Storage Design Patterns For Cloud-Native Apps

Anthony Ferrario, Pure Storage
We Live in a World of

Rapidly Changing Applications

Monolithic
- Client / Server
- Tight Clusters
- 1980-90s

Virtualized
- x86 + VIRTUALIZATION
- Flexible Consolidation
- 2000s

Cloud
- Scale-Out
- Stateless Cattle
- 2010s
Container Adoption is Exploding

Dev/Test  |  Web Services  |  Modern Database  |  Enterprise Apps

Stateless  →  Persistent
The 451 Take

'Cloud native' is a bit of a 'Lego' market right now with all kinds of building blocks being created by all kinds of interested groups and individual entities. What's clear, however, is that stacks using these blocks will emerge, and those not paying attention to the opportunities and challenges risk becoming irrelevant. All of the models place the developer at the center: CNCF believes the number of developers could increase from about 20 million globally today to 100 million in 10 years given the demand for ubiquitous service delivery. More levels of abstraction mean the infrastructure becomes 'invisible' but not less important; in fact, the paradox is that the more important the role infrastructure plays in our lives, the more important it becomes to shield users from having to directly interact with or even consider it.
Environment
Some Assumptions

• You’re looking at how to build the next generation of your infrastructure
• Your data center is still relevant
• This is meant to be descriptive, not prescriptive
• Please come talk to us if you have questions
Underlying Infrastructure

Physical Cluster

Virtualization

openstack™

Physical Cloud

IaaS
Compute Orchestration
What Storage Needs to Provide
Modern Environments Want to Consume

STORAGE-AS-A-SERVICE

SIMPLE • AUTOMATED • ELASTIC • RELIABLE • SHARED
Environment Connectivity
Characteristics to Consider

It’s Not Just About IOPs Anymore

Availability
• What are your business needs? How much resilience does your solution have?

Scalability
• What do you need today? Can your solution support your growth?

Simplicity
• How much work is it to manage? Scale? Recover from failures?

Performance
• Do you need low latency? High throughput?

Cost Economics
• How much will it cost you to build your solution? What about maintaining it?
1 - SDS
Software-Defined Storage
Leverage the Drives in Your Servers
SDS Review

Pros
- Easy to Get Started
- Support Cloud-Native Patterns
- Low Purchase Cost
- Strong Ecosystems
- High Availability

Cons
- Complexity With Scale
- Significant Ongoing Maintenance
- Inconsistent Performance
- Questionable Failure States
2 - Top of Rack
Top of Rack Storage V1
Take Advantage of Scale-Out App Design

Pod 1
Pod 2
Pod 3
Pod 4
Top of Rack Storage V2
Control Scale and Cost
Top of Rack Recap

**Pros**
- Enterprise Storage features
- Consistent Performance
- Familiar Infrastructure
- Support Cloud-Native Patterns
- Massive Scalability

**Cons**
- Higher Cost
- Relatively Complex at Small scale
- Can’t Run on Your Laptop
3 - Centralized
Centralized Storage V1
Utilize Enterprise Storage Appliance to Support Your Cloud

Multi-Zone Storage Connectivity
Centralized Storage V2
Support Enterprise Needs with HA storage

Storage available to all compute
Replication link
## Centralized Storage Recap

<table>
<thead>
<tr>
<th>Pros</th>
<th>Cons</th>
</tr>
</thead>
<tbody>
<tr>
<td>- Enterprise Storage features</td>
<td>- Limited Scalability</td>
</tr>
<tr>
<td>- Consistent Performance</td>
<td>- Requires Higher-level abstraction</td>
</tr>
<tr>
<td>- Familiar Infrastructure</td>
<td>- Higher Cost</td>
</tr>
<tr>
<td>- Leverage Appliance Data Reduction</td>
<td>- Can’t Run on Your Laptop</td>
</tr>
<tr>
<td>- Leave Apps Unchanged</td>
<td></td>
</tr>
</tbody>
</table>
How Pure Helps
Pure Service Orchestrator
Delivering Container Storage-as-a-Service

SMART PROVISIONING
- Automated container storage, on demand
- Policy-based provisioning
- Full integrated with Kubernetes, Docker

ELASTIC SCALING
- Scale across multiple arrays seamlessly
- File and block on shared infrastructure
- Add new storage with a single command

TRANSPARENT RECOVERY
- Automatic failover for unhindered service
- Self heals to ensure data access integrity
- Enterprise grade resiliency