

Resiliency at Scale in the Distributed Storage Cloud

Alma Riska
Cloud Infrastructure Group
EMC Corporation

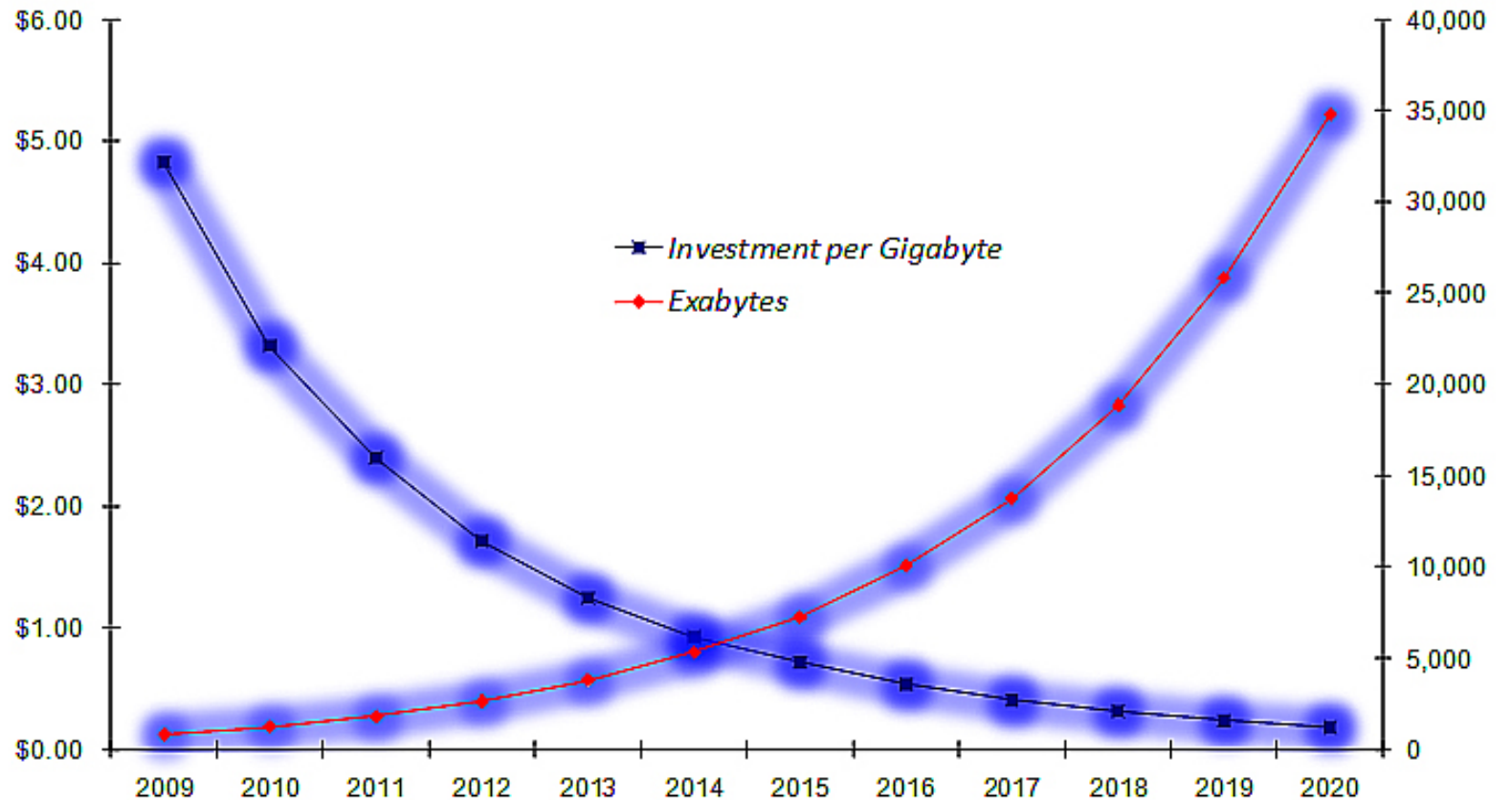
In collaboration with many at CIG
Erik Riedel, Matthew Geddes, Kartik Vandra

Wide topic but this talk will focus on

- ❑ Motivation
- ❑ Definition of resiliency in cloud storage
- ❑ Failures and challenges
- ❑ Redundancy schemes
 - ❑ Erasure coding
- ❑ Distribution of redundancy
 - ❑ Key to resilience at scale

Digital Content Creation & Investment

Investment per Gigabyte



Source: IDC Digital Universe Study, sponsored by EMC, May 2010

IT Infrastructure

Traditional Storage



	Today	Tomorrow
Workload	Transactional	Content
Application interaction	Mount point	API
Data management	Admin-driven/ manual	App-driven / policy automated
Data protection / Disaster protection	Additional best of breed products	Built-in data services
Storage scaling and management	File system limits / pre-planning	Unlimited namespace / dynamic
Infrastructure model	Single tenant / storage isolation	Native multi- tenancy throughout the stack

Active Object Storage



Changing View

❑ Cloud storage

- ❑ Goal: meeting SLOs at low(est) cost
- ❑ Mask lower hardware reliability with clever software
- ❑ More limited range of performance/availability (aim for “good enough” for a majority of apps, with some variance)
- ❑ Focus on OpEx (operating costs, human costs)

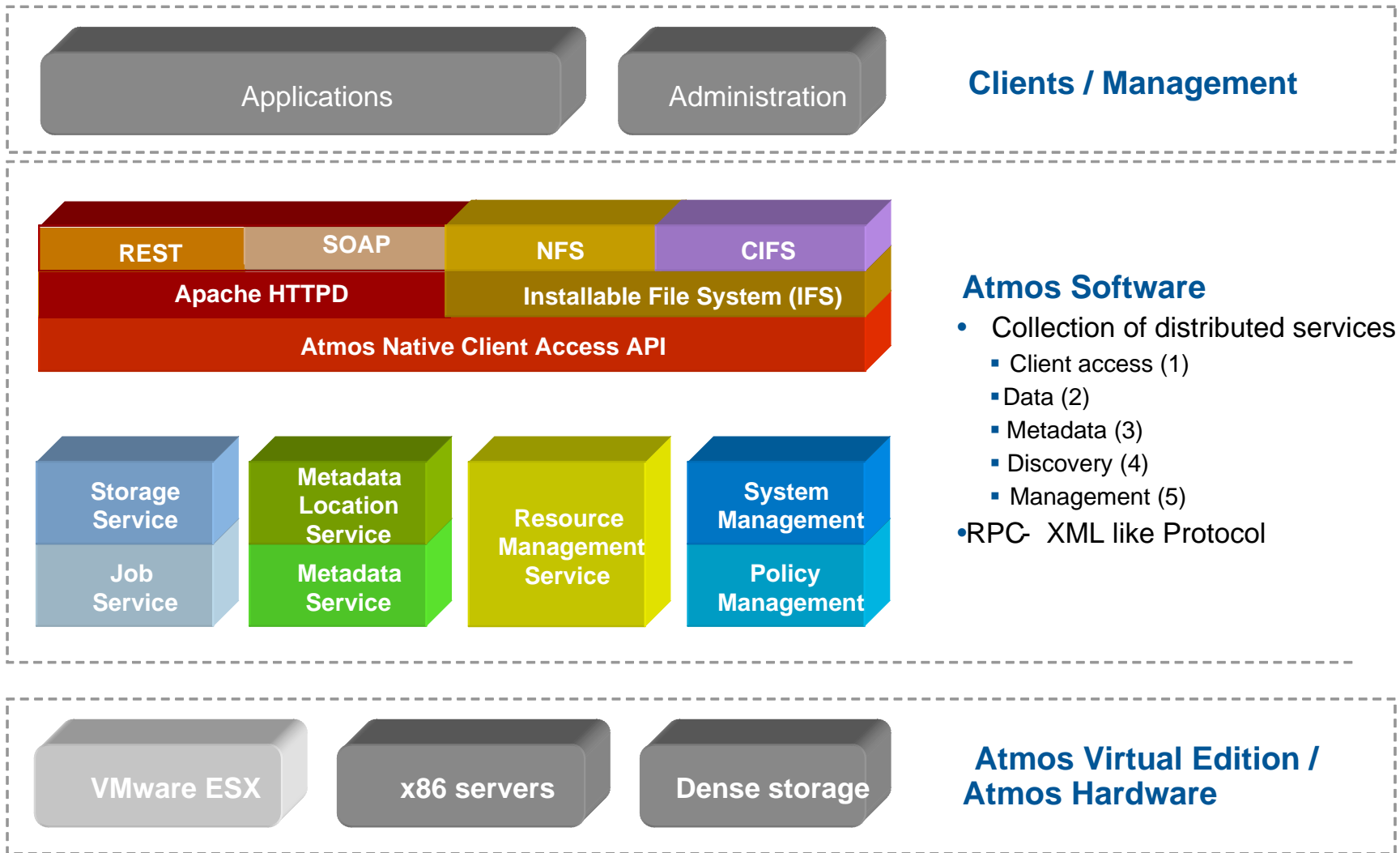
❑ Enterprise storage

- ❑ Goal: meeting SLOs, tolerate high(er) cost
- ❑ Expect high hardware reliability
- ❑ Wide range of performance/availability trade-offs (optimize for e.g. OLTP all the way to long-term archive)
- ❑ Focus on CapEx (purchase costs, human costs)

Reasonable cost investments:

- serviceable components where sensible (e.g. disks/power supplies)
- leverage HA components where sensible (e.g. rack power, connectivity)

Example: Atmos



Atmos Software

- Collection of distributed services
 - Client access (1)
 - Data (2)
 - Metadata (3)
 - Discovery (4)
 - Management (5)
- RPC- XML like Protocol

Scaled-out Systems

- ❑ Large amount of hardware
 - ❑ Thousands of disks
 - ❑ Tens to hundreds of servers
 - ❑ Significant amount of networking
- ❑ Wide range of applications
 - ❑ Driven by virtualized services
- ❑ Up-to millions of users

- ❑ Different from traditional systems
 - ❑ Amount of components
 - ❑ Weight components have on overall system



Storage Requirements

- ❑ Store **COST-EFFECTIVELY** massive amount of data
 - ❑ (Tens) PetaBytes
- ❑ Highly available
 - ❑ Minimum down time
- ❑ Reliably stored
 - ❑ Beyond the traditional 5 nines
- ❑ Ubiquitous access
 - ❑ Access from anywhere
- ❑ Consistent performance



Storage Resiliency in the Cloud

- ❑ Resiliency in this talk is
 - ❑ Data Reliability
 - ❑ Data is stored persistently in a device(s)
 - ❑ Data Availability
 - ❑ Devices that store the data are accessible
 - ❑ Data Accessibility
 - ❑ Retrieved data is returned to the user upon request
 - ❑ Metadata

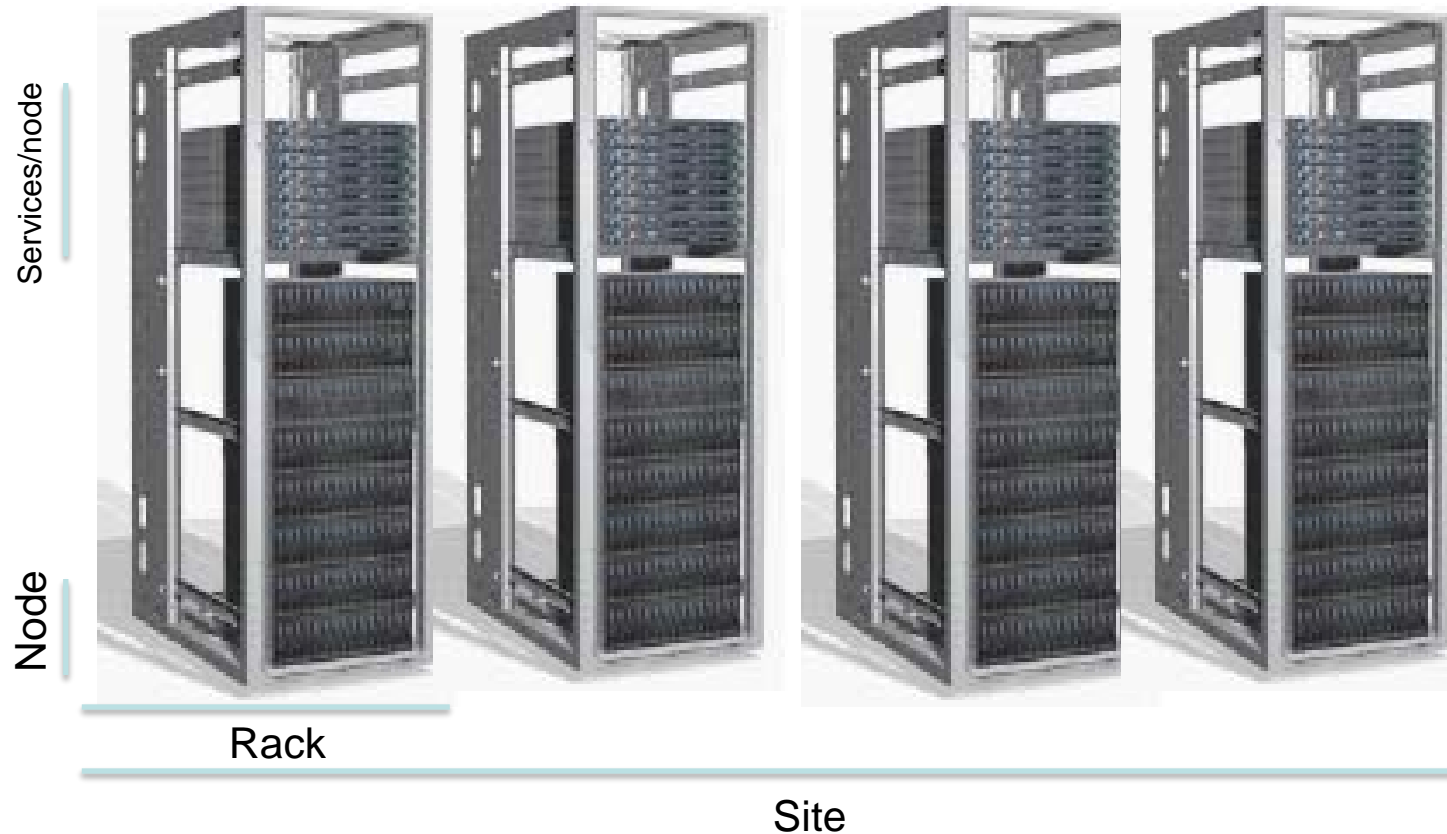
Meeting Cloud Storage Requirements

- ❑ Cost and capacity
 - ❑ Large capacity SATA HDD (2TB or more)
 - ❑ Directly attached HDDs

- ❑ Resiliency = Availability/Reliability/Accessibility
 - ❑ Redundancy
 - ❑ Hardware & Software services
 - ❑ Architecture of the system
 - ❑ How hardware resources are distributed

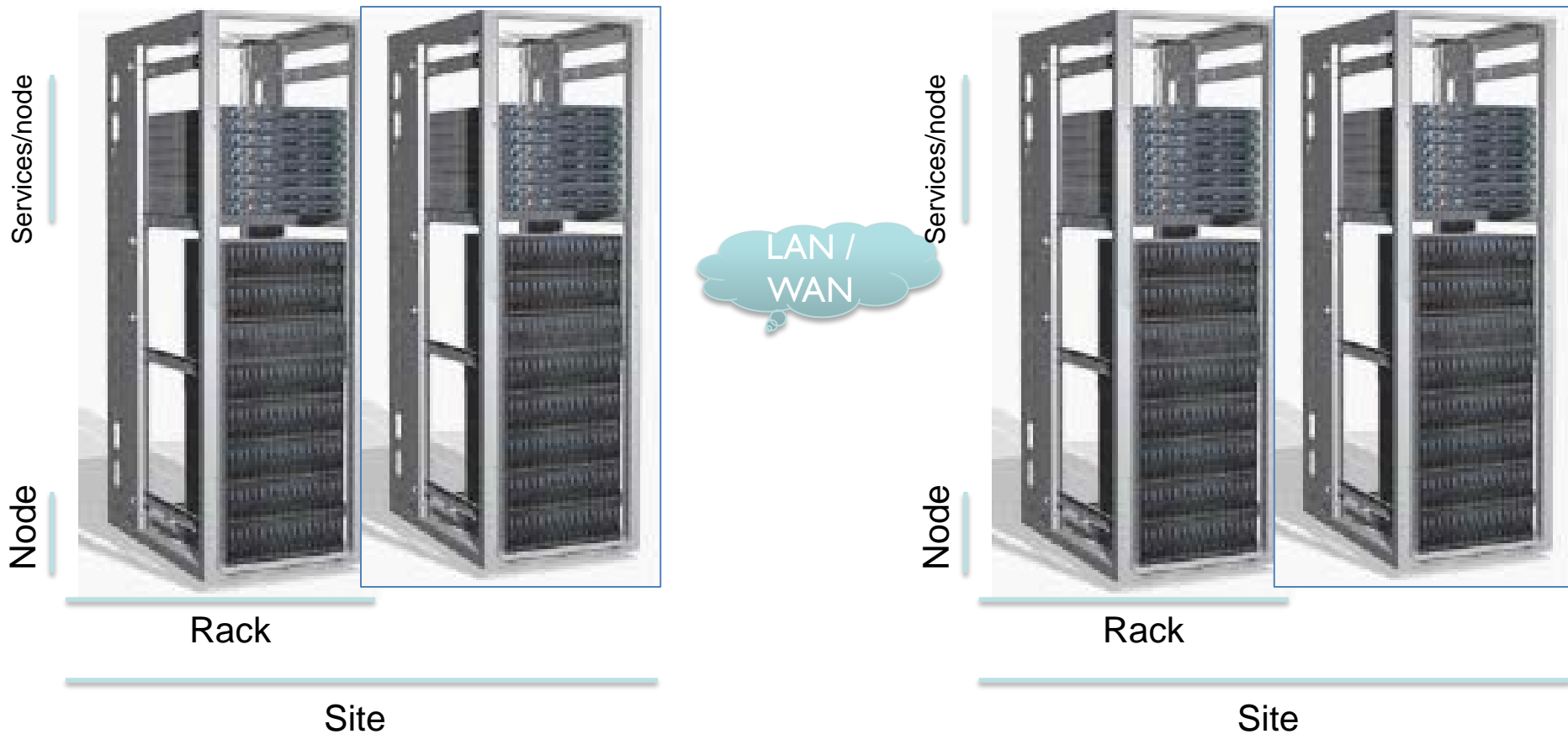
Cloud Storage Architecture

- 1 Pbyte of cloud storage
 - 480 2TB HDDs, 32 nodes, 1 site



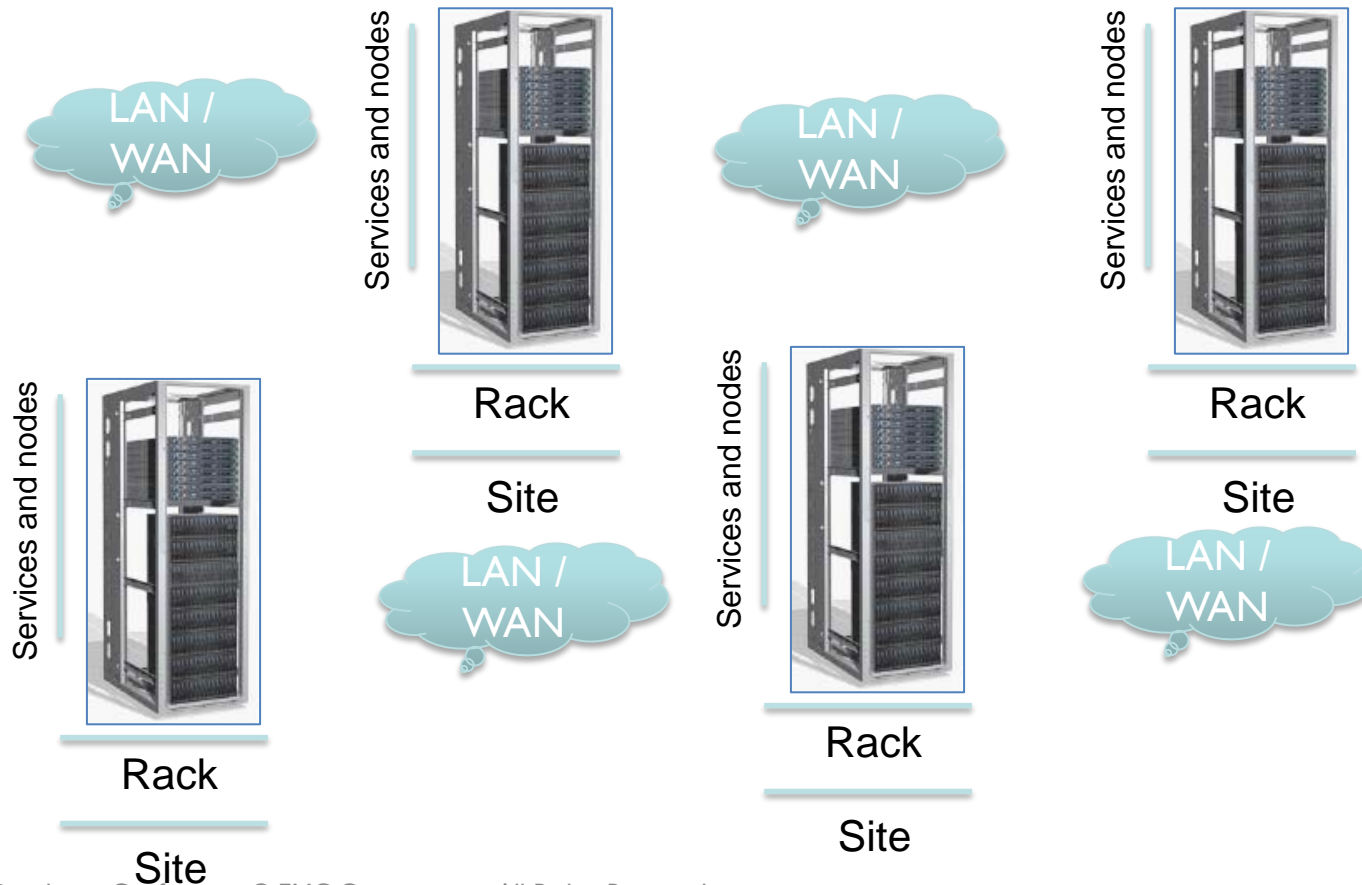
Cloud Storage Architecture

- 1 Pbyte of cloud storage
 - 480 2TB HDDs, 32 nodes, 2 site



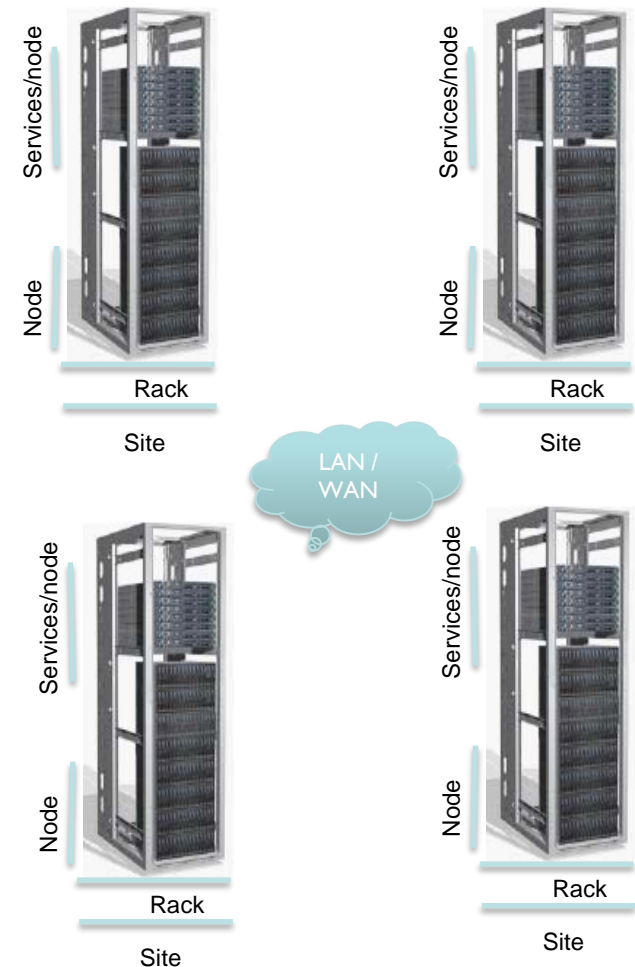
Cloud Storage Architecture

- 1 Pbyte of cloud storage
 - 480 2TB HDDs, 32 nodes, 4 site



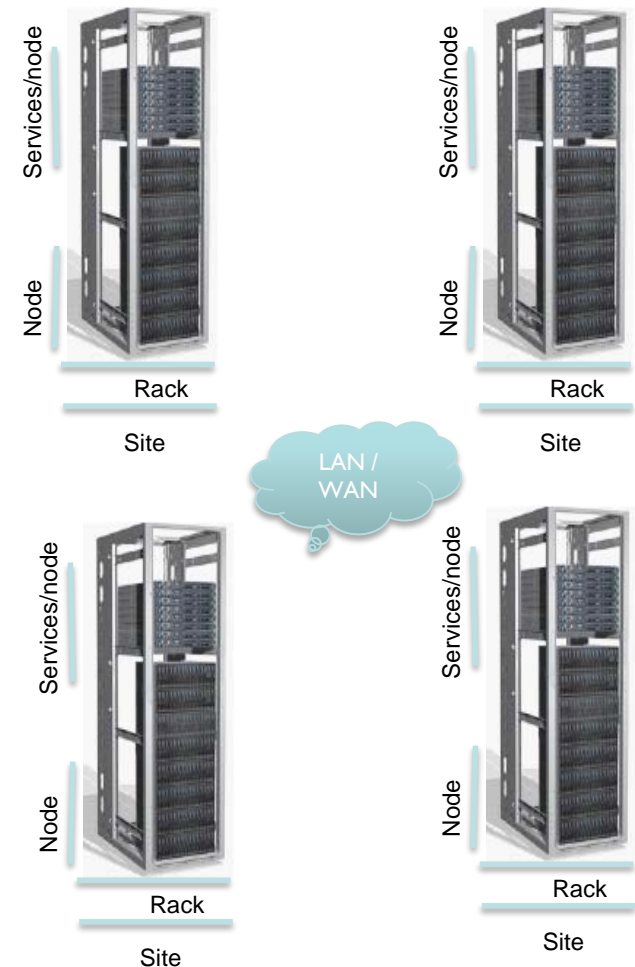
Failures in the Cloud

- ❑ Wide range
 - ❑ Data devices (HDDs)
 - ❑ Other components
 - ❑ Hardware
 - ❑ Network
 - ❑ Power outages
 - ❑ Cooling outages
 - ❑ Software
 - ❑ Drivers
 - ❑ Services (metadata)



Impact of Failures in the Cloud

- ❑ Data Reliability
 - ❑ Disk failures – directly
 - ❑ Other failures – indirectly
 - ❑ Time off-line
- ❑ Data Availability
 - ❑ All failures
- ❑ Data Accessibility
 - ❑ Service failures - directly
 - ❑ Metadata
 - ❑ Other failures - indirectly



Criticality of Failures in the Cloud

- ❑ Large scale, e.g node failure
- ❑ Make unavailable large amount of data simultaneously
- ❑ Systems have more components hence more failures
- ❑ System needs to consider high component unavailability

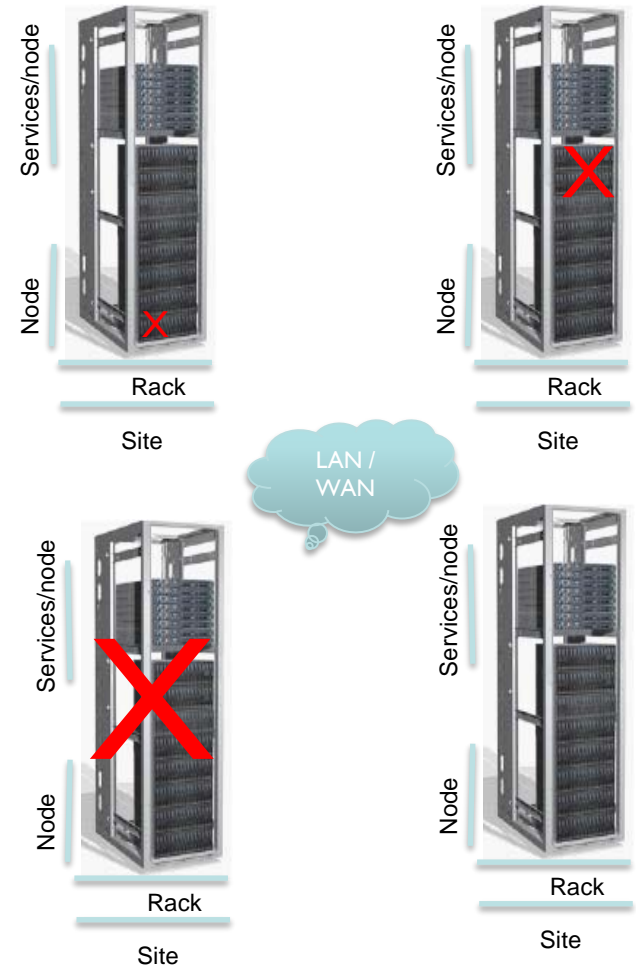
- ❑ Traditional storage experiences the similar failures
 - ❑ But at a different scale
 - ❑ E.g. dependability on network robustness

Challenges of Handling Failures in the Cloud

- ❑ Accurate identification of failures
 - ❑ Many failures have similar symptoms
 - ❑ Disk unreachable
 - ❑ But different root cause
 - ❑ Failure of disk, controller, power, network
- ❑ Isolation of failures
 - ❑ Avoid propagation or escalation of failures
 - ❑ A node failure becomes a multi-node unavailability
- ❑ Timely detection of failures
 - ❑ In a large system failures may go undetected

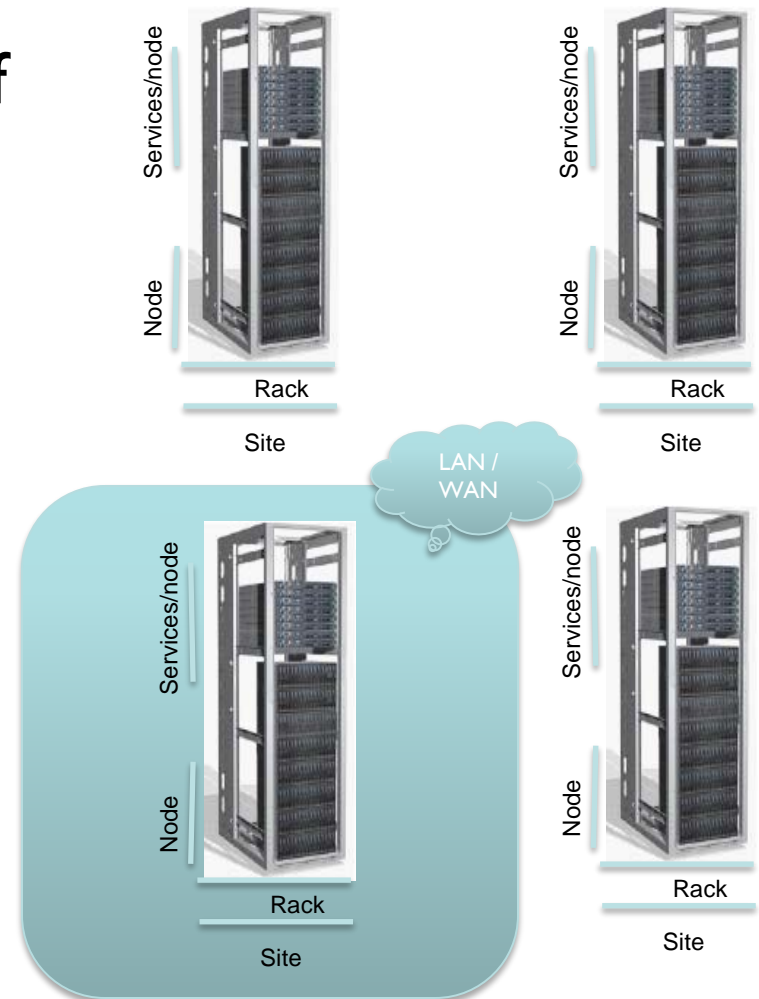
Fault Tolerance in Cloud Storage

- Transparency toward failures of
 - components
 - services
- Transparency varies by system



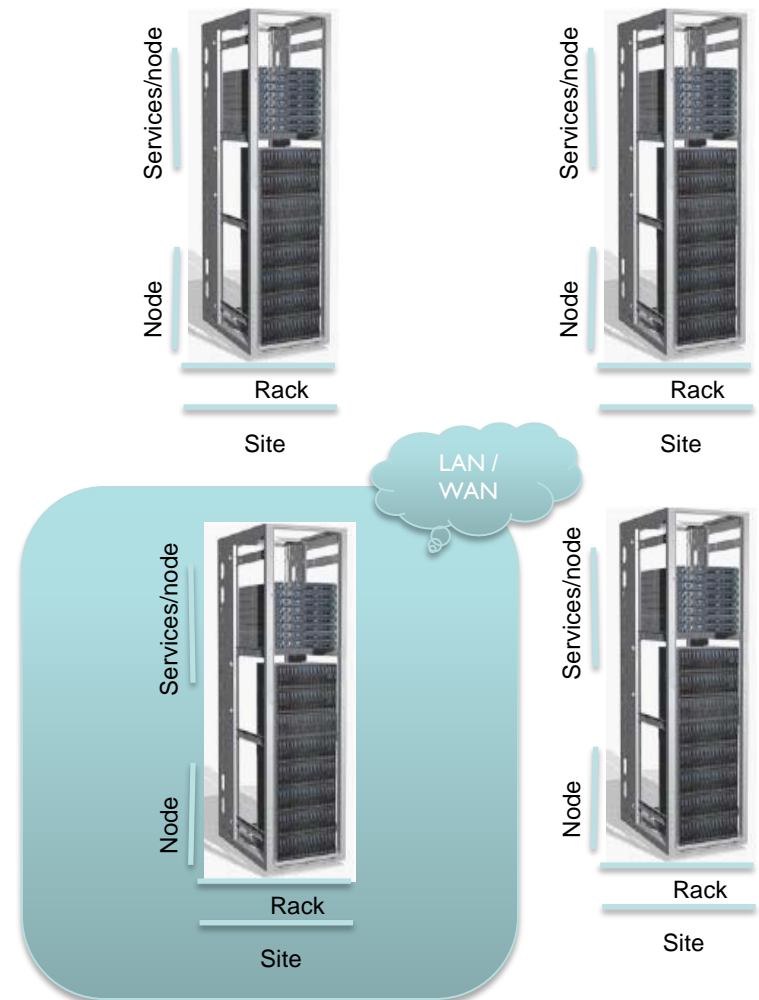
Fault Tolerance in Cloud Storage

- ❑ Transparency toward failures of
 - ❑ components
 - ❑ services
- ❑ Transparency varies by system
- ❑ Fault tolerance goal determines
 - ❑ Fault Domains



Fault Domains

- ❑ The hierarchy of the set of resources whose failure can be tolerated in a system
- ❑ Example: tolerate failures at
 - ❑ Disk-level
 - ❑ Node-level
 - ❑ Rack-level
 - ❑ Site-level
- ❑ Determines distribution
 - ❑ Architecture
 - ❑ Data / Services



Fault Tolerance and Redundancy

- ❑ Fault tolerance is primarily achieved via redundancy
 - ❑ More hardware and software than needed
- ❑ Achieving a fault tolerant goal depends
 - ❑ Amount of redundancy
 - ❑ Overhead in the system (cost and management)
 - ❑ Distribution of redundancy
- ❑ Examples
 - ❑ Redundancy of data (capacity overhead)
 - ❑ Redundancy of components
 - ❑ Power supplies, switches, network connections (cost)

Fault Tolerance and Redundancy

- ❑ Resilience at scale depends on
 - ❑ Handling failures in a variety of system components often not handled by the same owners
 - ❑ Eg. Network services (WAN)
 - ❑ Eg. Power/cooling of the center
 - ❑ Virtualization of hardware may hide lower-level details from systems like Atmos
 - ❑ Distribution of redundancy
 - ❑ Between data and other components
 - ❑ Rigorous definition of fault domains

Fault Tolerance and Redundancy

- ❑ Storage fault tolerance achieved mainly via data redundancy
 - ❑ Traditionally parity (RAID)
 - ❑ Often in the cloud is replication
 - ❑ Erasure coding (emerging)
- ❑ Storage systems deploy often pro-active measures
 - ❑ Monitoring/analysis/prediction of system's health
 - ❑ HDD SMART
 - ❑ Background scanning
 - ❑ Detection of silent failures
 - ❑ Verification of data content

Fault Tolerance and Data Replication

- ❑ Replicate data (including metadata) up to 3 times
- ❑ Pros
 - ❑ High reliability
 - ❑ High availability
 - ❑ Good performance and accessibility
 - ❑ Easy to implement
- ❑ Cons
 - ❑ High capacity overhead
 - ❑ Up to 200% in a 3-way replication

Replication in Scale-Out Cloud Storage

- ❑ Average case in a cloud storage system
 - ❑ Pbytes of usable capacity
 - ❑ Tens of Pbytes of physical capacity
- ❑ Does not scale with regard to
 - ❑ Cost
 - ❑ Resilience
 - ❑ With only 3 replicas resiliency goals are not always met

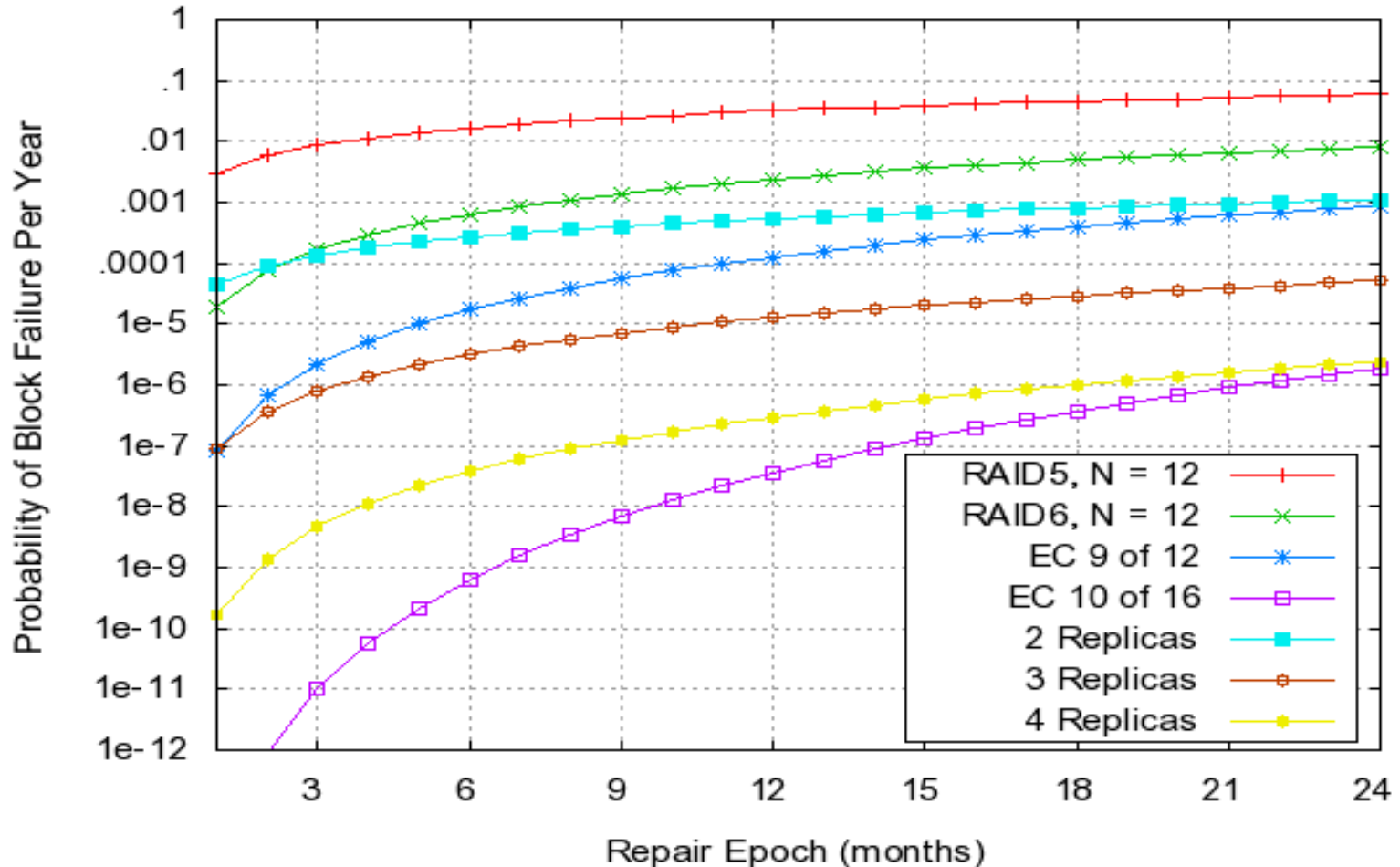
Erasure Coding Alternative to Replication

- ❑ Generalization of parity-based fault tolerance
 - ❑ RAID schemes
 - ❑ Replication is a special case
- ❑ Out of n data fragments
 - ❑ m are actual data
 - ❑ k are additional codes ($n=m+k$)
 - ❑ k missing fragments of data can be tolerated
 - ❑ Capacity overhead is k/m
 - ❑ Code is referred to as m/n code
 - ❑ RAID5 (4+1) is an 4/5 code

- ❑ Capacity overhead k/m
 - ❑ Overhead reduces as n increases
 - ❑ Same protection
- ❑ Complexity – computational and management
 - ❑ Increases as n increases
- ❑ Trade-off between
 - ❑ Protection
 - ❑ Complexity
 - ❑ Overhead
- ❑ Common codes are 9/12, 10/16, 6/8

EC vs. Other redundancy schemes

Fractional Block Loss Per Year (RAID vs. EC, MTTF: 375k hrs, destage)







- ❑ Data fragments distributed based on the system fault domains
- ❑ Placement of these fragments is crucial
 - ❑ Round-robin placement ensures uniform distribution of fragments
 - ❑ Assumed in the previous plot
- ❑ Placement of data fragments depends on
 - ❑ User requirements with regard to
 - ❑ Performance
 - ❑ Priorities

EC data placement in the Cloud

- ❑ Develop strategies to determine
 - ❑ Redundancy level in a system
 - ❑ Tolerance goal
 - ❑ Protection scheme (EC or replication)
 - ❑ System architecture
 - ❑ Number of sites, nodes etc.

Example

	2-way replication	3-way replication	9/12 Code	10/16 Code
Overhead	100%	200%	33%	60%
Site Failure Tolerance				

❑ Erasure Coding with larger number of fragments and low overhead makes sense for systems with large node and site count

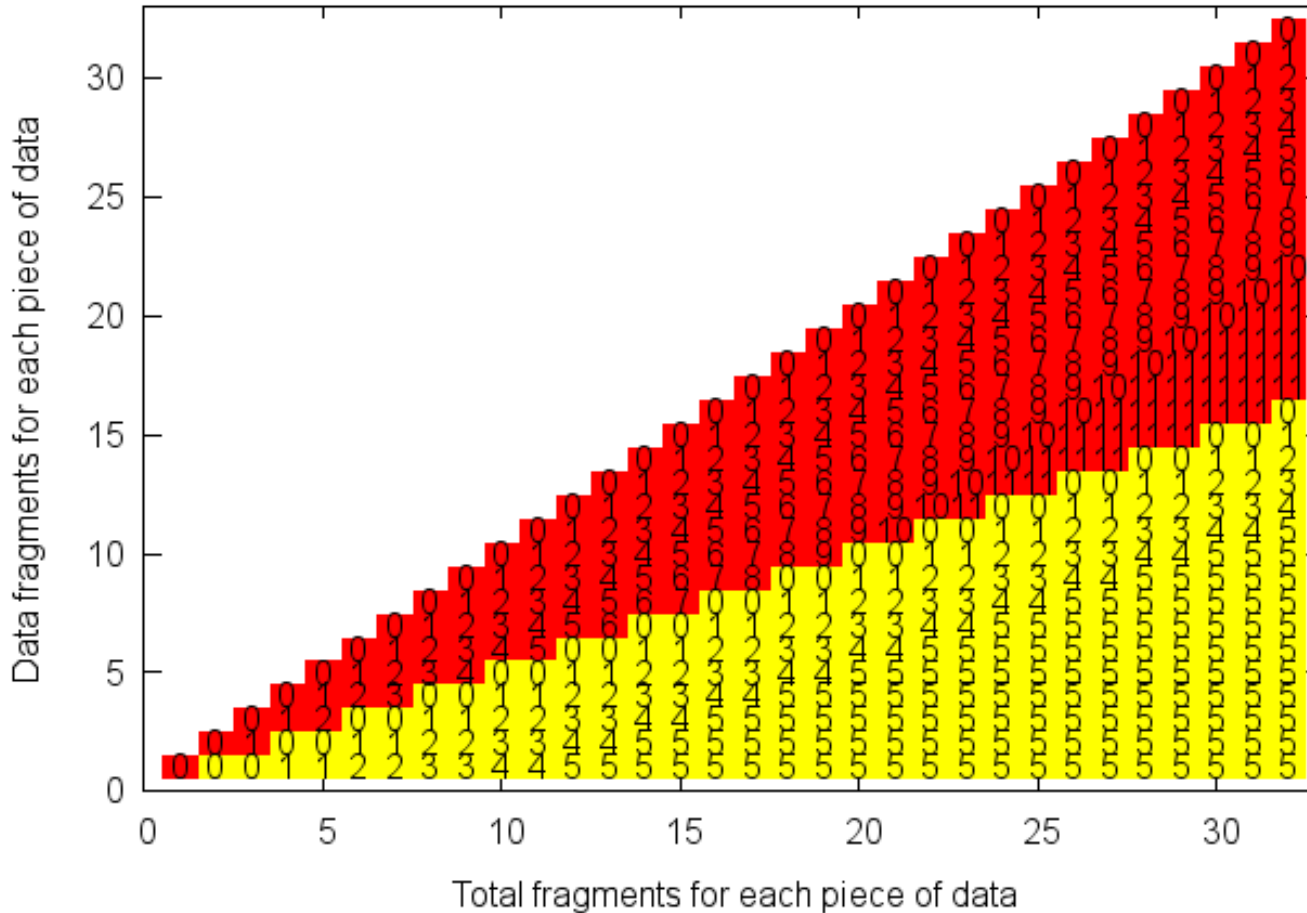
❑ Clients that have already multiple data centers can edeploy the system cost-effectively over multiple sites

EC data placement in the Cloud

- ❑ Homogeneous geographically distributed sites
 - ❑ Equal number of nodes and disks
 - ❑ Equal network delays between any pair of sites
 - ❑ Equal data priority
- ❑ Round robin distribution of the fragments across
 - ❑ Sites / nodes / disks
- ❑ Failures on disks / nodes / sites
 - ❑ Data unavailability or loss of redundancy

Failure Tolerance in 2 Site System

Site Failure Tolerance in a 2 site System with 6 nodes per site



In a two site system there is only one site failure tolerance

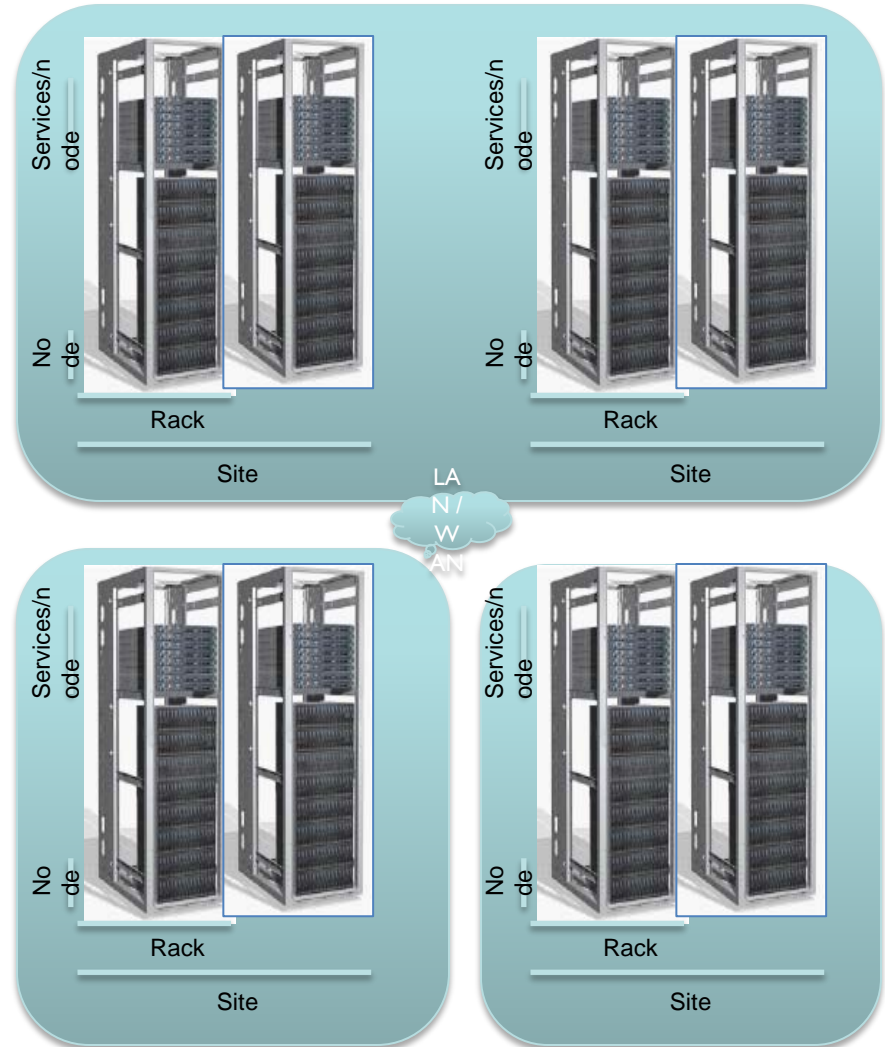
Each site has 6 nodes available

The numbers inside each (x,y) tuple are the number of nodes tolerated in addition to the sites tolerated

- ❑ As system evolve their resources become heterogeneous
 - ❑ Different node or site sizes
 - ❑ Different network bandwidth
 - ❑ Different data priority
 - ❑ location origin
- ❑ In such a case
 - ❑ Uniformity of data distribution not a requirement
 - ❑ The above factors (including performance) should determine data fragment placement

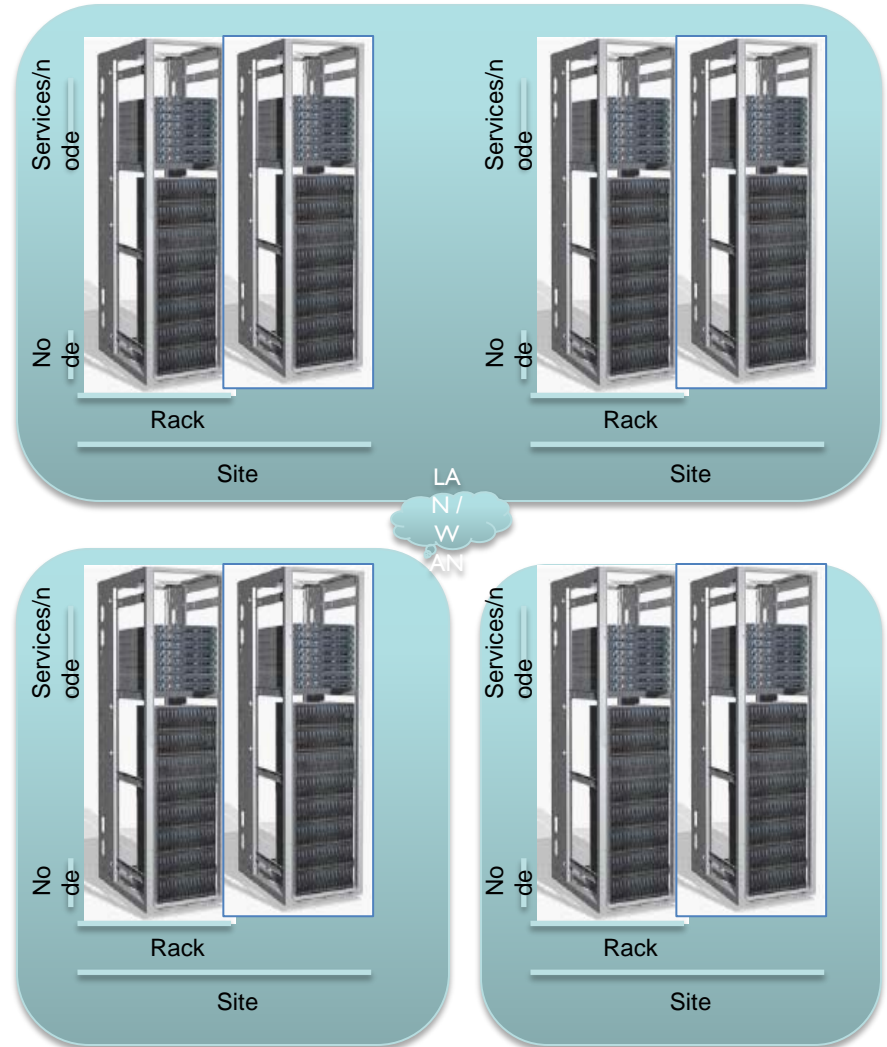
Abstraction of Heterogeneous Cloud Storage

- Group components based on affinity criteria
 - Network bandwidth
- Create homogeneous sub-cluster
- Determine redundancy for each sub-cluster
- Handle each sub-cluster independently
- Combine outcome for system-wide placement



Abstraction of Heterogeneous Cloud Storage - Example

- ❑ Two sites are close because they share faster network
- ❑ Data can be placed in any of the nodes in both sites and retrieving it will not suffer extra network delay
- ❑ If an 9/12 redundancy scheme is used
 - ❑ Uniform distribution of fragments would be detrimental to performance
 - ❑ If data primary location is the two-site subcluster the 9 data fragments can be placed in the two sites and the 3 codes in the other remote sites
 - ❑ Accessing the data is not affected by network bandwidth



Atmos – Policy-based Cloud Architecture

- ❑ Flexible policy settings for grouping resources and isolating applications /tenants
- ❑ Easily managing a large heterogeneous system
 - ❑ Hybrid protection schemes that combine multiple replication schemes
 - ❑ E.g., a two replications policy where
 - ❑ First replica is the original data (stored in the closest site to tenant)
 - ❑ Second replica is a 9/12 EC scheme that distributes the data in the rest of the sites for resilience

Fault Tolerance via proactive monitoring

- ❑ Monitoring the health of devices and services
 - ❑ Logging events
- ❑ Taking corrective measures before failures happen
 - ❑ Strengthen the resilience
 - ❑ Address without the redundancy affected by failure
- ❑ Example
 - ❑ Use of SMART logs to determine health of drives
 - ❑ Replace HDDs that are about to fail rather than failed

Fault Tolerance via proactive monitoring

- ❑ Verify in the background the validity of data, services and health of hardware
 - ❑ Critical aspect of resiliency in the cloud
- ❑ System are large and some portions maybe idle for extended periods of time
 - ❑ Failures and issues may go undetected
 - ❑ Ensure timely failure detection
 - ❑ Improve resilience for a given amount of redundancy

Conclusions

- ❑ Resilience at scale = reliability+availability+accessibility
- ❑ Wide range of large scale failures
- ❑ Redundancy aids resiliency at scale
- ❑ Erasure coding – efficient scaling of resiliency
- ❑ Proactive measures to ensure resiliency at scale