Cloud Storage

Jeff Darcy

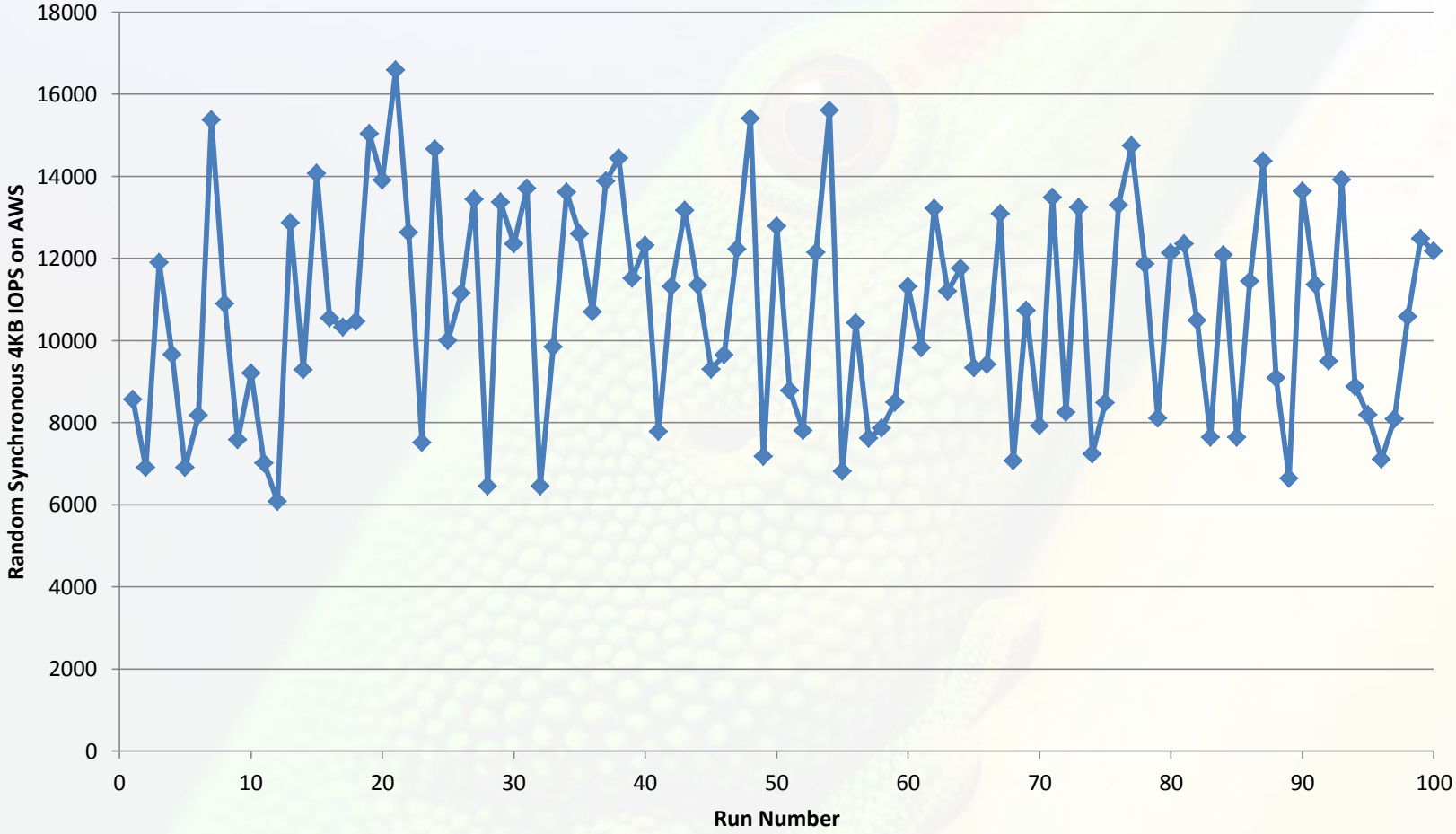
Red Hat

Which Is It?

For the cloud

In the cloud

Horizontal Scalability



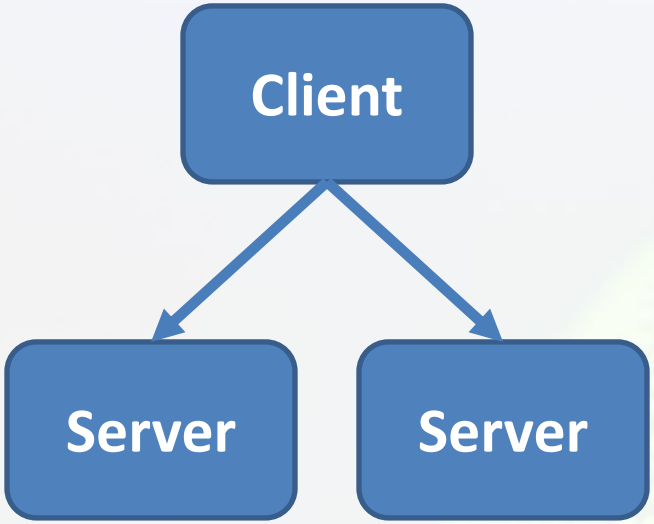
Autonomous Operation

- Adding machines is easy and cheap
- Adding people is hard and expensive
- Maximize machines/users/TB per operator
  - Automatic repair, rebalancing
  - Cluster-wide operations via single commands
  - External scripting and monitoring

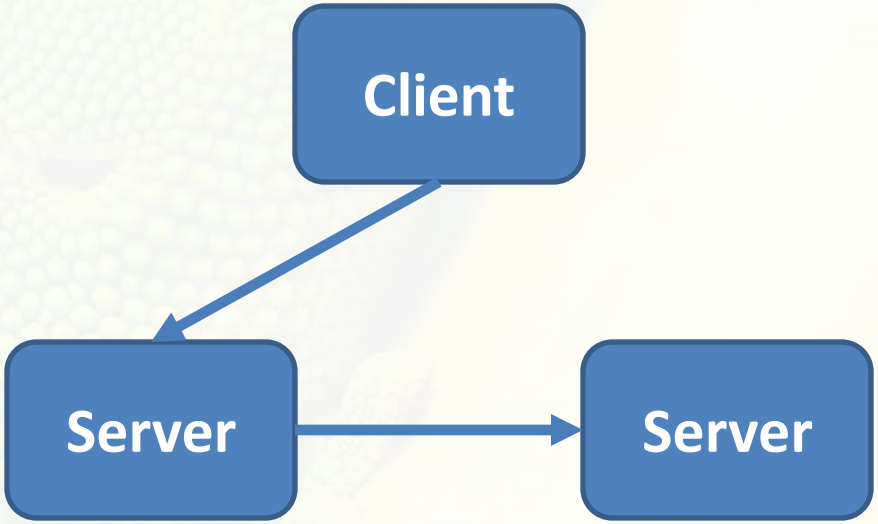
Performance Questions

- How many requests per operation?
  - Are requests batched and/or pipelined?
  - What are the main data flows?
- Are there single-node SPOFs/bottlenecks?
  - Lustre MDS, HDFS LameNode
- Are there more efficient APIs?
  - Blocks, objects, low-level library

Replication Flows



Fan out

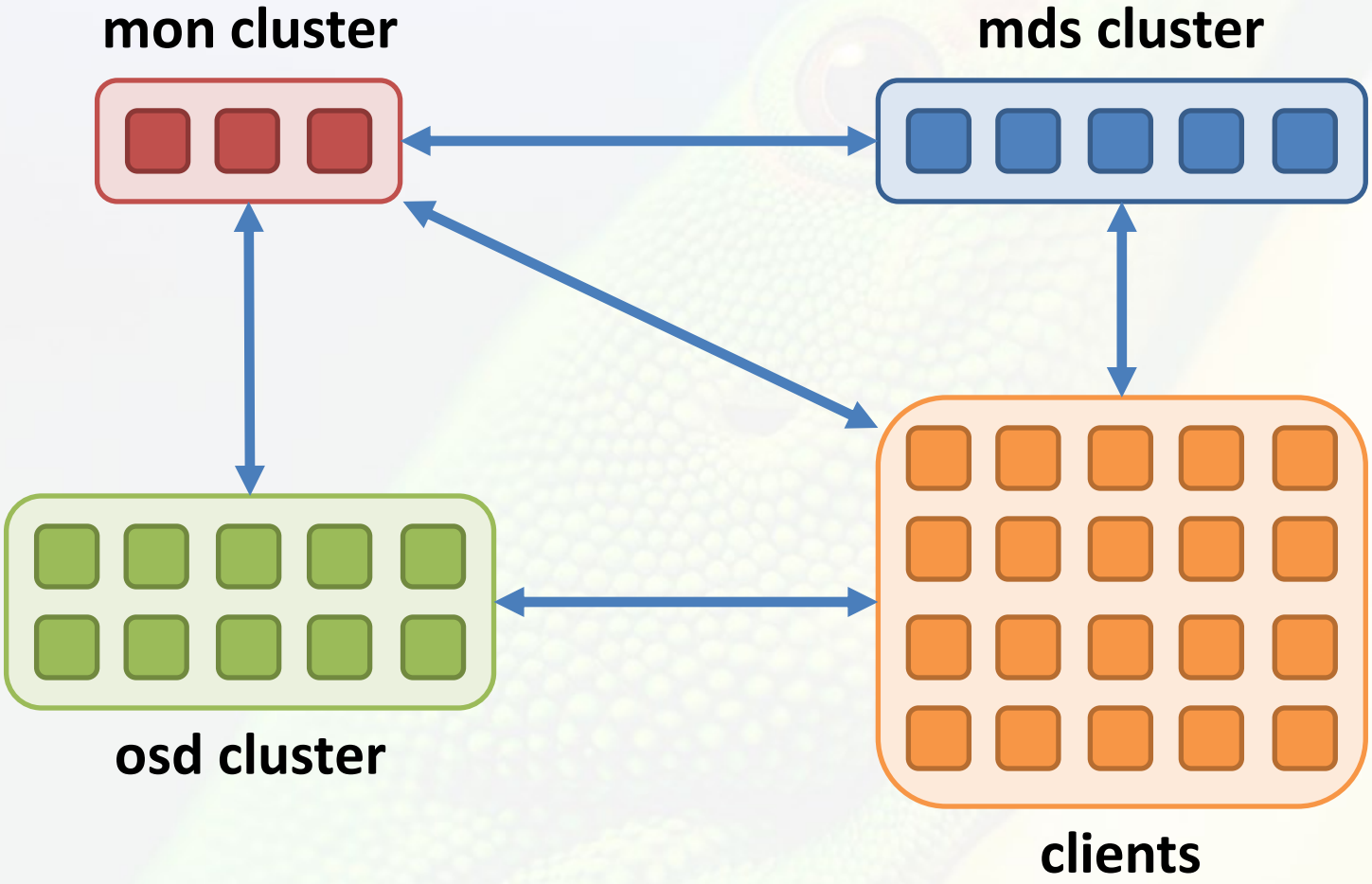


Chain

Other Questions

- Does it provide true durability?
  - Does it honor fsync/O_SYNC? (MooseFS doesn't)
  - Does it "complete" writes locally? (HDFS does)
- What are the consistency guarantees?
  - Strong, eventual, variable
- How comprehensible are the semantics?
  - POSIX, SQL vs. home-grown
- How sane are configuration/operations?

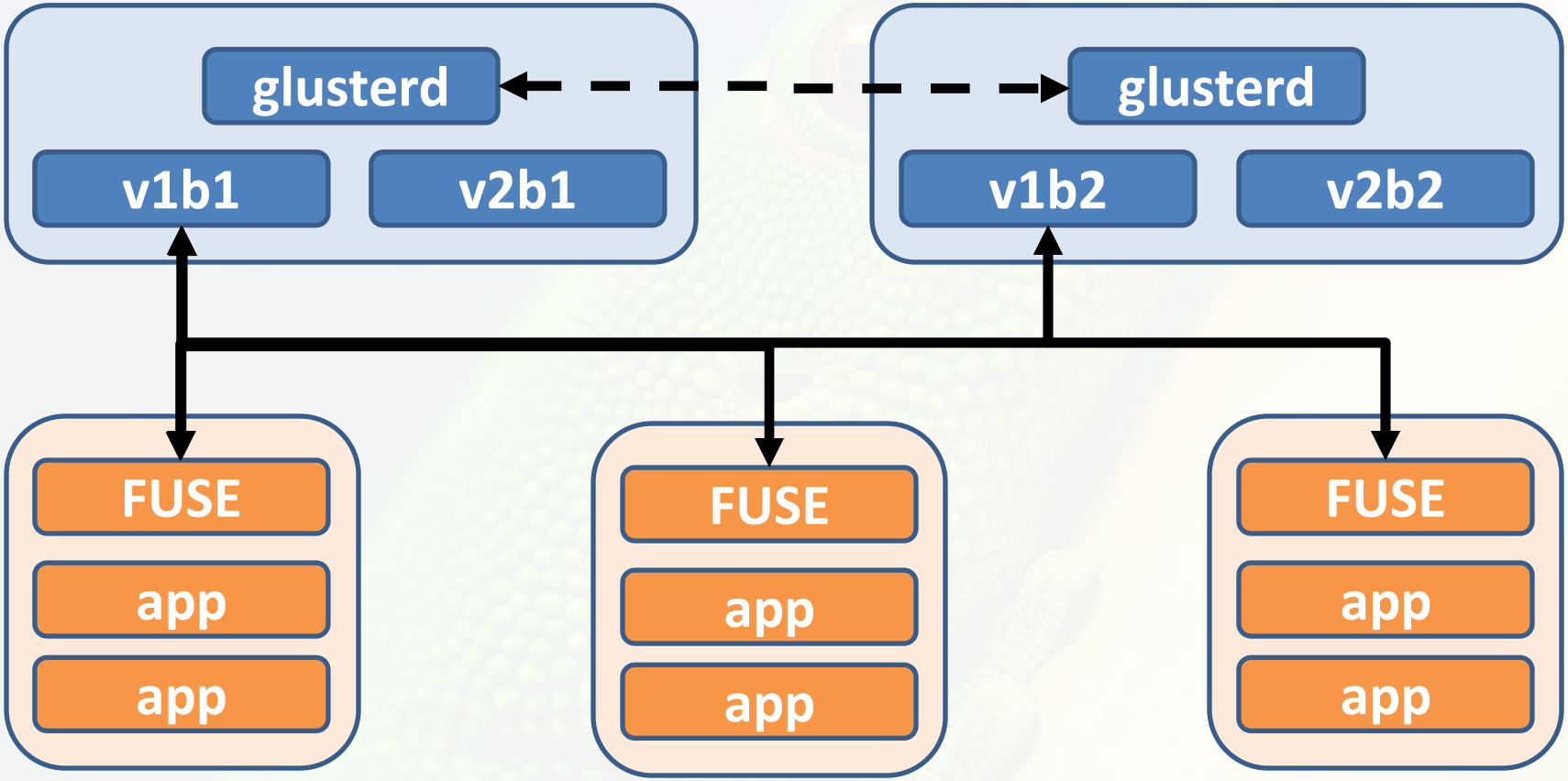
Case Study: Ceph (FS)



Ceph Features

- Native format: objects (T10-ish)
  - Block and S3/Swift object adapters
  - Filesystem: separate metadata, kernel client
- Automatic replication (chain/splay)
- Automatic rebalancing
- Snapshots
- Config files plus some automation

Case Study: GlusterFS



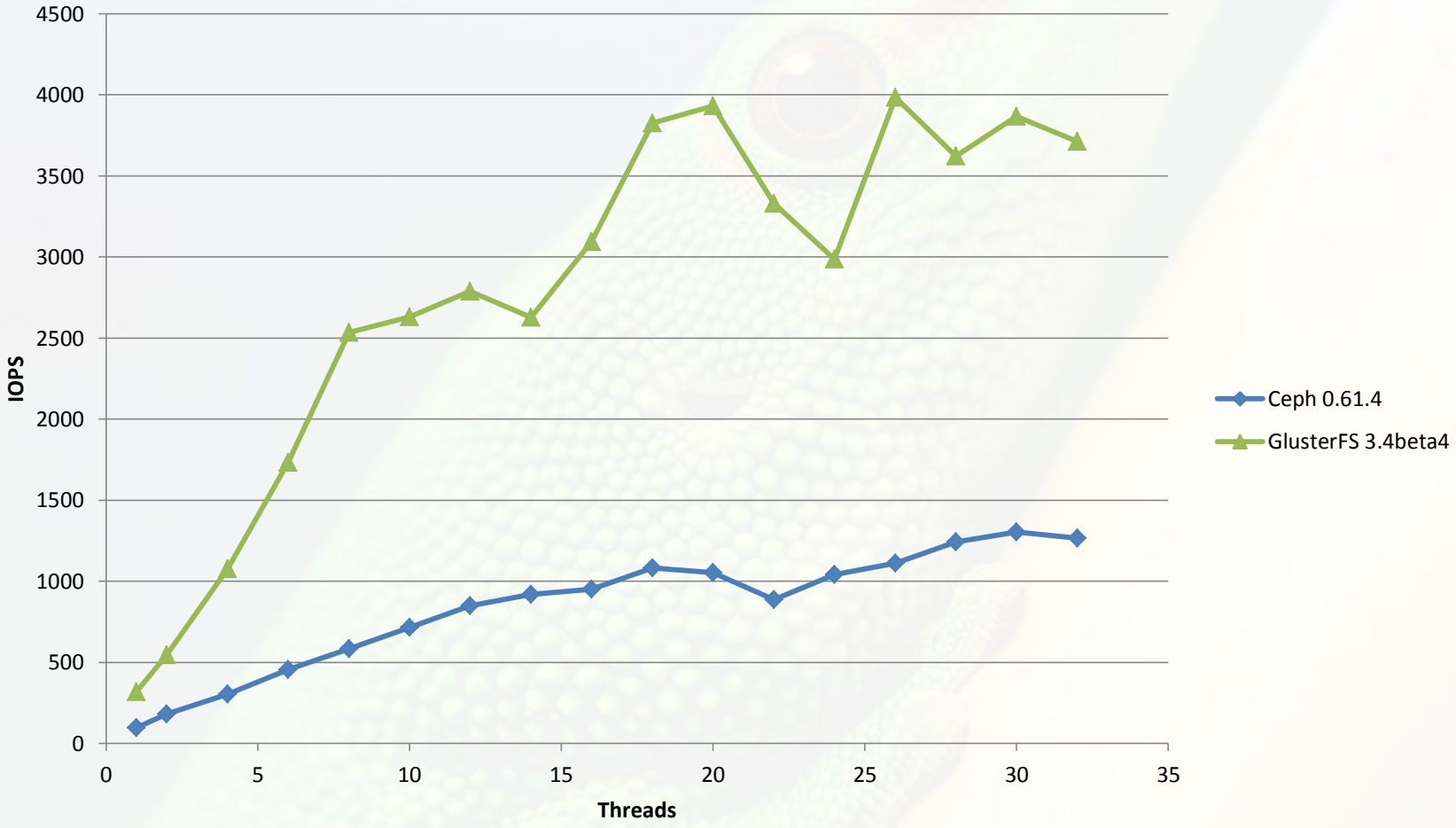
GlusterFS Features

- Native format: POSIX filesystem
  - Object/block layered on top
  - FUSE client, NFSv3
- Automatic replication (fan-out)
- Geo-replication for all data types
- Configuration/management via single CLI
- Modular structure eases feature addition
  - Multiple third-party “translators”

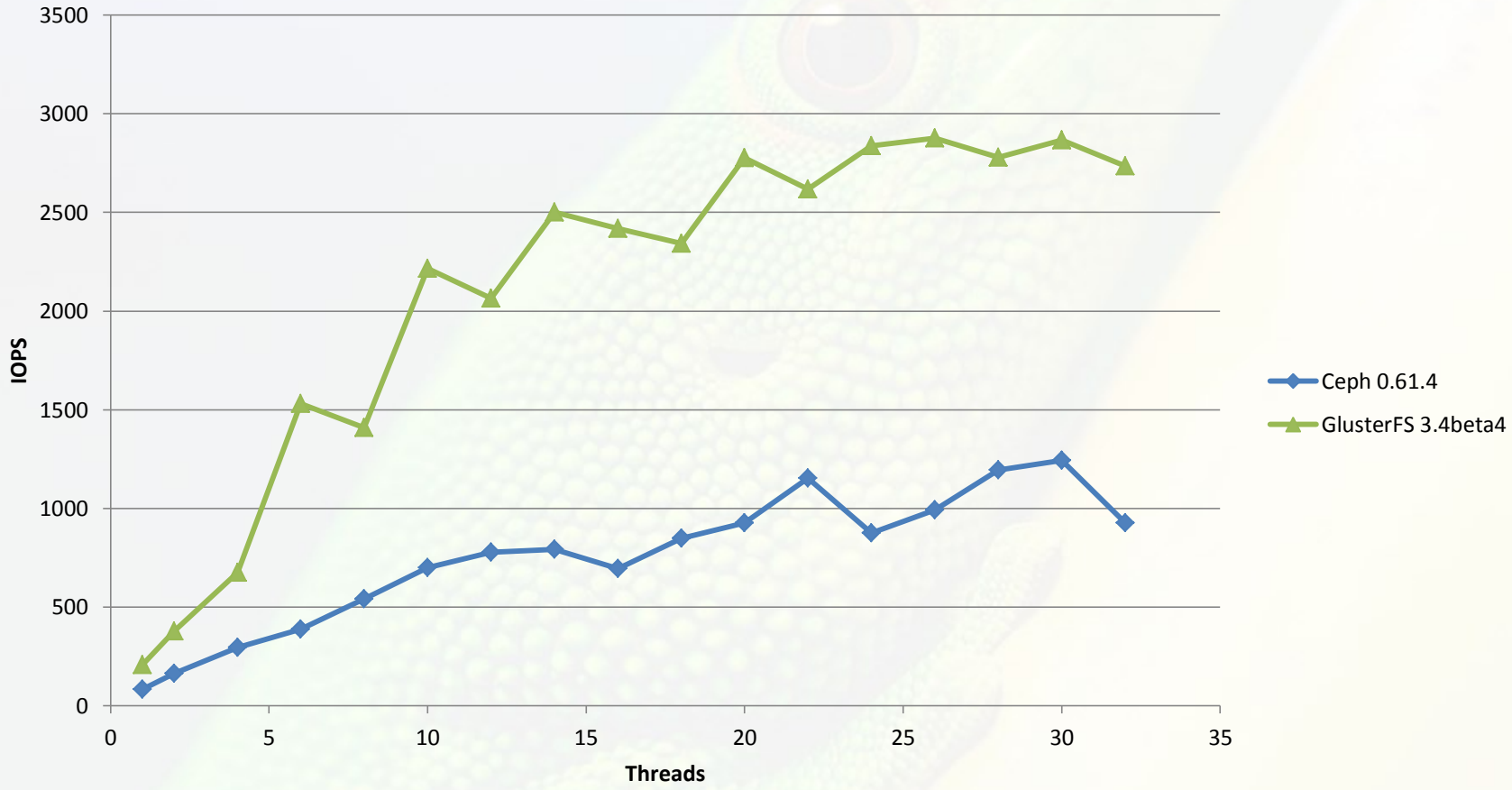
But How Do They Perform?

- Rackspace 15GB second-gen servers
  - 6 VCPUs
  - Measured ~13K local IOPS
  - Measured ~400Mb/s between systems
  - Centos 6.4, Ubuntu 13.04
- Random O_SYNC 4KB IOPS
- Ceph: kernel client + librados
- GlusterFS: FUSE client + libgfapi

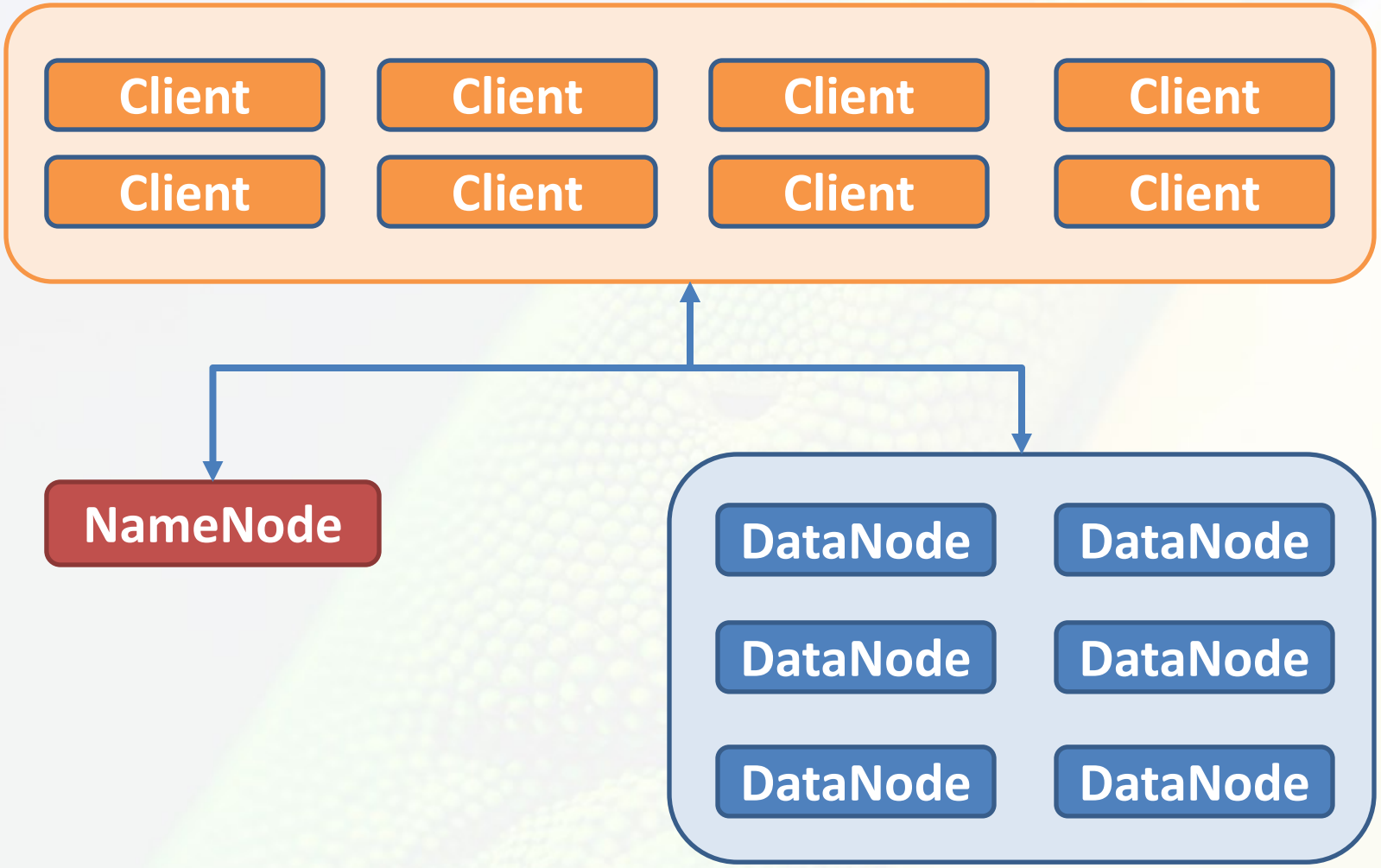
API Comparison



Filesystem Comparison



HDFS


Client
Client
Client
Client
Client
Client
Client
Client
NameNode
DataNode
DataNode
DataNode
DataNode
DataNode
DataNode

HDFS Misfeatures

- Single NameNode
  - HA/Federated NameNode exists (at last), maybe even in your Hadoop distribution, but not default
- Optimized for one workload
  - Sequential write-once, huge chunk size
- Not a “normal” filesystem
  - Missing features (xattrs), local-write cheating
- Requires copying to/from other storage

“Database” Options

- Cassandra
  - BigTable/columnar data model
  - CassandraFS, Brisk (Hadoop), CQL
- Riak
  - Document data model
  - Riak CS (Swift)
- Same scalability, operational simplicity, etc.
  - Many of the same algorithms under the covers

Conclusion

- Some features aren't optional
  - e.g. correct durability behavior
  - Beware of options that sacrifice correctness for performance
- Non-performance factors matter a lot
  - Open source and community, horizontal scalability, operational simplicity, additional APIs
- Pick the right tool for the job