Protecting Data with Declustered Parity

John Bent, Seagate Senior Director

This is your drive

This is your precious

Store your precious
on your drive.
Happiness forever.

This is your drive

This is your precious

Happiness Forever?

Unfortunately, disks fail

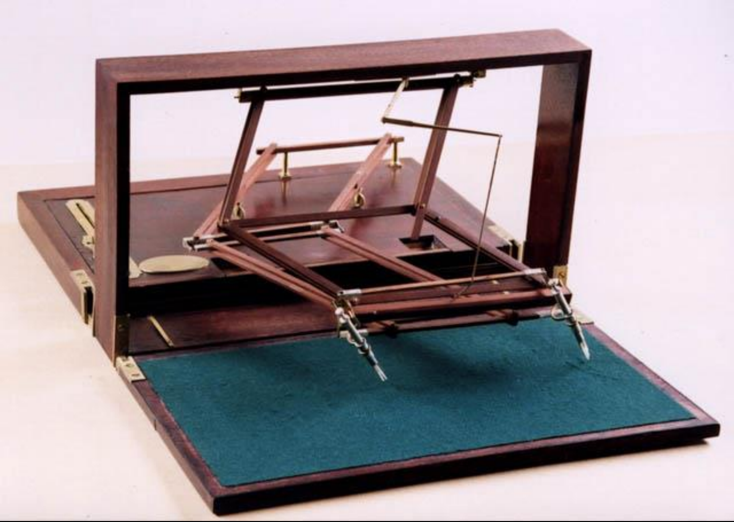
Some of John's Favorite Papers about Storage Failure

- Availability in Globally Distributed Storage Systems
  - https://static.googleusercontent.com/media/research.google.com/en//pubs/archive/36737.pdf
- An Analysis of Data Corruption in the Storage Stack
  - https://www.cs.toronto.edu/~bianca/papers/fast08.pdf
- Disk failures in the real world
  - http://www.cs.toronto.edu/~bianca/papers/fast07.pdf
- Mean time to meaningless
  - https://www.usenix.org/legacy/event/hotstorage10/tech/full_papers/Greenan.pdf

Also many good blog posts at backblaze.

This is Not Just a Modern Problem

As long as the world has stored data, the world has needed mechanisms to protect that data from the inevitable failure of the storage medium.



Polygraph: create a copy of your document as you write it.

Duplicating books before the invention of the printing press.

Replication Has Been Around For a Long Time

But high capacity overhead….

RAID: Modern Technique to Gain Durability without High Capacity Overhead

A **case** for **redundant arrays** of **inexpensive disks** (RAID)
DA Patterson, G Gibson, RH Katz - Proceedings of the 1988 ACM ..., 1988 - dl.acm.org
Increasing performance of CPUs and memories will be squandered if not matched by a similar performance increase in I/O. While the capacity of Single Large Expensive Disks (SLED) has grown rapidly, the performance improvement of SLED has been modest ...
Cited by 4397 Related articles All 146 versions

RAID5: Simple XOR Logic to Compute P(arity) from D(ata)

Terminology

Chunk: the contiguous piece of data stored to a single device

Chunk size: you can guess this one

Data chunk: also this one

Parity chunk: also this one

Stripe: the collection of chunks that form a contiguous logical unit of data

Layout: the particular devices on which a stripe is stored

Pattern: the shared configuration of all stripes (e.g. 8+2 or 3-way replication)

Full-stripe write: an important concept for performance (not discussed here)

Partial-stripe write: a write that is smaller than a full-stripe write

Read-modify-write: the performance inhibition caused by partial-stripe write

Pool: for parity declustered (not yet explained), all devices on which a stripe could be stored

How Long Does Rebuild Take?

Find replacement drive

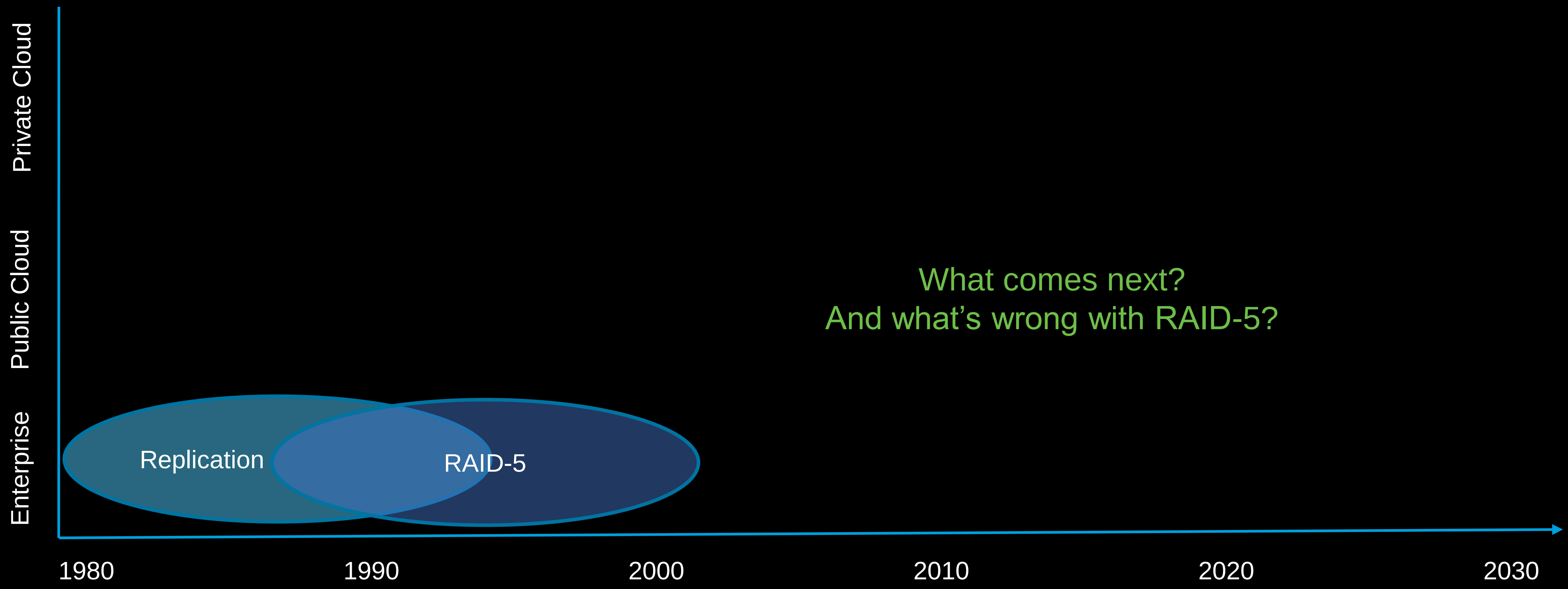
For each *chunk* that was on failed drive

- Read other chunks from that chunk's stripe from the sibling drives in the stripe layout
- Use parity to reconstruct the lost unit
- Write the lost unit to the replacement drive

$$t(rebuild) \approx t(write\ entire\ drive)$$

$$t(write\ entire\ drive) = capacity/bandwidth$$

The Evolution of Data Durability in Data Centers



What comes next?
And what's wrong with RAID-5?

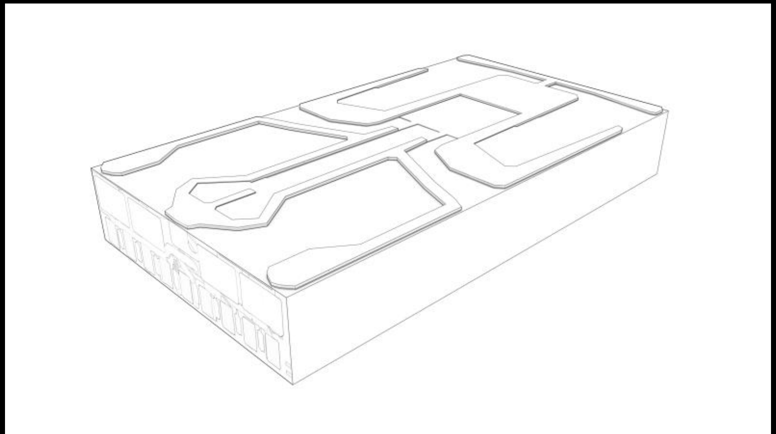
An Important Characteristic of Disks

$$growth(bandwidth) \approx \sqrt{growth(capacity)}$$

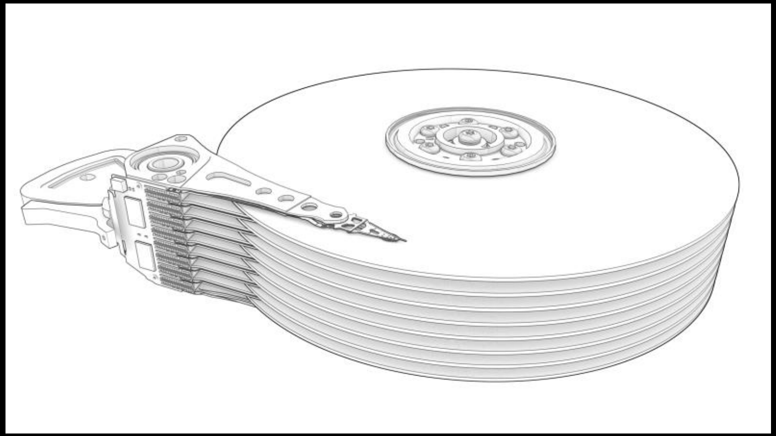
- alternately -

$$growth(capacity) \approx growth(bandwidth)^2$$

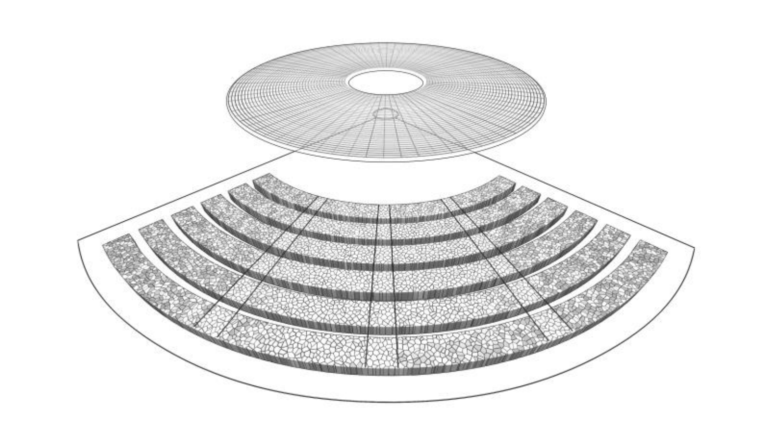
How to Increase Disk Capacity



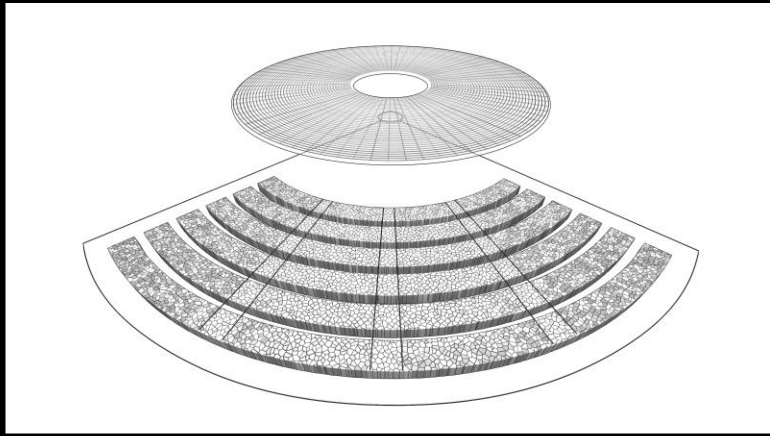
Increase the Form Factor



Insert more Platters



More Tracks per Surface

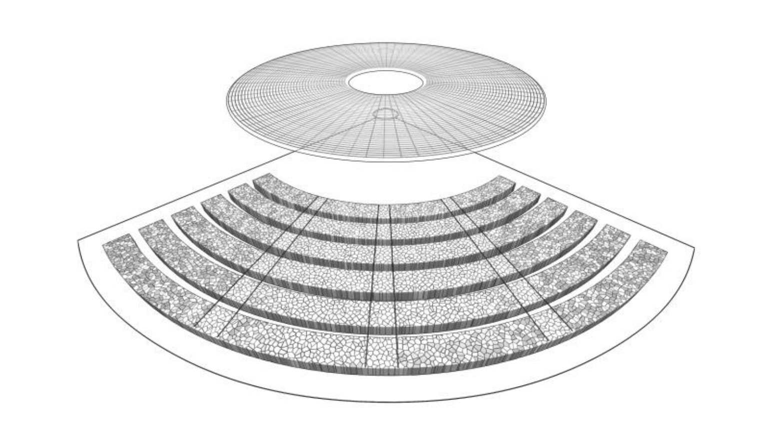


More Bits Per Track

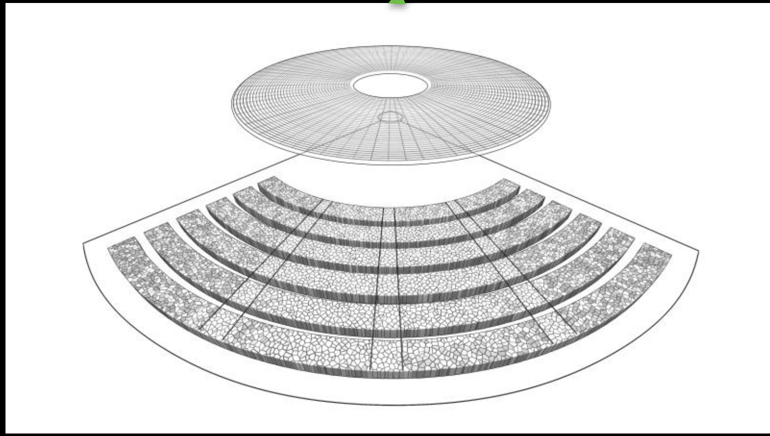
How to Increase Disk Bandwidth

Spin Faster

SAME!



Access More Tracks
Simultaneously



More Bits Per Track

$$growth(bandwidth) \approx \sqrt{growth(capacity)}$$

How to Increase Capacity

- ~~Increase the Form Factor~~
- ~~Insert more platters~~
- More bits per track
- More tracks per surface

How to Increase Bandwidth

- ~~Spin the surface faster~~
- ~~Access more tracks simultaneously~~
- More bits per track

$$growth(bandwidth) \approx \sqrt{growth(capacity)}$$

How to Increase Capacity

- ~~Increase the Form Factor~~
- ~~Insert more platters~~
- More bits per track
- More tracks per surface

How to Increase Bandwidth

- ~~Spin the surface faster~~
- ~~Access more tracks simultaneously~~
- More bits per track

$$bandwidth(disk_{n+1}) \approx bandwidth(disk_n) * growth(bits\ per\ track)$$

$$capacity(disk_{n+1}) \approx capacity(disk_n) * growth(bits\ per\ track) * growth(tracks\ per\ surface)$$

$$growth(tracks\ per\ surface) \approx growth(bits\ per\ track)$$

$$capacity(disk_{n+1}) \approx capacity(disk_n) * growth(bits\ per\ track)^2$$

$$growth(bandwidth) \approx \sqrt{growth(capacity)}$$

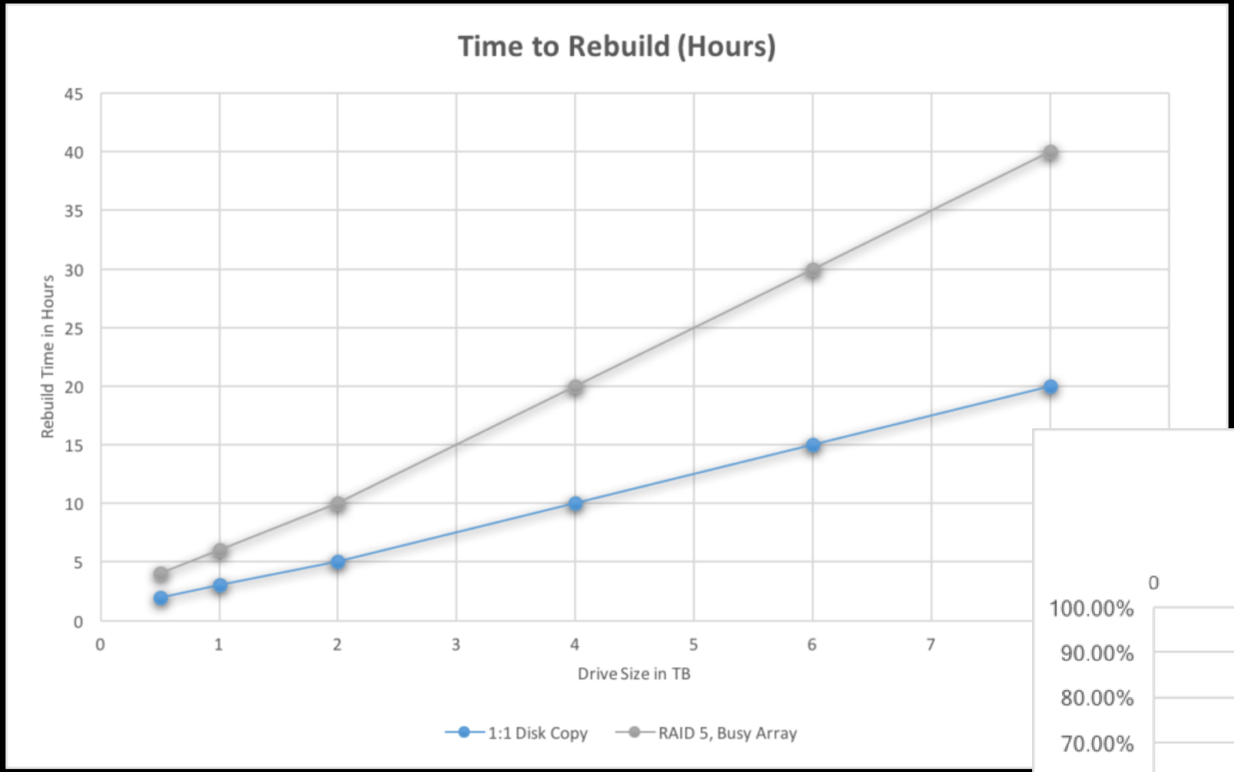
$$t(write\ entire\ drive) = capacity/bandwidth$$

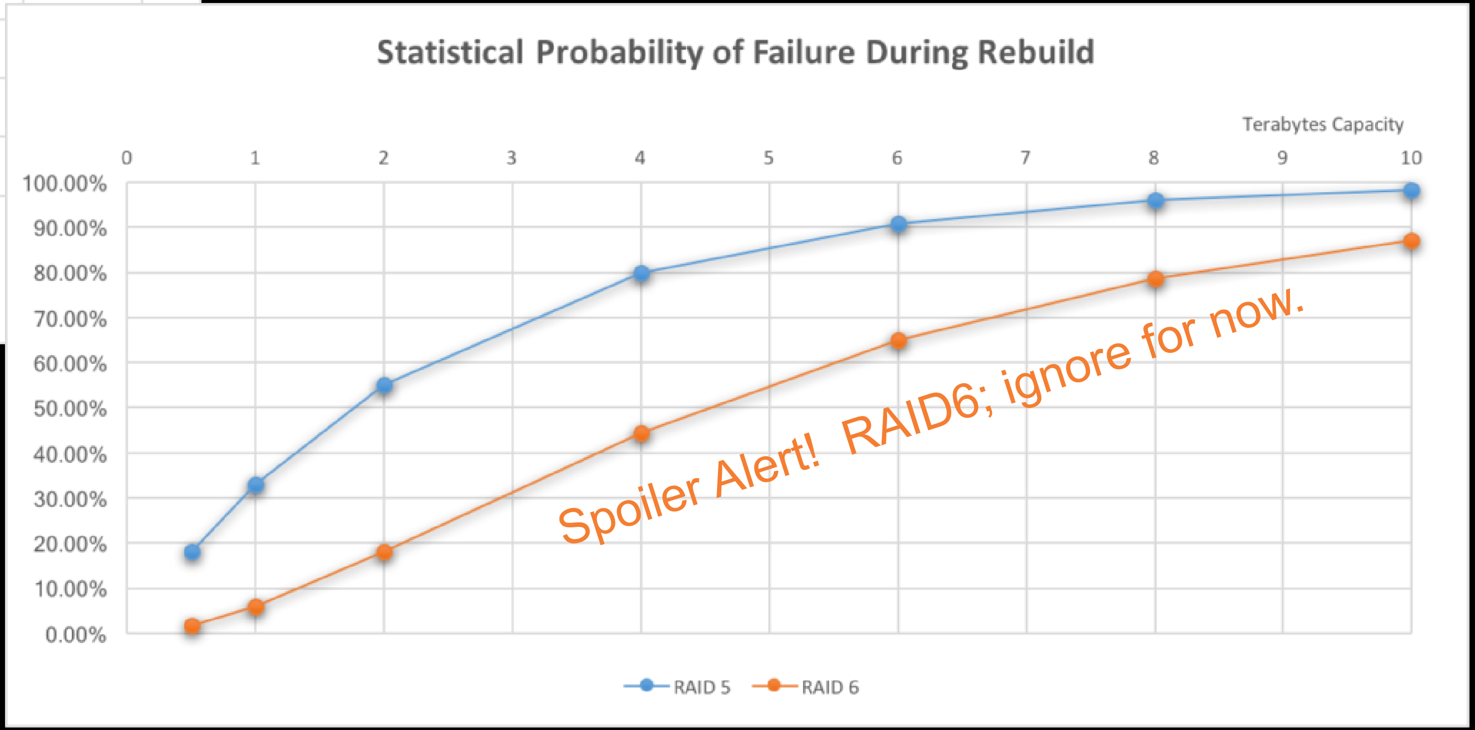
PLUS

$$growth(bandwidth) \approx \sqrt{growth(capacity)}$$

EQUALS

$$Probability(DataLoss, RAID5) \Rightarrow 1$$





Graphs: https://www.actualtech.io/really-outgrown-raid/

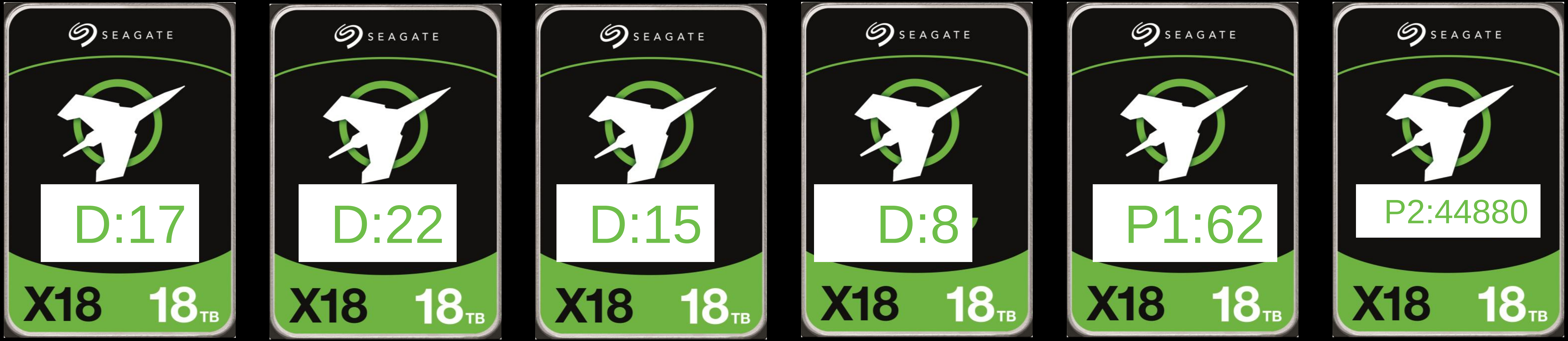
What Comes After RAID5? RAID6!

RAID6 Logic? High School Algebra! Two Equations, Two Unknowns.

* System understanding != Math understanding

P1 sum*

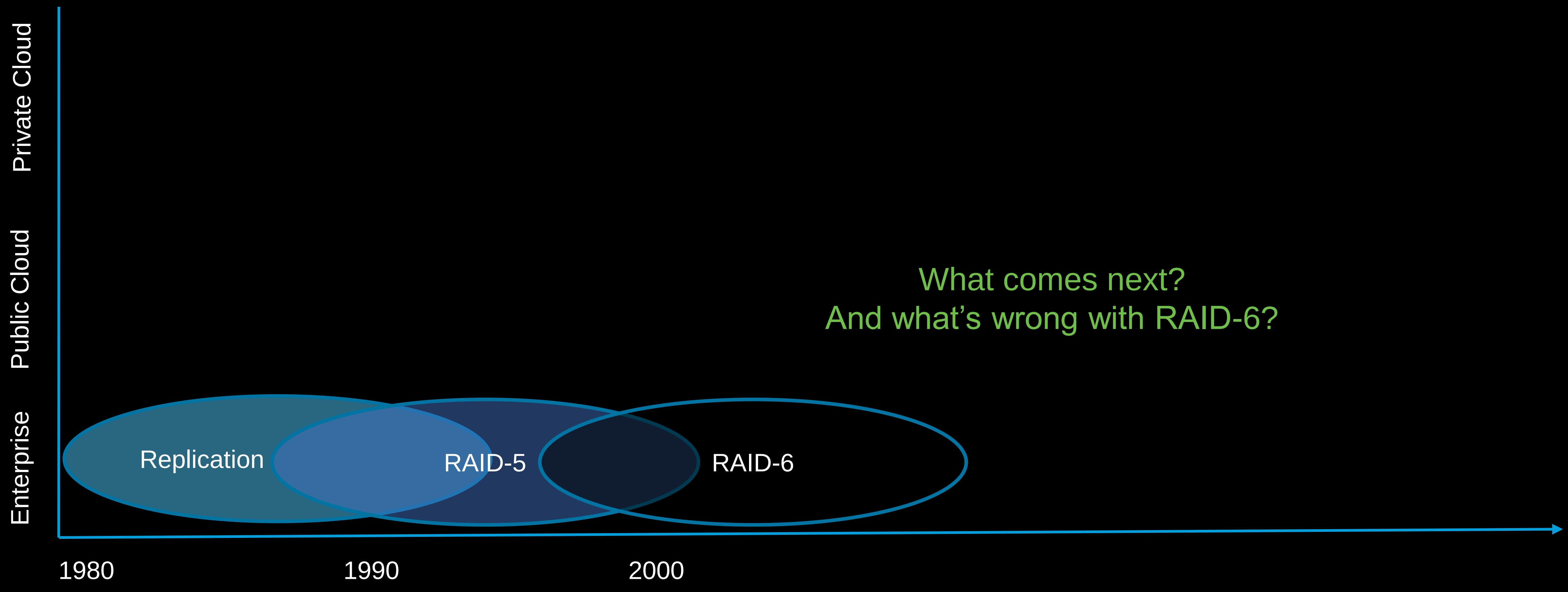
P2 product*



$$62 = 17 + x + 15 + y$$
$$44880 = 17 * x * 15 * y$$

$\xrightarrow{\text{(math)}}$ $x = 22 ; y = 8$

The Evolution of Data Durability in Data Centers



What comes next?
And what's wrong with RAID-6?

$$t(write\ entire\ drive) = capacity/bandwidth$$

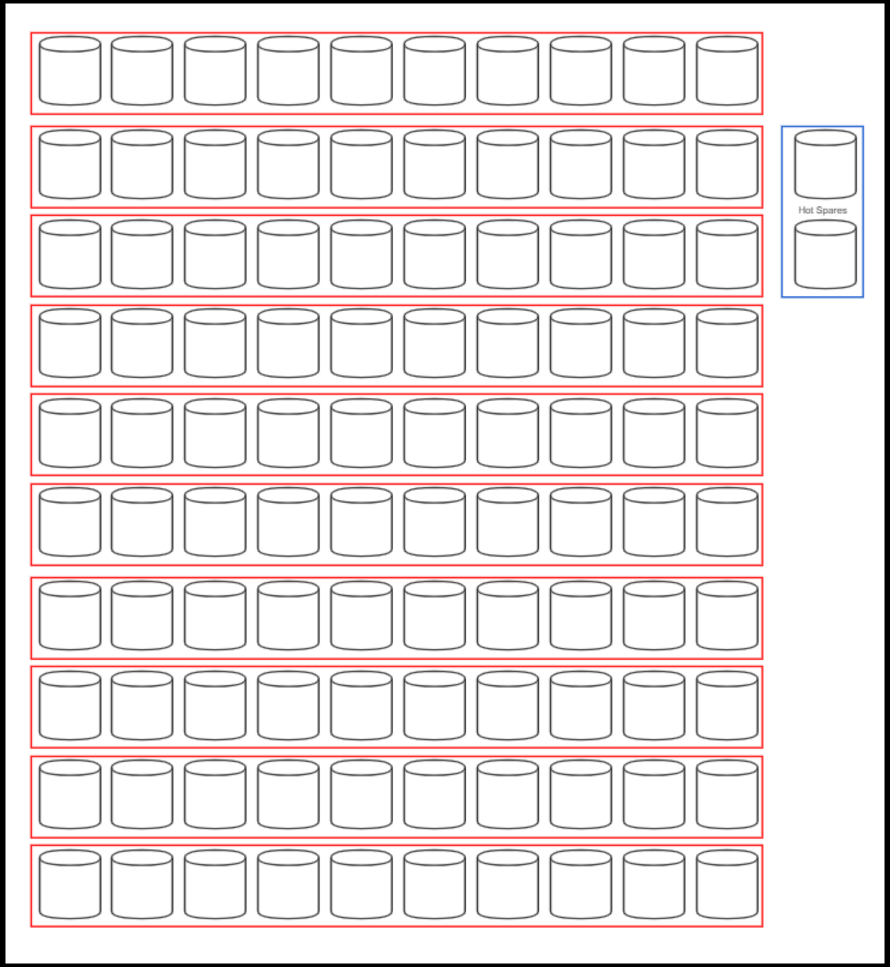
PLUS

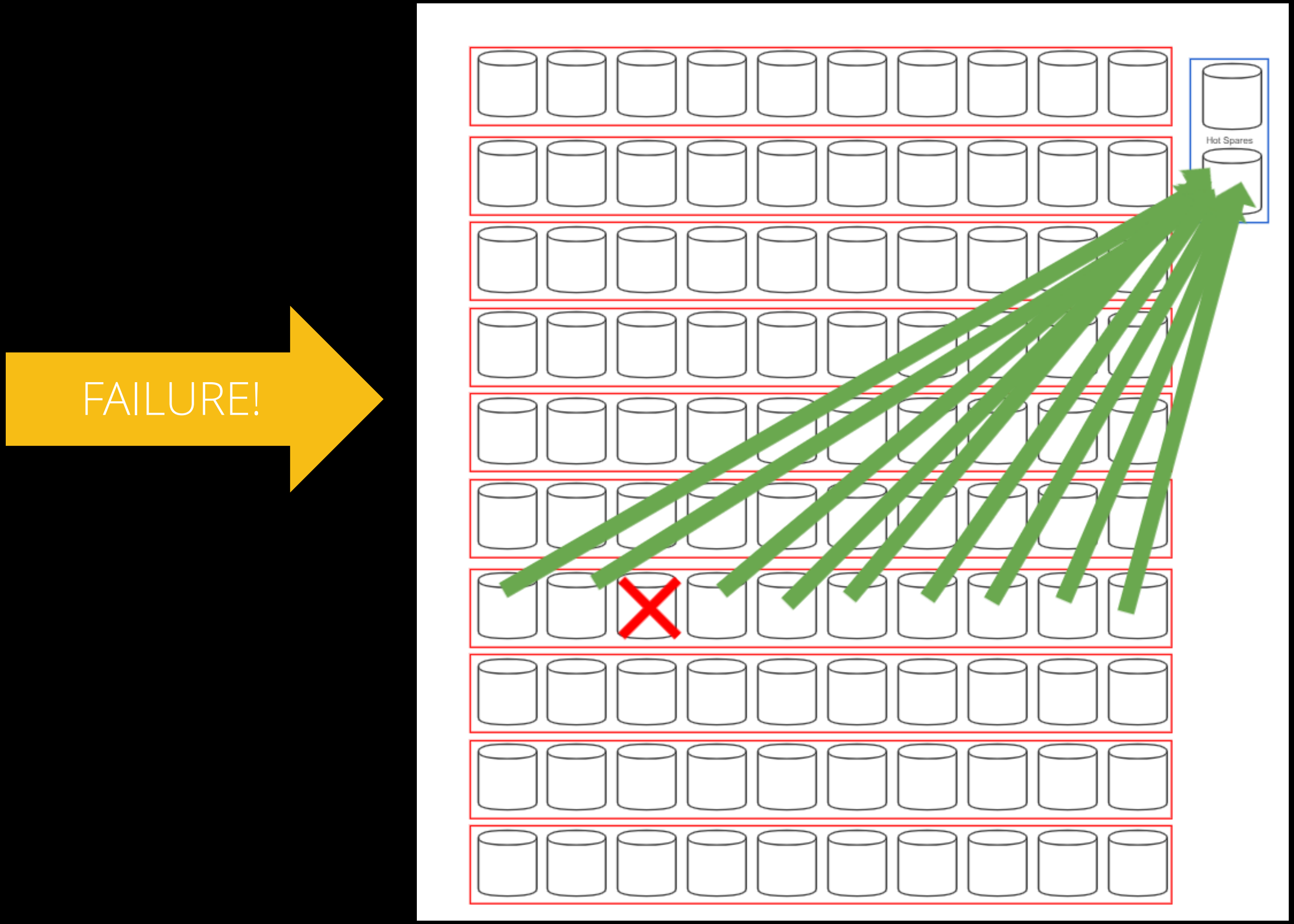
$$growth(bandwidth) \approx \sqrt{growth(capacity)}$$

EQUALS

$$Probability(DataLoss, RAID6) \Rightarrow 1$$

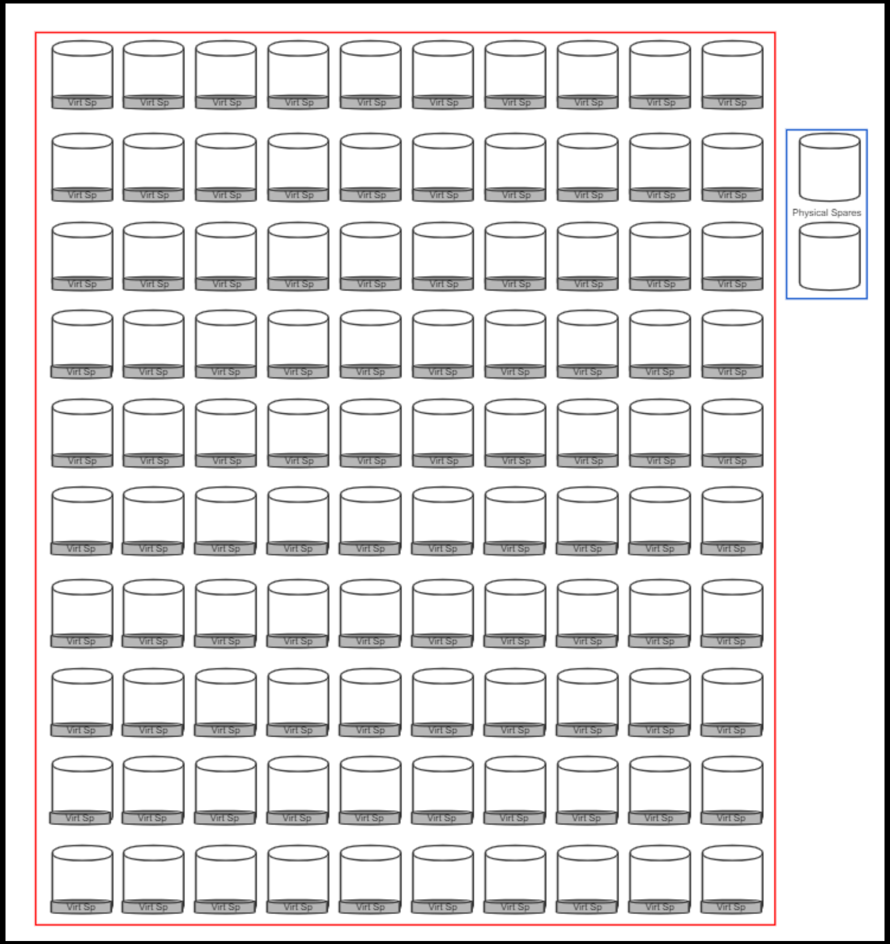
Finally, Onto Declustered! (wait, not quite)





100 disks, fixed groups, 2 spares

Finally, Onto Declustered!



100 disks, one declustered groups, 2 spares

Declustered Parity Key Concepts

Virtual spares
- N+K -> N+K+S

Replay Phases
- Following the rebuild Phase

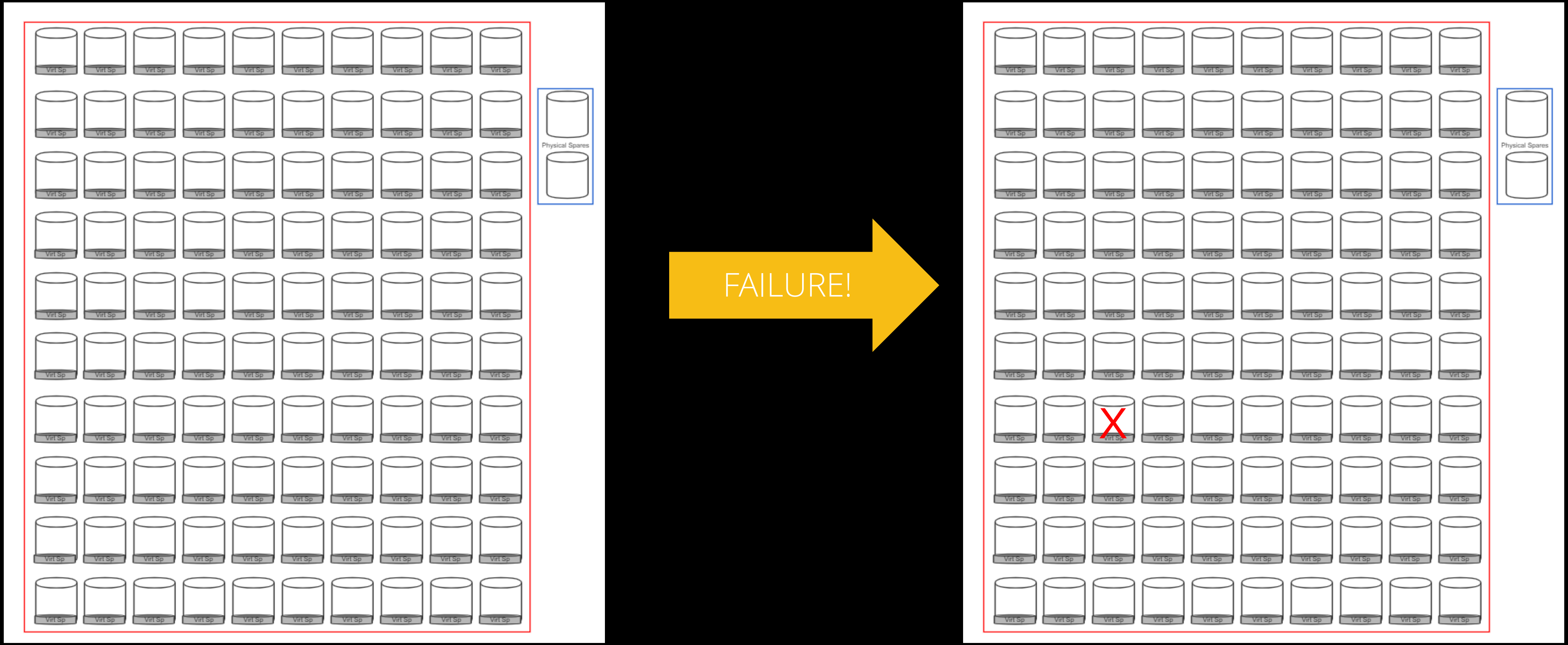
Pseudo-Random Layout
- Every stripe goes to a different random N+K drives

Rebuild is all-all
- E.g. 99 to 99

Rebuild time is time to write a disk's worth of data across all
- Scales with total population of drives

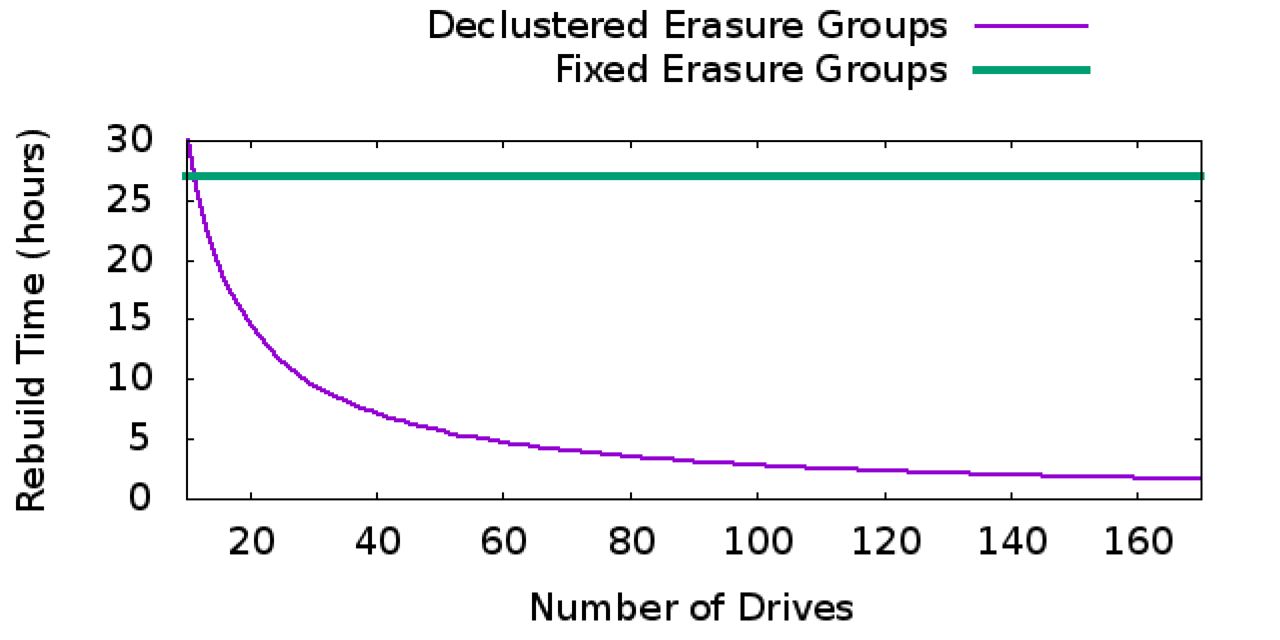
Finally, Onto Declustered!

Not drawing the 9702 arrows! Sorry!



100 disks, one declustered groups, 2 spares

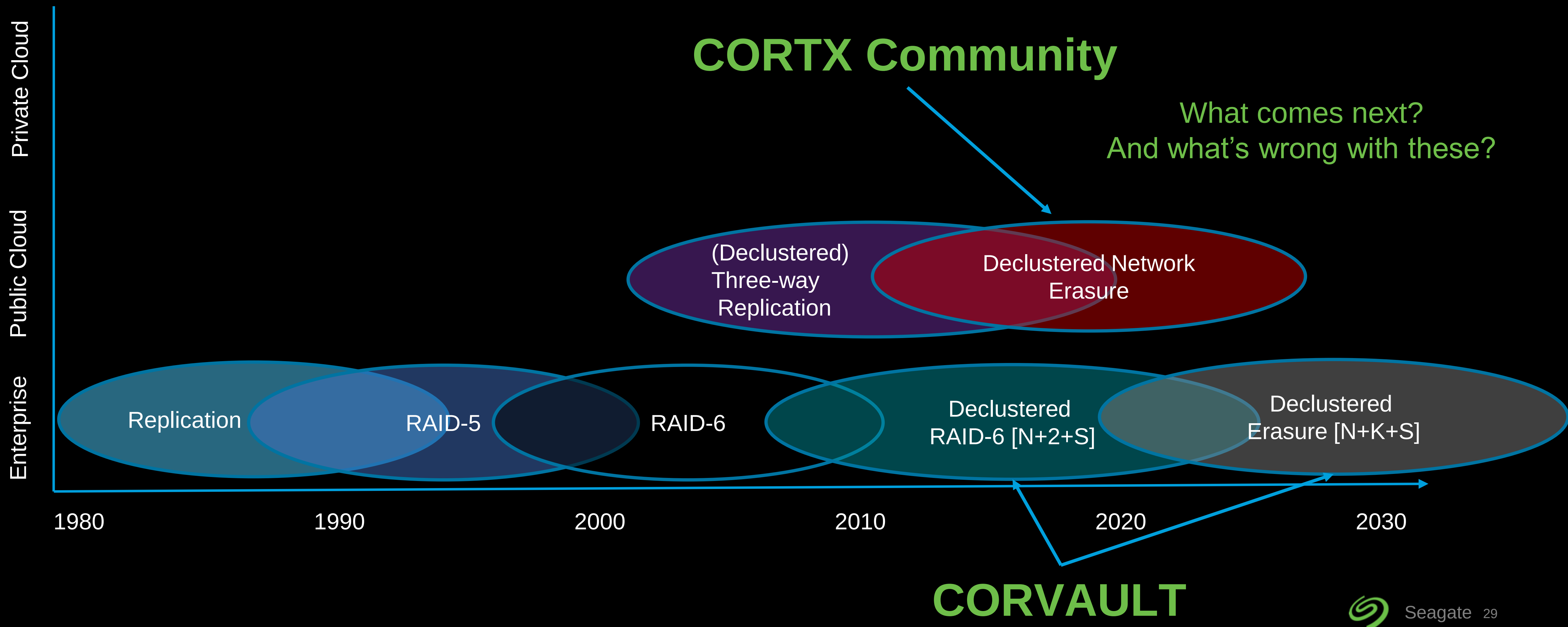
Rebuild Time Scales with Population



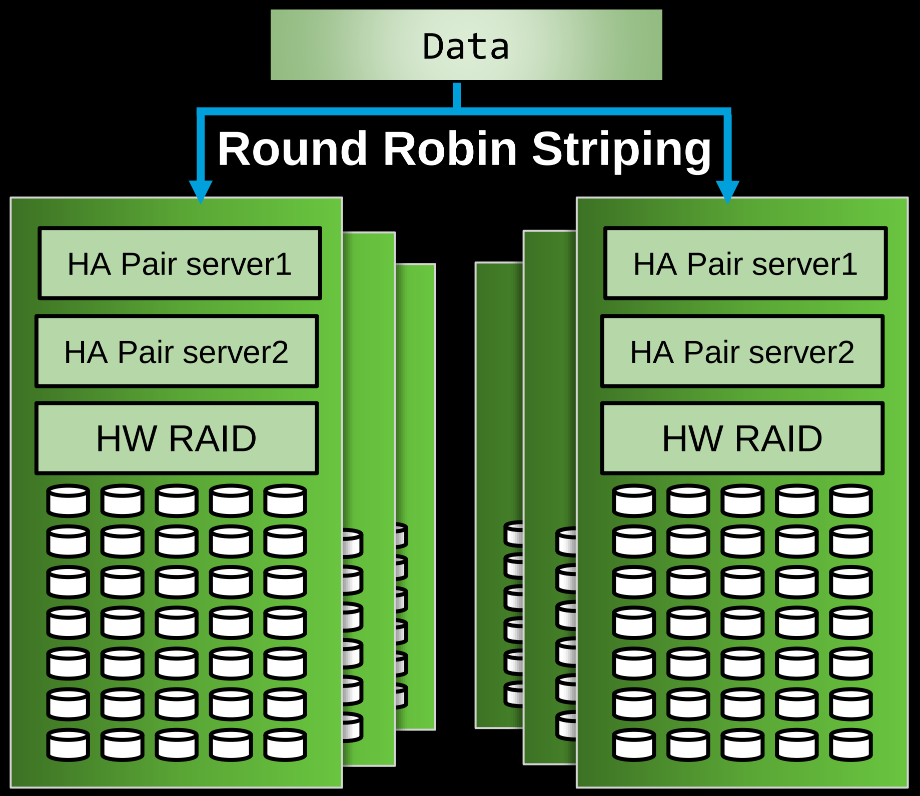
(https://cortx.link/declustered)

```
set terminal png font "Calibri,14" size 640,320

capacity_tb=20
capacity_mb=capacity_tb*1000*1000
hdd_mbps=200
seconds_in_hour=3600

amplification=10 # need to read 8 pieces and then write one
# for legacy, ignore amplification since you are just bound to one drive

dcr_rebuild_rate(x)=hdd_mbps*(x-1) # subtract out the failed drive
dcr_rebuild_hours(x)=((capacity_mb*amplification)/dcr_rebuild_rate(x)
legacy_rebuild_hours(x)=(capacity_mb/hdd_mbps)/3600

set yrange [0:30]
set xrange [10:170]

set key above
unset title
set ylabel 'Rebuild Time (hours)'
set xlabel 'Number of Drives'
set output 'declustered_rebuild.png'
plot dcr_rebuild_hours(x) lw 2 t "Declustered Erasure Groups", legacy
```

The Evolution of Data Durability in Data Centers

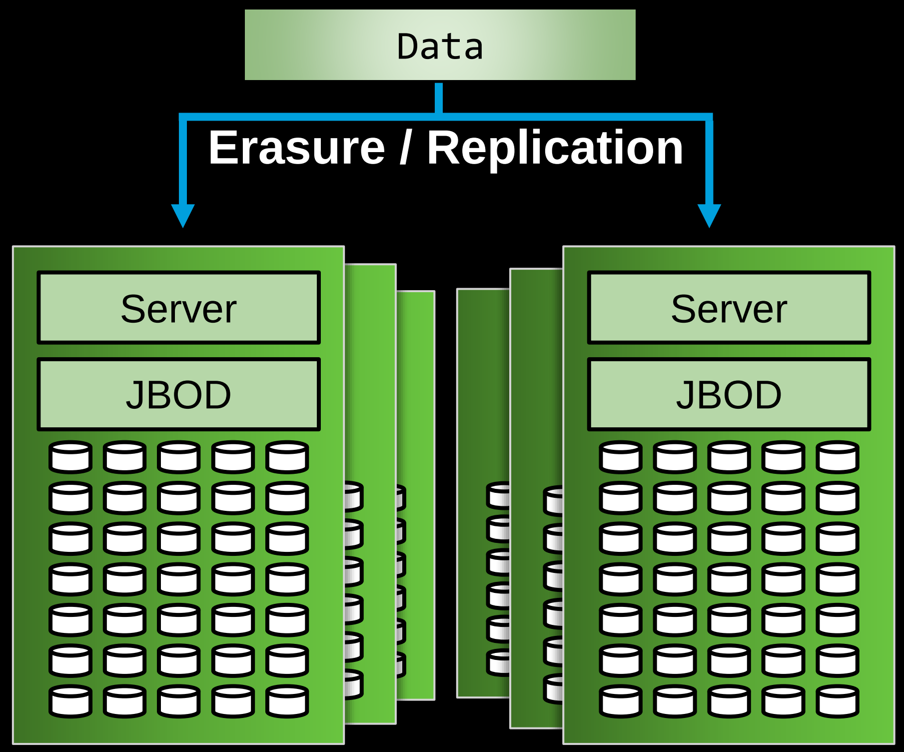


Two Existing Approaches for Data Durability and Availability



HW Local Declustered Erasure

Lustre, PVFS, BeeGFS, etc.



SW Network Declustered Erasure

Cloudian, ActiveScale, SwiftStack, Ceph, HDFS, etc.

Extreme Scale Requires Extreme Protection

Availability in Globally Distributed Storage Systems

Daniel Ford, François Labelle, Florentina I. Popovici, Murray Stokely, Van-Anh Truong,*
Luiz Barroso, Carrie Grimes, and Sean Quinlan
{ford,flab,florentina,mstokely}@google.com, vatruong@ieor.columbia.edu
{luiz,cgrimes,sean}@google.com
Google, Inc.

SPATIAL FAILURE BURST
Multiple simultaneous drive failures within a single rack.
Protect against these with erasure across enclosures.
Parity within enclosure is insufficient.

ASPATIAL FAILURE BURST
Multiple simultaneous drive failures across multiple racks.
Protect against these with erasure within enclosures.
Erasure across enclosures is insufficient..

No single tier of parity can protect against all these failures. Google knows about this and presumably had a team of PhD's implement a solution. Private cloud needs our help to solve this; we can do so with CORVAULT & tiered erasure.

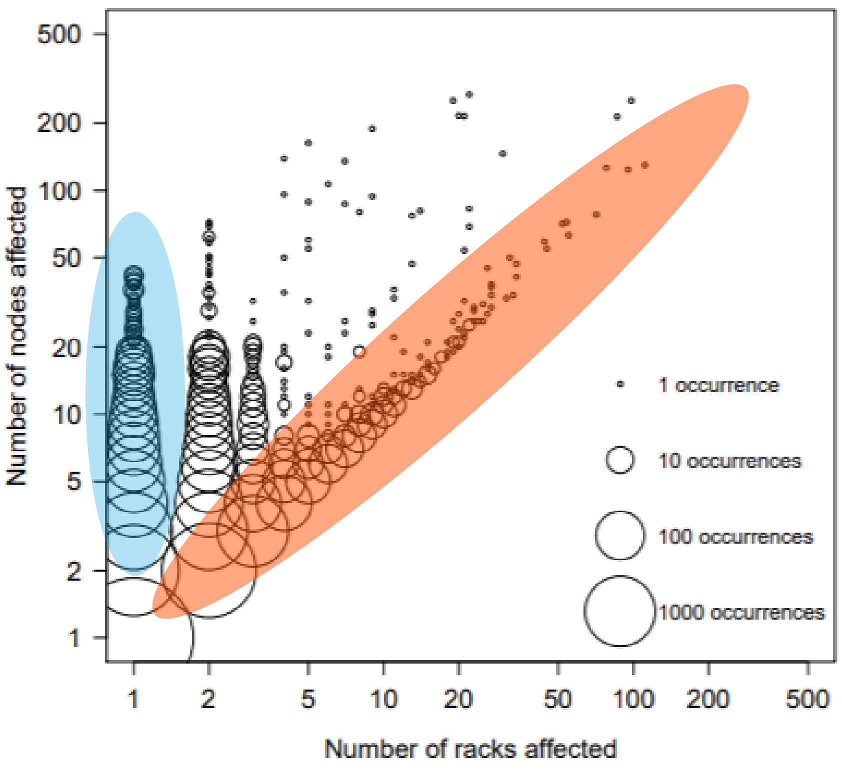


Figure 8: Frequency of failure bursts sorted by racks and nodes affected.

Two Existing Approaches for Data Durability and Availability

Data

Round Robin Striping

HA Pair server1

HA Pair server1

DURABILITY PROBLEM:
- Cannot protect against Aspatial Failure Bursts

AVAILABILITY PROBLEM:
- Cannot protect against HA server pair failure

COST PROBLEM:
- Hardware RAID is too expensive

HW Local Declustered Erasure

Lustre, PVFS, BeeGFS, etc.

Data

Erasure / Replication

DURABILITY PROBLEM:
- Cannot protect against Spatial Failure Bursts

AVAILABILITY PROBLEM:
- Server failure is handled but ‘blast radius’ is concern
- Typical configs recommend 2U12 – highly inefficient

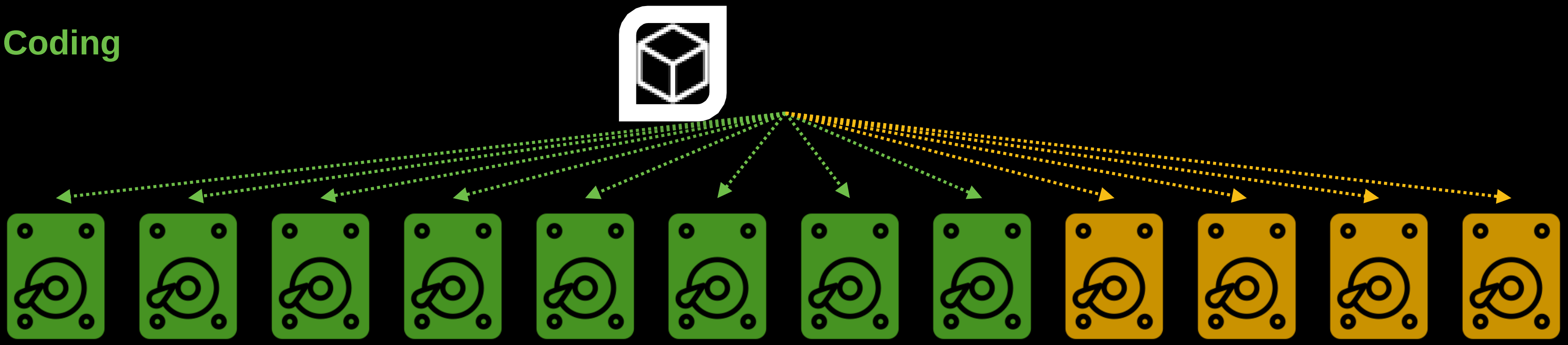
COST PROBLEM:
- Public cloud too expensive
- DIY too dangerous
- On-prem too expensive

SW Network Declustered Erasure

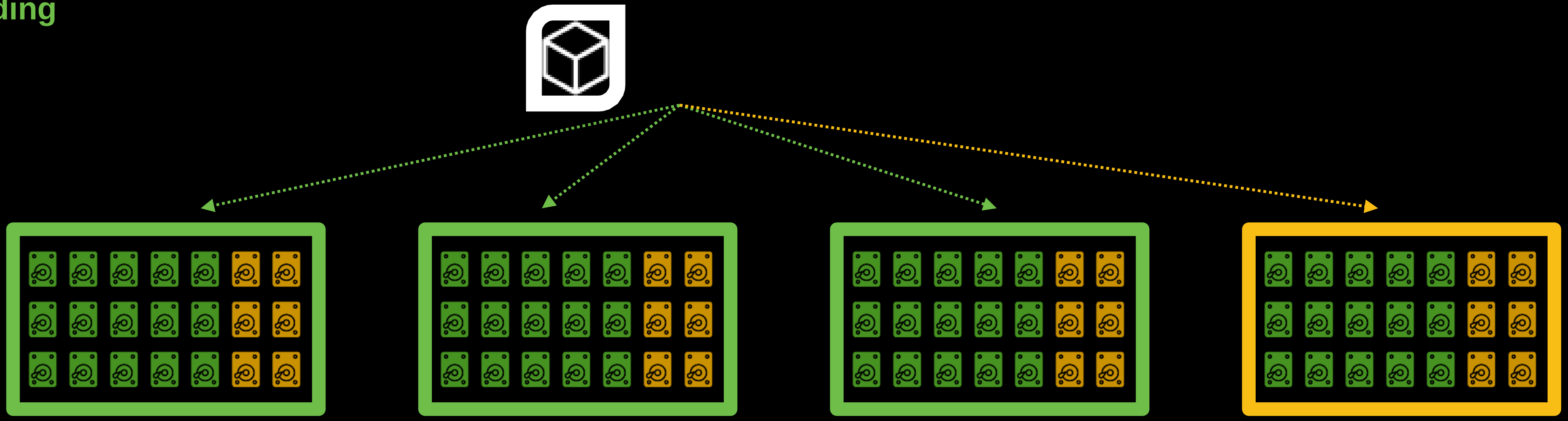
Cloudian, ActiveScale, SwiftStack, Ceph, HDFS, etc.

Multilevel Erasure Coding

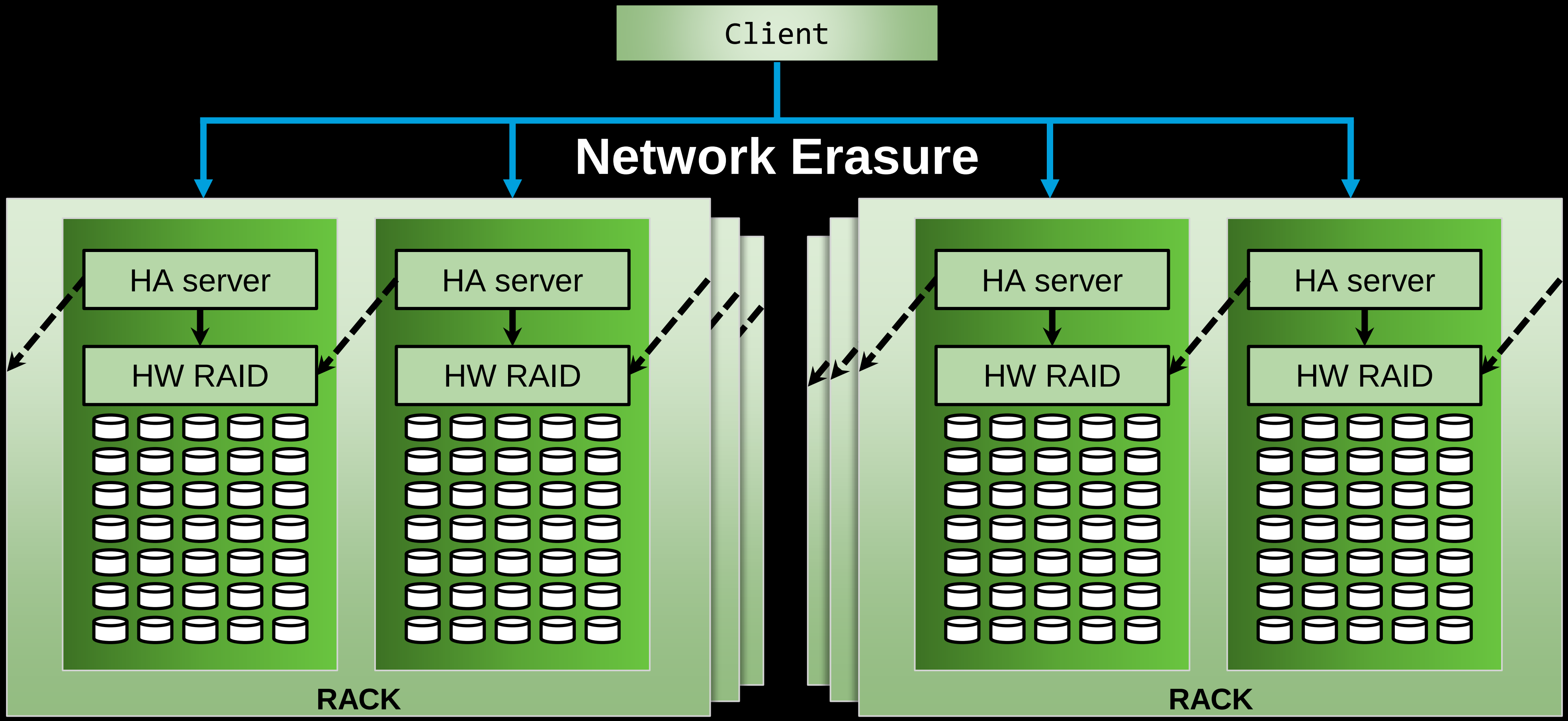
Single Level Erasure Coding



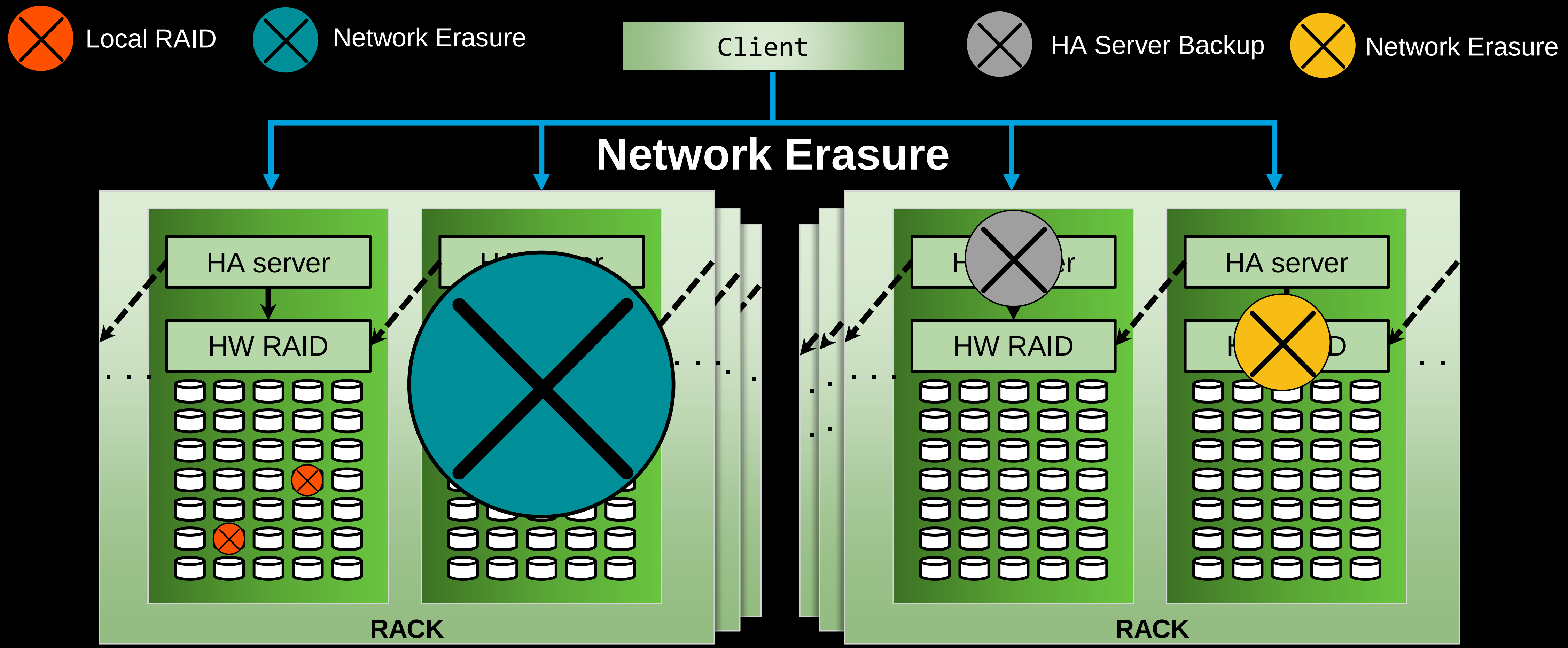
Hierarchical Erasure Coding



Hybrid Approach: Two Layers of Erasure

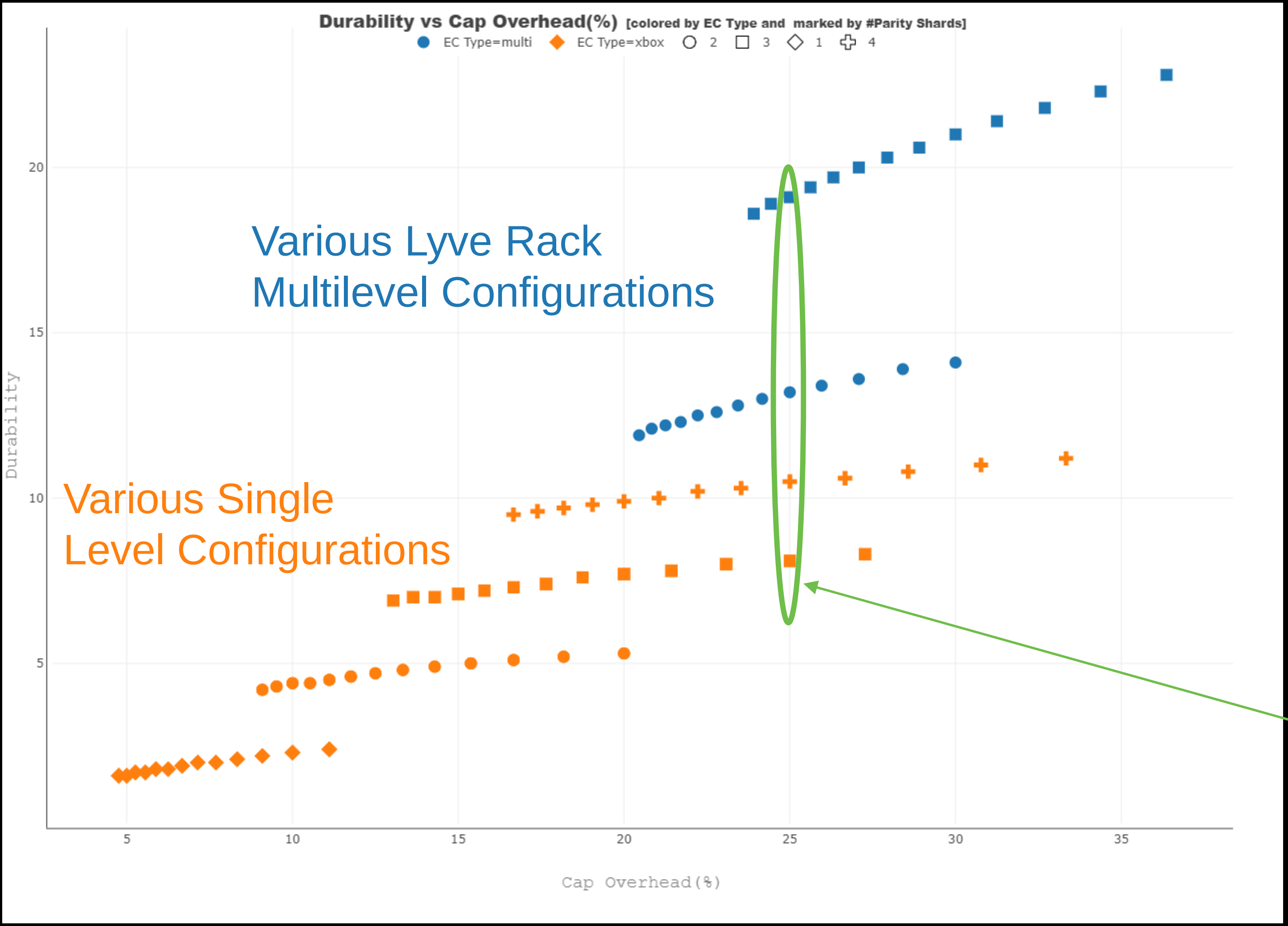


Hybrid Approach: Two Layers of Erasure



CORTX Hybrid Approach Maximizes Durability, Availability, and Density

The Value of Multilevel Erasure Coding



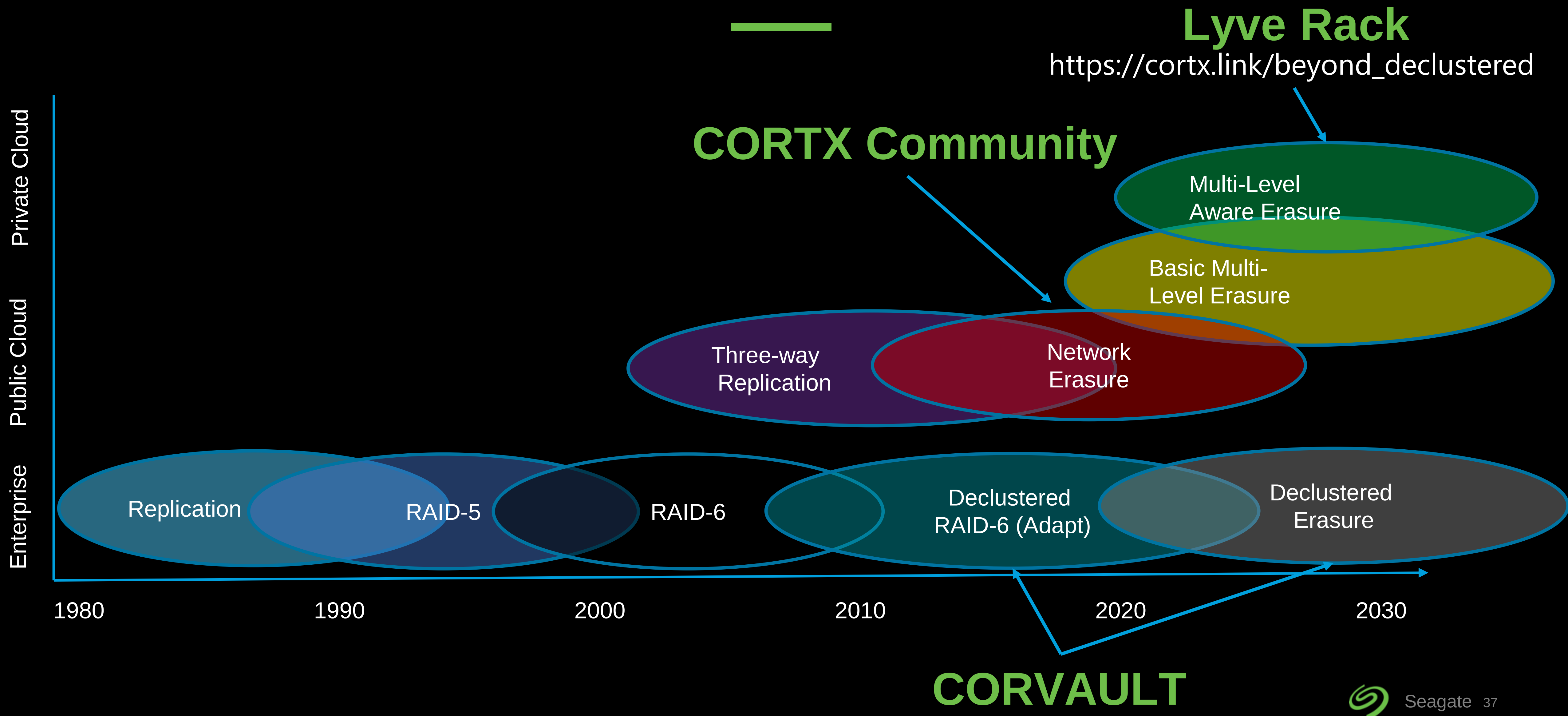
Basic rule: add more overhead to improve durability

Advanced rule: use multilevel to improve durability *without* increasing overhead

At comparable capacity overhead, the multilevel configuration provides many more 9's of durability.

S. Ölmez, "Reliability Analysis of Storage Systems With Partially Repairable Devices," https://ieeexplore.ieee.org/document/9424170

The Evolution of Data Durability in Data Centers



Thanks!

john.bent@seagate.com

https://cortx.io