :8080...")
http.ListenAndServe(":8080", nil)
```

# Example Upspin names

- gildaschbt@gmail.com/mifune/Shubun\_poster\_Toshiro\_Mifune.jpg
- augie@upspin.io/Images/camstream.mp4
- upspin-fosdem@yopmail.com/music/audiobinger/Audiobinger\_-\_Skid\_Row\_E\_P\_-\_20150623163633028.jpg

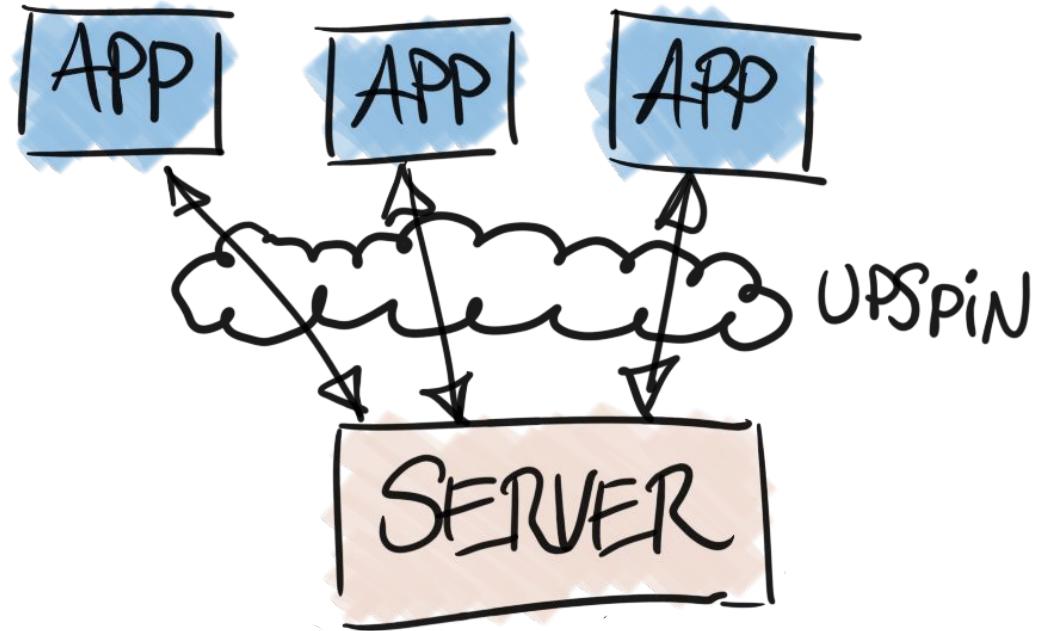


# Plan

- Upspin
  - Why Upspin?
  - Overview
- Upspin in practice as a user
  - Signing up and deploying your servers
  - Tools
- Upspin in practice as a developer
  - Developing a client
  - Also possible: server and storage
- Conclusion

# A server app

- Exposes Dir and Store interfaces to the clients
- Can check or not the access control
- Data source of your choice



# Upspin server vs. storage

```
// Storage is a low-level storage interface for services to store their data  
// permanently. Storage implementations must be safe for concurrent use.
```

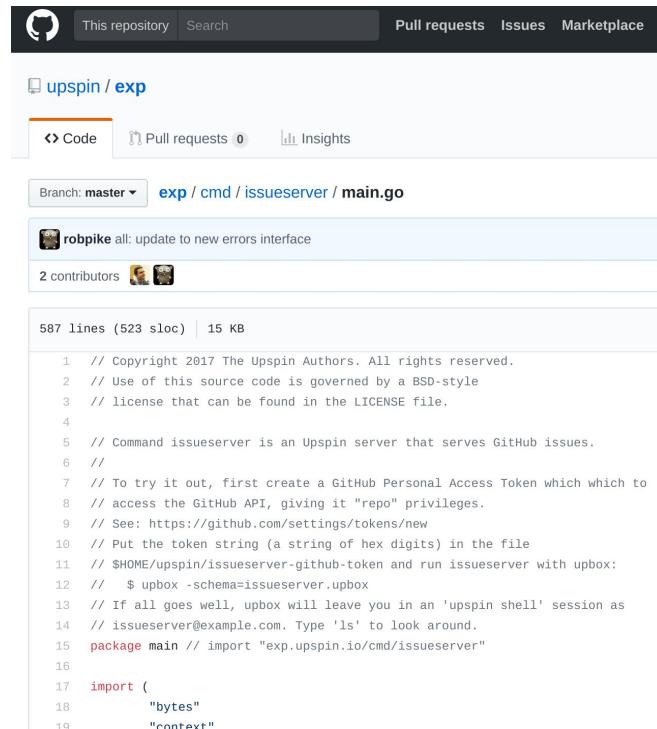
```
type Storage interface {  
    LinkBase() (base string, err error)  
    Download(ref string) ([]byte, error)  
    Put(ref string, contents []byte) error  
    Delete(ref string) error  
}
```

Disk, AWS, Google Drive,  
Google Cloud Platform,  
Dropbox, Openstack,  
Backblaze B2 Cloud Storage



# See examples at...

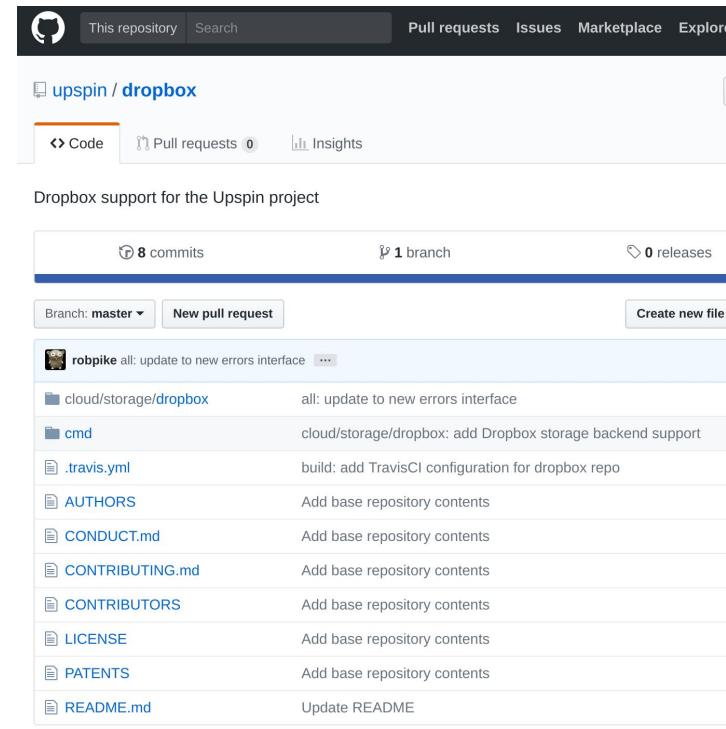
[exp.upspin.io/cmd/issueserver](https://exp.upspin.io/cmd/issueserver)



This screenshot shows the GitHub repository page for 'exp'. The main navigation bar includes 'This repository', 'Search', 'Pull requests', 'Issues', 'Marketplace', and 'Explore'. Below the navigation, there are tabs for 'Code', 'Pull requests (0)', and 'Insights'. A dropdown menu indicates the branch is 'master'. The current view is on the 'main.go' file under the 'cmd/issueserver' directory. The commit history shows a single commit by 'robpike' with the message 'all: update to new errors interface'. The file has 587 lines (523 sloc) and is 15 KB in size. The code content is as follows:

```
1 // Copyright 2017 The Upspin Authors. All rights reserved.
2 // Use of this source code is governed by a BSD-style
3 // license that can be found in the LICENSE file.
4 //
5 // Command issueserver is an Upspin server that serves GitHub issues.
6 //
7 // To try it out, first create a GitHub Personal Access Token which which to
8 // access the GitHub API, giving it "repo" privileges.
9 // See: https://github.com/settings/tokens/new
10 // Put the token string (a string of hex digits) in the file
11 // $HOME/upspin/issueserver-github-token and run issueserver with upbox:
12 //   $ upbox -schema=issueserver.upbox
13 // If all goes well, upbox will leave you in an 'upspin shell' session as
14 // issueserver@example.com. Type 'ls' to look around.
15 package main // import "exp.upspin.io/cmd/issueserver"
16
17 import (
18     "bytes"
19     "context"
20     "encoding/json"
21     "fmt"
22     "log"
23     "net/http"
24     "os"
25     "path/filepath"
26     "sync"
27     "time"
28 )
29
30 type IssuesServer struct {
31     ...
32 }
33
34 func NewIssuesServer(upbox *upbox.Upbox, token string) (*IssuesServer, error) {
35     ...
36 }
37
38 func (is *IssuesServer) ServeHTTP(w http.ResponseWriter, r *http.Request) {
39     ...
40 }
41
42 func (is *IssuesServer) Start() error {
43     ...
44 }
45
46 func (is *IssuesServer) Stop() error {
47     ...
48 }
```

[dropbox.upspin.io/cloud/storage/dropbox](https://dropbox.upspin.io/cloud/storage/dropbox)



This screenshot shows the GitHub repository page for 'dropbox'. The main navigation bar includes 'This repository', 'Search', 'Pull requests', 'Issues', 'Marketplace', and 'Explore'. Below the navigation, there are tabs for 'Code', 'Pull requests (0)', and 'Insights'. A dropdown menu indicates the branch is 'master'. The repository summary shows 8 commits, 1 branch, and 0 releases. The commit history shows a single commit by 'robpike' with the message 'all: update to new errors interface'. The repository contains several files and directories, each with a brief description:

- cloud/storage/dropbox: all: update to new errors interface
- cmd: cloud/storage/dropbox: add Dropbox storage backend support
- .travis.yml: build: add TravisCI configuration for dropbox repo
- AUTHORS: Add base repository contents
- CONDUCT.md: Add base repository contents
- CONTRIBUTING.md: Add base repository contents
- CONTRIBUTORS: Add base repository contents
- LICENSE: Add base repository contents
- PATENTS: Add base repository contents
- README.md: Update README

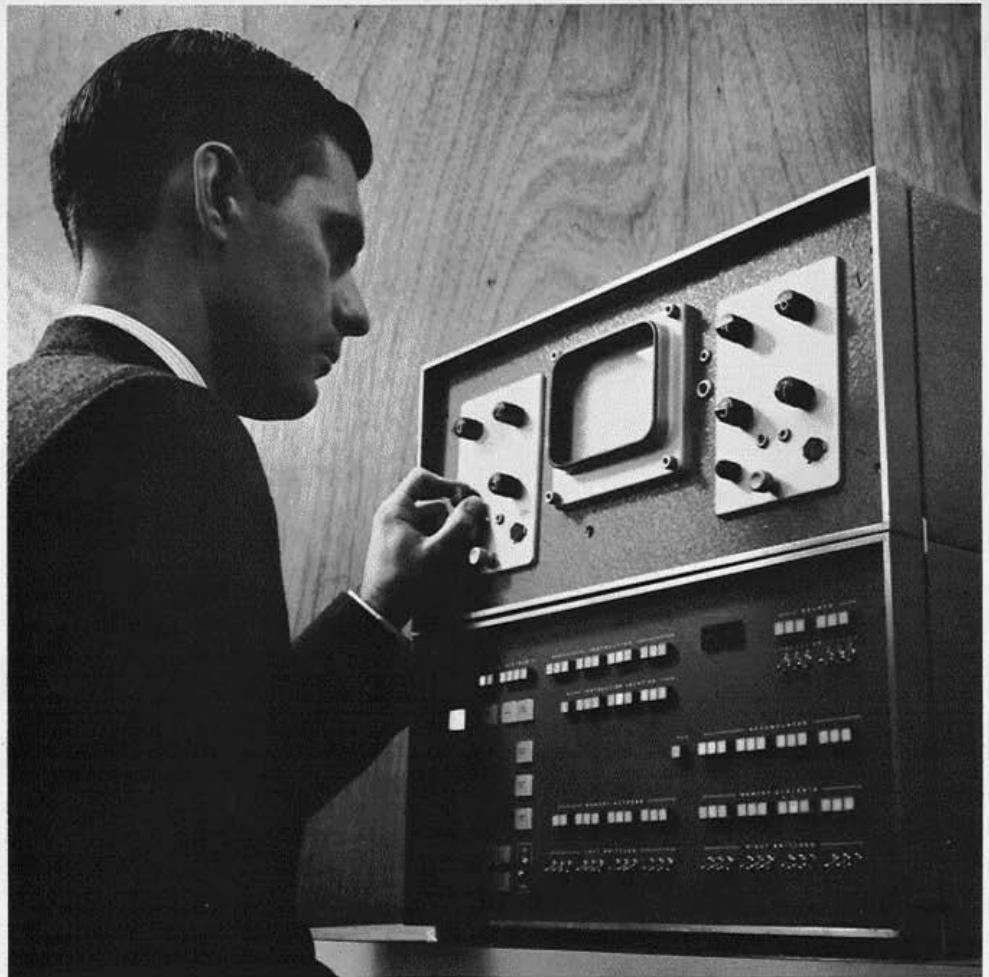
# Plan

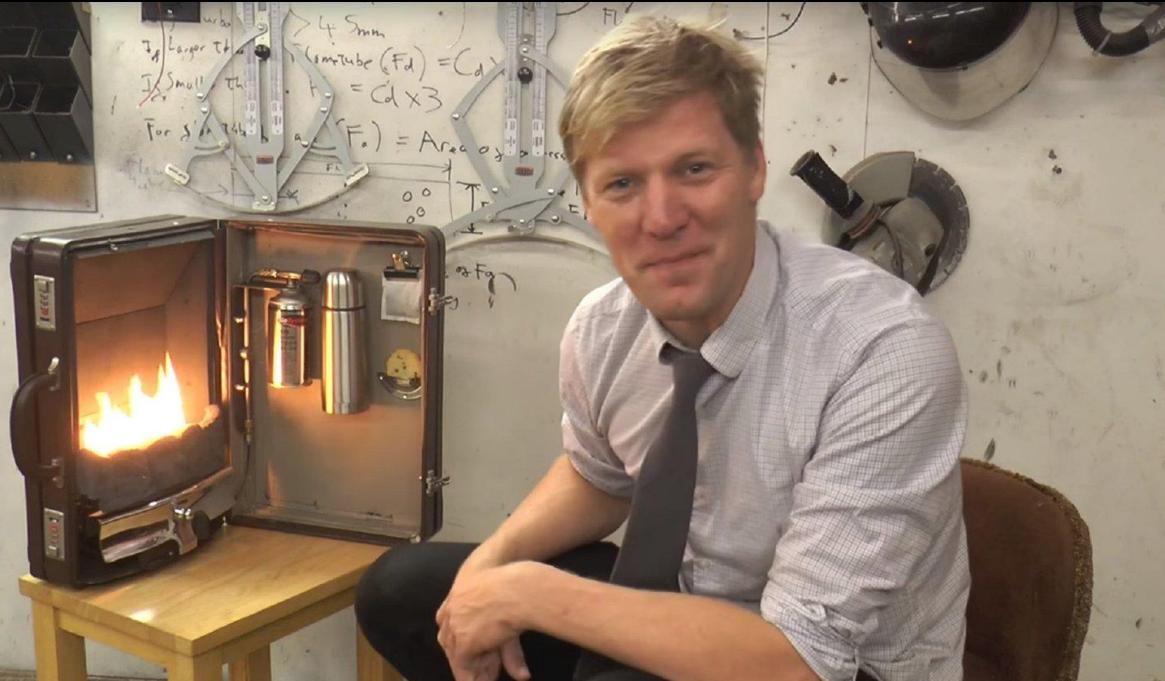
- Upspin
  - Why Upspin?
  - Overview
- Upspin in practice as a user
  - Signing up and deploying your servers
  - Tools
- Upspin in practice as a developer
  - Developing a client
  - Also possible: server and storage
- Conclusion

# Privacy and control



*More complexity*  
for the wide  
audience



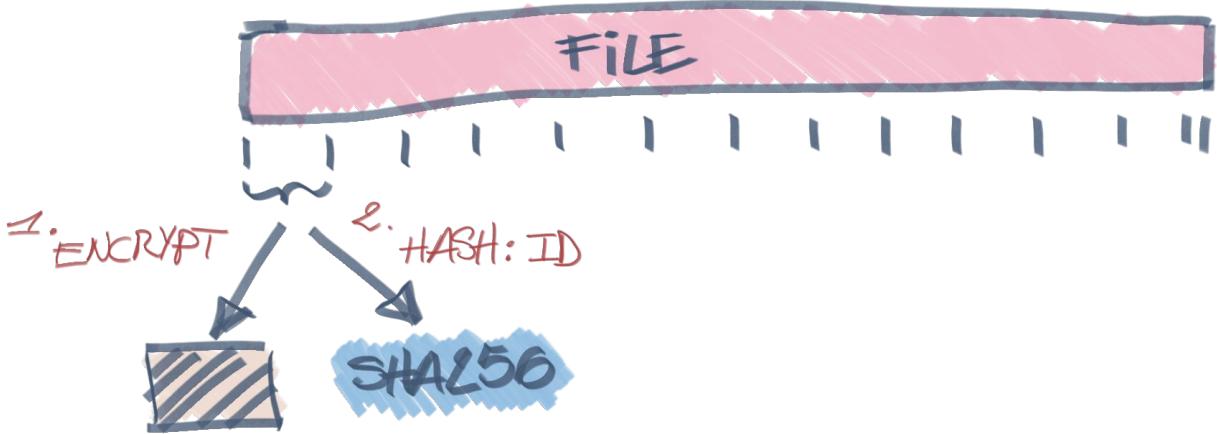


Great UX for  
everybody via  
collaboration

# Thank you

- Gildas Chabot ([github.com/gildasch](https://github.com/gildasch)), Leboncoin, Paris
- Augie's image is Copyright 2017 by [Renee French](#), All Rights Reserved.

# Annex: packing



- The file is cut in blocks
  - 1MB by default
- The block is encrypted
- The encrypted block is hashed to create the *ref*

# Annex: Comparaisons with kbfs and ipfs (WIP)

	kbfs	ipfs	upspin
End-to-end encryption	✓	✗	✓
Storage solution	✓	✓	✗
Private sharing	✓	✗	✓
“Universal” protocol	✗	✓ (http gateway)	✓
Naming system	✓	✓ (via ipns)	✓
Data available for ever (no deletion)	✗	✓	✗
Cache & offline access	✗	✓	~

# Jan 3, 2016 – Jan 18, 2018 Annex: Upspin history

Contributions to master, excluding merge commits

