Los Alamos
NATIONAL LABORATORY

EST. 1943

Delivering science and technology
to protect our nation
and promote world stability

NNSA
National Nuclear Security Administration

Operated by Los Alamos National Security, LLC for the U.S. Department of Energy's NNSA

MarFS

Tiered Parity: When Flat Doesn't Fit

David Bonnie

May 2017

Overview

- **Types of drive failure**
  - Random drive failure
  - Correlated drive failure
    - Spacially versus non-spacially correlated
  - Systemic drive failure
- **“Flat” RAID / Erasure**
  - When does it make sense?
  - What happens as we scale up?
- **Tiered Erasure**
  - How to perform efficiently?
  - Our solution in MarFS

Drive failures -

- **Random drive failure:**
  - Generally due to age, defects, trauma
  - Handful of drives at a time, generally not correlated with any particular event
  - Traditional schema handle this quite well, especially well in the case of distributed rebuild
  - In modern systems, not a huge driver for system design unless rebuilds are slow or failure domain is very large

Drive failures -

- **Correlated drive failure:**
  - A batch of failures that have a common root
  - Correlated in space –
    - Server failure, local corruption, fire, electrical event, etc
    - All contained within a small special area
    - This can be mitigated with clever placement algorithms
  - Correlated in time –
    - Drive failure due to an electrical event (power loss) or physical trauma (jolt of some sort)
    - Failures all over the system, no real locality
    - This is *not* solved by clever placement algorithms – can be avoided with very strong erasure, replication, multi-site storage…but this is expensive in both storage and compute

Drive failures -

- **Systemic drive failure**
  - Firmware problems, bad batch of drives, bugs in storage stack…
  - The only way to avoid these if they happen quickly is through very costly measures that aren't available in many environments
    - Replication across sites with different hardware
    - Hardware variety such that any one source doesn't break protection scheme
    - Copy everything to offline storage and have a good way to get all of it back quickly
  - Basically, unless you have unlimited resources, give up. ☺

“Flat” protection schema

- **Typical RAID sets (+2/+3 protection)**
  - Great for small scale, but if striping over many sets, failure is almost guaranteed
  - If not striping over many sets, no performance or capacity
- **Distributed erasure**
  - Each object or block is encoded to some K+M, then placed within the system
  - No related chunks on any drive, no related chunks on the same server if desired
  - Hashed layout of K+M over the set of drives -> as the system scales, M+1 failures essentially guarantee some data loss
  - Restricted placement can help (smaller failure domains)

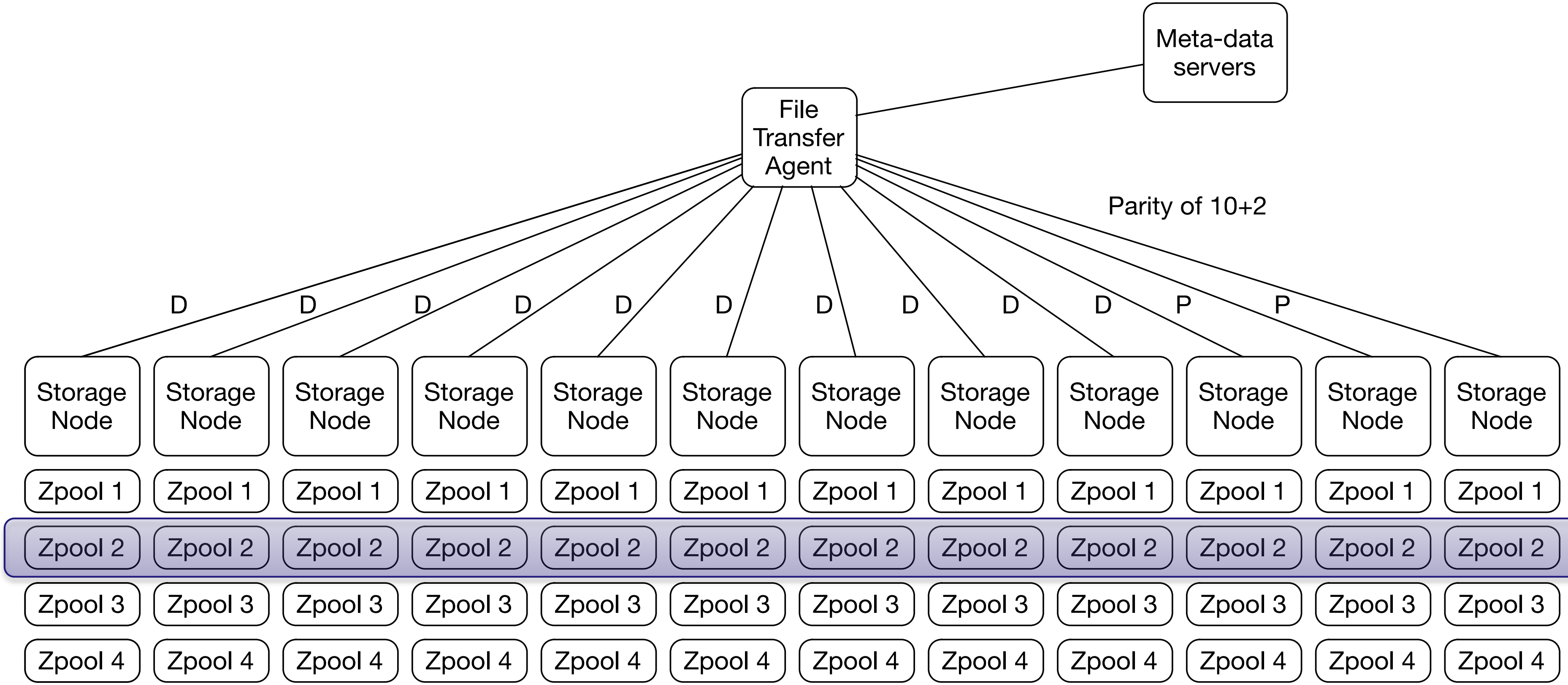
Tiered Erasure

- **Why?**
  - As +M increases, computation cost increases as well
  - Very large values of M offer great protection but require large data blocks and expensive/slow computation beyond what current CPUs are capable of
- **What?**
  - “Multi-Component Repositories” in MarFS speak
  - Top level:
    - Intel ISA-L for AVX-optimized K+M at the top level (10+2 in our deployment)
    - Storage clients operate on 1 GB, chunk it into 10x 100 MB data blocks+ 2x 100 MB parity blocks
  - Bottom level:
    - ZFS RAIDZ3 at the bottom level (17+3) using AVX
    - 100 MB chunks are large enough for efficient storage bandwidth

Tiered Erasure

- **Benefits:**
  - Tightly controlled failure domains (very configurable)
  - Rebuilds kept local except for catastrophic failure at bottom level
  - Vastly improved protection against the “shotgun effect” of scattered drive failure after an event
- **Tradeoffs:**
  - Not very applicable to small-block write or random write (latency + RMW)
  - If a data loss event does happen, it affects more of the data instead of a small subset
  - Potentially higher storage overhead due to multiple layers
    - In our case, ~30% overhead on 240 drives, equal to 170+70 overhead flat

Tiered Erasure Example



Each Zpool is a 17+3

Storage nodes in separate racks

Multiple JBODs per Storage Node

Data and Parity are round-robined to storage nodes

Storage Nodes NFS export to FTAs

Questions?