New Fresh Open Source Object Storage

OpenIO

Jean-Francois Smigielski
SNIA SDC 2016
Santa Clara, CA
@OpenIO

Lille, Tokyo, San Francisco, Montreal, Madrid
Achievements

E-mails + Consumer Cloud >> major french ISP

Strong SLAs (latency, availability)

15+PiB, 60M end users

50Gib/s IN+OUT
#idea
2006: The painful days

- Too much data
- Growth acceleration
- Silos, sharding, migrations ...
- Too many operators

$\text{TCO too high}!$
Powering Real apps

Large stacks of interfaces become I/O-unfriendly

Necessity of an App. Aware solution with pragmatic Software Defined approach
“Double Humped” data

Mixed data profiles
- Most of the capacity by large cold files
- Most of the I/O ops on small hot files

E.g. mailbox index vs. Contents.

Colocating seems a bad idea.
JIT invest. appreciated

No big investment at day-0
Start small
Grow when necessary
Grow without limit
Recycle HW
JIT invest' problem: heterogeneity

Up-to-date HW
HW recycling
HW decommissioning
New vendor deals
Etc ...

Need for **versatility** adaptation!
Real humans managed

Strong End-User orientation

Observed Daylight rythm

Goal of flexibility

OK for admin.  Users online
Real humans served

Major ISP use-case
- Emails, Videos, Archives
- $10^8$ users / ISP
- $10^6$ contents / user
- $< 100$ms to download 1MiB
- $< 20$ms latency

Data life-cycle
- Recent data used
- Large data ignored
- Buzz effects

Room for caches & multiple tiers
#Architecture
Go for an Elastic Storage

Split hot/cold, offload cold
Independant low cost drives
Software Defined glue
The idea (to avoid)

Consistent Hashing Algorithms!

- Scale up → Rebalancing
- Decommissioning → Rebalancing

Both happen every week...
Locations not fixed?
Choose + Remember

Conscience

Directories
Distributes recent snapshots of your platform

Discovers the services
Qualifies them
Load-Balances on them

The Conscience
The directories

Remember polled locations for

$10^8 \text{ [user]} \times 10^6 \text{ [content/user]}$

Divided into trivial problems

- Naming indirection!
- 1 directory of users services
- 1 directory of contents / user
The directory of services

Layered as a hash table
- 1st level highly static
- 2nd level sharded in 64k slots

Replicated (replicas + backlinks)

SQLite powered: 1 file/shard replica

Sharded yet small & non-volatile → cached
The **containers**

Object view

Versions, Hardlinks, Properties

Efficient listing

Notifications → async mgmt

Replicated (replicas + backlinks)

SQLite powered : 1 file/replica

Map\langle name, Set\langle @\rangle \rangle
SDK's model

Client App.

- oio-sds client
- data-client
- dir-client

proxy-data

- Erasure coding
- Replication

proxy-dir

- Caching
- Load-balancing

http

http+json
Where latency matters
Connectors

OSS:
SDK: Python, C, Java
Interfaces: S3, Swift

Specific Editions:
Email: cyrus, dovecot, zimbra
Media: tailor-made HTTP
Filesystem: FUSE, NFS, ...
#Tiering
#Flexibility
Directory of users
Directories of contents
Pointers everywhere!
Storage Policy

= 

Storage Pool + Data Protection

“Where”

“How”
Storage pools
« Where »
Fine load-balancing
Tag-based, Fallbacks
Distance constraints
Geo-distribution

Data Protection
« How »
Erasure code ?
Replication ?
Plain ?
**Configuration**

« When »

Set upon an upload

Per content > container > account

Managed asynchronously

**Tiering rules**

« Why »

Dynamic storage policy
(still under active development)

Filter on metadata
(Mime-Type, size, ctime, mtime, ...)

« Why »
Immediate benefits

Possible partitions...

- Platform (high-end, low-end)
- Users set (gold, regular)

All QoS elements!

All allow cost optimisations!
Why public tiers?

- TCO still too high for ultra-cold data
- Alternative to tape archiving
- Ultra-cold tier
Public tiers: How?

Dedicated Storage Policy

Embedded connector

Asynchronism to cope with limited bandwidth
First partner
Backblaze B2
#serverless
#storage
Still...

Too much HW complexity

Too much sysadmin required

Technical “lasagna” with bugs and latencies on each layer

Could we drop servers?
The Kinetic Opportunity

TCP/IP/Ethernet connected drives
Direct access to data
No host required
Sorted map of <Key,Value>
The perfect tier!

Same vision!

Sleek protocol

No host required

Direct access to the data (when it matters)

Proxied access (when enough)
Meet the Kinetic Open Storage Group and OpenIO at the Plugfest

Sonoma Room, 09/20
Apps need more than just storage

Processing colocated to data
Metadata handling
Full text search

We call this **Grid for Apps**
... out of scope of SDC
http://openio.io

#scalable
#opensource
#objectstorage
#tiering
#serverless
#hybrid
#cloud