# Plan

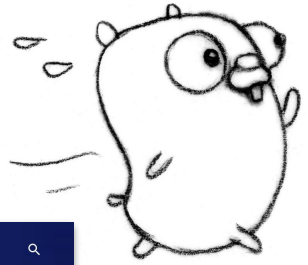
- Upspin
  - Why Upspin?
  - Overview
- Upspin in practice as a user
  - Signing up and deploying your servers
  - Tools
- Upspin in practice as a developer
  - Developing a client
  - Also possible: server and storage
- Conclusion

# 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

Feb. 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.

command center

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 Renee French

Oct. 2017

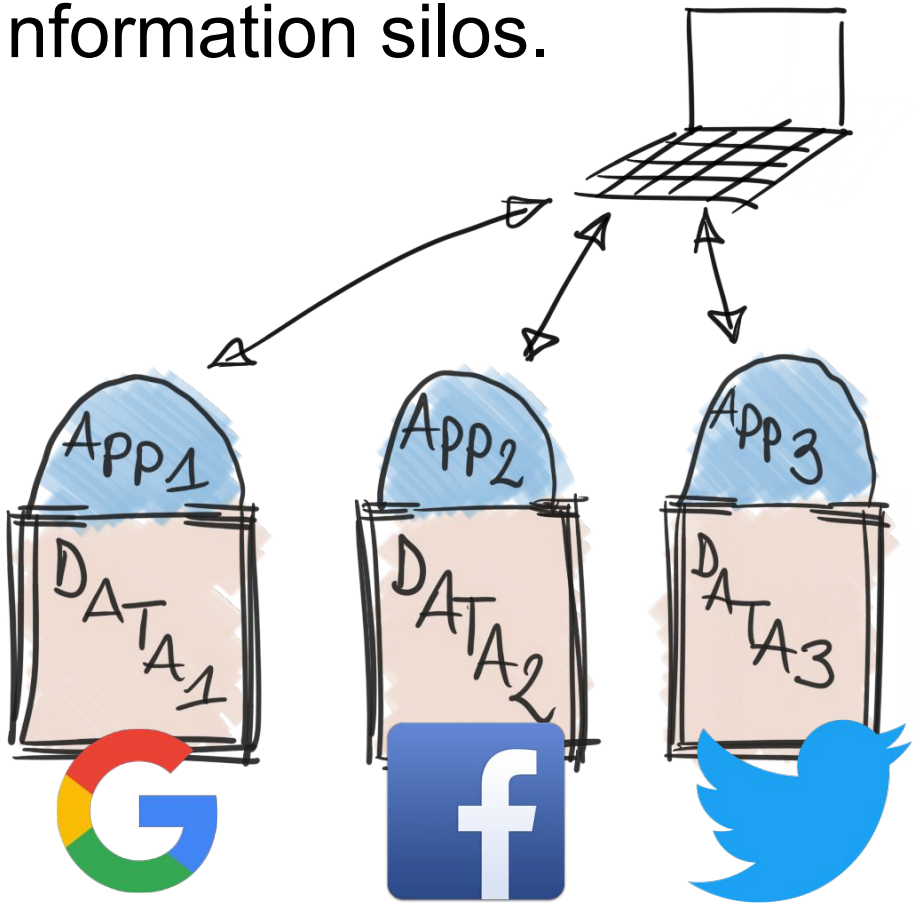
**Manifesto**

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.”

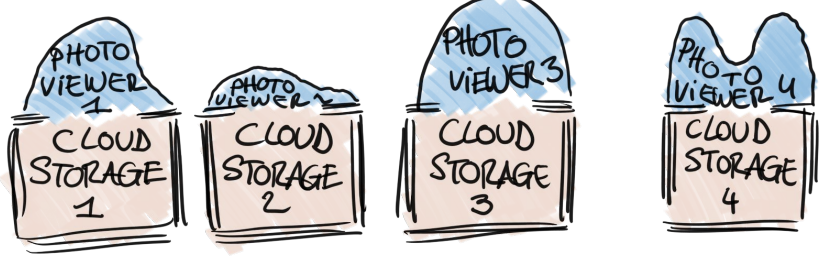
— Upspin manifesto



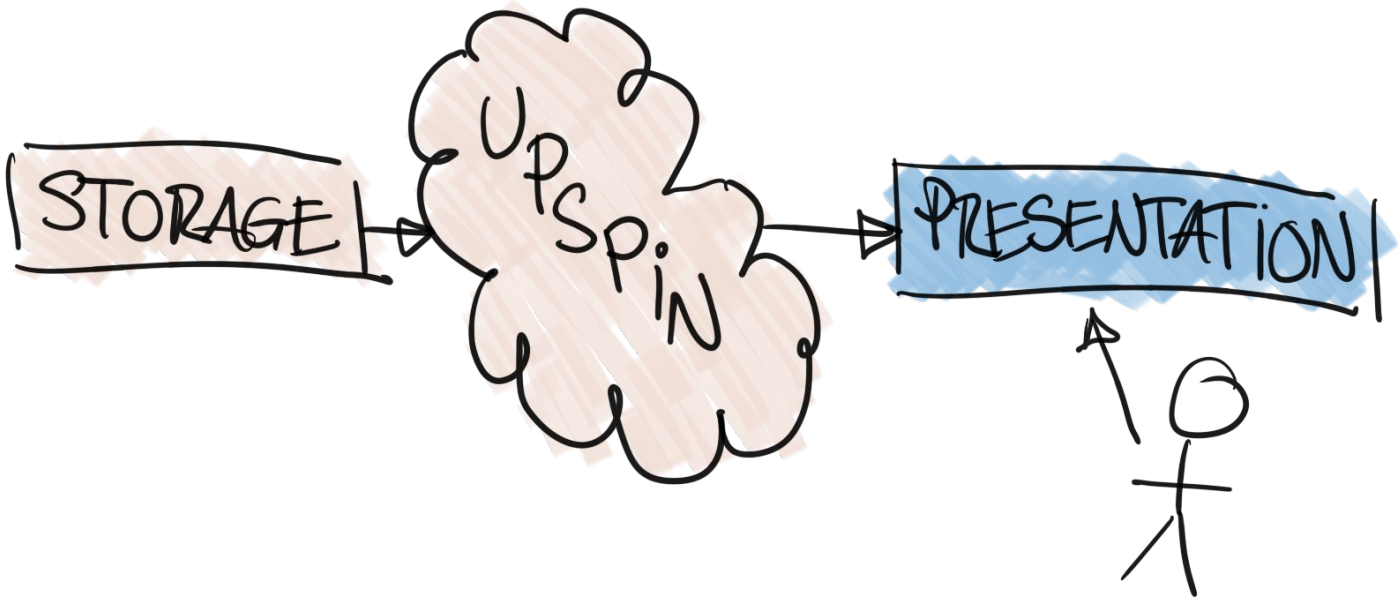
# “You don't own your data any more”

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

# Mediocre apps too!

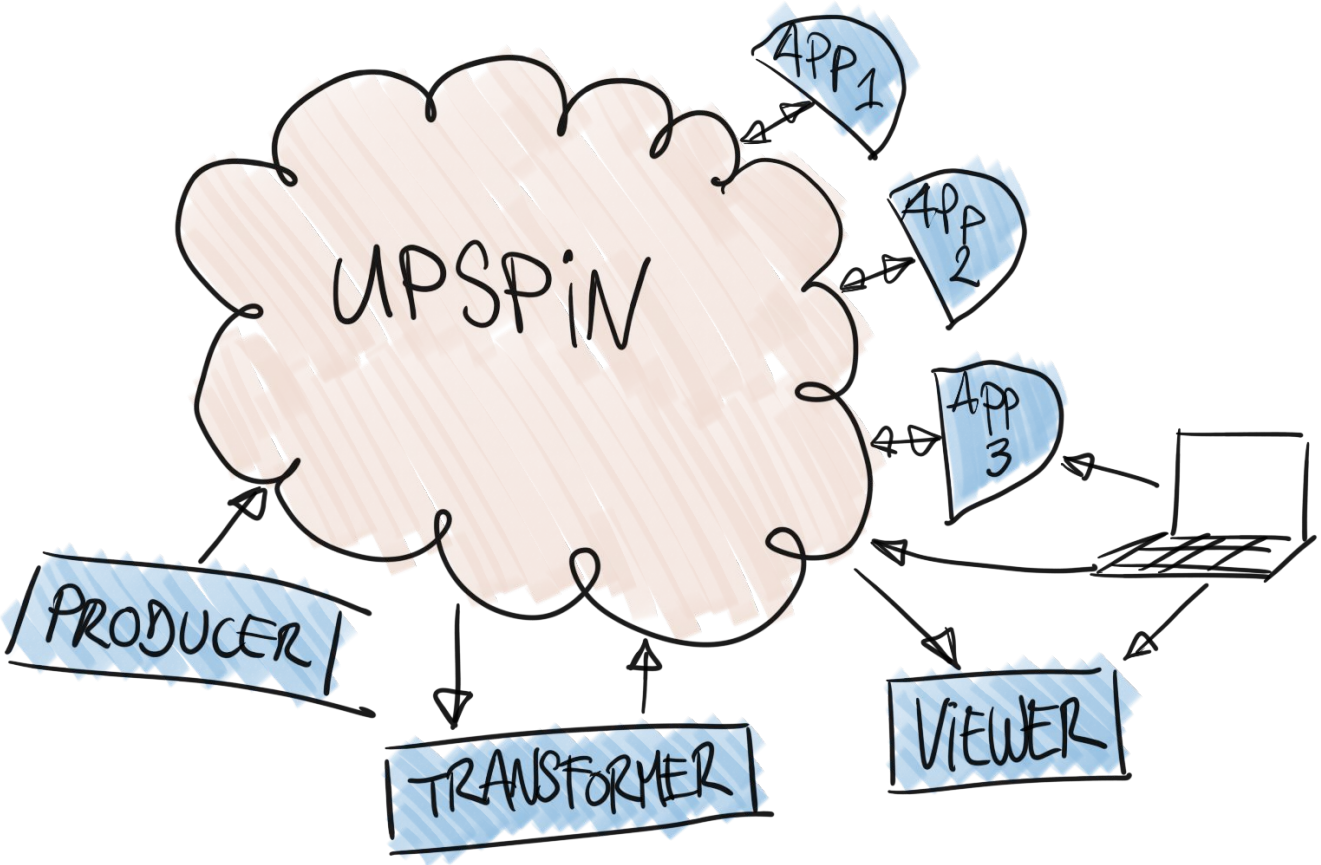



- Developping Google Photos takes a lot of time
- Very few cloud storage providers can compete
- Let's keep *storage* and *presentation* 2 separate businesses



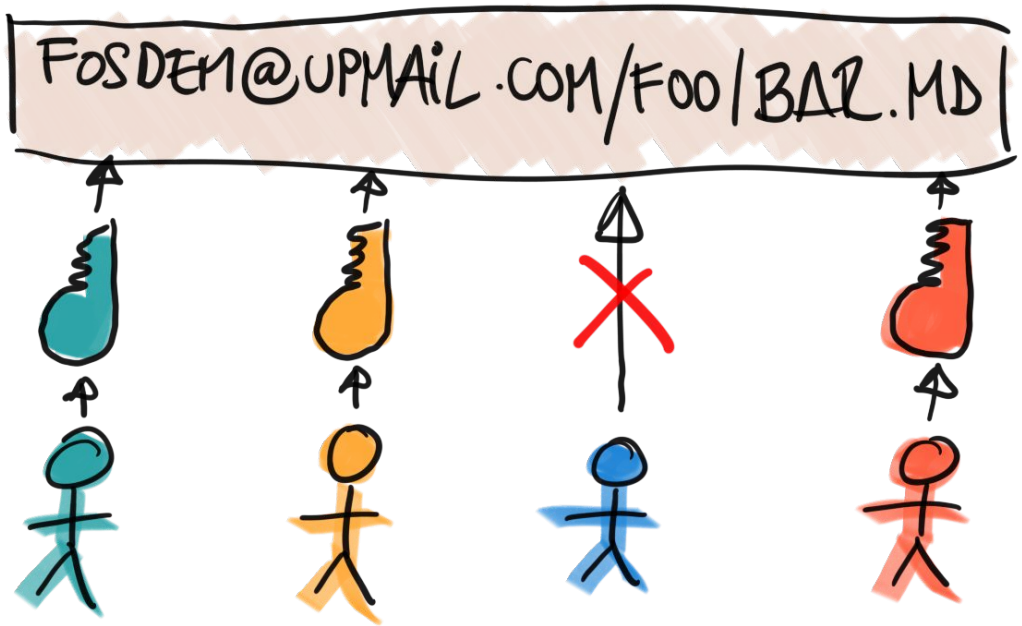
# A *universal* data source

- 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



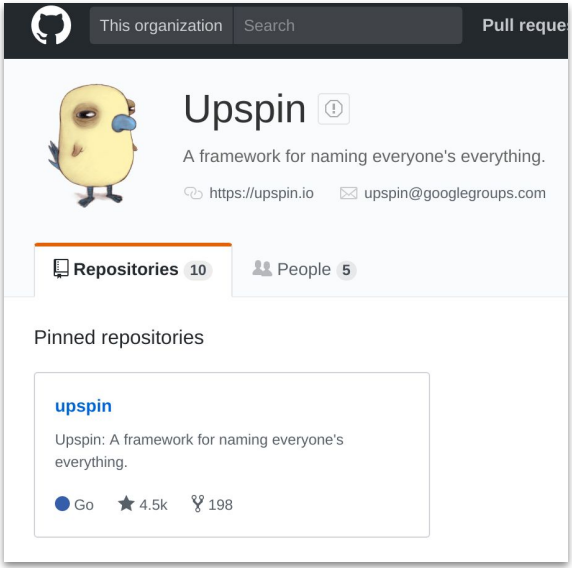
# Plan

- 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

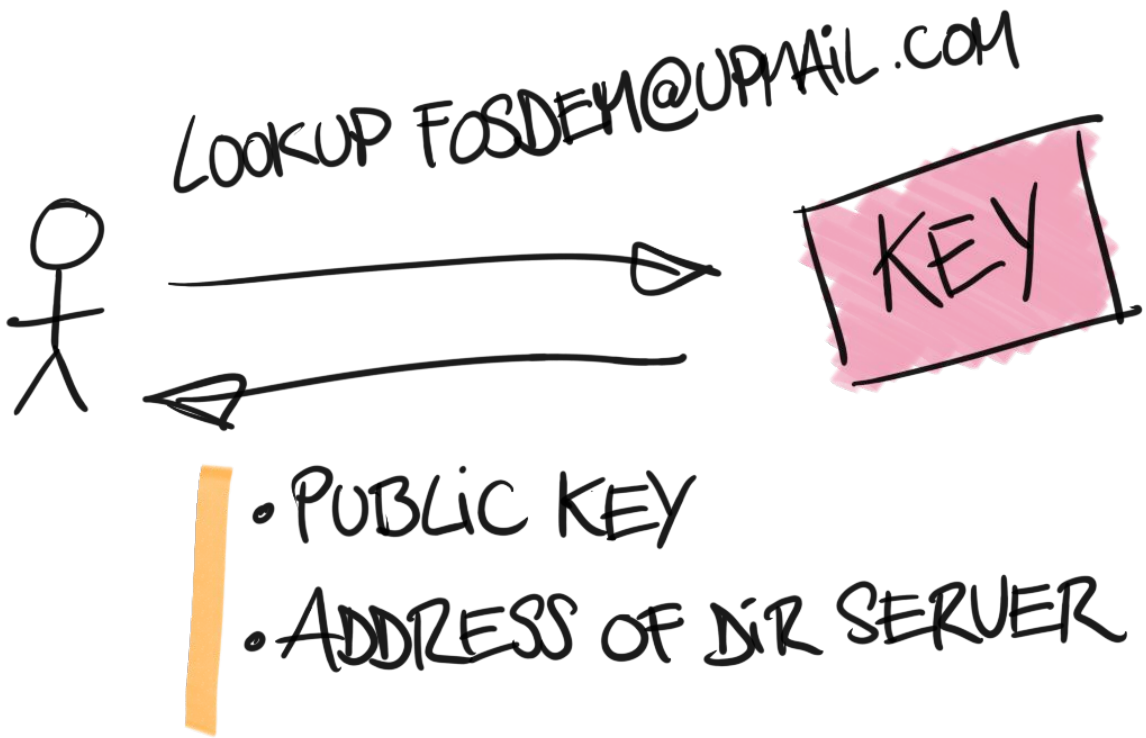
```
// The StoreServer saves and retrieves data without interpretation.
type StoreServer interface {
     Get(ref Reference) ([]byte, *Refdata, []Location, error)
     Put(data []byte) (*Refdata, error)
     Delete(ref Reference) error
}
```

# DirServer

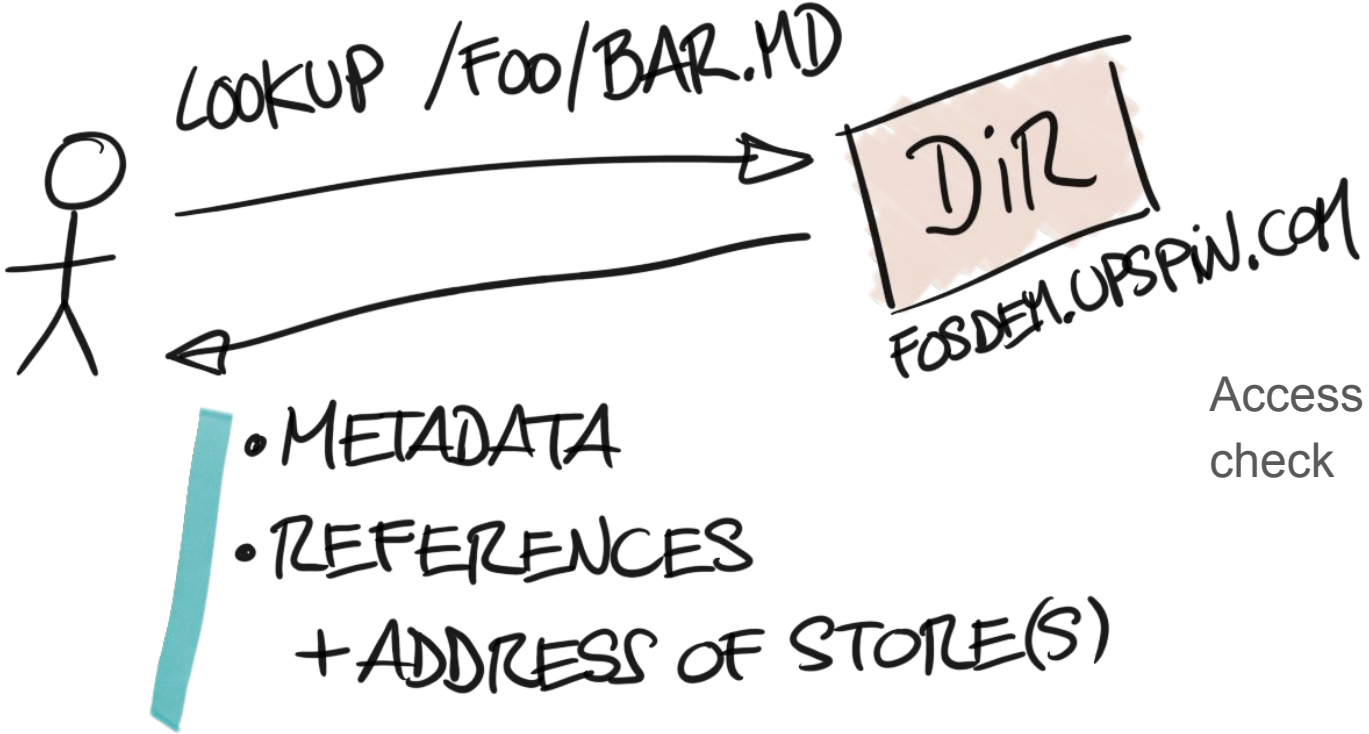
from upspin/upspin.go

```
// 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

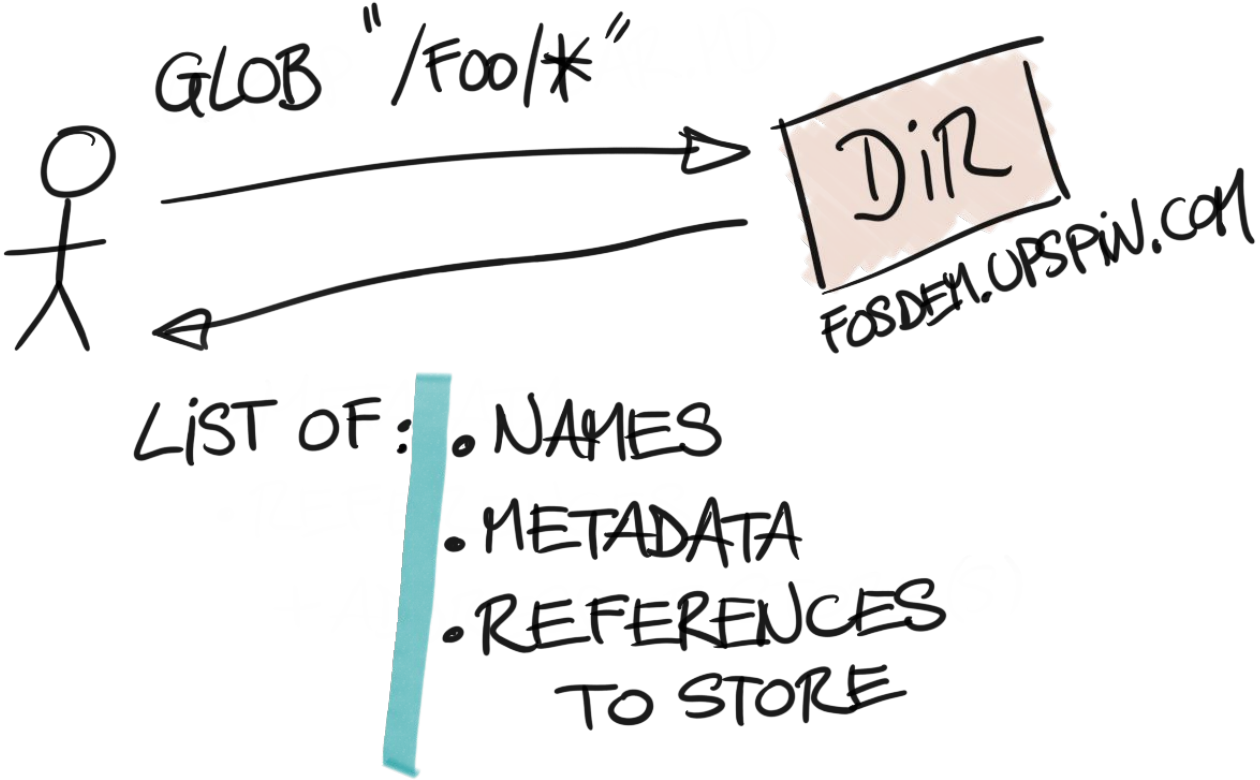



Access control check

# Getting a file: 3. Get file


The data is most likely encrypted

# Listing a directory: “glob”



# Controling 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

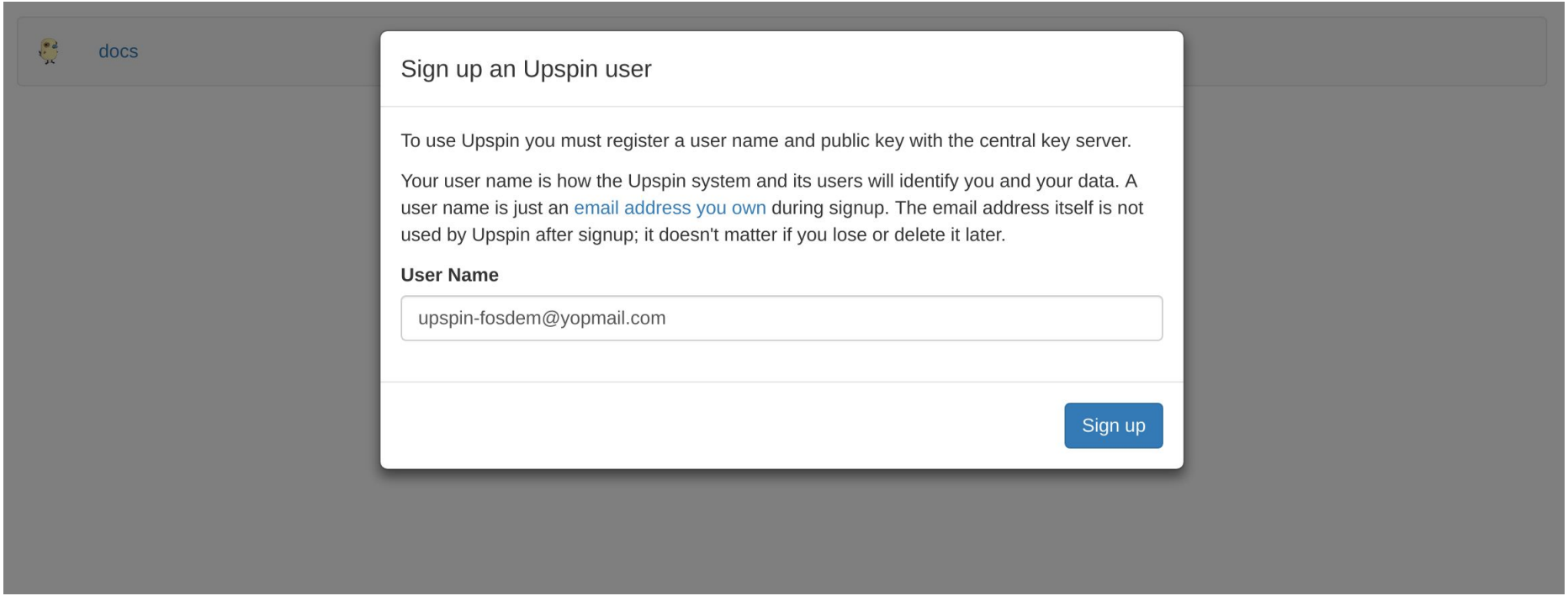
- Create a key pair
- Verify your e-mail
- Put a new user into the Key server — key.upspin.io

# Deploying the Dir and Store servers

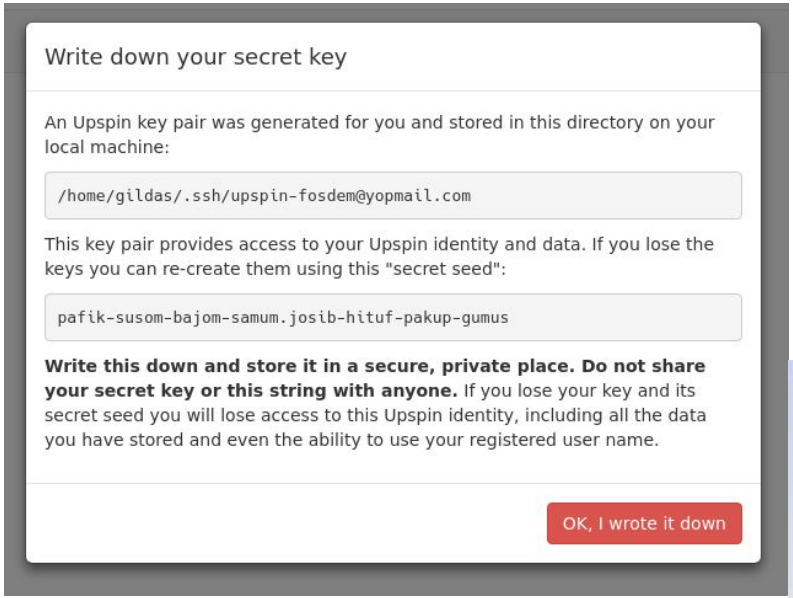
- Install the dir and store servers on a host
- Make them accessible from the Internet
  - With a valid HTTPS certificate
- Update the user in the Key server

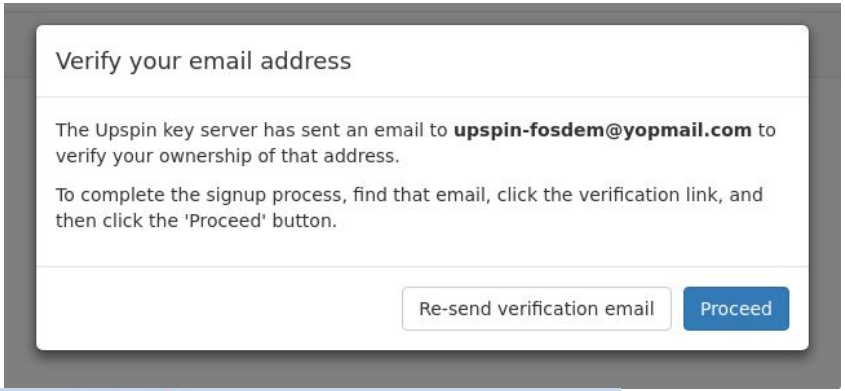
# Signup process with `upspin-ui`

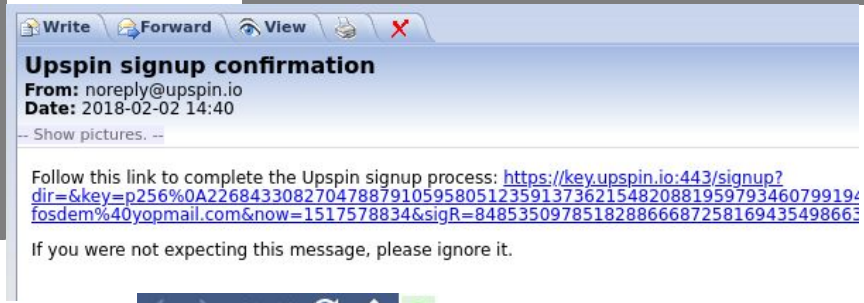
- Binaries for `upspin-ui` on upspin.io or install with `go get`

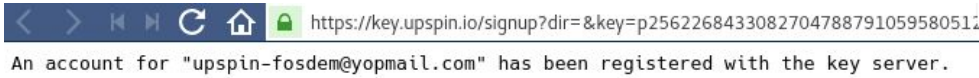


# Signup process with `upspin-ui`



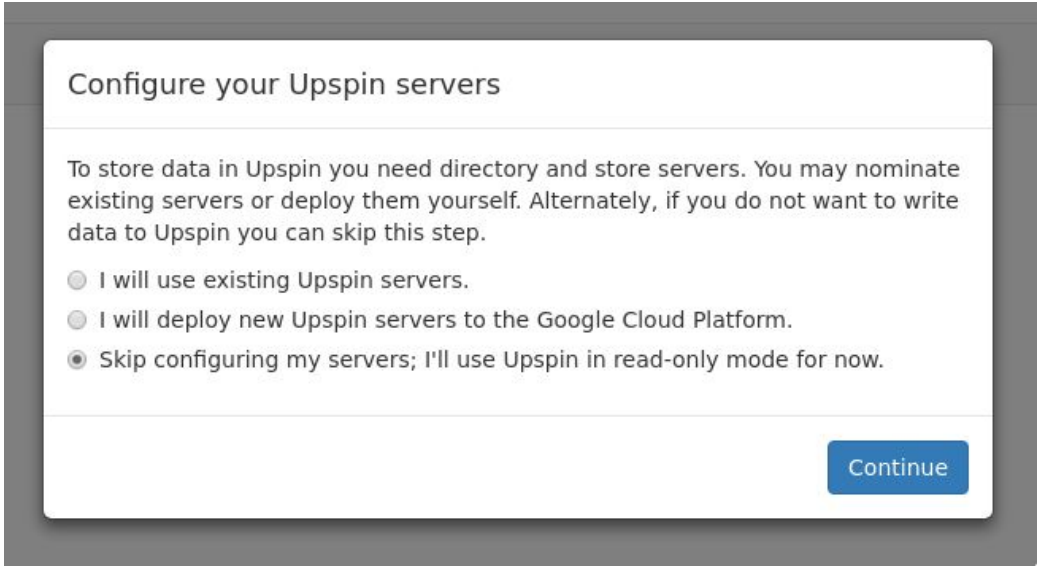



# Server deployment

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.upspin.services:443

storeserver: remote,8cda9311ce4bed564f1004cf4dd864b7.upspin.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
105958051235913736215482088195979346079919434878345387231 13\n244208627096763940125471651969979701966
7424537210468907354999335353170075715 9\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
105958051235913736215482088195979346079919434878345387231 13\n244208627096763940125471651969979701966
7424537210468907354999335353170075715 9\n"}
SHA256:741474f8e4e4f8c1036d4dcc3024806177ff88159a10c0ec983b22cab8666fbd
```

# 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.service
s:443"],"Stores":["remote,8cda9311ce4bed564f1004cf4dd864b7.upspin.services:443"],"PublicKey":"p256\
n22684330827047887910595805123591373621548208819597934607991943487834538723113\n244208627096763940125471651969979701966742453721046890735499933535317007571 59\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.service
s:443"],"Stores":["remote,8cda9311ce4bed564f1004cf4dd864b7.upspin.services:443"],"PublicKey":"p256\
n22684330827047887910595805123591373621548208819597934607991943487834538723113\n244208627096763940125471651969979701966742453721046890735499933535317007571 59\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`

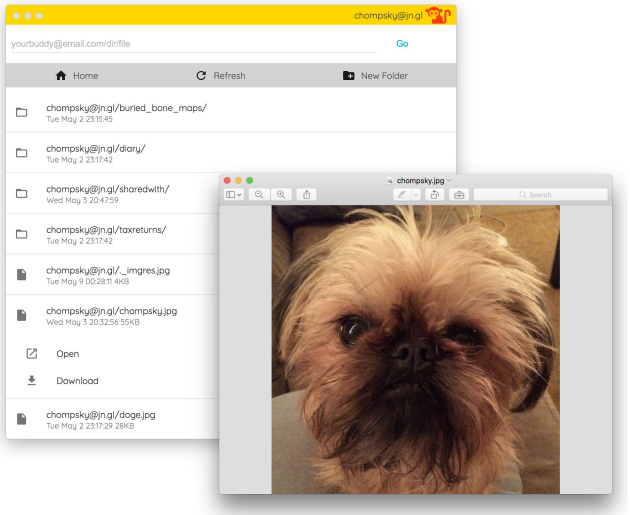
# Upspin and upspin-ui

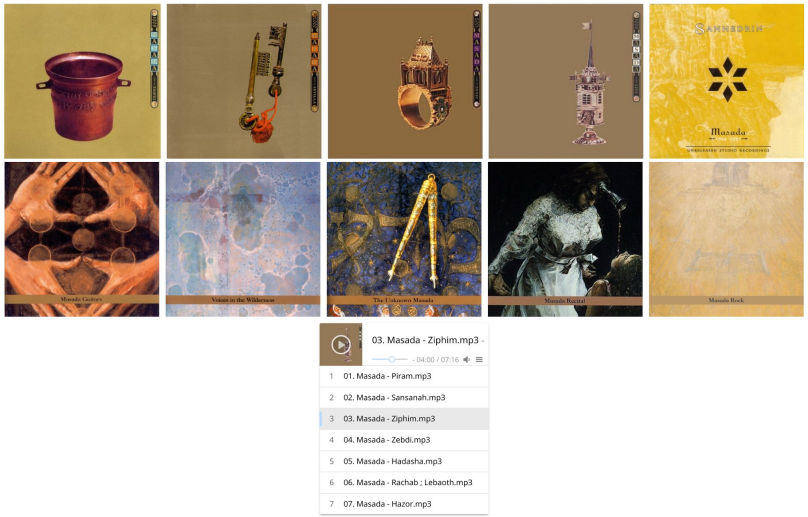
- Two other basic commands
- `go get upspin.io/cmd/upspin` and `go get augie.upspin.io/cmd/upspin-ui`
- upspin-ui is a web file explorer
- upspin give access to all the possible operations

```
$ upspin -help
Upspin commands:

shell (Interactive mode), config, countersign, cp, createsuffixeduser,
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



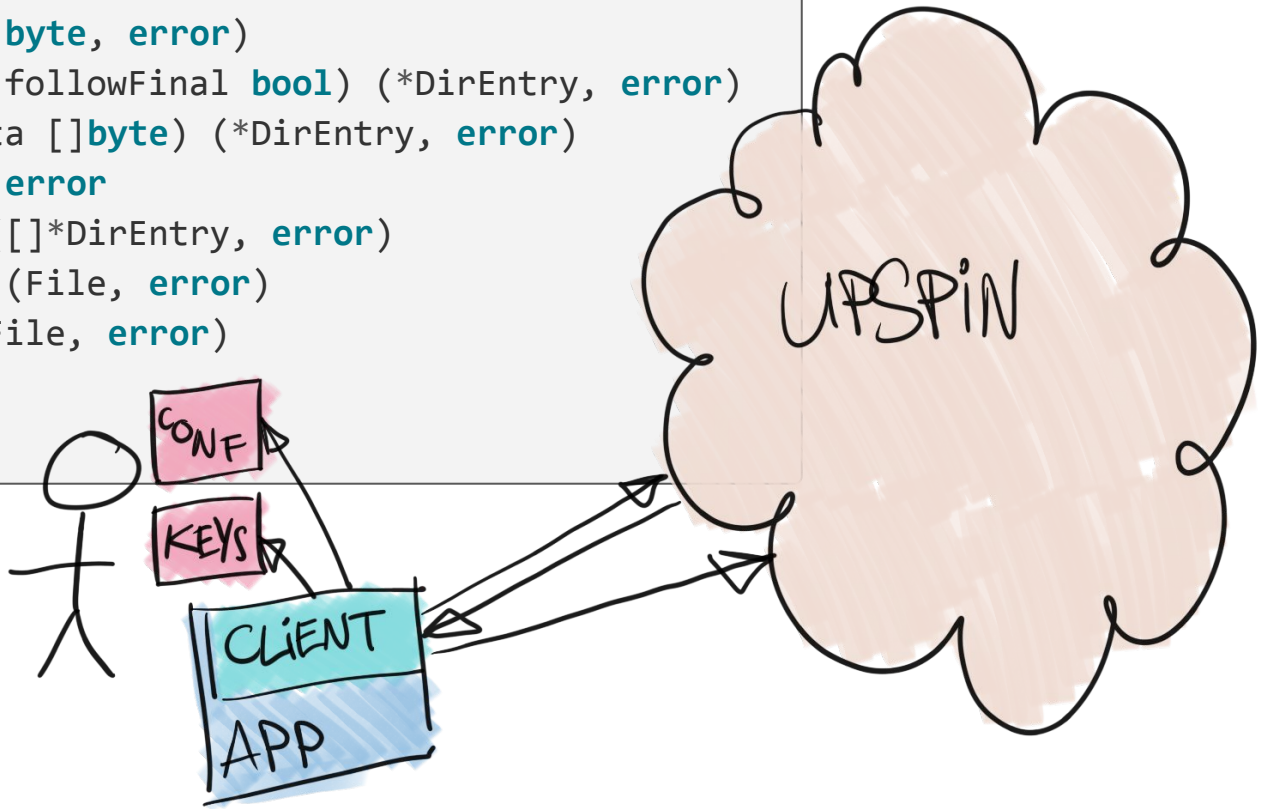



# Plan

- 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

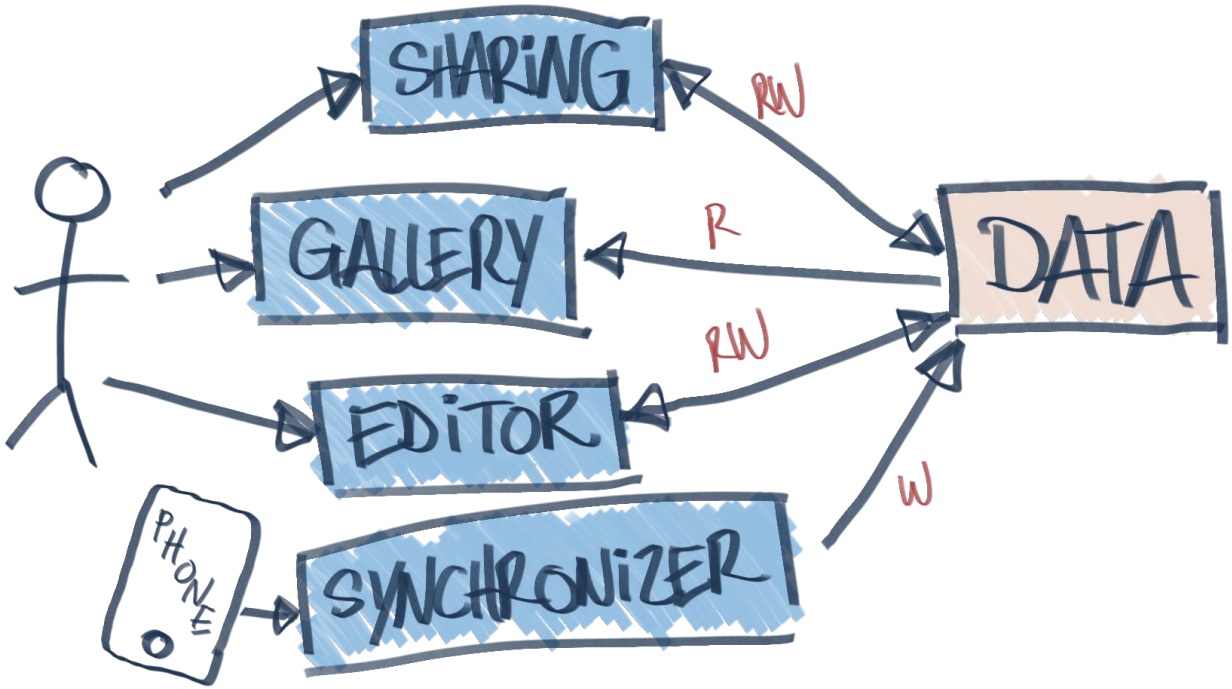
# A client app

```
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

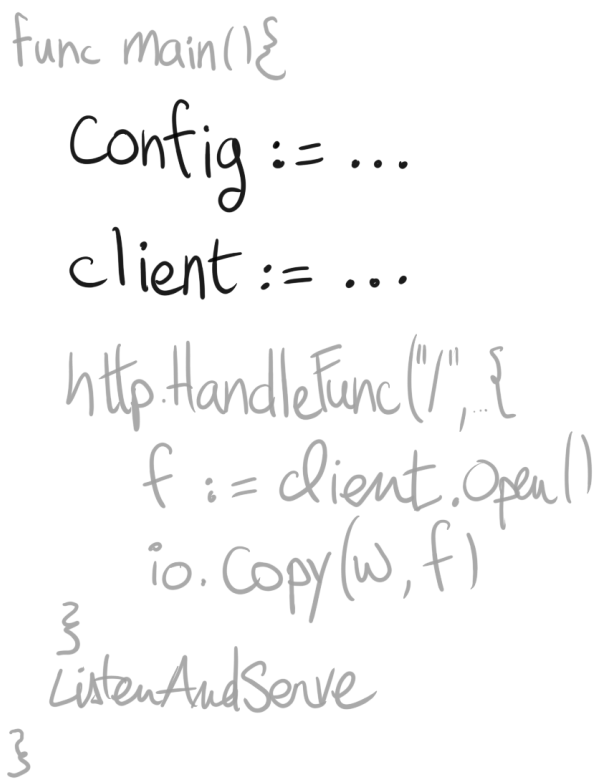
- Synchronization on your phone: write-only
- Photo gallery: read-only
- Sharing organizer: read & write (copy to shared folder)
- Photo editor: read & write
- ...



# Building a client app

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

# Initialize your Upspin client



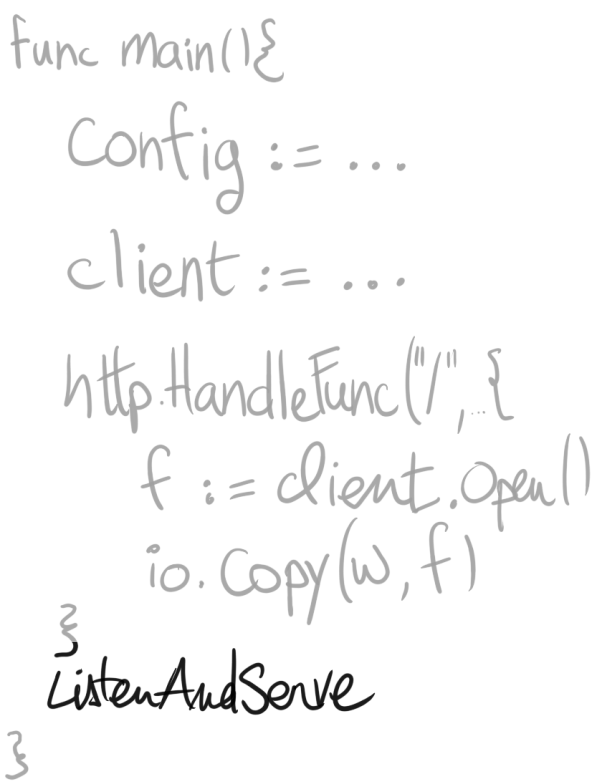

```go
// upspin.io/config
cfg, err := config.FromFile("config")
// check error

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
    }
})
```

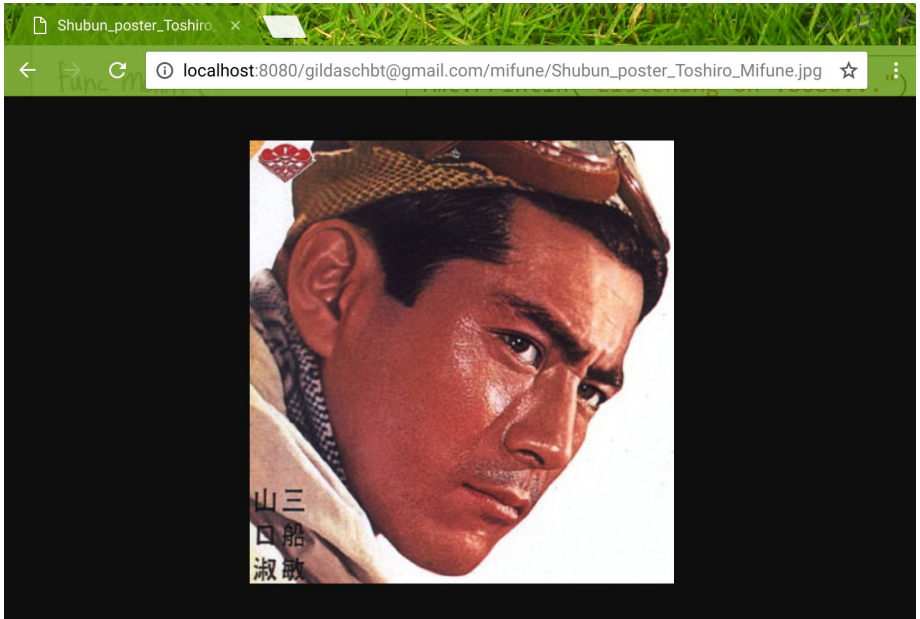
# ListenAndServe as you usually do




```
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_EP_-_20150623163633028.jpg

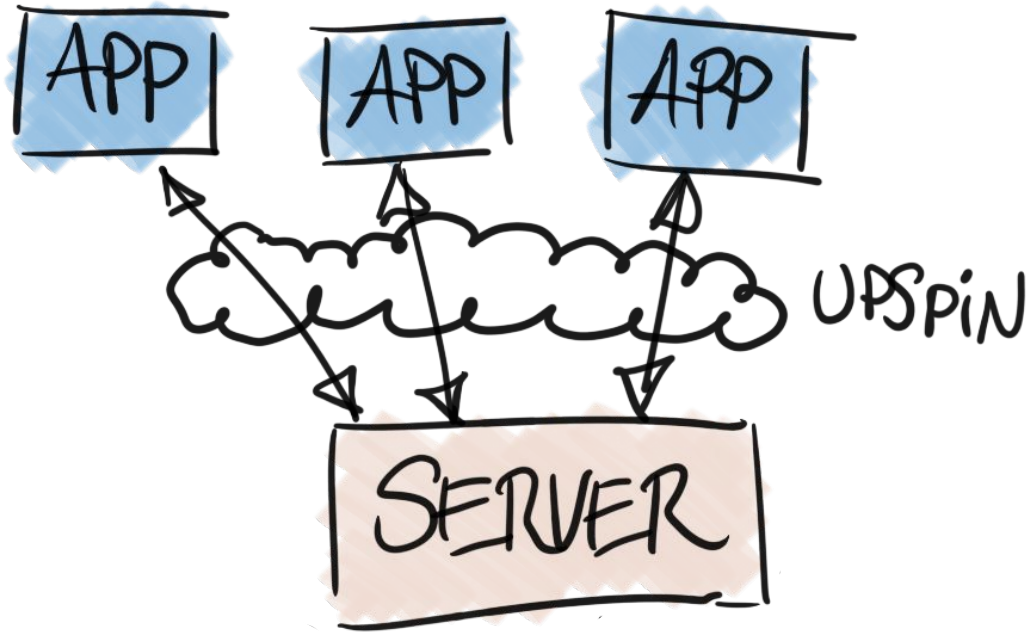


# Plan

- 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



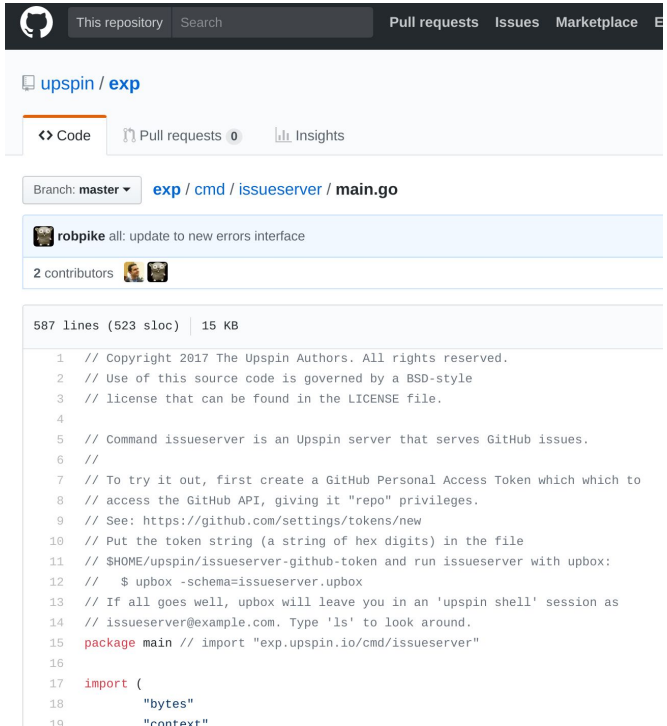
# Upspin server vs. storage

```
// 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
}
```

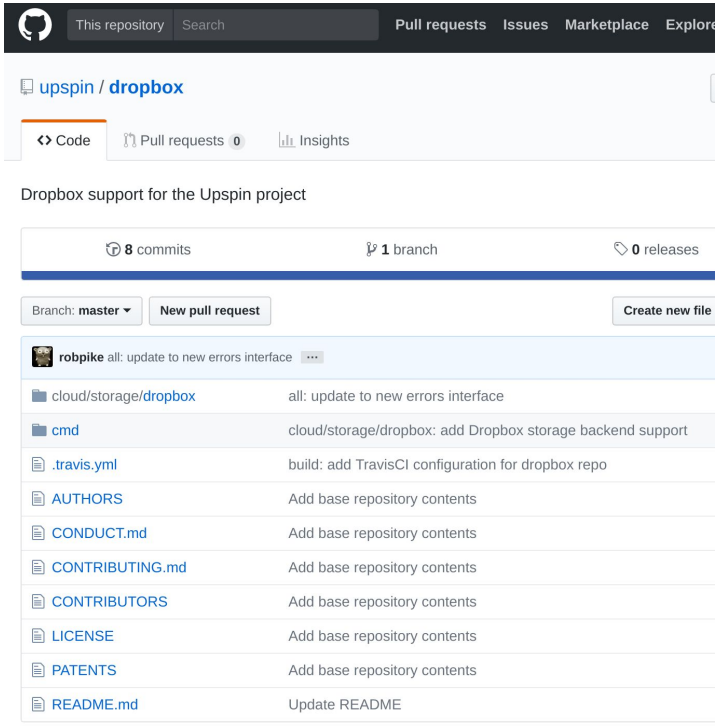
Disk, AWS, Google Drive,
Google Cloud Platform,
Dropbox, Openstack,
Backblaze B2 Cloud Storage

# See examples at...

`exp.upspin.io/cmd/issueserver`




`dropbox.upspin.io/cloud/storage/dropbox`

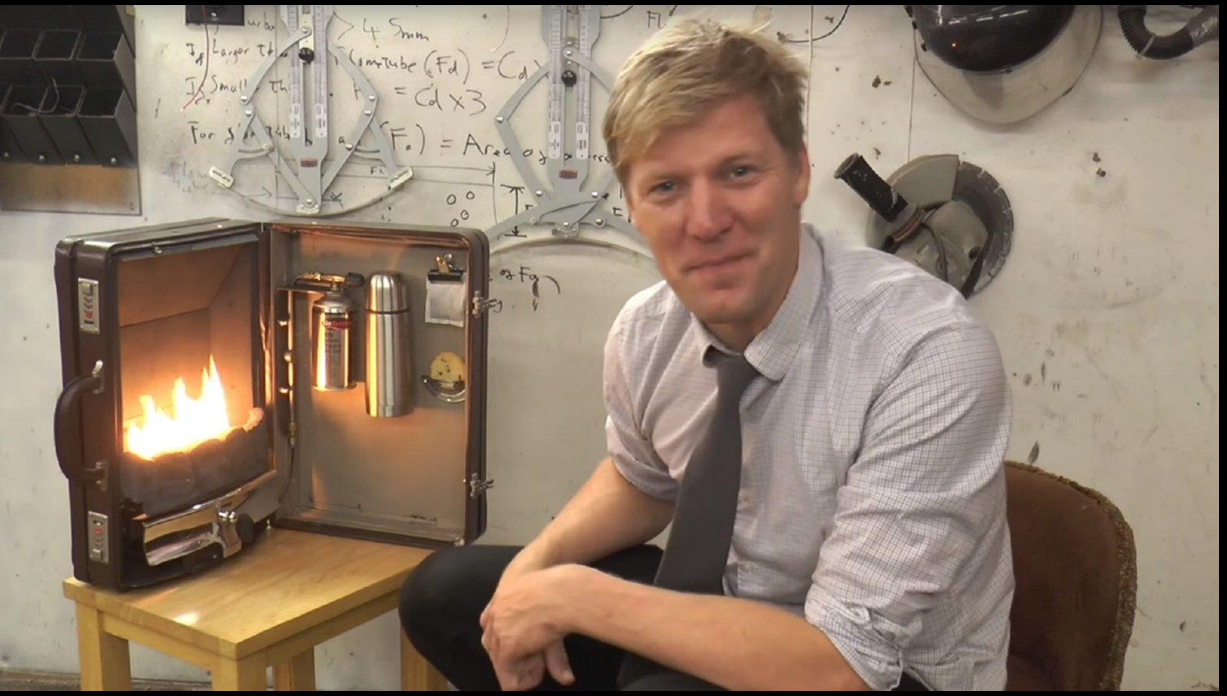


# Plan

- 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

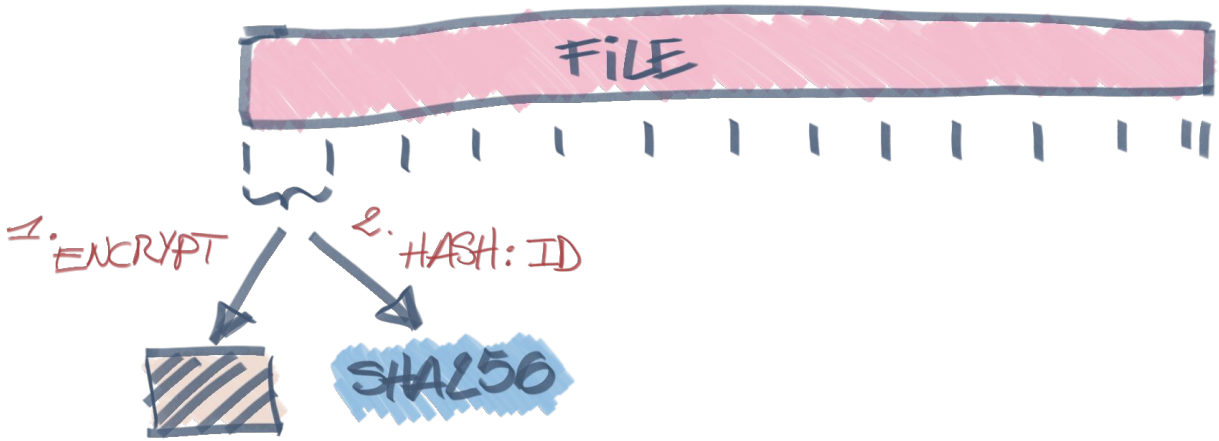


Great UX for everybody via collaboration

# Thank you

- Gildas Chabot (github.com/gildasch), Leboncoin, Paris

# 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)

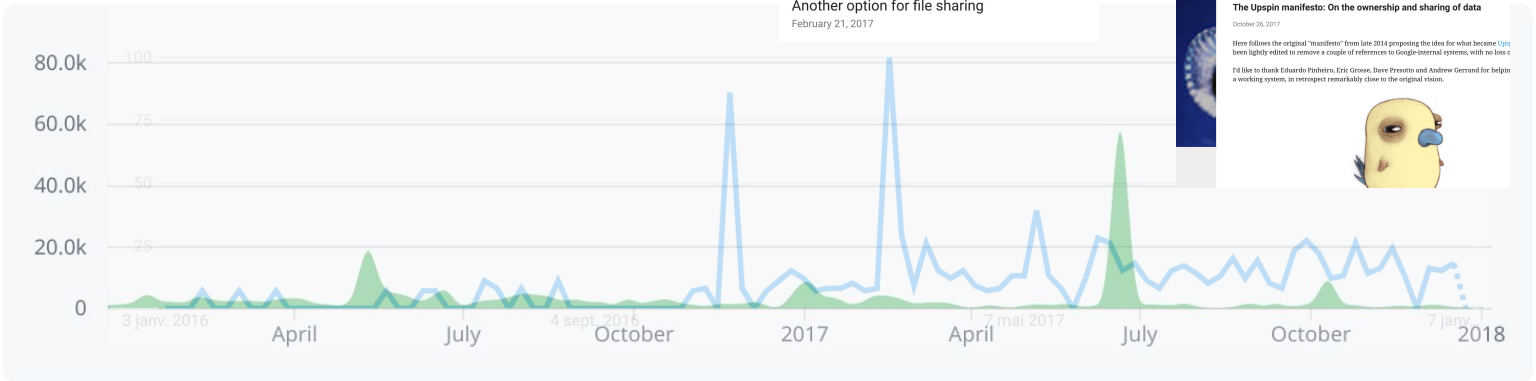
| | kbfs | ipfs | upspin |
|---|---|---|---|
| End-to-end encryption | ✔ | ✗ | ✔ |
| Storage solution | ✔ | ✔ | ✗ |
| Private sharing | ✔ | ✗ | ✔ |
| “Universal” protocol | ✗ | ✔ (http gateway) | ✔ |
| Naming system | ✔ | ✔ (via ipns) | ✔ |
| Data available for ever (no deletion) | ✗ | ✔ | ✗ |
| Cache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

# Annex: Upspin history

Jan 3, 2016 – Jan 18, 2018

Contributions: **Additions**

Contributions to master, excluding merge commits




Google Security Blog

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

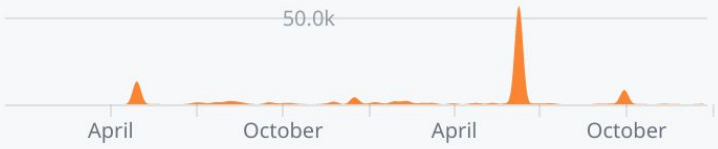
Another option for file sharing

February 21, 2017




**adg** #1

642 commits 164,657 ++ 72,489 --




**robpike** #2

574 commits 65,191 ++ 43,741 --