Cloud Filesystem

Jeff Darcy
for BBLISA, October 2011

What is a Filesystem?

- “The thing every OS and language knows”
- Directories, files, file descriptors
- Directories within directories
- Operate on single record (POSIX: single byte) within a file
- Built-in permissions model (e.g. UID, GID, ugo·rwx)
- Defined concurrency behaviors (e.g. fsync)
- Extras: symlinks, ACLs, xattrs

Are Filesystems Relevant?

- Supported by every language and OS natively
- Shared data with rich semantics
- Graceful and efficient handling of multi-GB objects
- Permission model missing in some alternatives
- Polyglot storage, e.g. DB to index data in FS

Network Filesystems

- Extend filesystem to multiple clients
- Awesome idea so long as total required capacity/performance doesn't exceed a single server
  - ...otherwise you get server sprawl
- Plenty of commercial vendors, community experience
- Making NFS highly available brings extra headaches

Distributed Filesystems

- Aggregate capacity/performance across servers
- Built-in redundancy
  - ...but watch out: not all deal with HA transparently
- Among the most notoriously difficult kinds of software to set up, tune and maintain
  - Anyone want to see my Lustre scars?
- Performance profile can be surprising
- Result: seen as specialized solution (esp. HPC)

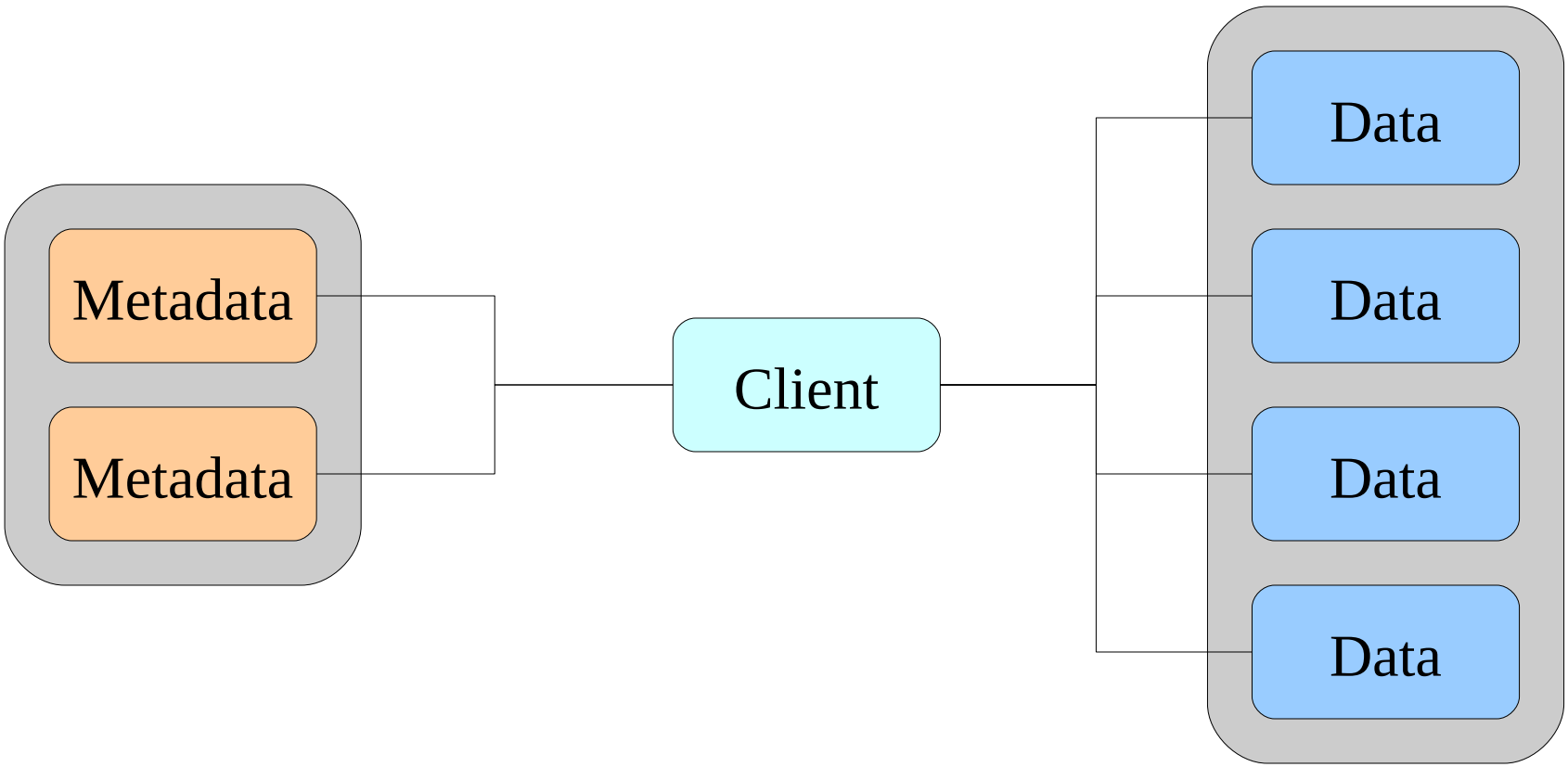
Example: NFS4.1/pNFS

- pNFS distributes data access across servers
- Referrals etc. offload some metadata
- Only a protocol, not an implementation
  - OSS clients, proprietary servers
- Does not address metadata scaling at all
- Conclusion: partial solution, good for compatibility, full solution might layer on top of something else

Example: Ceph

- Two-layer architecture
- Object layer (RADOS) is self-organizing
  - can be used alone for block storage via RBD
- Metadata layer provides POSIX file semantics on top of RADOS objects
- Full-kernel implementation
- Great architecture, some day it will be a great implementation

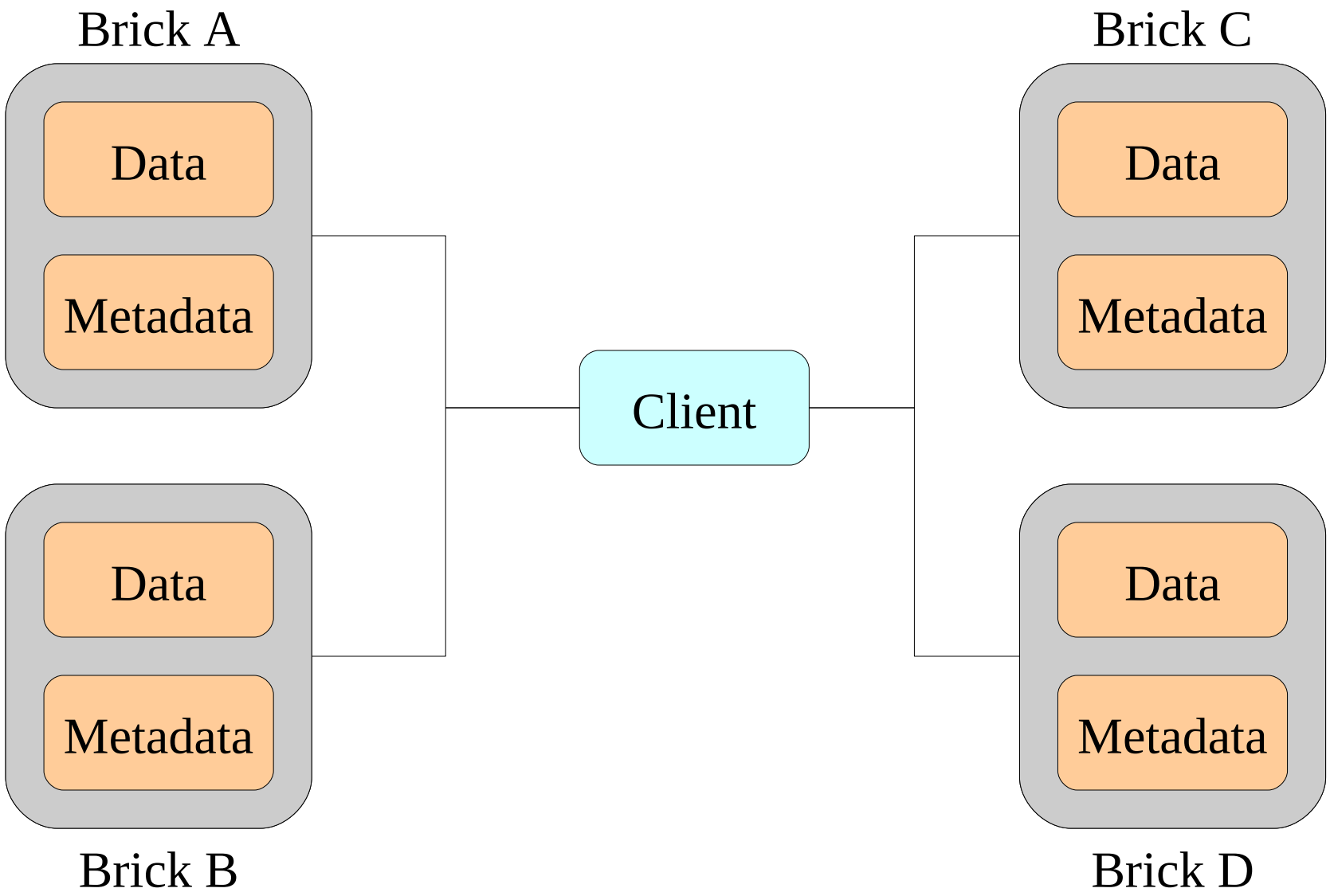
Ceph Diagram



Example: GlusterFS

- Single-layer architecture
  - sharding instead of layering
  - one type of server – data and metadata
- Servers are dumb, smart behavior driven by clients
- FUSE implementation
- Native, NFSv3, UFO, Hadoop

GlusterFS Diagram



OK, What About HekaFS?

- Don't blame me for the name
  - trademark issues are a distraction from real work
- Existing DFSes solve many problems already
  - sharding, replication, striping
- What they don't address is cloud-specific deployment
  - lack of trust (user/user and user/provider)
  - location transparency
  - operationalization

Why Start With GlusterFS?

- Not going to write my own from scratch
  - been there, done that
  - leverage existing code, community, user base
- Modular architecture allows adding functionality via an API
  - separate licensing, distribution, support
- By far the best configuration/management
- OK, so it's FUSE
  - not as bad as people think + add more servers

HekaFS Current Features

- Directory isolation
- ID isolation
  - “virtualize” between server ID space and tenants'
- SSL
  - encryption useful on its own
  - authentication is needed by other features
- At-rest encryption
  - Keys **ONLY** on clients
  - AES-256 through AES-1024, “ESSIV-like”

HekaFS Future Features

- Enough of multi-tenancy, now for other stuff
- Improved (local/sync) replication
  - lower latency, faster repair
- Namespace (and small-file?) caching
- Improved data integrity
- Improved distribution
  - higher server counts, smoother reconfiguration
- Erasure codes?

HekaFS Global Replication

- Multi-site asynchronous
- Arbitrary number of sites
- Write from any site, even during partition
  - ordered, eventually consistent with conflict resolution
- Caching is just a special case of replication
  - interest expressed (and withdrawn) not assumed
- Some infrastructure being done early for local replication

Project Status

- All open source
  - code hosted by Fedora, bugzilla by Red Hat
  - Red Hat also pays me (and others) to work on it
- ~~Close collaboration with Gluster~~
  - ~~they do most of the work~~
  - ~~they're open-source folks too~~
  - ~~completely support their business model~~
- “current” = Fedora 16
- “future” = Fedora 17+ and Red Hat product

Contact Info

- Project
  - http://hekafs.org
  - jdarcy@redhat.com
- Personal
  - http://pl.atyp.us
  - jeff@pl.atyp.us

Cloud Filesystem

Jeff Darcy
for BBLISA, October 2011

What is a Filesystem?

- “The thing every OS and language knows”
- Directories, files, file descriptors
- Directories within directories
- Operate on single record (POSIX: single byte) within a file
- Built-in permissions model (e.g. UID, GID, ugo·rwx)
- Defined concurrency behaviors (e.g. fsync)
- Extras: symlinks, ACLs, xattrs

Are Filesystems Relevant?

- Supported by every language and OS natively
- Shared data with rich semantics
- Graceful and efficient handling of multi-GB objects
- Permission model missing in some alternatives
- Polyglot storage, e.g. DB to index data in FS

Network Filesystems

- Extend filesystem to multiple clients
- Awesome idea so long as total required capacity/performance doesn't exceed a single server
  - ...otherwise you get server sprawl
- Plenty of commercial vendors, community experience
- Making NFS highly available brings extra headaches

Distributed Filesystems

- Aggregate capacity/performance across servers
- Built-in redundancy
  - ...but watch out: not all deal with HA transparently
- Among the most notoriously difficult kinds of software to set up, tune and maintain
  - Anyone want to see my Lustre scars?
- Performance profile can be surprising
- Result: seen as specialized solution (esp. HPC)

Example: NFS4.1/pNFS

- pNFS distributes data access across servers
- Referrals etc. offload some metadata
- Only a protocol, not an implementation
  - OSS clients, proprietary servers
- Does not address metadata scaling at all
- Conclusion: partial solution, good for compatibility, full solution might layer on top of something else

Example: Ceph

- Two-layer architecture
- Object layer (RADOS) is self-organizing
  - can be used alone for block storage via RBD
- Metadata layer provides POSIX file semantics on top of RADOS objects
- Full-kernel implementation
- Great architecture, some day it will be a great implementation

Example: GlusterFS

- Single-layer architecture
  - sharding instead of layering
  - one type of server – data and metadata
- Servers are dumb, smart behavior driven by clients
- FUSE implementation
- Native, NFSv3, UFO, Hadoop

OK, What About HekaFS?

- Don't blame me for the name
  - trademark issues are a distraction from real work
- Existing DFSes solve many problems already
  - sharding, replication, striping
- What they don't address is cloud-specific deployment
  - lack of trust (user/user and user/provider)
  - location transparency
  - operationalization

Why Start With GlusterFS?

- Not going to write my own from scratch
  - been there, done that
  - leverage existing code, community, user base
- Modular architecture allows adding functionality via an API
  - separate licensing, distribution, support
- By far the best configuration/management
- OK, so it's FUSE
  - not as bad as people think + add more servers

HekaFS Current Features

- Directory isolation
- ID isolation
  - “virtualize” between server ID space and tenants'
- SSL
  - encryption useful on its own
  - authentication is needed by other features
- At-rest encryption
  - Keys **ONLY** on clients
  - AES-256 through AES-1024, “ESSIV-like”

HekaFS Future Features

- Enough of multi-tenancy, now for other stuff
- Improved (local/sync) replication
  - lower latency, faster repair
- Namespace (and small-file?) caching
- Improved data integrity
- Improved distribution
  - higher server counts, smoother reconfiguration
- Erasure codes?

HekaFS Global Replication

- Multi-site asynchronous
- Arbitrary number of sites
- Write from any site, even during partition
  - ordered, eventually consistent with conflict resolution
- Caching is just a special case of replication
  - interest expressed (and withdrawn) not assumed
- Some infrastructure being done early for local replication

Project Status

- All open source
  - code hosted by Fedora, bugzilla by Red Hat
  - Red Hat also pays me (and others) to work on it
- ~~Close collaboration with Gluster~~
  - ~~they do most of the work~~
  - ~~they're open-source folks too~~
  - ~~completely support their business model~~
- “current” = Fedora 16
- “future” = Fedora 17+ and Red Hat product

Contact Info

- Project
  - http://hekafs.org
  - jdarcy@redhat.com
- Personal
  - http://pl.atyp.us
  - jeff@pl.atyp.us