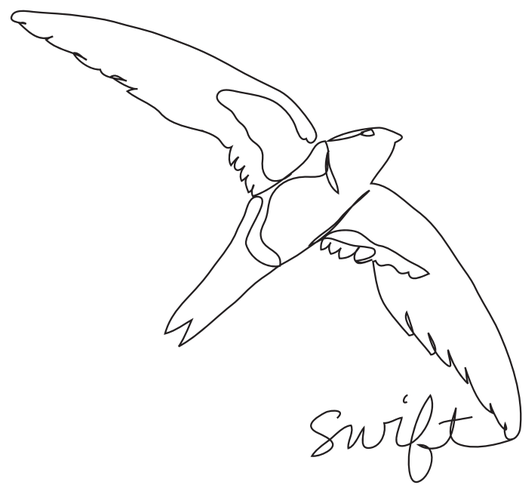
Scalable Object Storage for Public and Private Clouds

John Dickinson
@notmyname



Who Am I?

relatively new to storage. I don't consider myself a "storage guy" (I'm not even sure what that means) <click>

Almost four years ago, I joined Rackspace as part of their Cloud Files team <click>

While there, we built an object storage engine called Swift <click>

Swift was contributed to the OpenStack project, and has since attracted many more contributors and seen widespread deployment <click>

Last year, I joined a company called SwiftStack, where I'm working today. We build tools to help people solve storage problems, specifically around on-premise deployments of Swift. <click>

Early this year, I became more involved in the Open Compute project. At SwiftStack, we see it as a way for our customers to have access to cutting-edge hardware that both lowers costs and enables new storage applications

I'm Director of Technology at SwiftStack, and currently I'm serving as both the Project Technical Lead for OpenStack Swift and the Open Compute Storage committee chair

Who Am I?

relatively new to storage. I don't consider myself a "storage guy" (I'm not even sure what that means) <click>

Almost four years ago, I joined Rackspace as part of their Cloud Files team <click>

While there, we built an object storage engine called Swift <click>

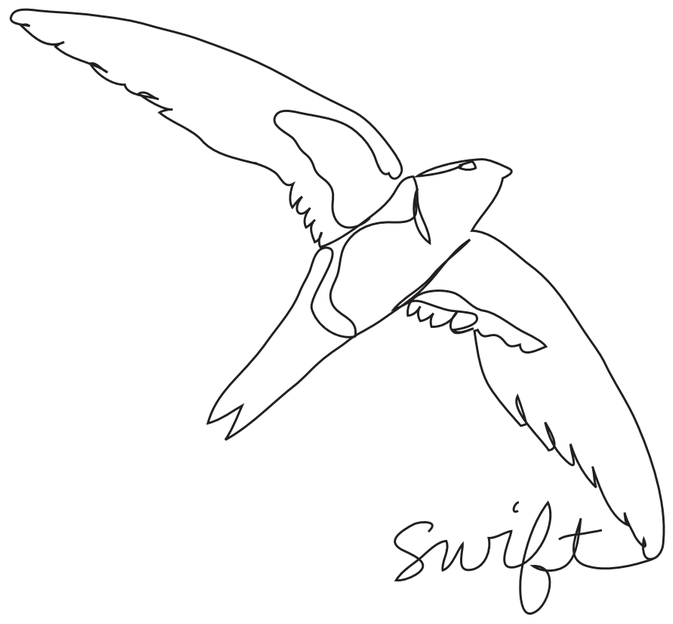
Swift was contributed to the OpenStack project, and has since attracted many more contributors and seen widespread deployment <click>

Last year, I joined a company called SwiftStack, where I'm working today. We build tools to help people solve storage problems, specifically around on-premise deployments of Swift. <click>

Early this year, I became more involved in the Open Compute project. At SwiftStack, we see it as a way for our customers to have access to cutting-edge hardware that both lowers costs and enables new storage applications

I'm Director of Technology at SwiftStack, and currently I'm serving as both the Project Technical Lead for OpenStack Swift and the Open Compute Storage committee chair

Who Am I?



relatively new to storage. I don't consider myself a "storage guy" (I'm not even sure what that means) <click>

Almost four years ago, I joined Rackspace as part of their Cloud Files team <click>

While there, we built an object storage engine called Swift <click>

Swift was contributed to the OpenStack project, and has since attracted many more contributors and seen widespread deployment <click>

Last year, I joined a company called SwiftStack, where I'm working today. We build tools to help people solve storage problems, specifically around on-premise deployments of Swift. <click>

Early this year, I became more involved in the Open Compute project. At SwiftStack, we see it as a way for our customers to have access to cutting-edge hardware that both lowers costs and enables new storage applications

I'm Director of Technology at SwiftStack, and currently I'm serving as both the Project Technical Lead for OpenStack Swift and the Open Compute Storage committee chair

Who Am I?

relatively new to storage. I don't consider myself a "storage guy" (I'm not even sure what that means) <click>

Almost four years ago, I joined Rackspace as part of their Cloud Files team <click>

While there, we built an object storage engine called Swift <click>

Swift was contributed to the OpenStack project, and has since attracted many more contributors and seen widespread deployment <click>

Last year, I joined a company called SwiftStack, where I'm working today. We build tools to help people solve storage problems, specifically around on-premise deployments of Swift. <click>

Early this year, I became more involved in the Open Compute project. At SwiftStack, we see it as a way for our customers to have access to cutting-edge hardware that both lowers costs and enables new storage applications

I'm Director of Technology at SwiftStack, and currently I'm serving as both the Project Technical Lead for OpenStack Swift and the Open Compute Storage committee chair

Who Am I?

relatively new to storage. I don't consider myself a "storage guy" (I'm not even sure what that means) <click>

Almost four years ago, I joined Rackspace as part of their Cloud Files team <click>

While there, we built an object storage engine called Swift <click>

Swift was contributed to the OpenStack project, and has since attracted many more contributors and seen widespread deployment <click>

Last year, I joined a company called SwiftStack, where I'm working today. We build tools to help people solve storage problems, specifically around on-premise deployments of Swift. <click>

Early this year, I became more involved in the Open Compute project. At SwiftStack, we see it as a way for our customers to have access to cutting-edge hardware that both lowers costs and enables new storage applications

I'm Director of Technology at SwiftStack, and currently I'm serving as both the Project Technical Lead for OpenStack Swift and the Open Compute Storage committee chair

Who Am I?

relatively new to storage. I don't consider myself a "storage guy" (I'm not even sure what that means) <click>

Almost four years ago, I joined Rackspace as part of their Cloud Files team <click>

While there, we built an object storage engine called Swift <click>

Swift was contributed to the OpenStack project, and has since attracted many more contributors and seen widespread deployment <click>

Last year, I joined a company called SwiftStack, where I'm working today. We build tools to help people solve storage problems, specifically around on-premise deployments of Swift. <click>

Early this year, I became more involved in the Open Compute project. At SwiftStack, we see it as a way for our customers to have access to cutting-edge hardware that both lowers costs and enables new storage applications

I'm Director of Technology at SwiftStack, and currently I'm serving as both the Project Technical Lead for OpenStack Swift and the Open Compute Storage committee chair

Everyone has data.

What motivates me to work on things like OpenStack and Open Compute? It comes down to four things:

Everyone has data <click>

It's always growing.

You should have ownership of everything that touches your data.

Open systems make this possible.

Wednesday, May 8, 13

This is what motivates me for OpenStack and Open Compute.

Open systems give users control, in an ownership sense, of what happens to their data. Users can not only see specs of the hardware and download the source code, they can make changes and contribute them back, get involved in the community developing these things, and even influence the overall design and governance of these projects.

That’s having ownership over your data.

What's the Use Case?

What is the problem we need to solve? It's a lot easier to build something once we know what problems it's supposed to solve (and it's a lot more personally satisfying when what you are building is actually useful to people).

Why do we need an object storage system at all?

Any unstructured data that can grow without bound.

documents, scientific data sets, web content, mobile device storage, application use data, user-generated content, medical imaging

This, of course, is a huge definition and leaves a lot of questions unanswered. It says nothing about the consistency model, the throughput requirements, the deployment patterns, and etc.

Swift, specifically

- Built for scale
- Optimized for durability, availability, and concurrency

This means it's really good for service providers, application developers (especially SaaS apps), and mobile devices.

Swift is great for storing large amounts of data (either aggregate bytes used or numbers of objects), and it really shines when it stores data that must be available and with high aggregate concurrency demands.

Why not block or file storage?

- Block is low-level and doesn't solve the "where do I put my data" question
- Filesystems are nice*, but have trouble when scaling

*not really

Wednesday, May 8, 13

The main issues that block and file systems struggle with as they grow is in maintaining consistency. Both storage access patterns require a consistent view of the world, and thus when deployed across a large cluster, hardware failures (which are expected and common) cause issues in availability, either in increased errors or in additional latency.

So by relaxing some of the constraints imposed by, eg POSIX, object storage systems can grow to massive scale and still remain both available and performant.

Main areas where things are relaxed:
partial file (object) updates
availability of file (object) locking

Learning Swift from Example Use Cases

so now we've covered, in general, what swift is good for.

I'd like to spend some time talking about Swift by describing some representative use cases

First, I'll look at "cheap and deep" to demonstrate swift's ability to scale and it's focus on durability
Next I'll talk about the web content use case to demonstrate swift's modular design and simple api
Finally I'll cover a mobile use case to talk about swift's support for very high concurrency across the data set

Swift Terminology

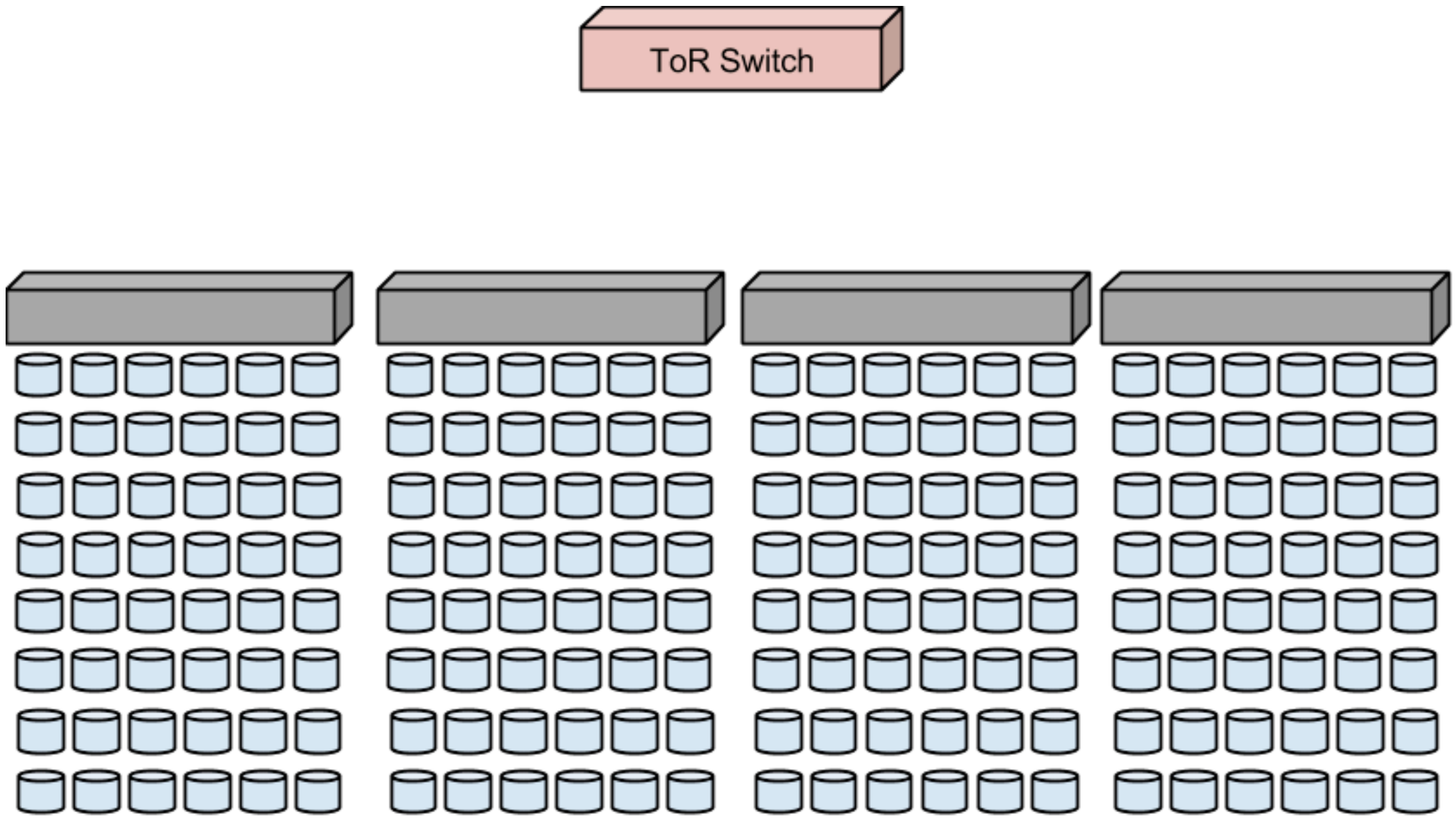
- Proxy - provides the API, coordinates requests to storage servers
- Account - part of the overall namespace for one user
- Container - user-defined segment of the account namespace
- Object - the data that is stored

but first a quick pause to give a very brief intro to some basic Swift vocabulary

Large Data Sets

backups
large scientific data sets (prohibitively expensive to replace)
    biomedical research
    astronomical data
geospacial data
storage for video production

Large Data Set Example

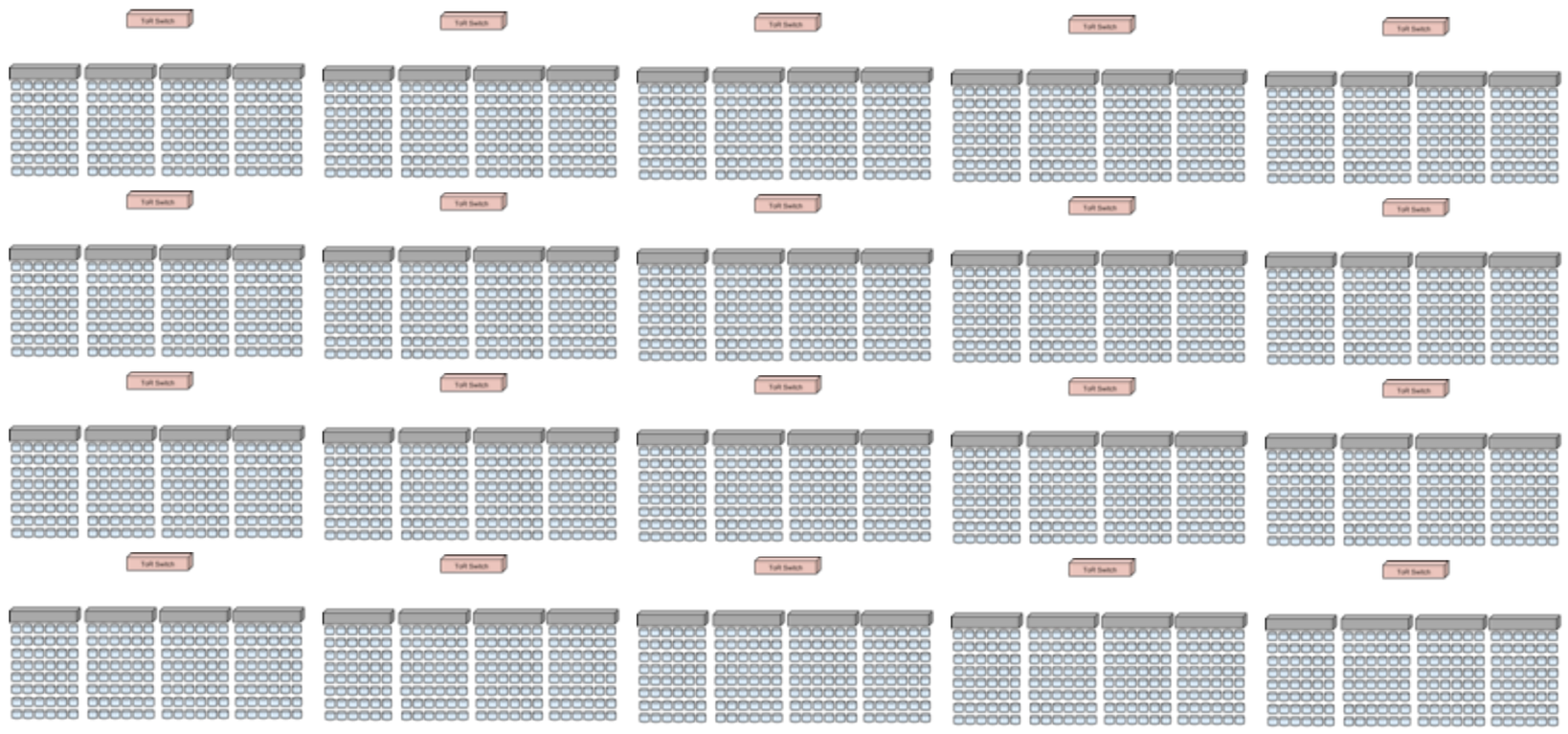


Large, dense nodes (eg JBODs to head units).
connected in a rack through a ToR switch
each of these nodes cloud be running all of the swift services. we'll look at how to separate things in other use cases

<click>

Build out many of these racks until you have enough space for your data
<click>

Large Data Set Example



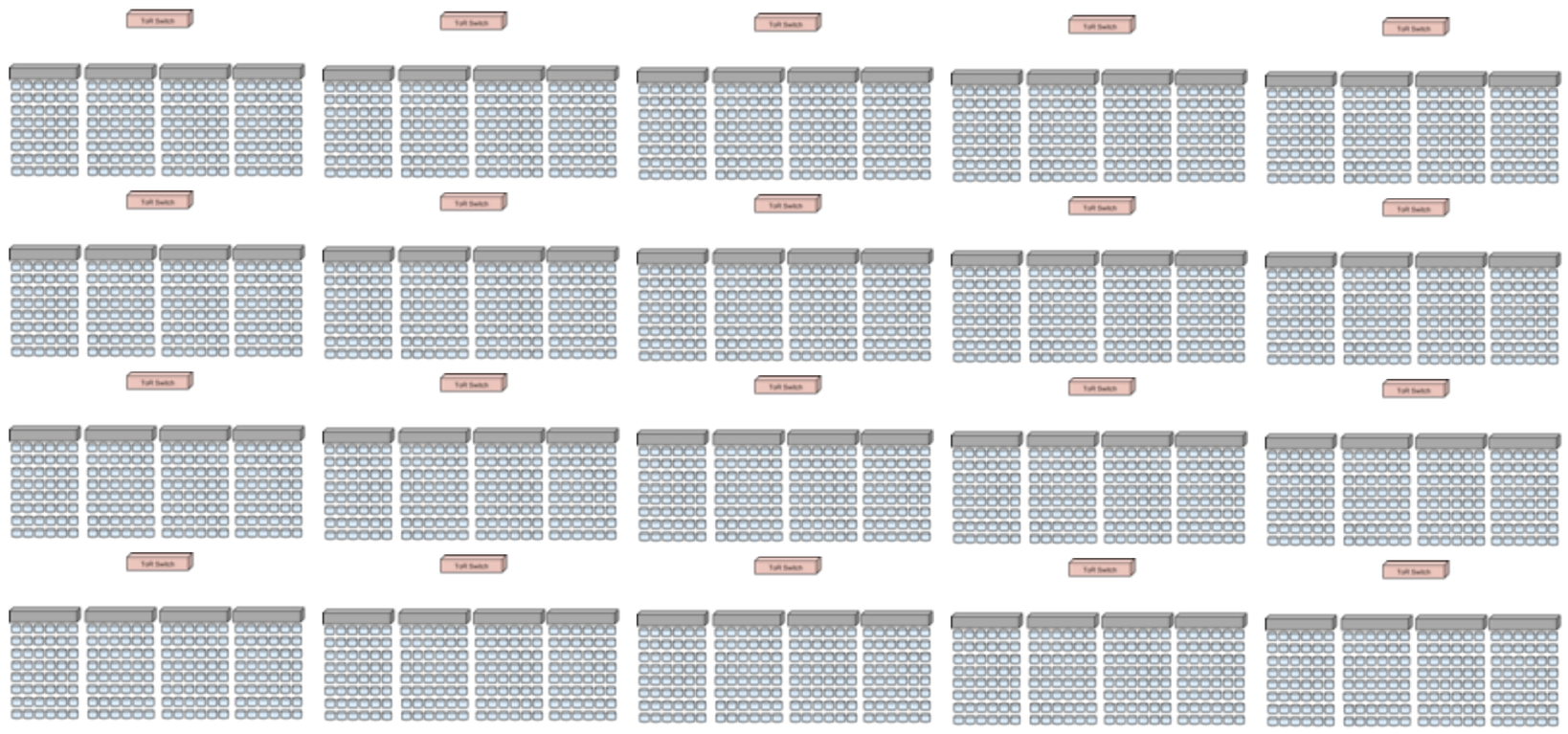
Wednesday, May 8, 13

Large, dense nodes (eg JBODs to head units).
connected in a rack through a ToR switch
each of these nodes cloud be running all of the swift services. we'll look at how to separate things in other use cases

<click>

Build out many of these racks until you have enough space for your data
<click>

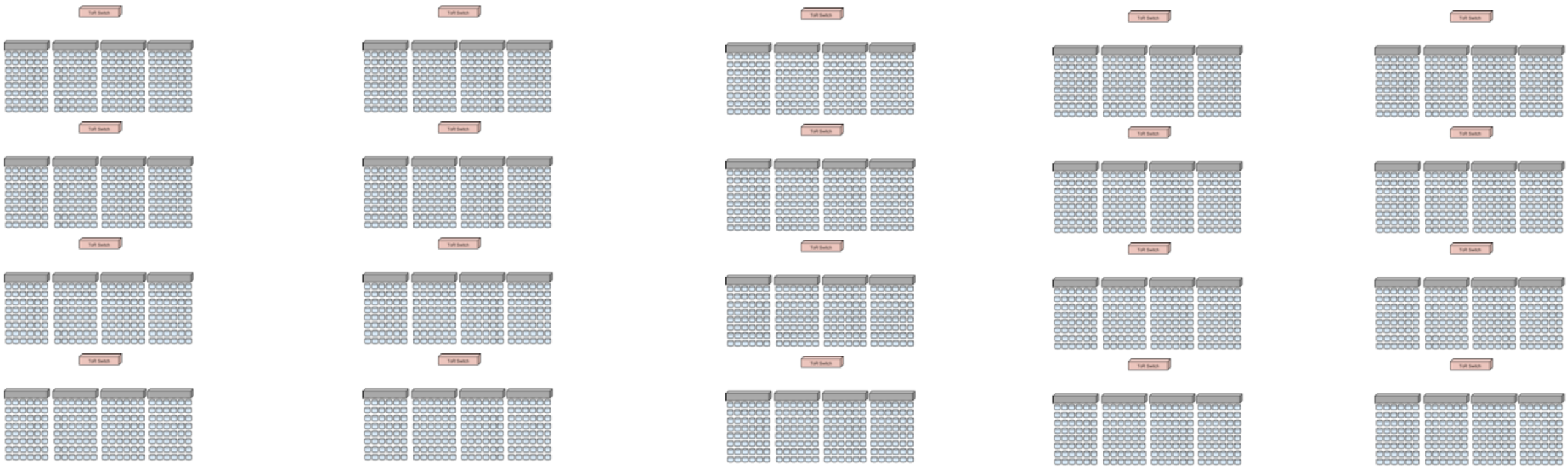
Large Data Set Example



Group your racks into availability zones, based on the other infrastructure you have.

eg, put them on separate power, HVAC, different DC rooms, even different DCs

Large Data Set Example



Group your racks into availability zones, based on the other infrastructure you have.

eg, put them on separate power, HVAC, different DC rooms, even different DCs

Swift Placement Rules

- "As Unique As Possible"
  - Region
  - Zone
  - Server
  - Drive

Wednesday, May 8, 13

this demonstrates one of the key features of swift: replicas of your data are stored across the cluster so that each replica is in as isolate as possible from other failures

it's a breadth-first search across the hierarchy of the cluster

Durability Concerns

- Active integrity checking
  - Checksum on reads
  - Scanning the data
- Continual replication processes

Wednesday, May 8, 13

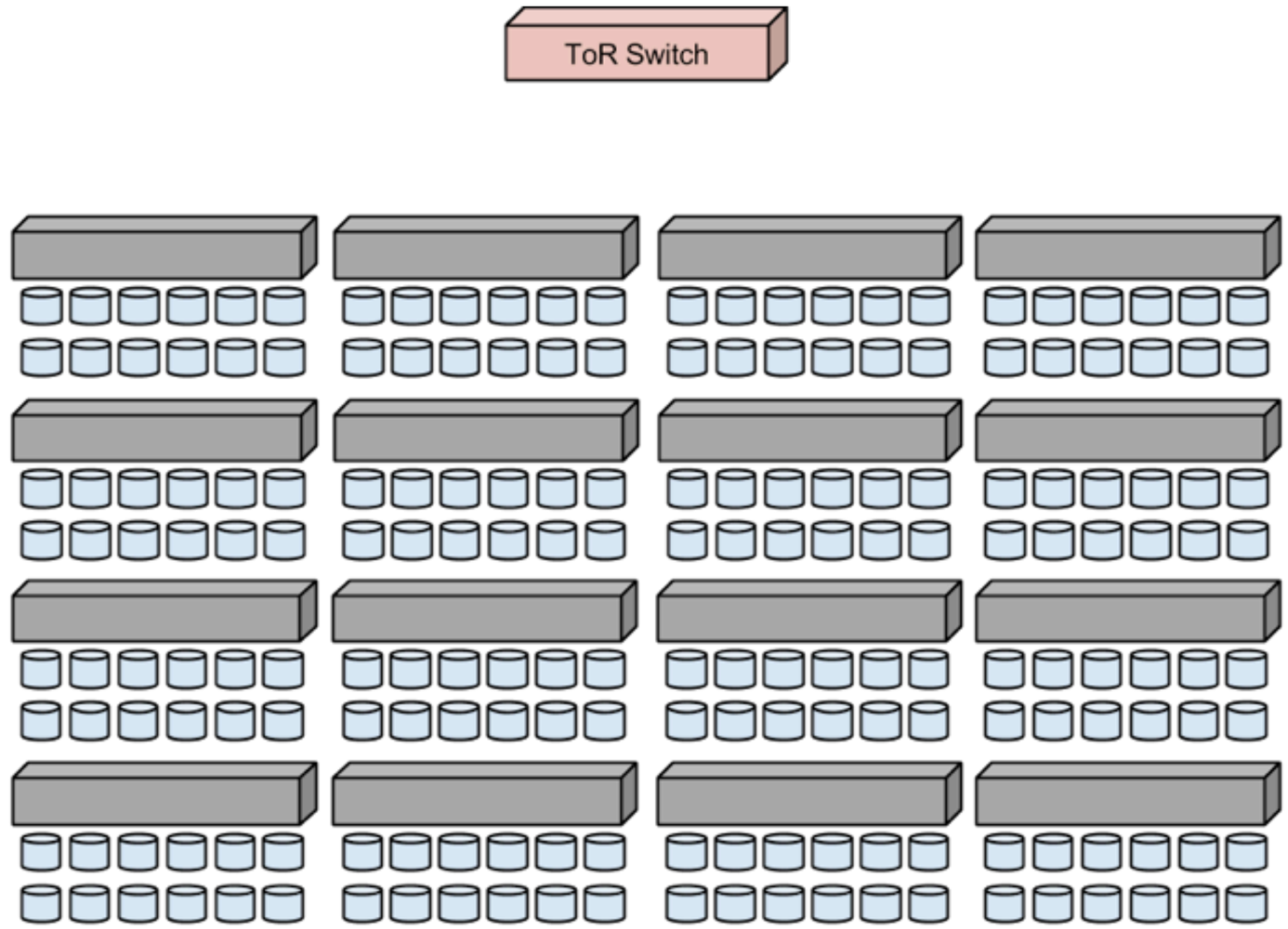
hardware fails (nearly four thousand drives in the example)

Swift's active integrity checking is accomplished by auditing the data on disk and ensuring that the data is properly replicated

Web Content

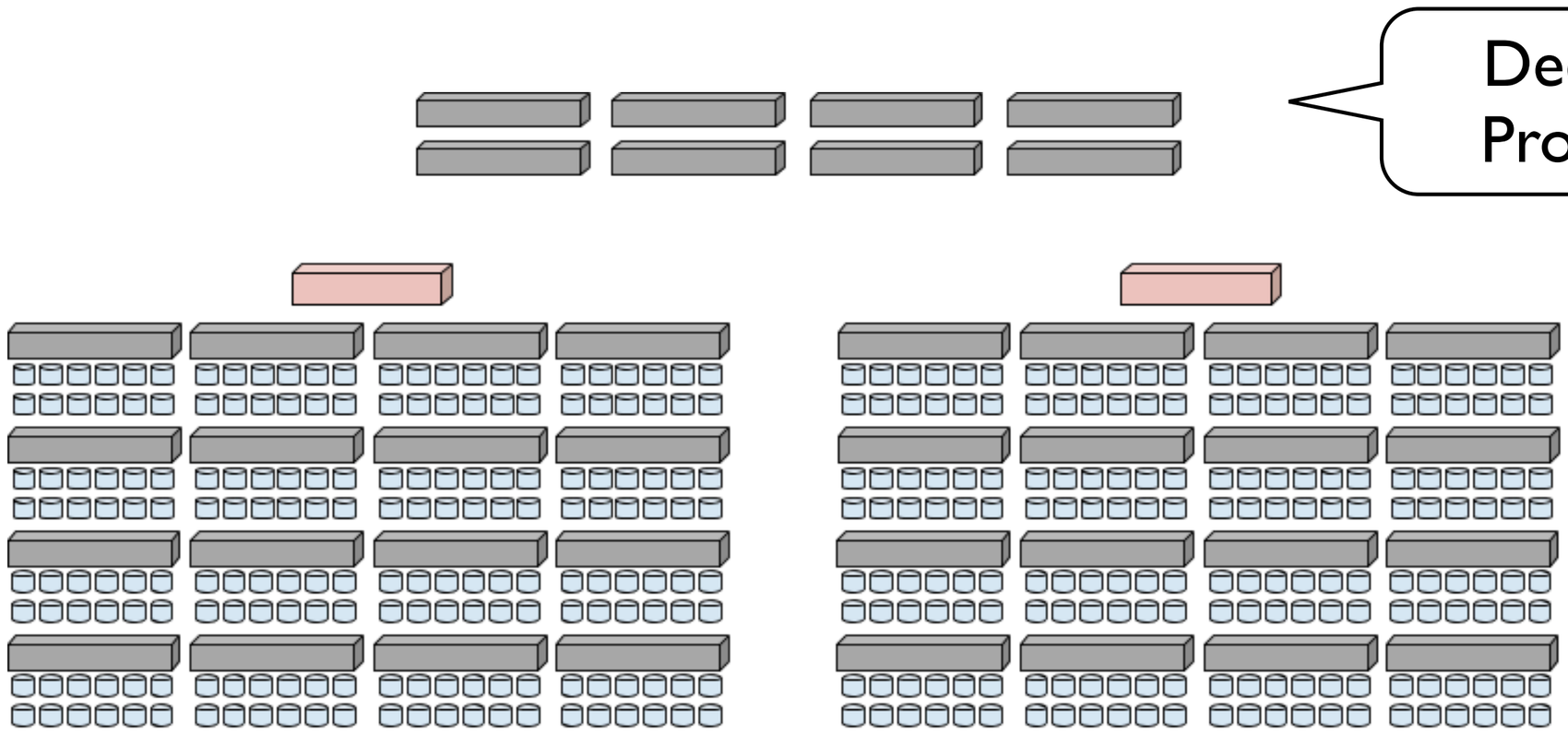
video storage
website content

Web content example



much less dense nodes

Web content example



Wednesday, May 8, 13

separate out a proxy tier to allow for more client requests relative to the storage.

this let's you optimize SKUs on each and could also help with your networking

this brings up an important point about swift <click>

Modular Design of Swift

- Proxy
- Account
- Container
- Object

Four major services in swift

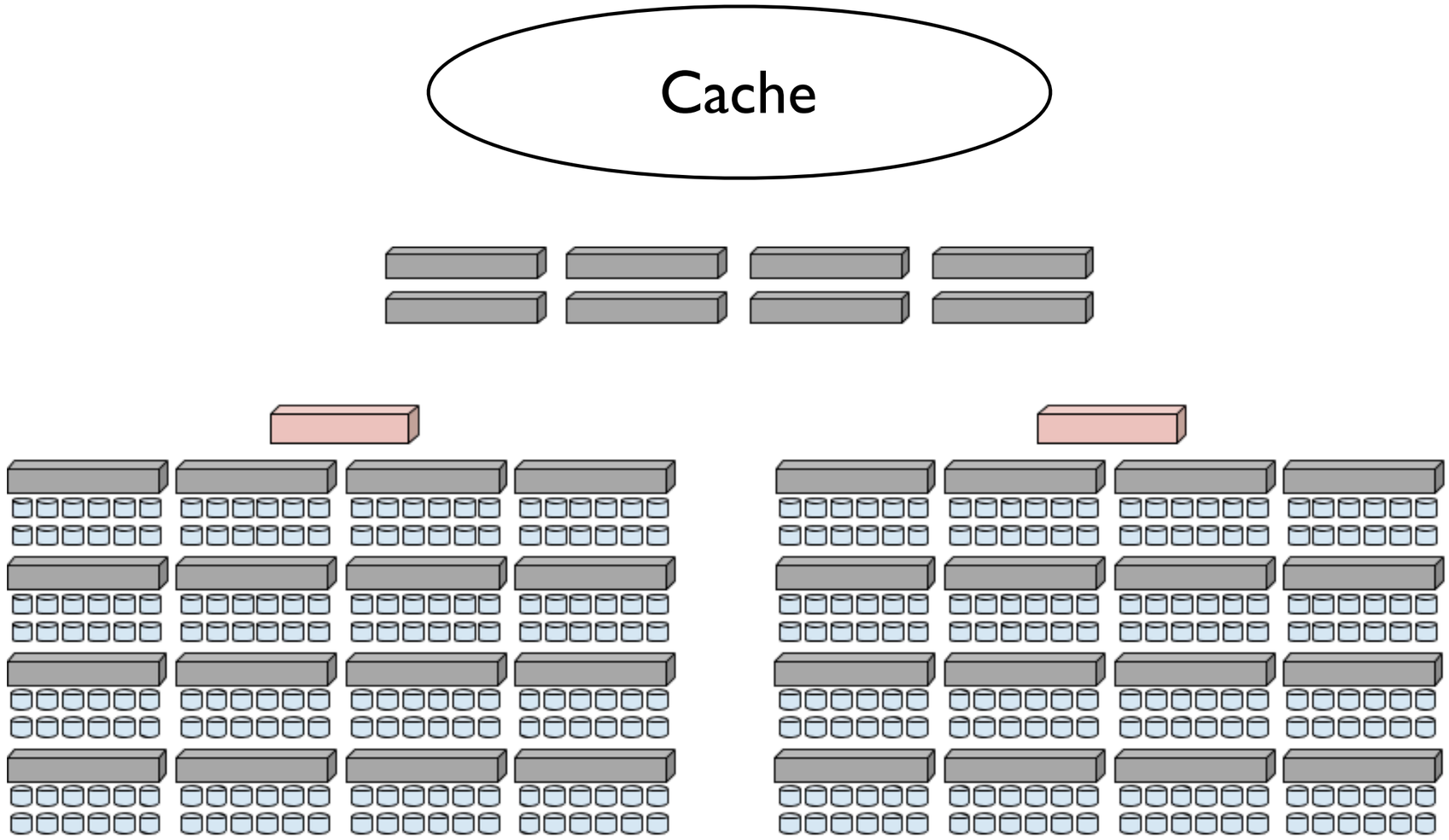
proxy: font-end requests (API and coordination with storage nodes)

account: per-user namespace. stores aggregated stats and a listing of containers

container: per account namespace. stores aggregated stats and a listing of contianers
containers are often the unit of implementation for features (eg ACLs are per-container)

object: stores the data on a storage volume

Web content example



Add in a caching layer to accelerate requests for hot content

The reason this is easy is because the Swift's REST API.

Swift API

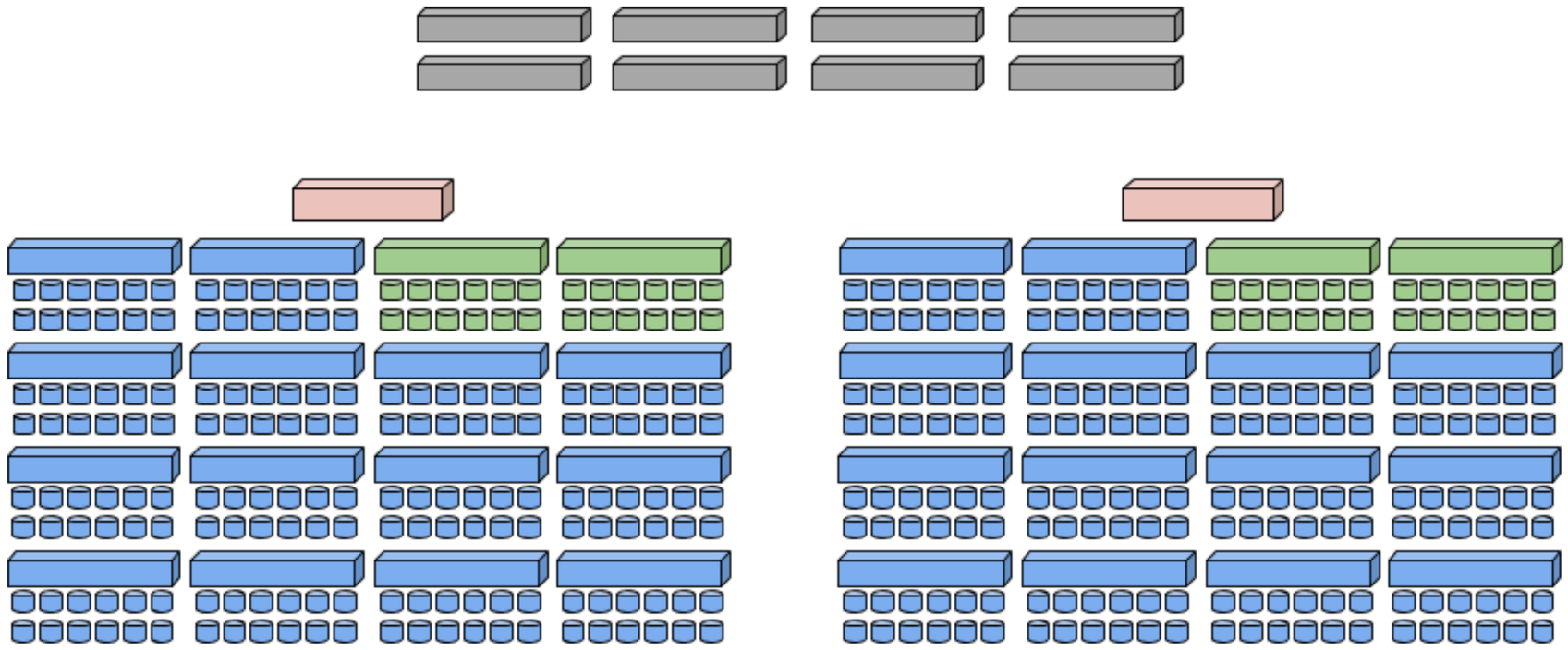
- Standard HTTP verbs and response codes
- Easy for devs to use
- Easy to integrate with other solutions, like caching

Mobile Content

millions of devices
user-generated content (eg image uploads. not just instagram, but depositing checks, scanning receipts, reading books, playing games)
phone backups

commonality: very wide access to data that must be available. not a lot of contention on a particular piece of data

Mobile content example



Wednesday, May 8, 13

separate proxy layer (high concurrent throughput)

separate tier for account + container servers
much less space needed for these, but having their own dedicated servers (and especially fast drives!!) means writes are faster

This again demonstrates swift's modular design and how it can be used to optimize a cluster for a particular workload.

But the other thing to point out is that this shows how swift scales as the cluster gets bigger <click>

Swift is designed so that there is no central authority or other single point of failure. Each connection is independent, and that means you can grow the cluster by adding more servers and get better overall performance

Mobile content example

separate proxy layer (high concurrent throughput)

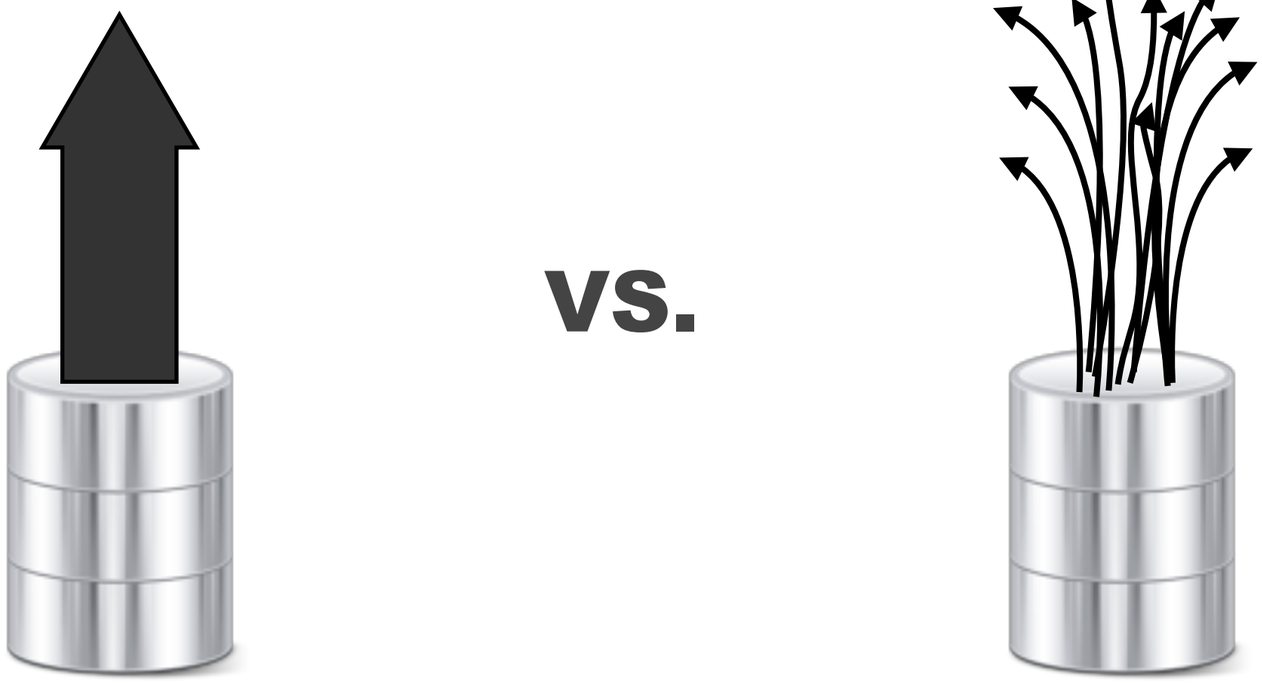
separate tier for account + container servers
much less space needed for these, but having their own dedicated servers (and especially fast drives!!) means writes are faster

This again demonstrates swift's modular design and how it can be used to optimize a cluster for a particular workload.

But the other thing to point out is that this shows how swift scales as the cluster gets bigger <click>

Swift is designed so that there is no central authority or other single point of failure. Each connection is independent, and that means you can grow the cluster by adding more servers and get better overall performance

Swift is Designed for Concurrency



vs.

Wednesday, May 8, 13

we've seen a lot of adoption of Swift in the past three years, especially in the use cases I've just talked about

these are deployers or contributors

we know of more private clouds, and I know there are many I don't know (eg contributor emails at gmail)

most of these are service providers, known because they announce their public clouds

my vision is for everyone to use swift every day, whether they know it or not

Building and Running Swift

Initial Considerations

- What tiers are you going to build out?
- Focus on density or performance?
- Hardware selection
- Challenge: networking

hardware selection:

use off-the-shelf components (head unit + JBOD)

commodity drives (Swift works around failures)

Running the Cluster

- Things to monitor
- Handling failures
- Operational focus

things to monitor, beyond standard system metrics:
    how are the consistency processes doing?
    response times
    any hardware failures?

handling hardware failures
    swift can work in a “lights out” DC
    replace drives when you can, maybe
    replace servers when you can, definitely

Swift was built and is still developed with a focus on sys admins. The people writing it are responsible for keeping it up and running. Yes, there are lots of pieces, but the overall design is straightforward and each piece does one thing.

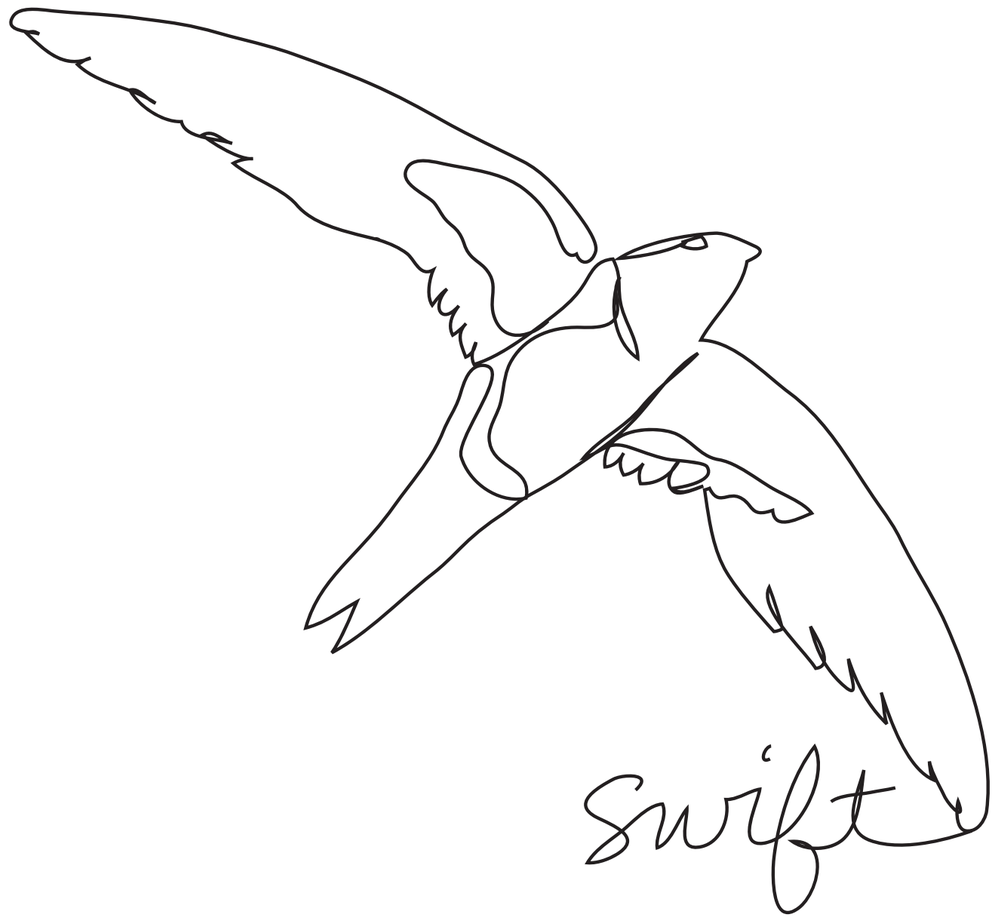
Growing the Cluster

- Monitoring shows you where to grow
- Modular design allows you to grow where you need

in large deployments, you generally define your unit of expansion as a rack or set of racks, already configured to your particular ratios for storage, proxies, etc

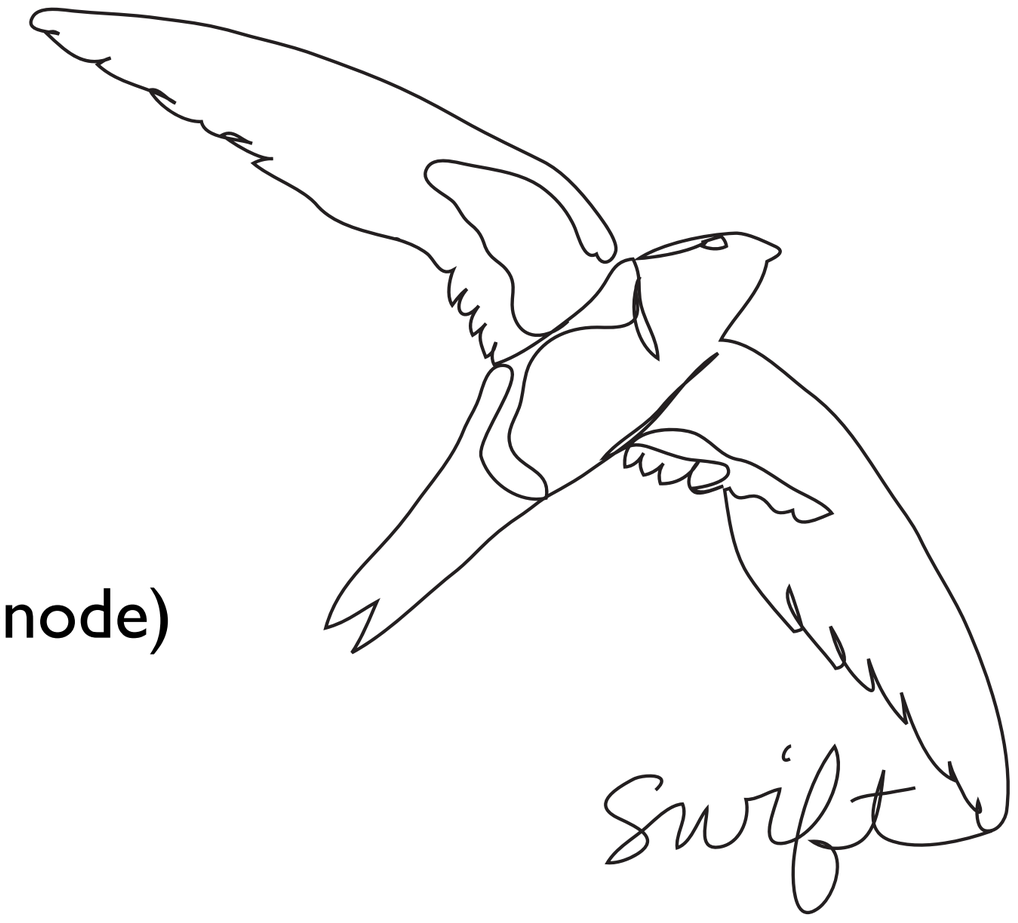
statsd monitoring is deeply integrated in swift
custom tools provided by swift
verbose logging, with the ability to follow a single request through the cluster

Summary



Swift is an open system that gives users ownership of their data. It's good for any unstructured data set that can grow without bound. Swift is built for scale and optimized for durability, availability, and concurrency.

Questions?



@notmyname
#openstack-swift (freenode)

http://swiftstack.com

Swift is an open system that gives users ownership of their data. It's good for any unstructured data set that can grow without bound. Swift is built for scale and optimized for durability, availability, and concurrency.