SNIA

SNW

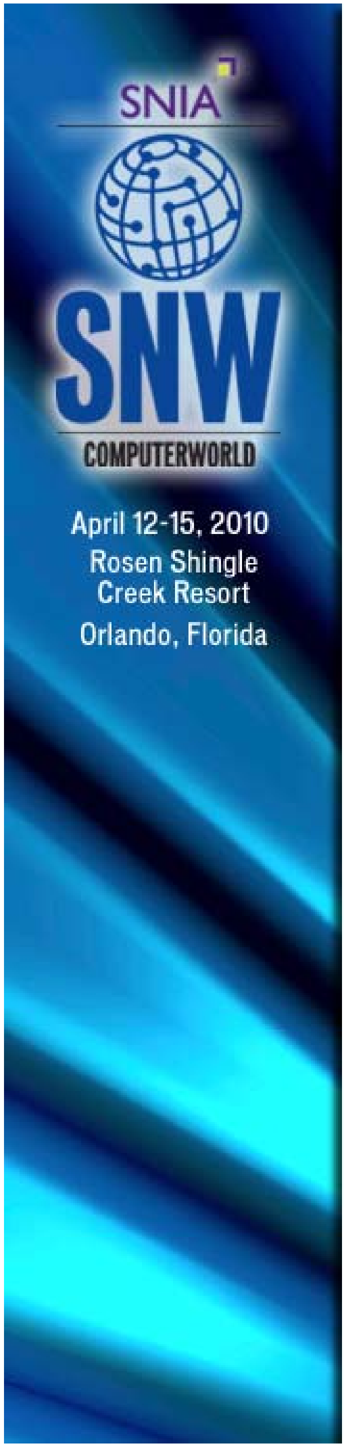
COMPUTERWORLD

April 12-15, 2010
Rosen Shingle
Creek Resort
Orlando, Florida

Building and Managing a Multi-Petabyte System

Jeff Whitehead

CTO, Zetta, Inc.



Storage Evaluation Criteria

- **Capacity** – how many TB can it hold?
- **Availability** – Is the system working all the time?
  - System failures are to be included, as well as planned maintenance (firmware upgrade, data center moves, etc)
- **Data Integrity** – Don’t lose the data, and *is the data exactly the same as it was written out?*
  - Also, what classes of situation can give rise to needing to restore from backup (data center destruction?)
  - If you do need to restore, how long does the restore take, and how current is the data
- **Cost – TCO**
  - Initial Capital Purchase
  - Data Center Space and Power
  - Support Costs
  - Cost of supporting systems (backups, network, monitoring)
  - Complexity and project opportunity cost
- **Performance & Bandwidth**
  - How many users/threads can it handle?
  - Does performance remain consistent during degraded states?
- **Scalability**
  - How large (and how hard) is it to grow to either larger capacity or higher performance?

SNIA

SNW

COMPUTERWORLD

April 12-15, 2010
Rosen Shingle
Creek Resort
Orlando, Florida

Typical Enterprise IT Storage Server

Server or Appliance providing 2-40TB of storage via NAS (NFS, CIFS) protocols

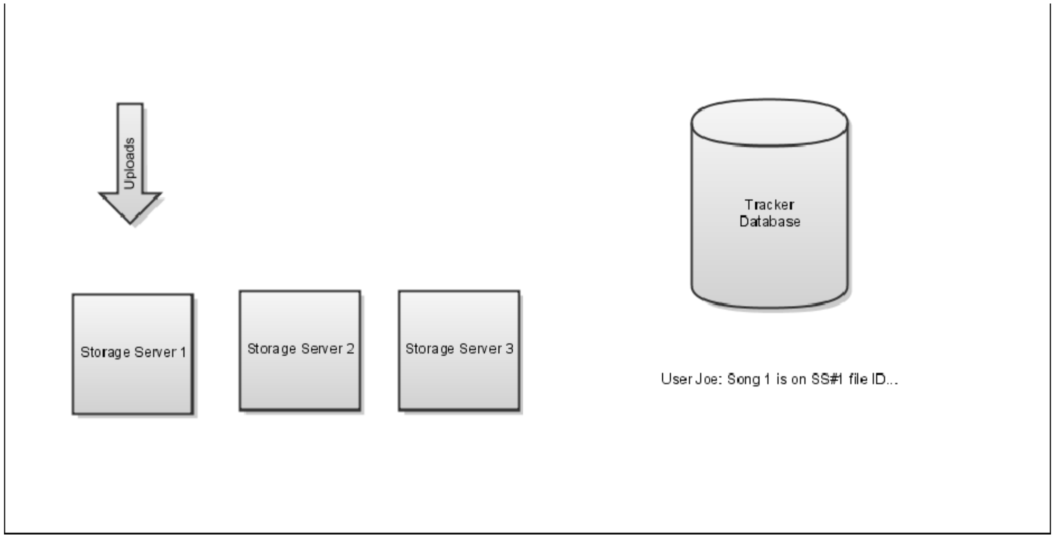
Criteria	
Primary Data Integrity	RAID encodes data across multiple disks so that if a single disk fails, data is not lost
Availability	It's just one box – perhaps with some redundant components, but not a HA system
Secondary Data Integrity	Typically backed up with 3rd party software to a 3rd party device (e.g. netbackup & tape library)
Scalability	Limited to what fits "in the box"

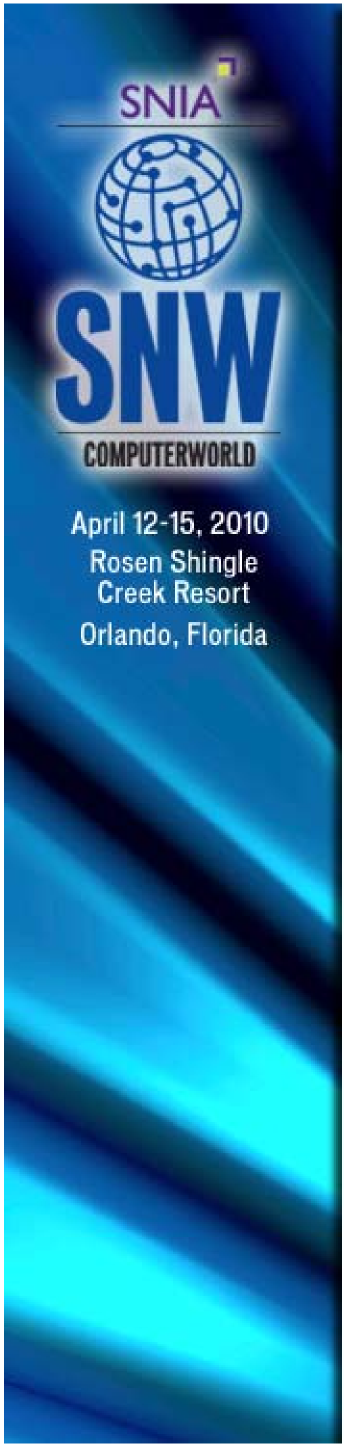
Scaling Beyond the Single Box

Application Partitioning is a powerful technique used to take a large workload and segment it such that multiple smaller computers can address the problem.

Application Partitioning applied to the typical Enterprise IT Storage Server to solve large scale storage problems gives rise to non-obvious problems.

Example: ***Large Media Archive***



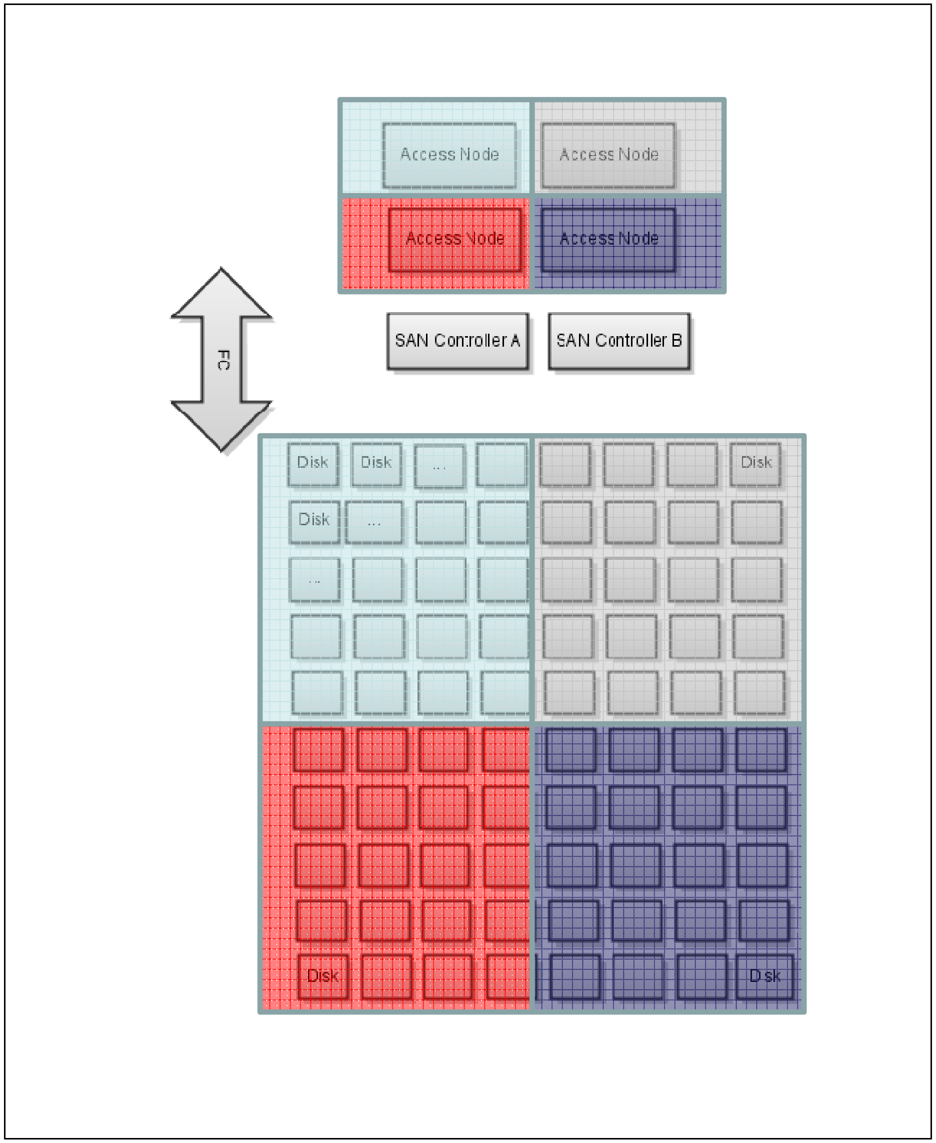


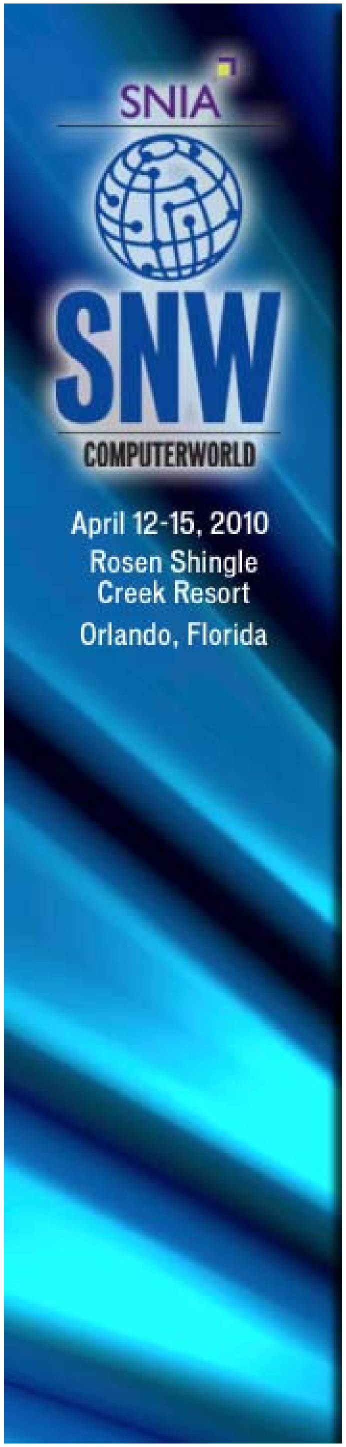
Large Media Archive Problems

Category	Issue
Availability	Since users will have files spread across all storage servers, **all storage servers (and tracker db) must be functioning for the data set to be available. In other words, the more storage server nodes, the worse overall system availability is.**
Performance	Uploads limited by tracker DB performance and write rate of a single server
Scalability	Limited by Tracker DB
Managability	Each server is a separate management domain
Cost	Optimized for footprint and initial capital; development team needed to support Tracker DB / Application and availability suffer

Large Media on SAN

Partitioned.



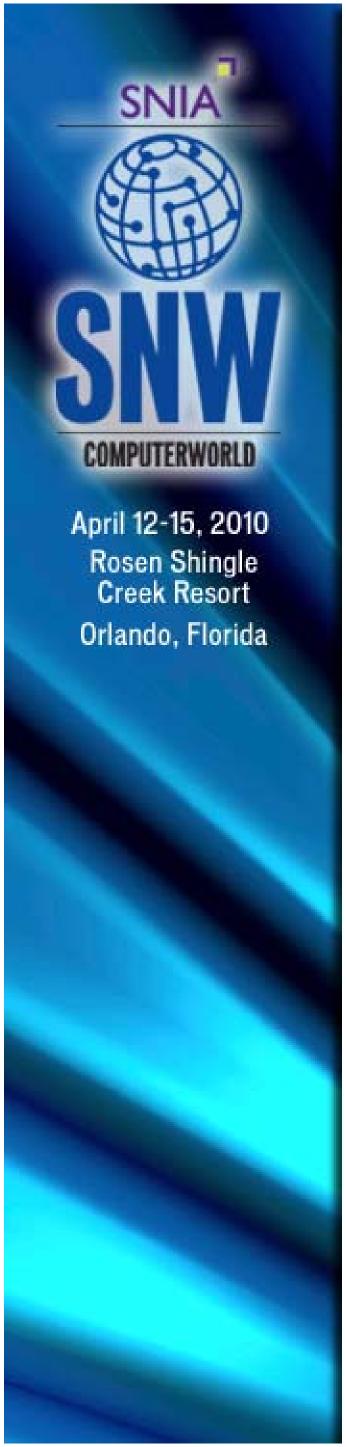


Large Media on SAN

Approach: **Go to traditional SAN architecture**
The base storage unit is much larger than a what a single server could hold, and the SAN provides raid protection, etc.

Pro:
- Higher performance per unit, larger “bucket size”
- Enterprise SANs usually have mechanisms to prevent silent data corruption

Con:
- Cost & Complexity many times that of single servers
- Biggest SANs still only about “N” PB
- If workload is distributed across multiple SAN heads then if any of them are down entire data set is down
- If workload isn’t distributed across multiple SAN heads there is inherent scalability limitation (performance & capacity)
- NAS head will be bottleneck, absent clustering or other (expensive, complicated) technologies



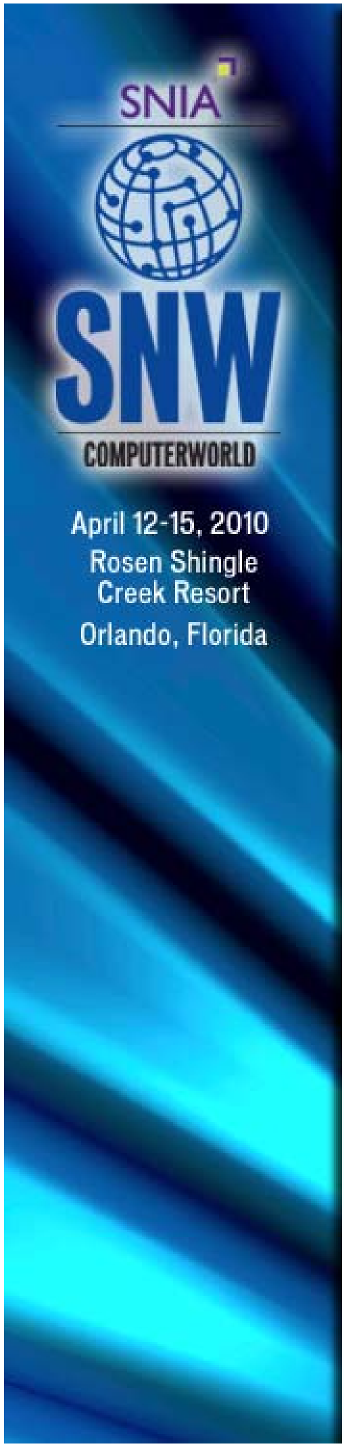
Large Media Archive– Distributed File Replication

Approach: ***Smart Software Layer*** provides ***Unified Namespace.***

Open source software such as HadoopFS, Gluster, MogileFS, and commercial softare such as Parascale, IBRIX sell the premise that you can aggregate the standard file server.

Each package has it’s own issues, however:

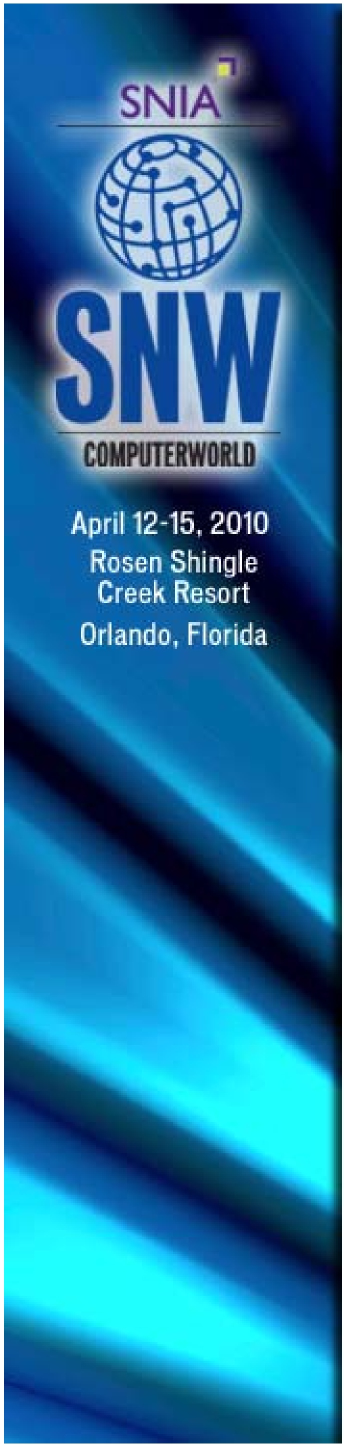
- Data Integrity (and often availability) is assumed to be provided at a lower layer (ie, a raid card in each server, or a SAN below the server) **or** replicates files across servers (better for availability and read performance, worse for cost / opex)
- Software complexity is significant enough to require **dedicated personnel** with **specialized skills**
- Consistency models / API may not be compatible with enterprise applications



Challenges of Petabyte Scale

- Performance
- Data Integrity / Data Permanence
    - Bit Error Rate / Silent Data Corruption Issues
    - Mean Time To Data Loss
- Availability
    - More nodes means more problems
    - More disks == more disk failures
- Data Migration
    - Sheer volume of data poses migraton challenges, and ensu errors do not get included

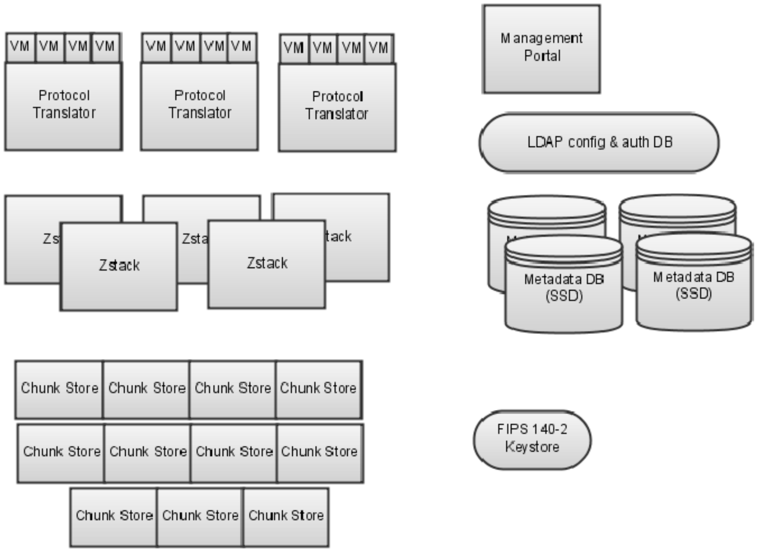
- Backups
- Data Center Power and Cooling
- Capacity Management



Zetta Design Objectives

- Data Integrity
- Strong consistency (read-after-write, respect sync()), POSIX Compatible
- Multi Tenant (virtualize IO performance as well as footprint)
- Tiered Design, with independent Horizontal Scalability (ie thin provisioning at all levels)
- Commodity Hardware Components
- Continuous Availability (failures, releases, scale out, moves, always consistent on disk)
- Ethernet/IP backend (as opposed to Infiniband / FC)
- Strong Technical and Procedural Security

Zetta Implementation



ZettaFS Distributed File System

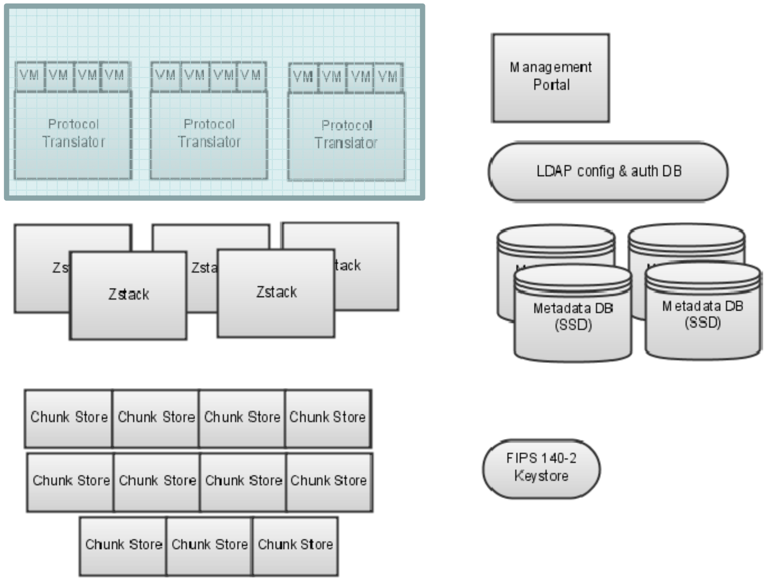
All elements implemented as network services

Centralized Metadata, holds ‘inode’ equivalents (on SSD)

10Gbps low latency ethernet

Basic unit of storage is a “chunk,” striped across discrete nodes

Zetta Implementation



Protocol Translator ==“NAS Head”

Xen VM-ZettaFS appears as local file system

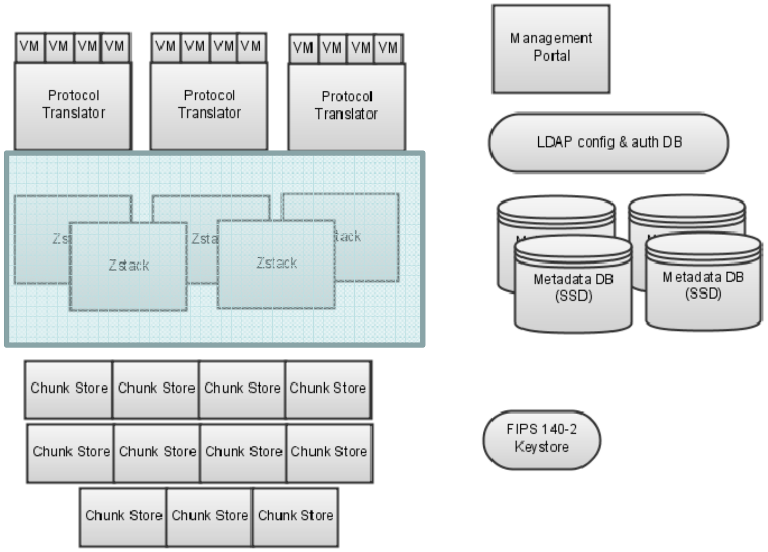
Pulls config and authentication creds from LDAP

QoS management

Caching

Reference Synchronization

Zetta Implementation



Zstack
==“RAID Controller”

Reed-solomon chunk encoding / recovery

Write cache (local SSD & consensus quorum protocol)

Metadata management

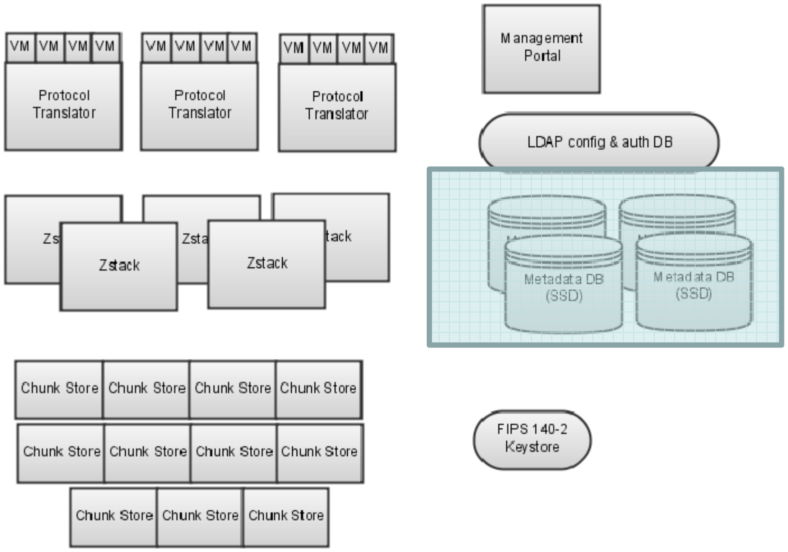
Lock Manager

Replication

Chunk placement rebalancing/optimization

SNIA
SNW
COMPUTERWORLD
April 12-15, 2010
Rosen Shingle Creek Resort
Orlando, Florida

Zetta Implementation



Metadata DB

N+3 protection

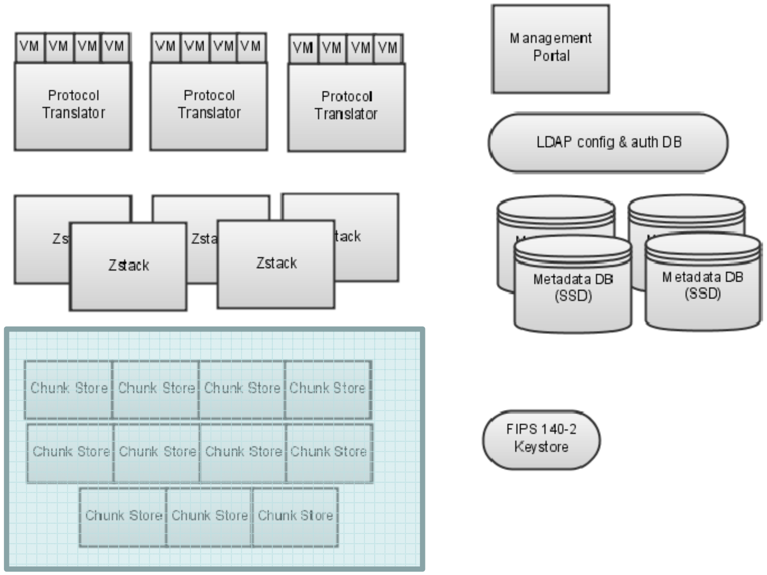
Volume -> file maps

File -> chunk maps

Raid stripe maps

Scalable / partitioned (except for filenames in a given volume currently constrained to one instance)

Zetta Implementation



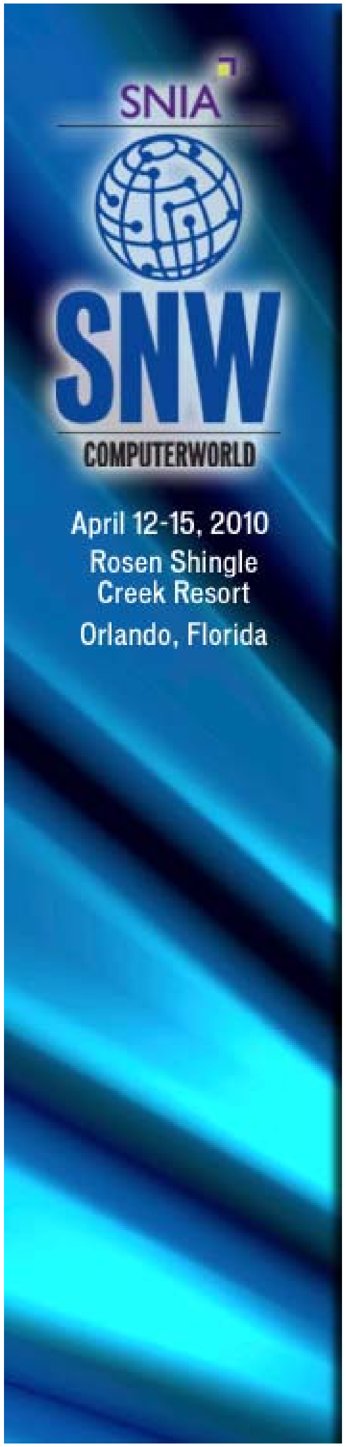
Chunk Stores
== "Disks"

Caching Layer

Encryption / Decryption – 100% on-disk encryption

Hash validation on read

Background hash validation



Other Key Features

- Clustered mount capabilities
- Entire system designed for high concurrency / throughput (as opposed to single transaction latency)
- End to end data validation
- Typical enterprise feature set: snapshots, replication, etc
- Undo/Redo filesystem capabilities (CDP)
- Site to Site replication (geodiverse data protection)
- **Zetta is the Ideal Architecture to meet Petabyte Scale Challenges**

Other Key Features

Thank you!

Jeff Whitehead

jw@zetta.net

http://www.zetta.net