Tahoe-LAFS

September 8, 2014
Lightning Talks at hackerspace.gr

Αλέξανδρος, 0x350EBE881E75241E

Tahoe-LAFS

- Brian Warner, Zooko Wilcox-O'Hearn, Daira Hopwood
- Free open source, python
- Network storage system built around the principle of least authority
- The absolutely necessary control over the data for each subsystem

Tahoe-LAFS Goals

- Confidentiality, Integrity, Availability
- Separate who has access and control of the data from who hosts the data (least authority)
- Increase availability without extending your security perimeter.
- A distributed, host-independent, secure, fault-tolerant, cloud-like storage system (wow!)

Tahoe-LAFS gateway

- A gateway between the application (client) and the storage provides all the necessary operations.
- The boundary of our security perimeter
- Confidentiality via encryption
- Integrity via hashing
- Reliability via erasure-coding
- Key-value storage scheme

Confidentiality

- Encrypt every file before storing
- Different AES key for every file
- Put the key in the filehandle, use as storage-index
- storage-index= sha2(key)

- Key management?
- A file-handle can locate, retrieve and decrypt a file

Integrity

- Hash ciphertext before storing
- Check hash upon retrieval, wrong hash => read failure

- Encryption key + ciphertext's hash = capability to retrieve, verify, decrypt and read the file
- filecap = (key, SHA2(ciphertext))

Availability

- Erasure coding, Reed-Solomon
- Split ciphertext to N shares
- k out of N return the original
- Send different share to different storage nodes, tolerate failure of some nodes
- Overhead

Gimme code to understand

```
def PUT(value):
  ciphertext = AESenc(key, value)
  SI = SHA2(key)
  shares = FEC(ciphertext)
  for i,server in enum(servers):
    server.storage[SI] = shares[i]
  filecap = (key, SHA2(ciphertext))
  return filecap
```

```
def GET(filecap):
  (key, hash) = filecap
  SI = SHA2(key)
  shares = someservers.storage[SI]
  ciphertext = unFEC(shares)
  assert(SHA2(ciphertext) == hash)
  return AESdec(key, ciphertext)
```

URI:CHK:kuteanogafkqnmkqemjnbhi6um:rllaeflaiel55yje422o4tx4l45wugpp6cbsorq67n6e3oy67xsq:2:4:7588

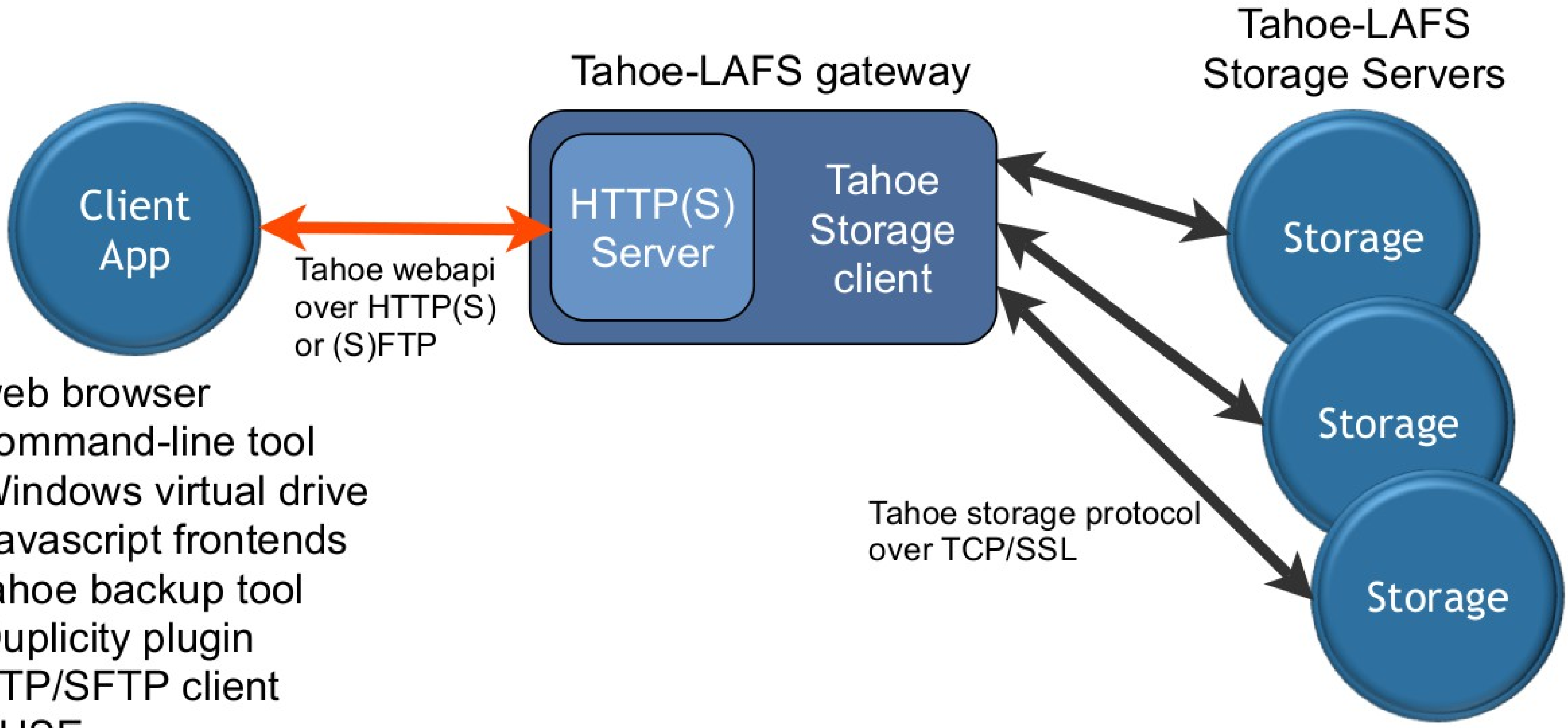
Storage Grid

- Multiple storage nodes
- An introducer lets client know about storage nodes
- Client's gateway talks to all nodes
- Different entities can contribute storage to a grid
- Redundant Array of Independent Clouds
- Gain availability without delegating control of your data

File types and capabilities

- Immutable files
  - read cap
- Mutable files
  - read cap
  - write cap
- Also directories, “dircaps”, map a child to a cap
- One can give a readcap cap to a peer while retaining writecap for herself

Overview


Tahoe-LAFS gateway
Tahoe-LAFS Storage Servers
Client App
Tahoe webapi over HTTP(S) or (S)FTP
HTTP(S) Server
Tahoe Storage client
Storage
Storage
Storage
Tahoe storage protocol over TCP/SSL

- web browser
- command-line tool
- Windows virtual drive
- Javascript frontends
- tahoe backup tool
- Duplicity plugin
- FTP/SFTP client
- FUSE

Usage

- Backup - duplicity backend, (s)ftp, fuse
- Sharing files/directories with a friend
- Tahoe-LAFS + (Tor | i2p)
- Tahoe-LAFS in Tails persistent volume
- Takedown resistant web hosting (onion + tahoe ftw)
- Other things a distributed, secure filesystem can do

Moar

- https://tahoe-lafs.org
- https://tahoe-lafs.org/trac/tahoe-lafs/attachment/wiki/News/tahoe-RSA-slides.pdf
- https://mailman.boum.org/pipermail/tails-dev/2014-June/005956.html
- http://killyourtv.i2p.us/tahoe-lafs/

Thanks!