

OceanStore

An Architecture for Global-Scale Persistent storage

Presented by:
Amir Khella

Motivation

- Computing everywhere
- Connectivity everywhere
- But where is persistent data?
- Requirements for persistent infrastructure:
 - Connectivity
 - Secure
 - Durability
 - Information divorce from location

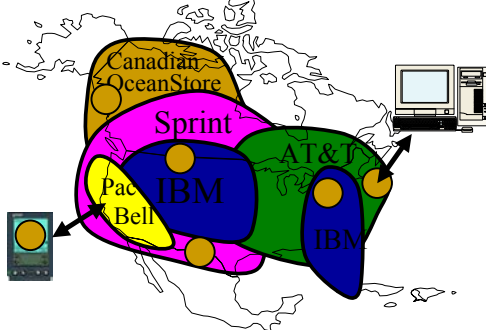
Presentation layout

- OceanStore goals
- System overview
- Applications
- System Architecture

What is OceanStore?

- A model in which information is free to migrate to wherever it is needed.
- Storage should survive major disasters and regional outages
- Two unique goals:
 - Non-trusted infrastructure
 - Nomadic data

The OceanStore system



Non-trusted Infrastructure

- Servers may crash without warning or leak information to third parties.
- All information that enters the infrastructure must be encrypted.
- Servers must be allowed to practice in protocols for distributed consistency management (one class of servers we can trust with carrying out protocols but not trust with the content of data)

Nomadic Data

- Data that is allowed to flow freely.
- Data can be cached anywhere , anytime (promiscuous caching).
- Continuous introspective monitoring is used to discover relationships between objects.

System Overview

- Fundamental unit is the persistent object
- Each object has a GUID
- Objects are replicated and stored on multiple servers (floating replicas)
- Objects are modified through *updates*
- Updates create a new version, consistency is maintained through versioning
- Objects are in Active or Archival forms

Applications

- Groupware and PIM tools
- Very large digital libraries and repositories for scientific data
- New streaming applications such as sensor data aggregation and dissemination.

System Architecture

- Naming
- Access control
- Data location and routing
- Update model
- Deep archival storage
- OceanStore API
- Introspection

Naming

- Each object has a GUID (secure hash of the owner's name key and some other human-readable name)
- Some objects act as directories , mapping human readable names to GUIDs
- GUIDs identify other OceanStore entities such as servers and archival fragments
- Objects that are replicas of each other have the same GUID

Access Control

- Reader restriction:
 - Encrypt non-public data and distribute the key to users with read access
 - Problem: There is no way to make a reader forget what he has read
- Writer restriction:
 - Through ACLs specified for each object by its owner
 - ACL can be another object or value indicating a default
 - Remember that each user has a signing key, ACLs use that key for granting access.

Data Location and Routing

- Distributed routing
- Attenuated bloom filters
- Wide-scale distributed data location

Distributed routing

- Messages are labeled with the destination GUID, a random number and a small predicate
- Destination IP does not appear in the message
- Routing layers finds the nearest node matching the predicate and having the desired GUID
- Messages begin routing from node to node along the distributed data structure till reaching the destination node
- As a result of the routing algorithm, entities accessed frequently will reside closer to where they are used

Attenuated bloom filters

- Probabilistic distributed algorithm based on hill climbing
- An attenuated bloom filter of depth D can be viewed as an array of D normal Bloom filters (Bloom filter???)
- An attenuated Bloom filter is stored for each directed edge in the network
- Query is routed along the edges whose filters indicate the presence of the object at the smallest distance.

Wide scale distributed data location

- Variation of Plaxton's randomized hierarchical distributed data structures.
- Every server is assigned a random and unique node-ID
- These IDs are used to construct a mesh of neighbor links (the node-IDs are divided into chunks of four bits and links are constructed using these chunks of bits)

Achieving fault tolerance

- Hashing GUIDs with some salt values, and mapping the result to several different root nodes
- The gain is redundancy and making it difficult for DoS attacks to target a range of nodes
- Bad links are detected and routing is continued by jumps to the neighbor nodes
- The infrastructure continually detects and repairs neighbor links

Update Model

- High degree of write sharing
- To allow concurrent updates and avoid locking, OceanStore uses conflict resolution update model
- The problem:
 - Conflict resolution requires the ability to perform server side computation on data.
 - OceanStore : Trust no server with the data

Update format and semantics

- List of predicates associated with actions
- To apply an update:
 - A replica evaluates each of the updates predicates in order
 - If any of the predicates evaluate to true, the actions associated with the earliest true will be atomically performed on data object
 - Update is said to commit, otherwise it's said to abort
 - Updates are always logged
- Model is extended to work on ciphertext which is encrypted information

Serializing updates

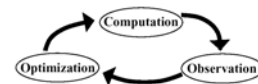
- Conflict resolution concept:
 - Series of updates
 - Choose a total order
 - Apply updates in that order
- All updates are required to pass through a master replica.. Problem!!
 - Replace master replica with a primary tier of replicas which cooperate to choose the final commit order for the updates
 - Add secondary replicas tier to serve as communication layer for the primary replicas.
 - They also contain both tentative and committed updates.
 - They timestamp tentative updates and pass them to the primary tier which uses the timestamp in ordering decision.

Deep Archival Storage

- Erasure coding is employed for archival
 - Treats input data as a series of fragments n , and transforms them into greater number of fragments ($2n$ or $4n$) generated in parallel.
 - Any n of the generated fragments are sufficient to reconstruct the original data
 - Spread code fragments widely, hence it is very unlikely that enough servers will be down to prevent the retrieval of data
- To preserve erasure nature of fragments, hierarchical hashing is used to verify each fragment.
- To maximize survivability, administrative domains are ranked according to their reliability and trustworthiness and data is distributed accordingly

Introspection

- Millions of servers with varying connectivity, disk capacity, and power.
- Introspection is employed.
- Augmenting the system's normal operation (computation) with observation and optimization.



Uses of Introspection

- Cluster recognition
 - Identify and group closely related files
 - Graphs constructed at each client by the even handlers
 - Periodic algorithm to consumes graphs and detects clusters
- Replica management
 - Adjusts the number and location of floating replicas to service access requests efficiently
 - Create additional neighbor replicas to the overloaded ones to alleviate load
 - Eliminate replicas that have fallen into disuse
- Other uses
 - Improves routing structure, ensures availability and durability of storage fragments,...