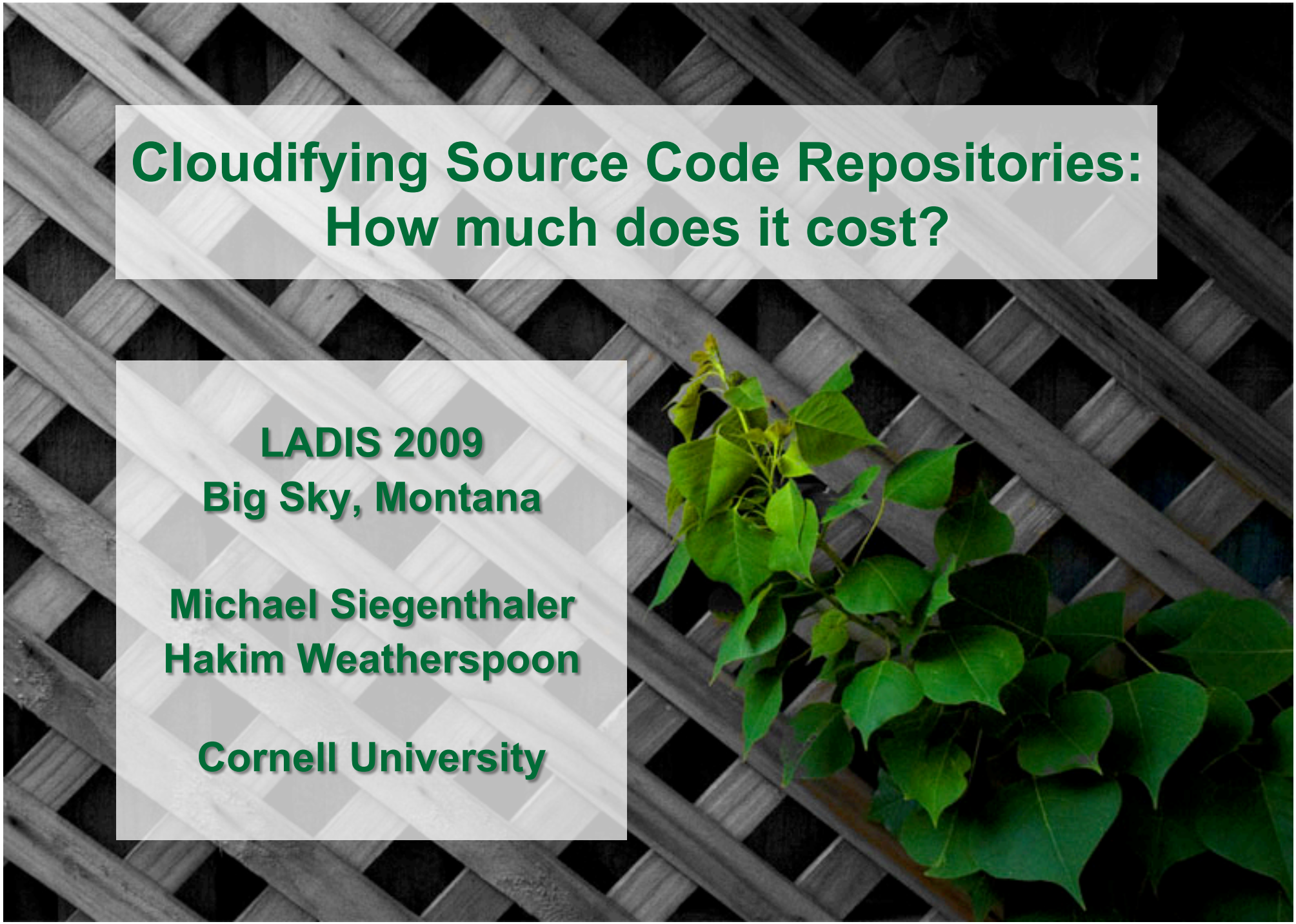


Cloudifying Source Code Repositories: How much does it cost?

LADIS 2009
Big Sky, Montana

Michael Siegenthaler
Hakim Weatherspoon

Cornell University

A Brief History of Cloud Computing

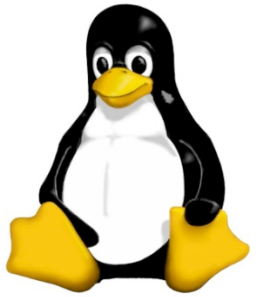
- Large scale
- Application-specific architectures
- Developed for in-house use

- Available for general usage
- Inexpensive, even for small or medium scale deployments

What is Revision Control?

- Repository for data (source code)
  - All changes are tracked by date and author
  - Branching and merging
- Why move it to the cloud?
  - Resilient storage
  - No physical server to administrate
  - Scale to larger communities (SourceForge)

Available Tools



- Subversion, revision control system
  - Free, open-source
  - Very popular
  - Rigid consistency model

- Amazon S3, cloud storage service
  - Eventual consistency

- Yahoo ZooKeeper, coordination service
  - Free, open-source

Various alternative solutions exist…

Cloud Computing

- Subversion etc.
- Repository stored persistently in the cloud
- One true, consistent repository exists

P2P

- GIT etc.
- Repository stored at every client
- Many repository copies, converging eventually

Outline

- Costs of using cloud storage for revision control
- Architecture of a simple solution
- Performance evaluation

How to Measure Costs

- Each revision stored as two files on disk
  - Revision data (diff against earlier revisions)
  - Revision properties (author, log message…)
- Calculate bandwidth, per-transaction, and storage costs of pushing each revision into S3 over time

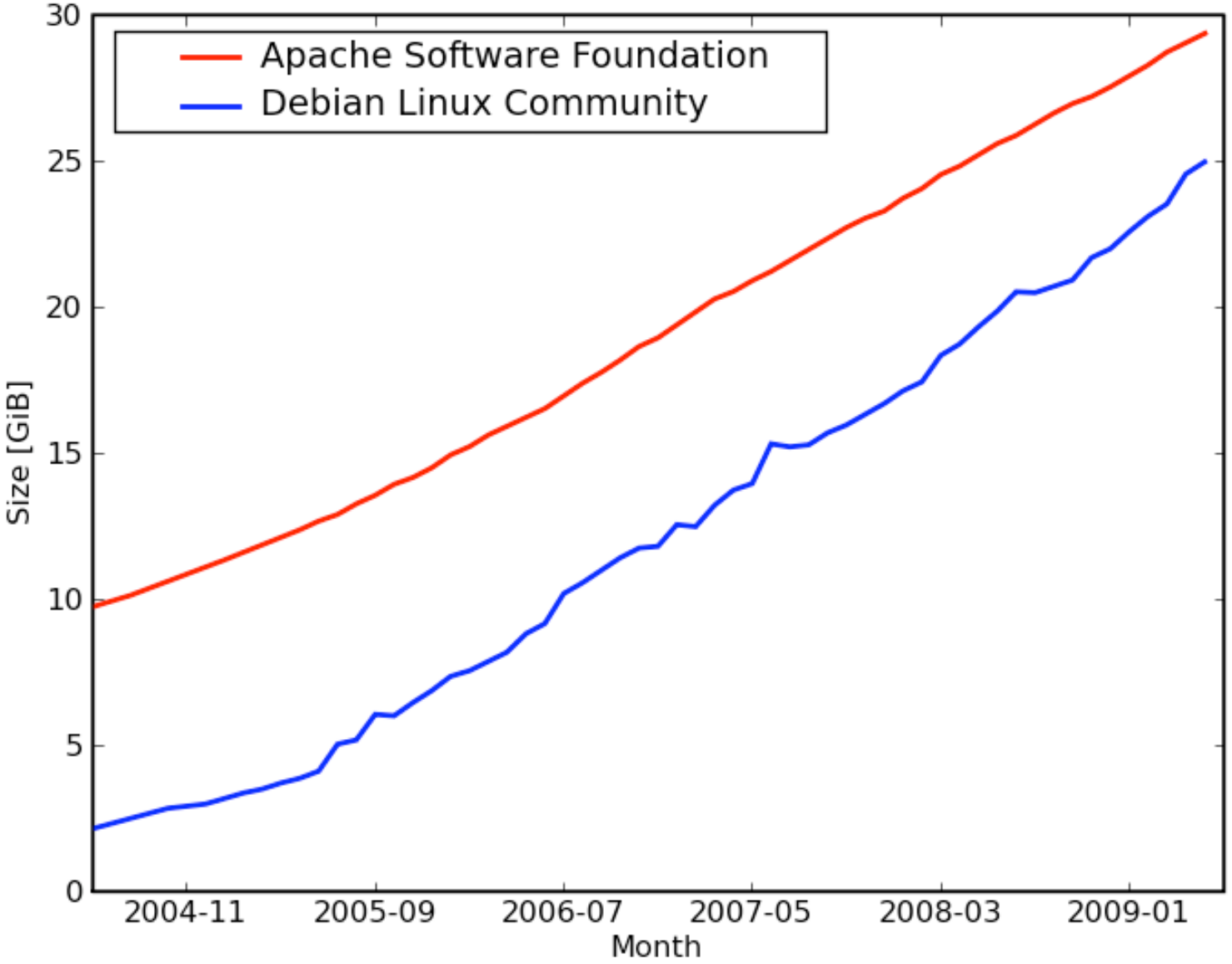
Description	Price
Monthly storage	$0.15 per GiB
Bandwidth in	$0.10 per GiB
Bandwidth out	$0.17 per GiB
Per 1000 writes	$0.01
Per 10,000 reads	$0.01

Storage Costs

Software Project	Monthly Cost
SquirrelMail	$0.03
phpMyAdmin	$0.04
Subversion	$0.08
Mono	$0.57
KDE	$7.35

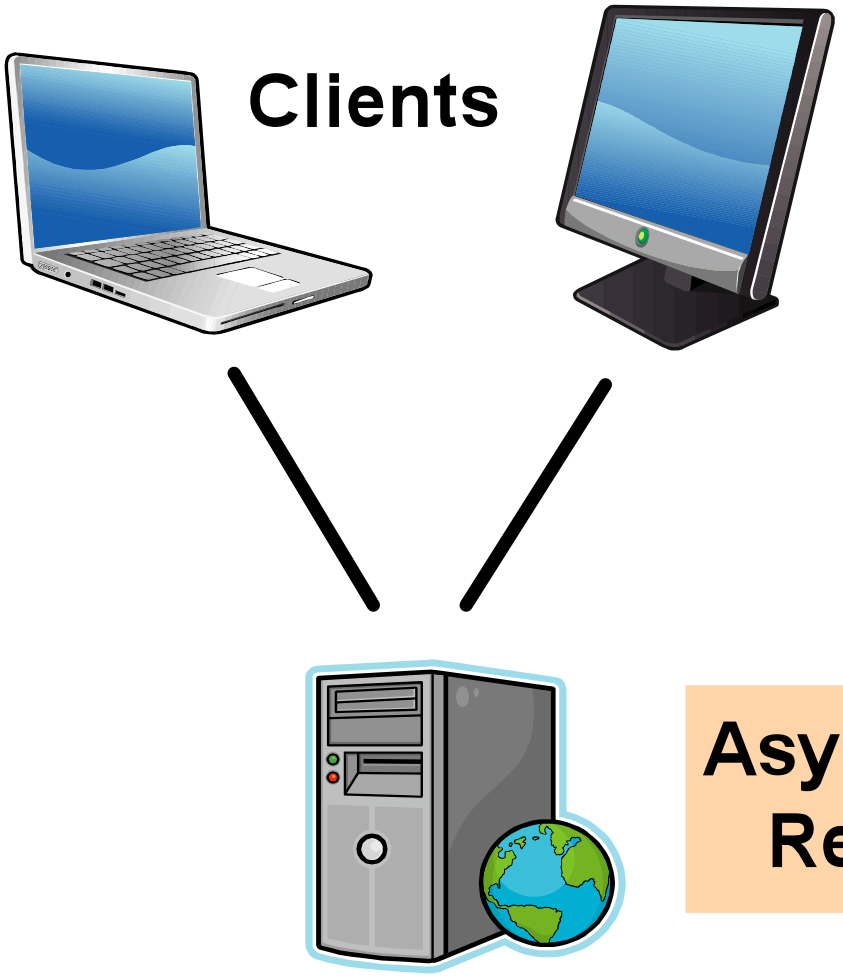
Hosting Community	Monthly Cost
Debian Linux Community	$3.89
Apache Software Foundation	$4.58

Storage Trends

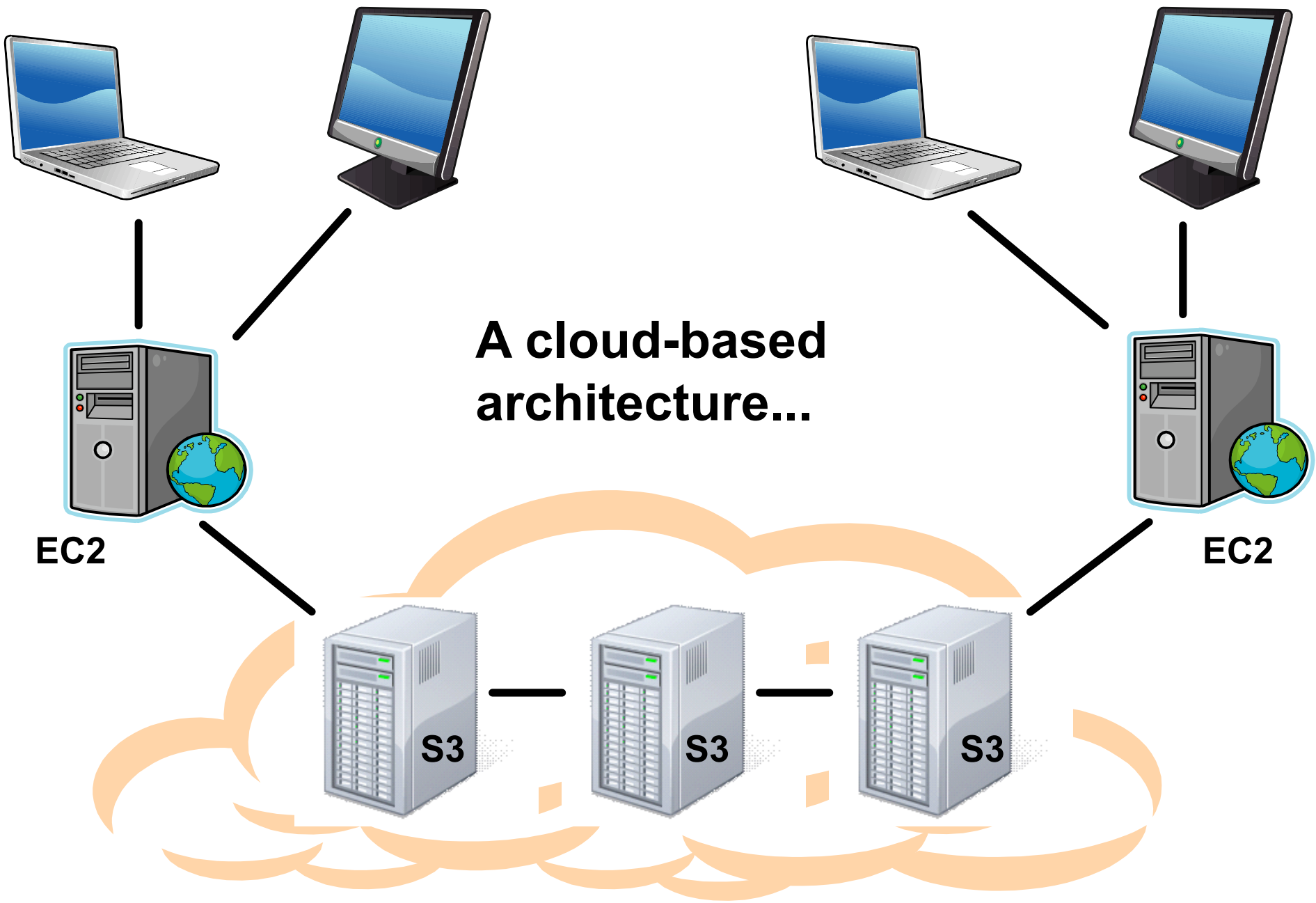


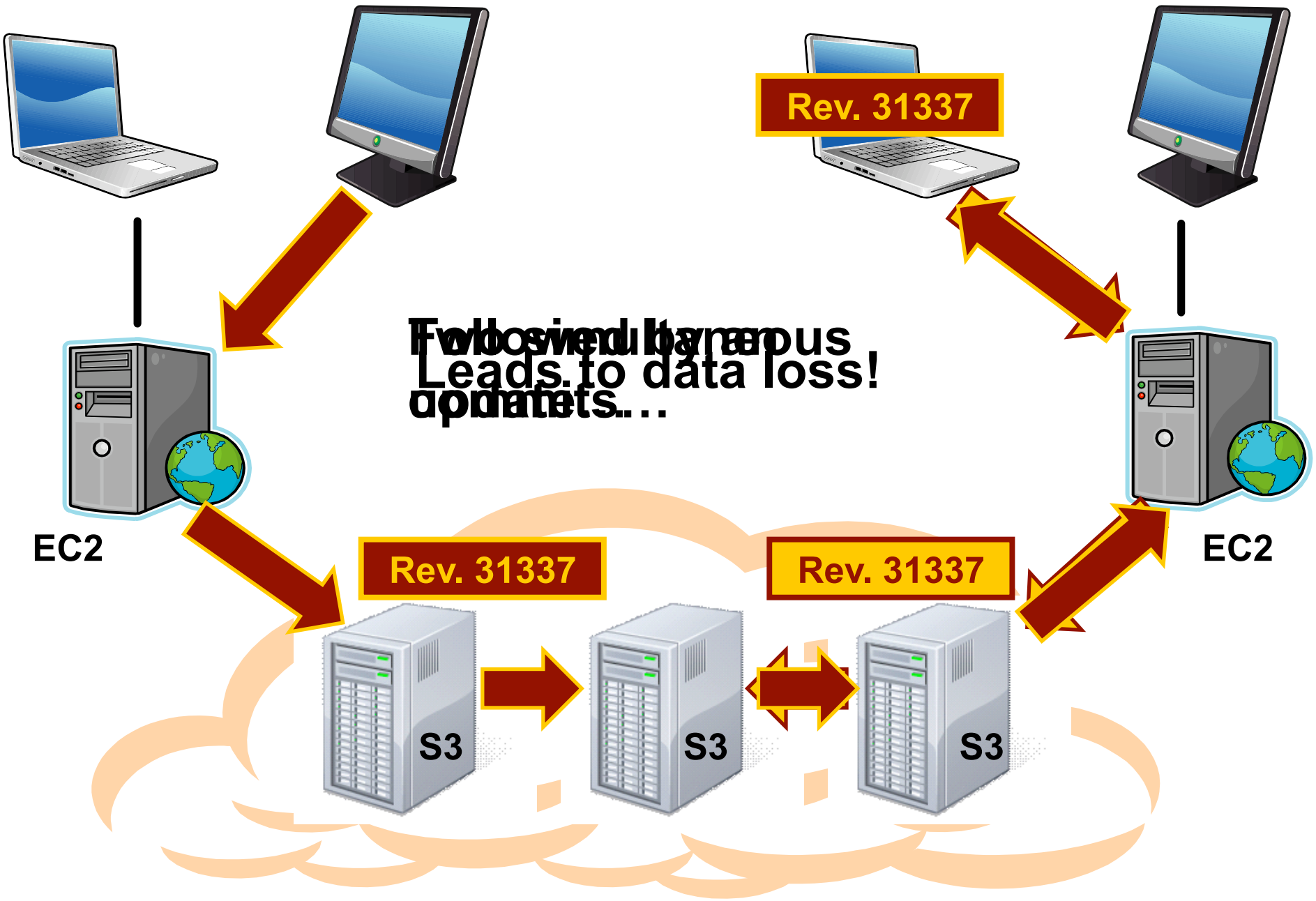
Outline

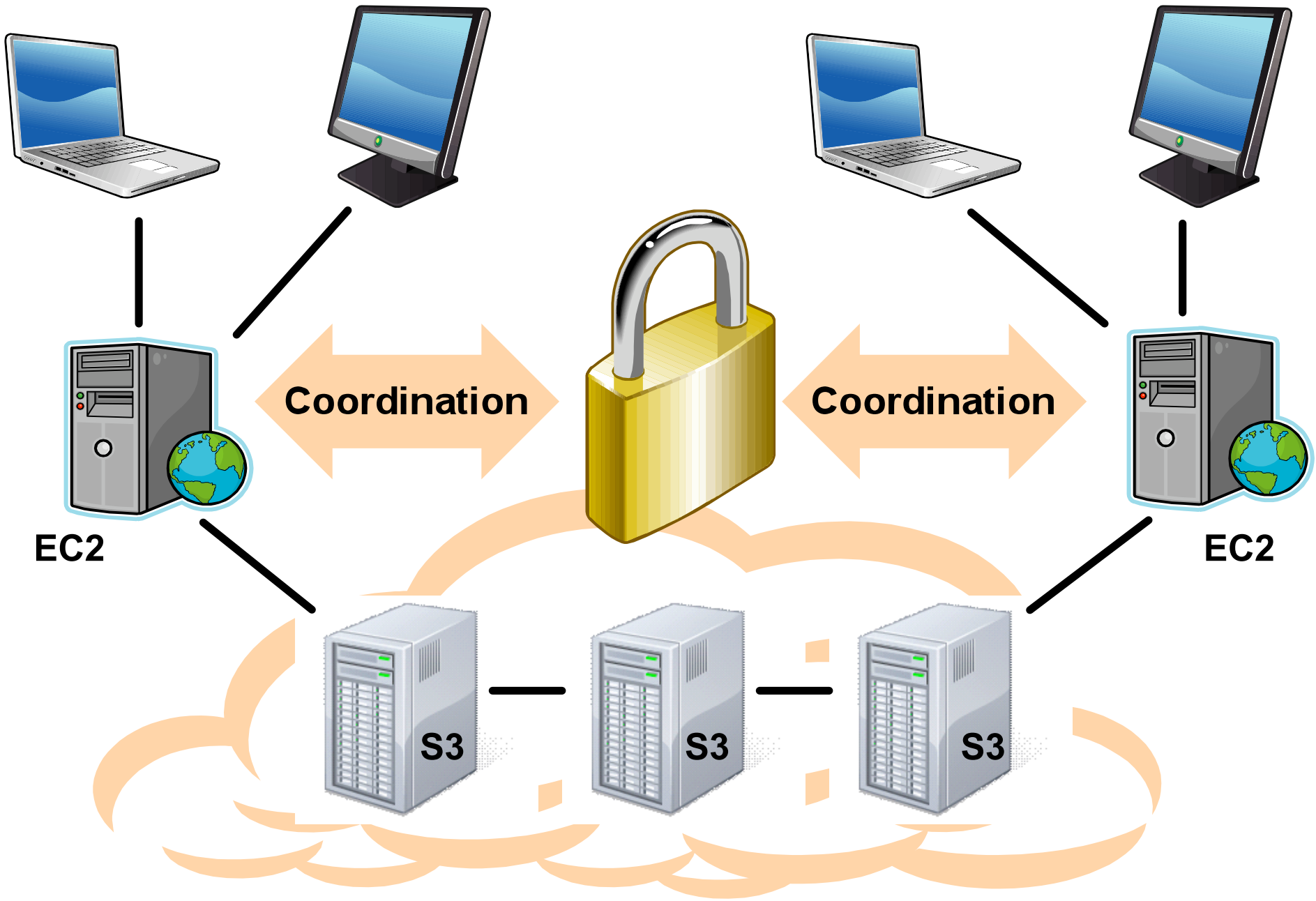
- Costs of using cloud storage for revision control
- Architecture of a simple solution
- Performance evaluation



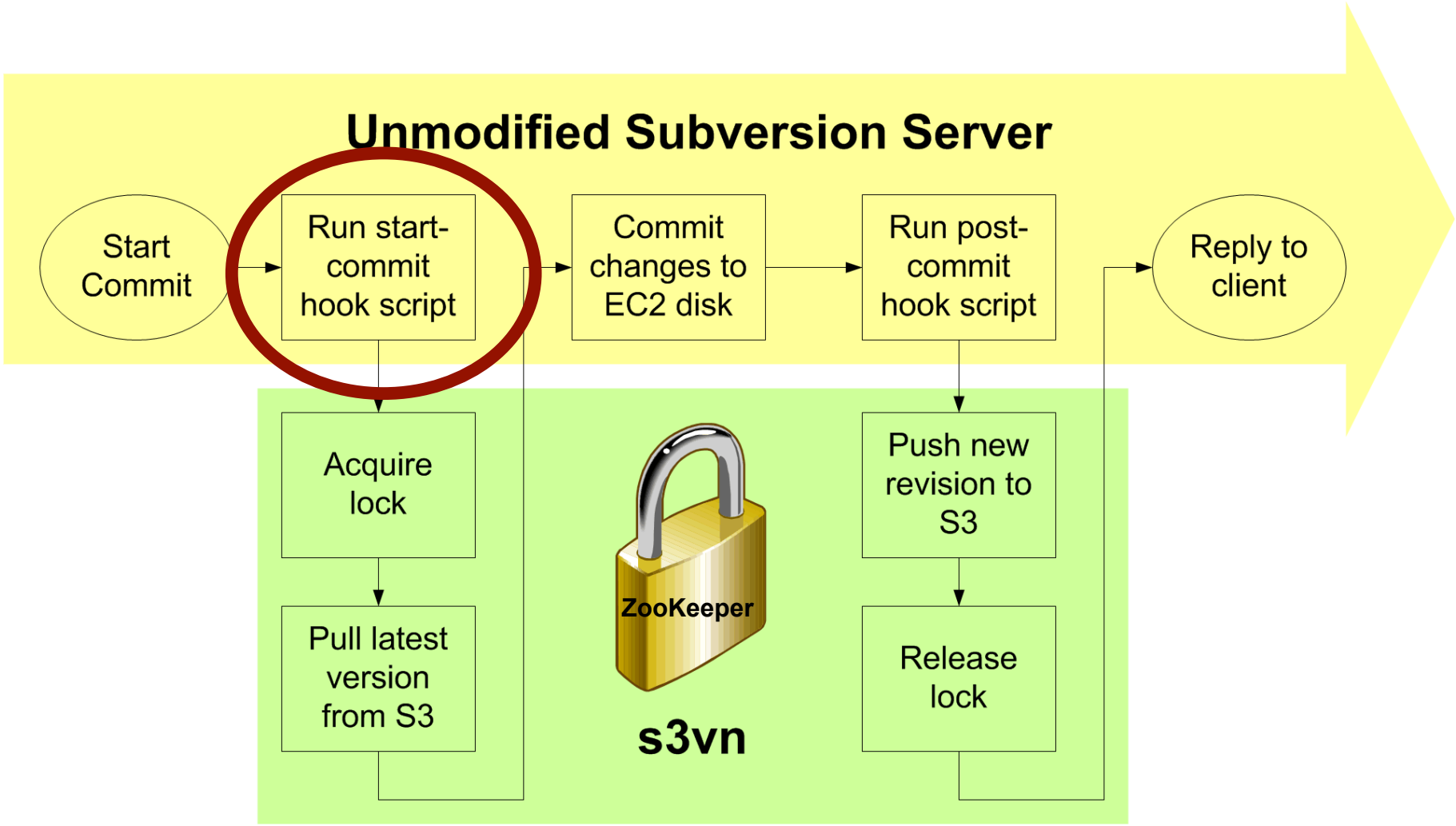
Today's architecture for source code revision control...

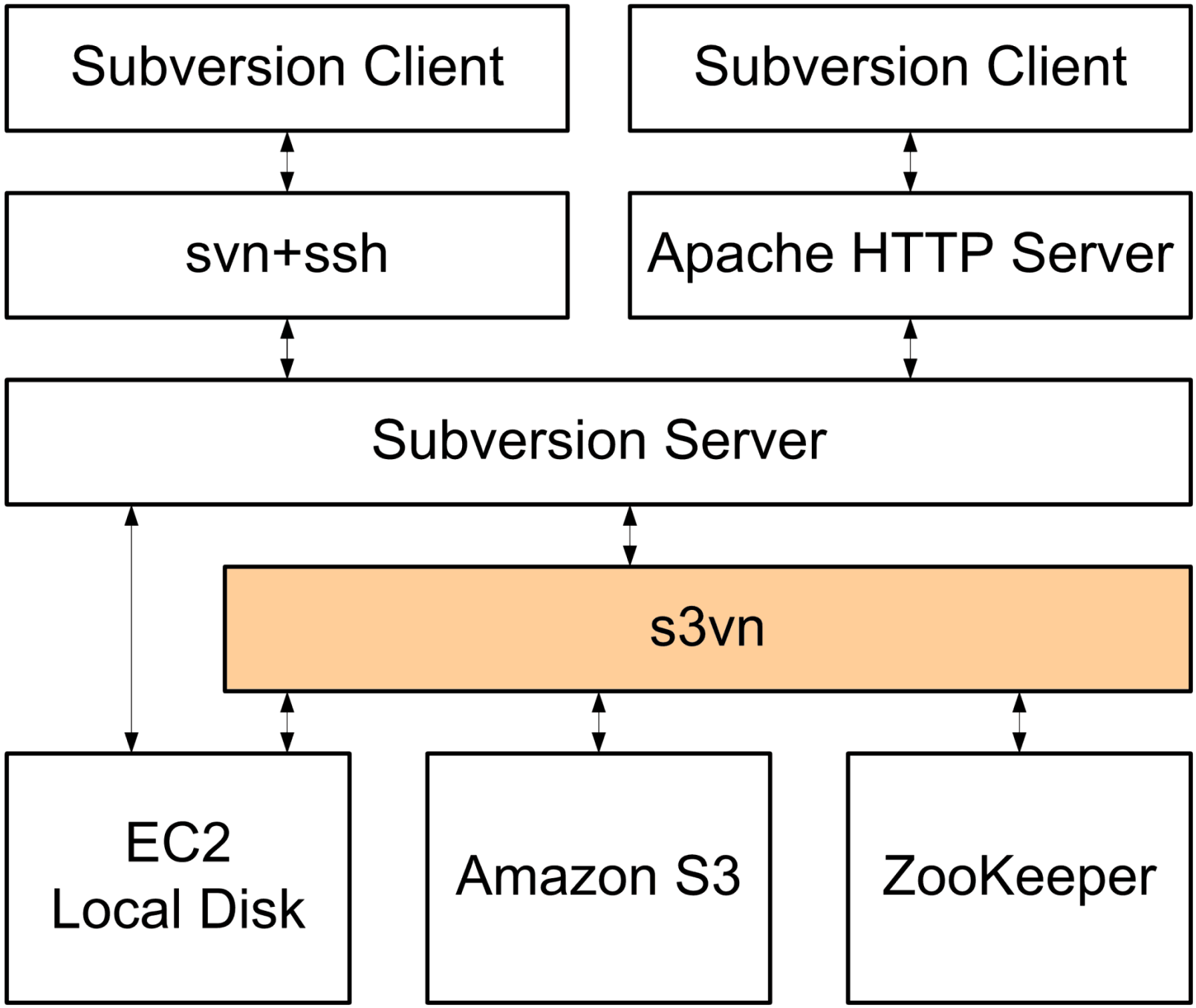

A cloud-based architecture...
EC2
EC2
S3
S3
S3


Rev. 31337
Leads to data loss!
EC2
Rev. 31337
Rev. 31337
S3
S3
S3
EC2


Coordination
Coordination
EC2
EC2
S3
S3
S3

Commit Process




Subversion Client
Subversion Client
svn+ssh
Apache HTTP Server
Subversion Server
s3vn
EC2
Local Disk
Amazon S3
ZooKeeper

How ZooKeeper is Utilized

- Acquire a lock by creating a node with an atomically increasing sequence number `/s3vn/<repo>/lock/lock-<seq>`
- List contents of `/s3vn/<repo>/lock` and wait if a node with a lower number than ours exists
- Store current revision number: `/s3vn/<repo>/current`
- Delete the lock node to release the lock

Outline

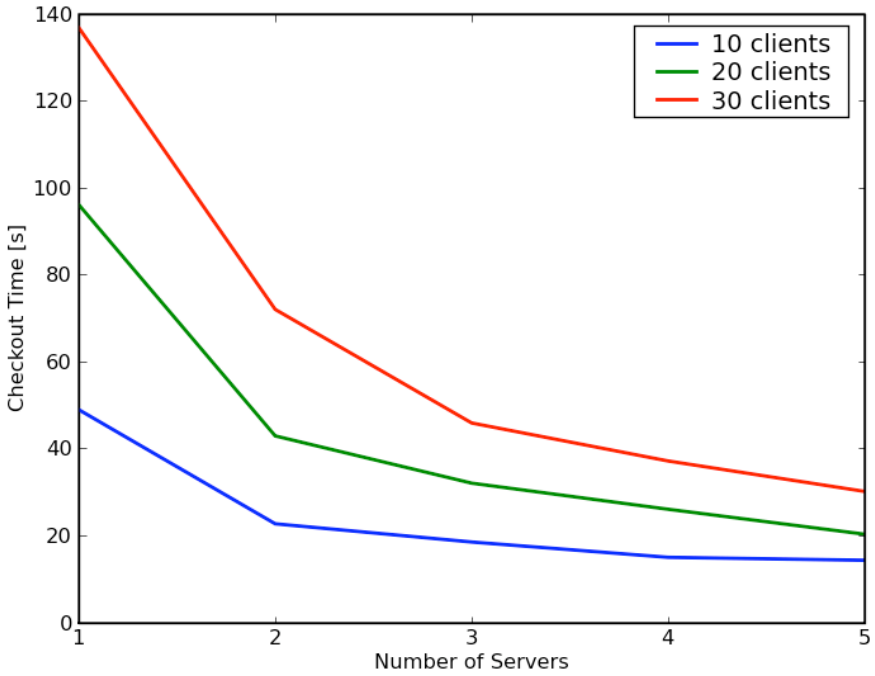
- Costs of using cloud storage for revision control
- Architecture of a simple solution
- Performance evaluation

Usage Observations

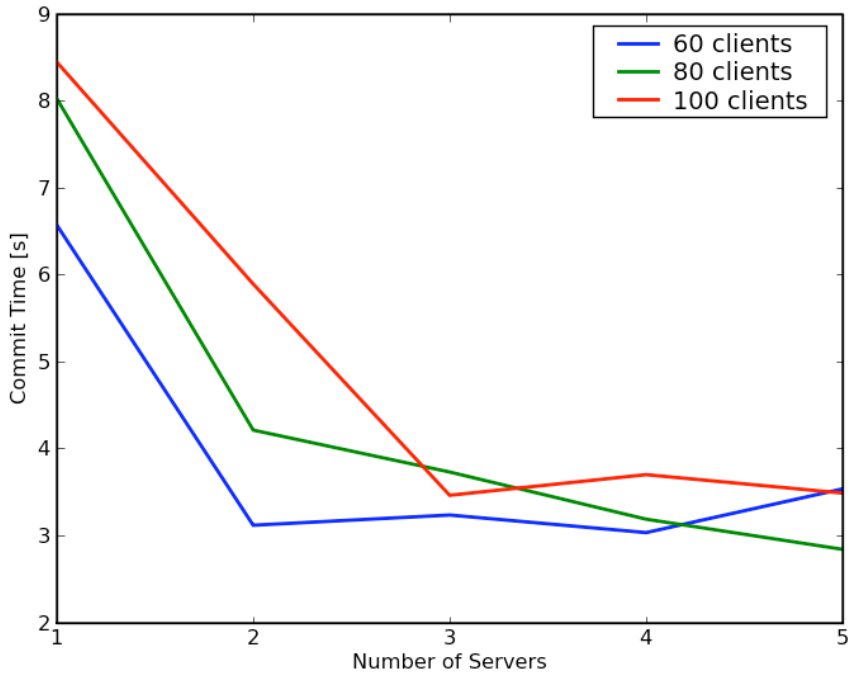
- Apache Foundation
  - 1 repository, 74 projects
  - Average 1.10 commits per minute
  - Maximum 7 commits per minute
- Debian community
  - 506 repositories
  - Average 1.12 commits per minute (aggregate)
  - Maximum 6 commits per minute (aggregate)

Results

Checkouts (Reads)



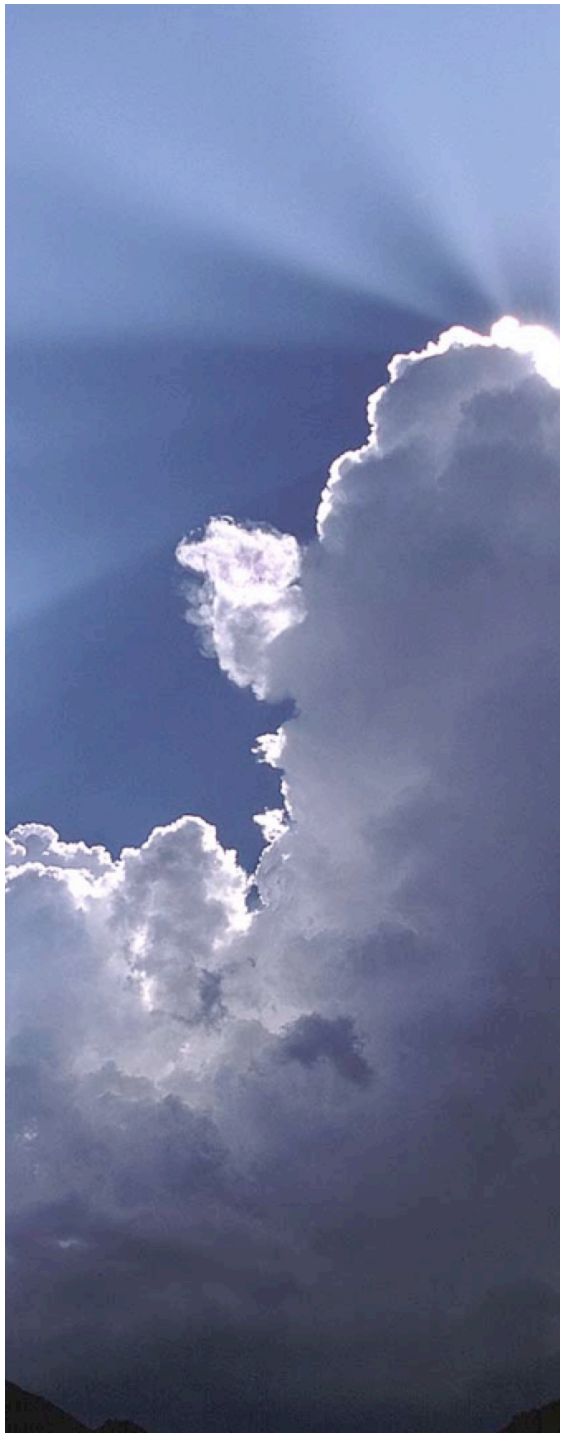
Commits (Writes)



- Adding servers improves the user experience

Conclusion

- Storing source code repositories in the cloud is feasible…
- …and very inexpensive
- Only minor changes to existing revision control systems are necessary to robustly take advantage of cloud storage



Lock Service: ZooKeeper

- Open source tool developed by Yahoo!
- Tree namespace with storage in nodes
  - **Sequence** nodes: automatically append a sequence number
  - **Ephemeral** nodes: disappear when the session that created them is closed
  - Clients can **watch** a node for changes
- All clients see changes in same order

s3vn Components

- **mkrepo**: Create a repository in an S3 bucket
- **fetchrepo**: Copy a repository from S3 to the local disk
- **updaterepo**: Background process to fetch changes from S3 as they are made
- **start-commit-hook**: Acquire a global write lock when a new revision is committed
- **post-commit-hook**: Upload the new changes to S3 and release the write lock