



# Designing Ceph Clusters: What's the Right Durability?

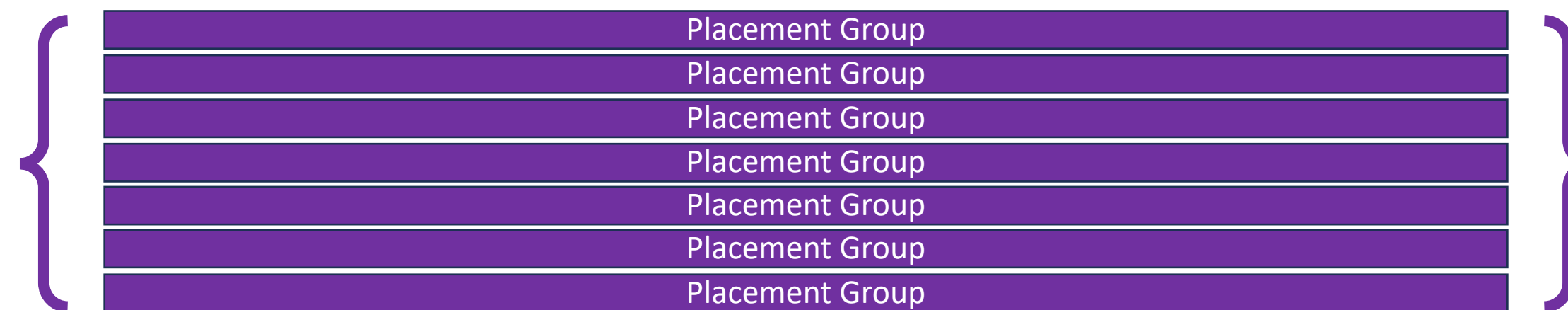
Ceph Days Seattle 2026

Steve Umbehocker, CTO/CEO, OSNexus

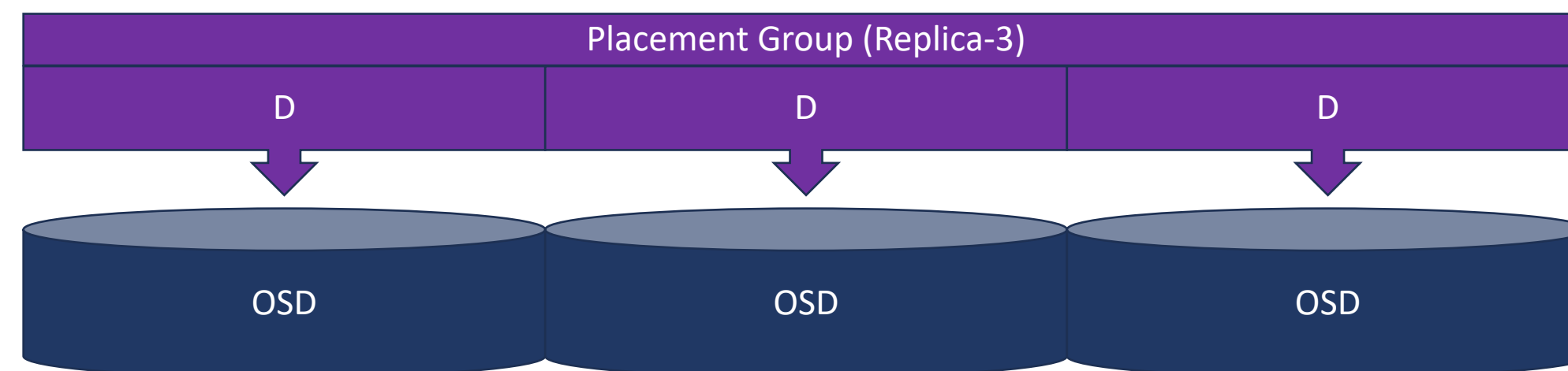
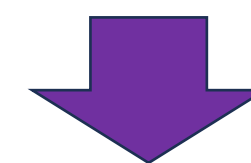
# Understanding the Layers



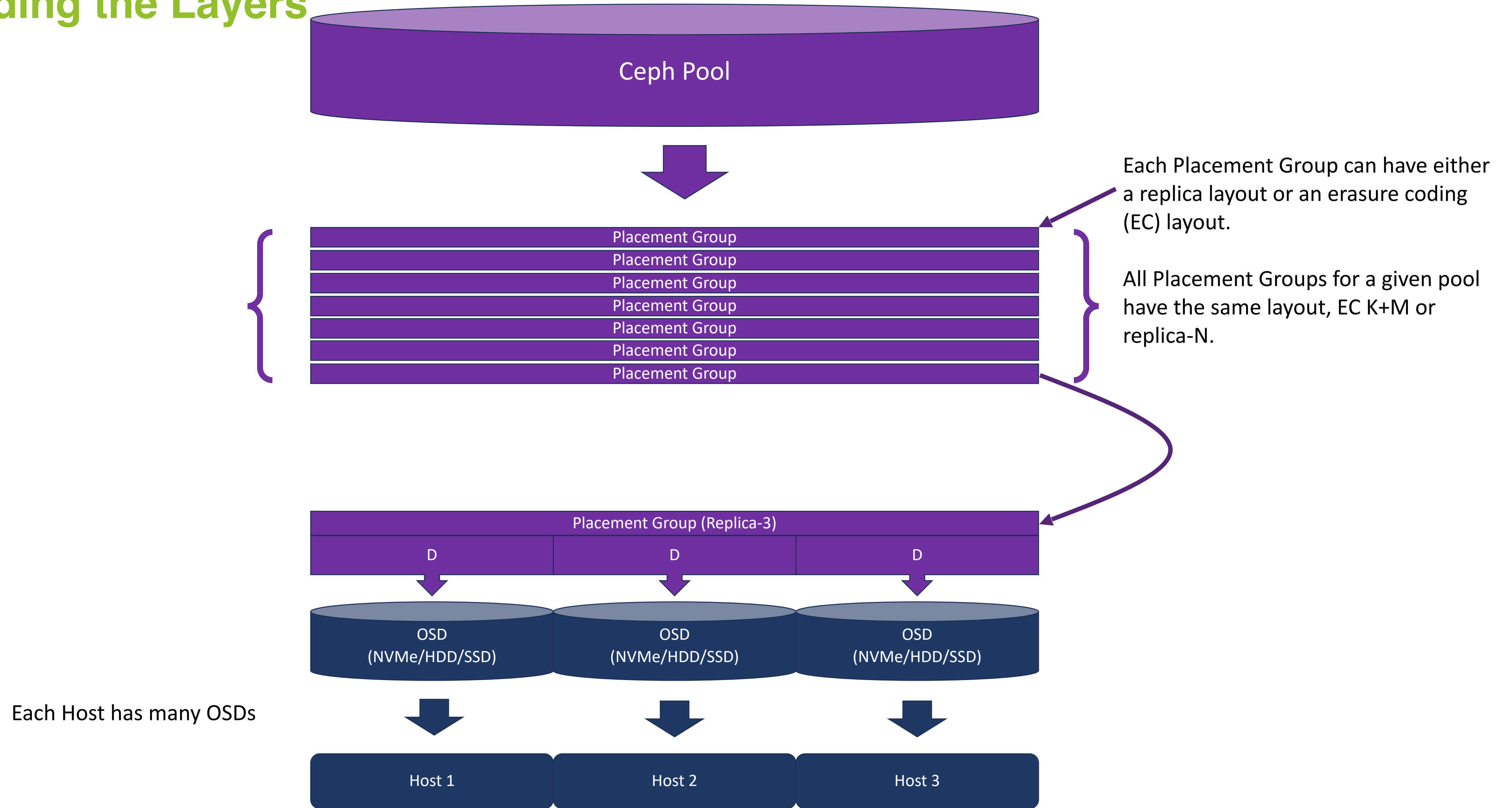
Each Ceph Pool is comprised of many placement groups



Each Placement Group references a series of OSDs

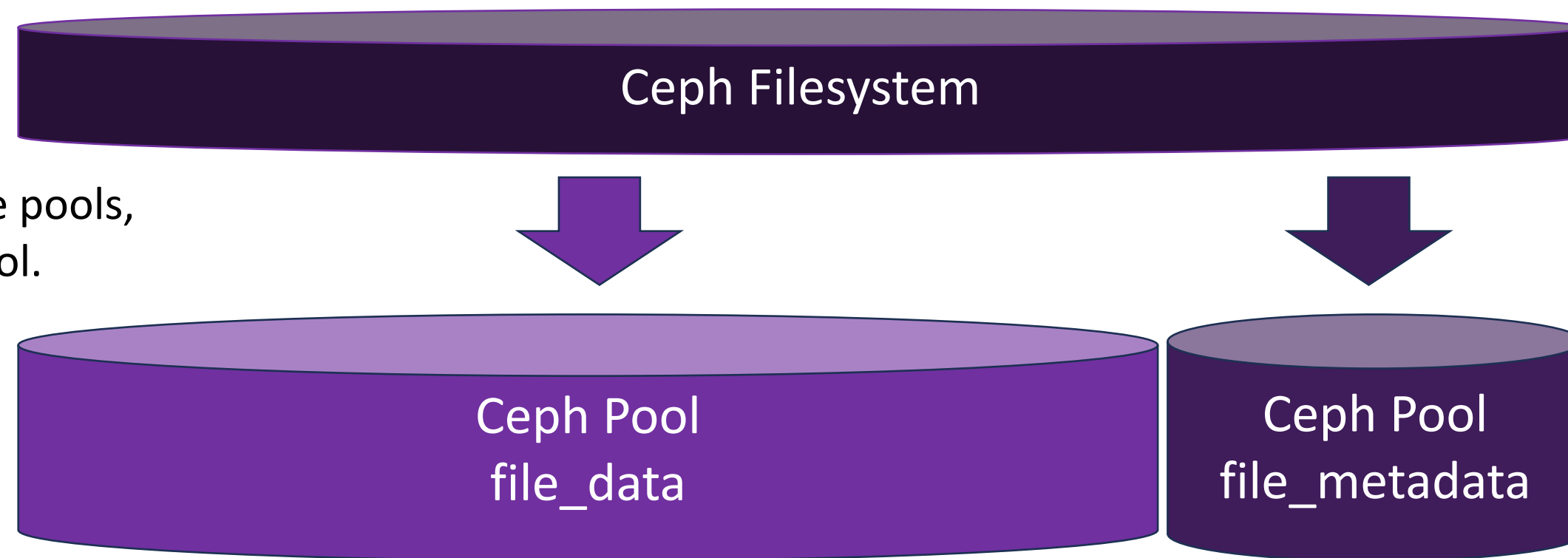


# Understanding the Layers

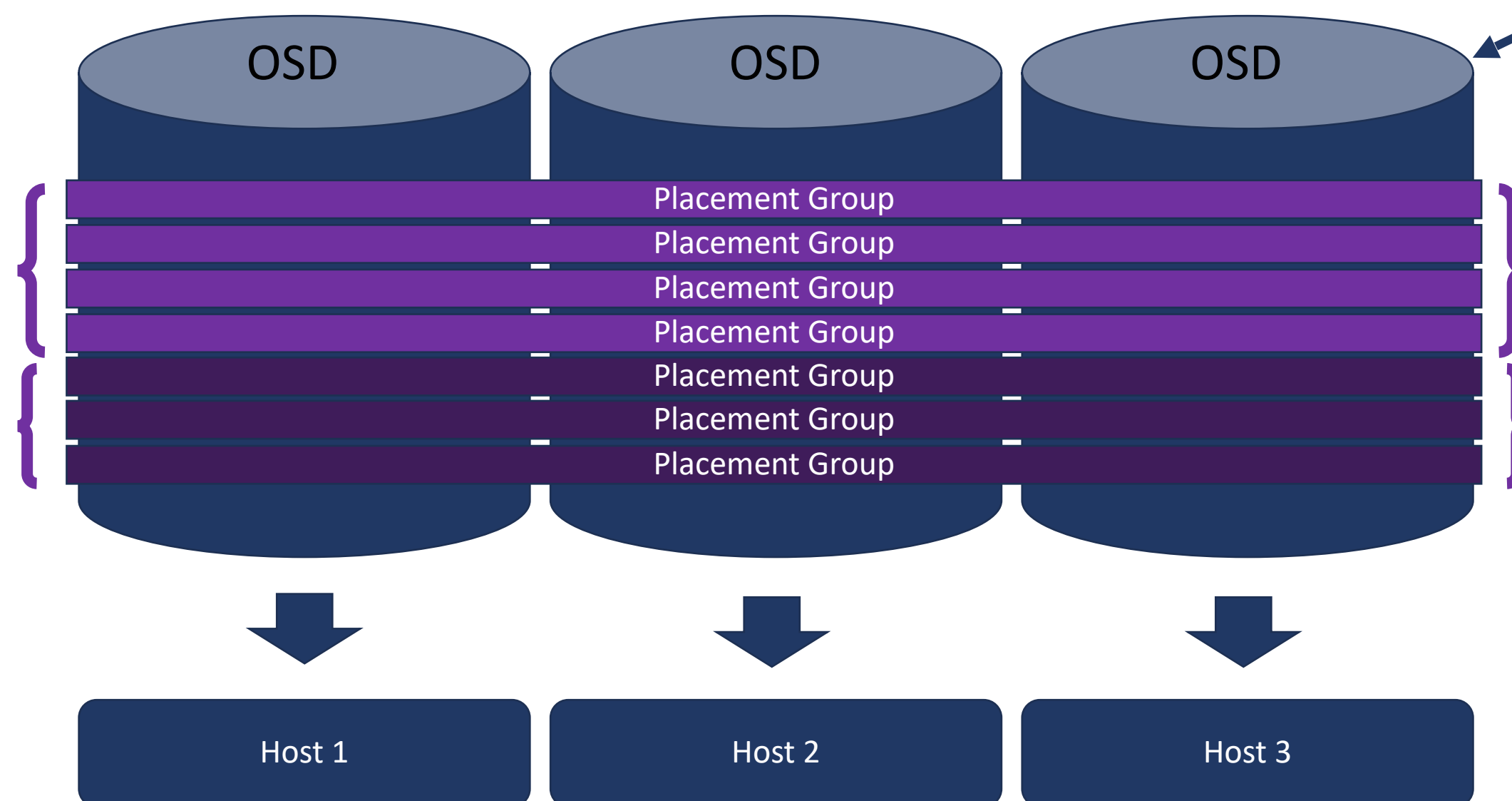


## Understanding the Layers

Ceph Filesystems are made up of multiple pools, at least one data pool and a metadata pool.

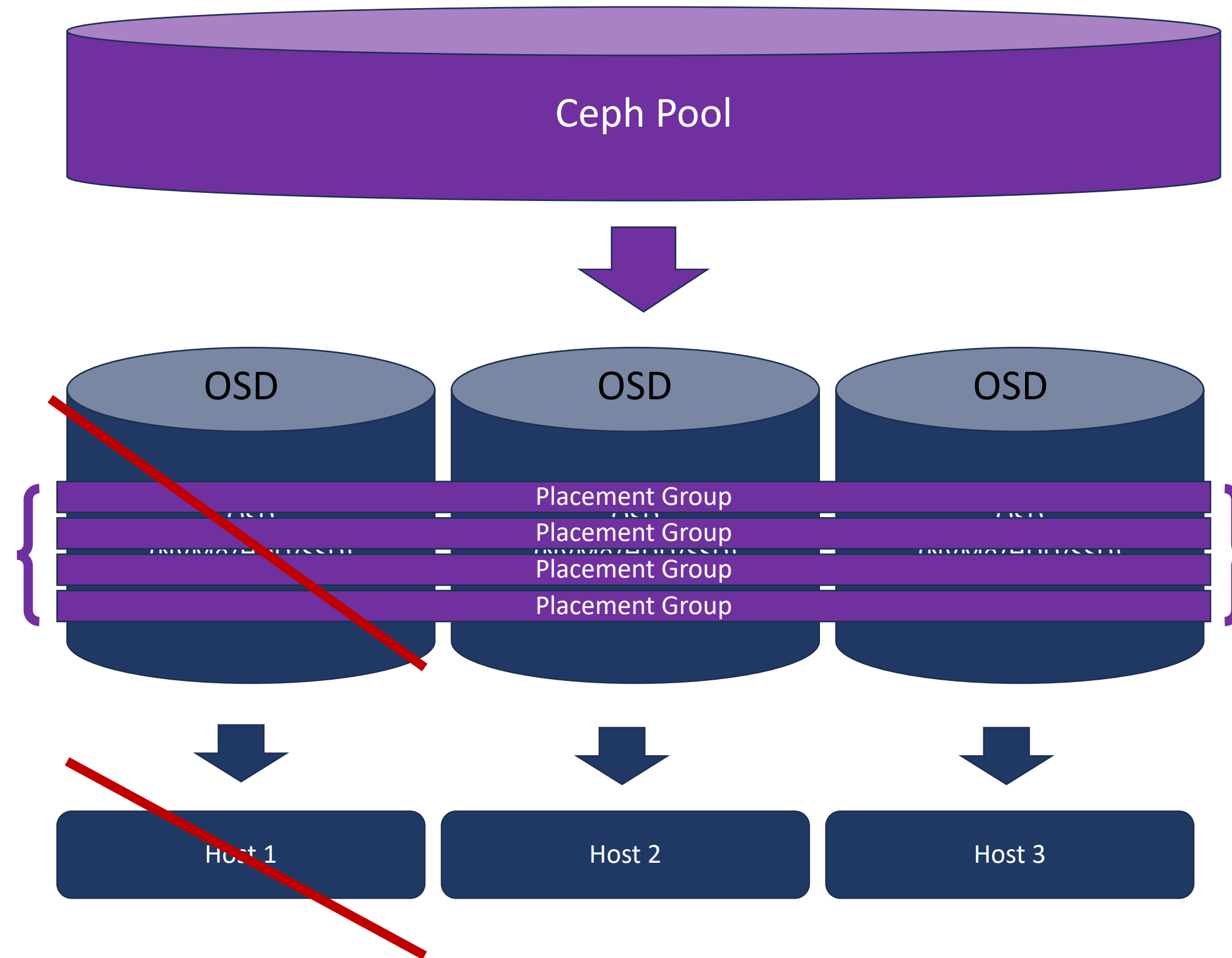


OSDs are a shared resource and are often used by many pools.

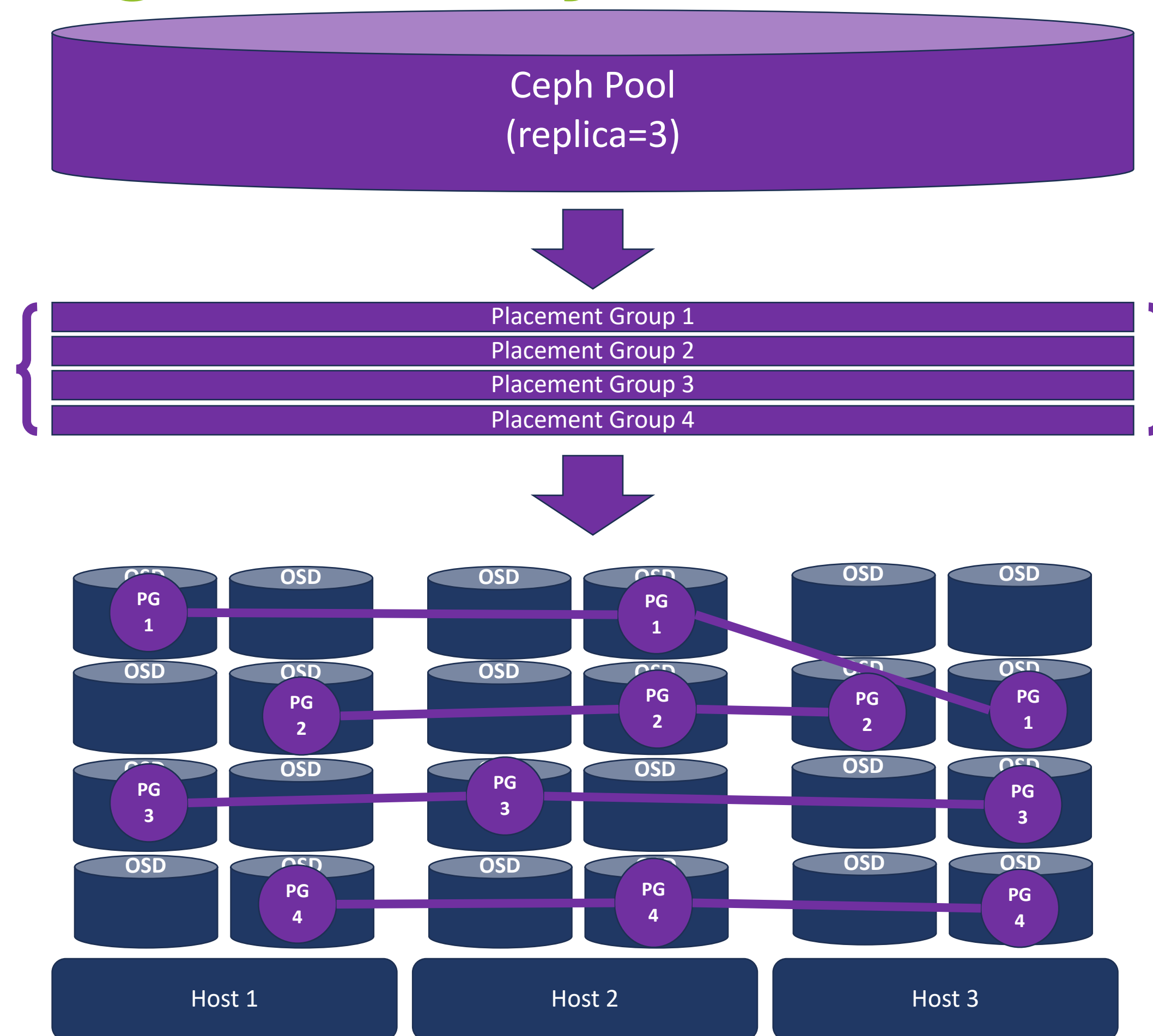


OSDs may be created from any media type including NVMe, SAS, NL-SAS, and SATA. Only enterprise and/or datacenter grade media should be used.

# Understanding Durability



# Understanding Durability



## The Core Formula

- PG Length = Replica count or K+M for EC pools
- Base PG count ( $PC_{base}$ ) =  $(100 * OSDs) / (PG_{length})$
- Scaled PG count ( $PC_{scaled}$ ) =  $PC_{base} * 50\%$
- Total PG count =  $PC_{scaled}$  rounded up (nearest base 2)
- PGs per OSD =  $PC_{total} / (OSDs / PG_{length})$

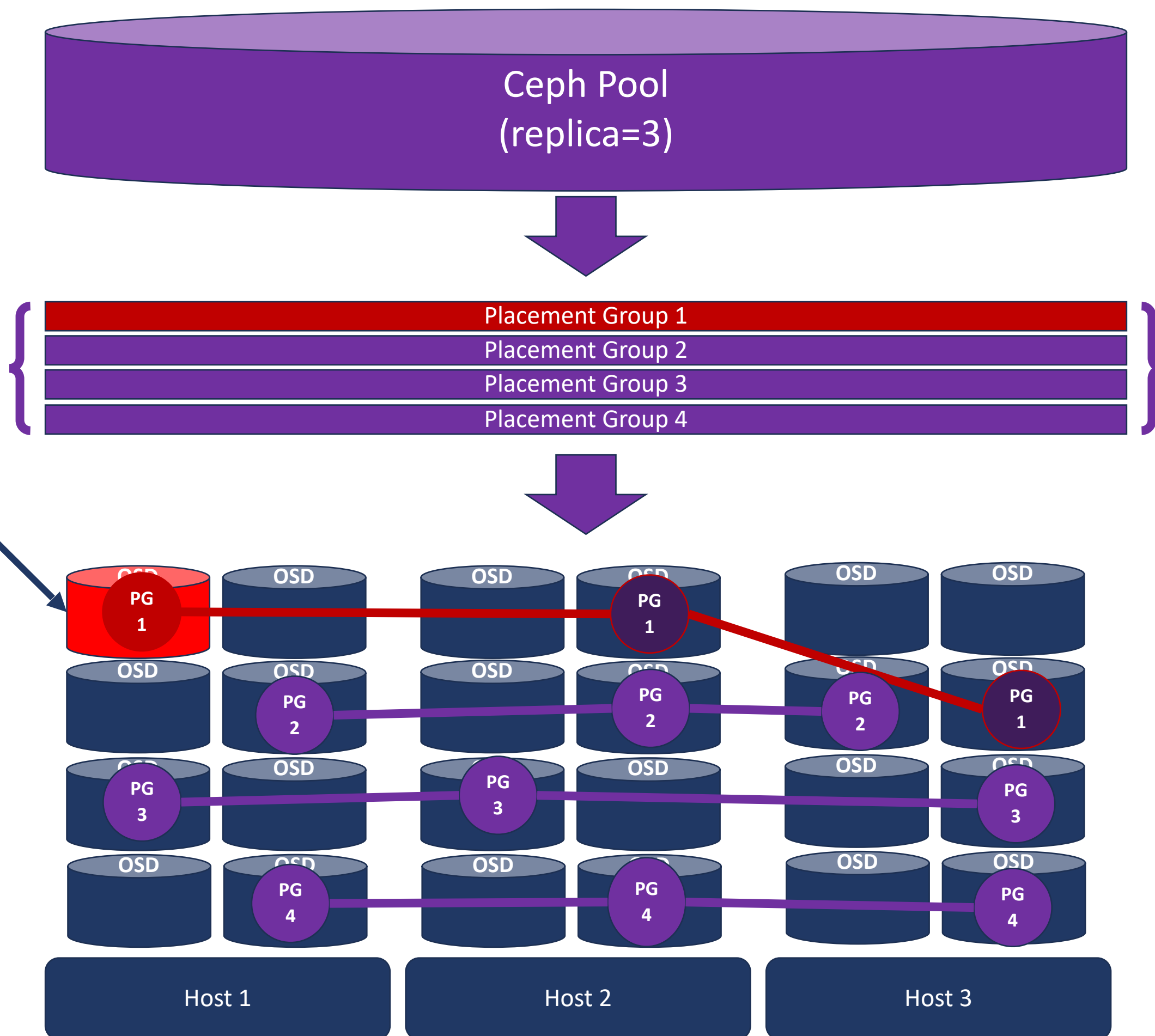
## Example

- PG Length = 3
- Base PG count ( $PC_{base}$ ) =  $(100 * 24) / 3 = 800$
- Scaled PG count ( $PC_{scaled}$ ) =  $PC_{base} * 50\% = 400$
- Total PG count =  $PC_{scaled}$  rounded up (nearest base 2) = 512
- PGs per OSD =  $512 / (24 / 3) = 512 / 8 = 64$

```
ceph osd pool create <pool-name> <pg_num> <pgp_num> replicated [crush-rule-name]
```

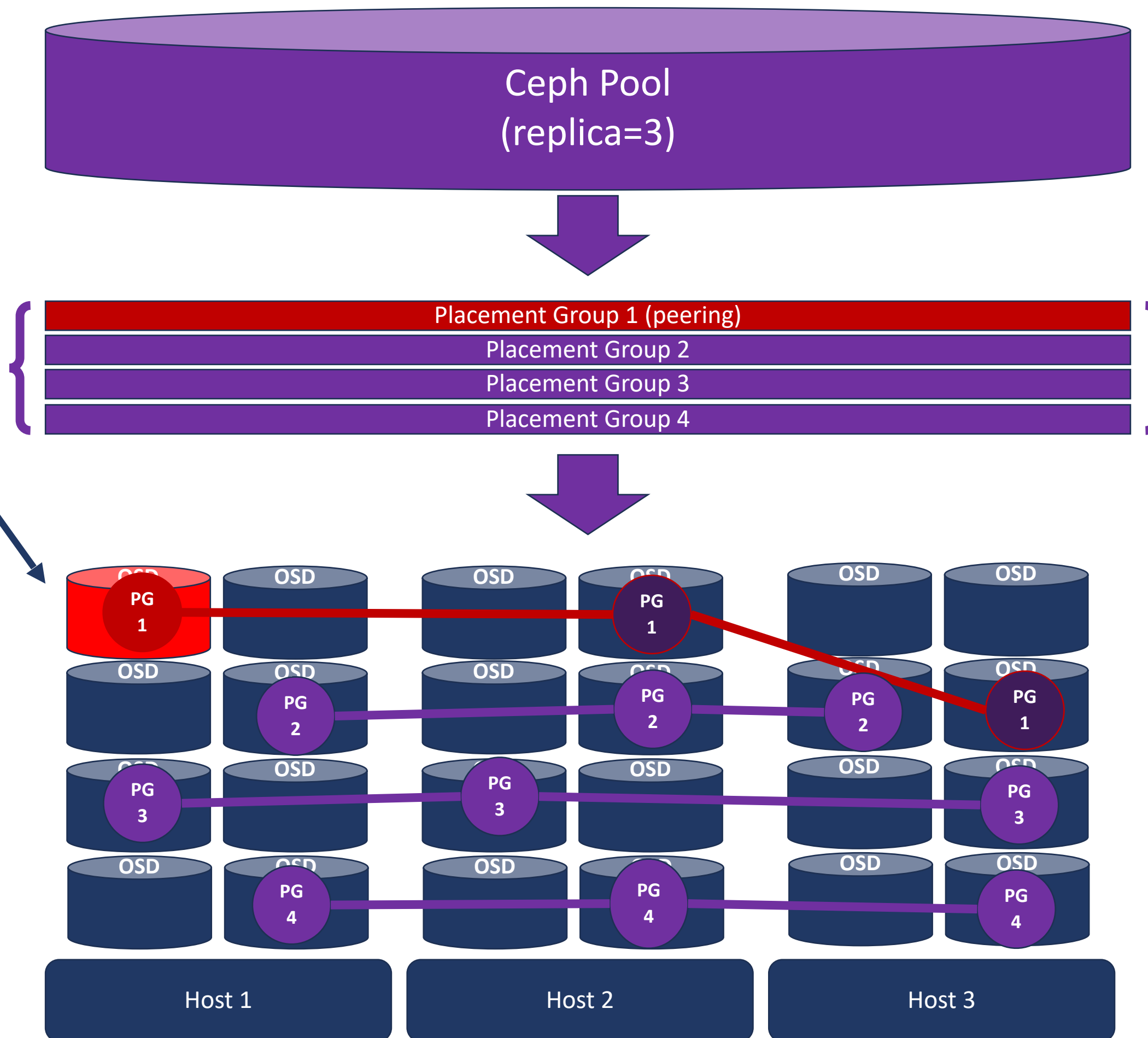
# Understanding Durability

1. When the media underlying an OSD fails it's initially marked **'down'**.
2. There's a brief window where the affected PGs will go into a **'peering'** state, usually just a few seconds but during this time I/O is paused.
3. After 10 minutes the OSD is marked **'out'**. This causes the degraded PGs to be remapped to use the remaining good OSDs.
4. If there are not enough OSDs remaining in a PG (**min\_size param**) the pool will stop.



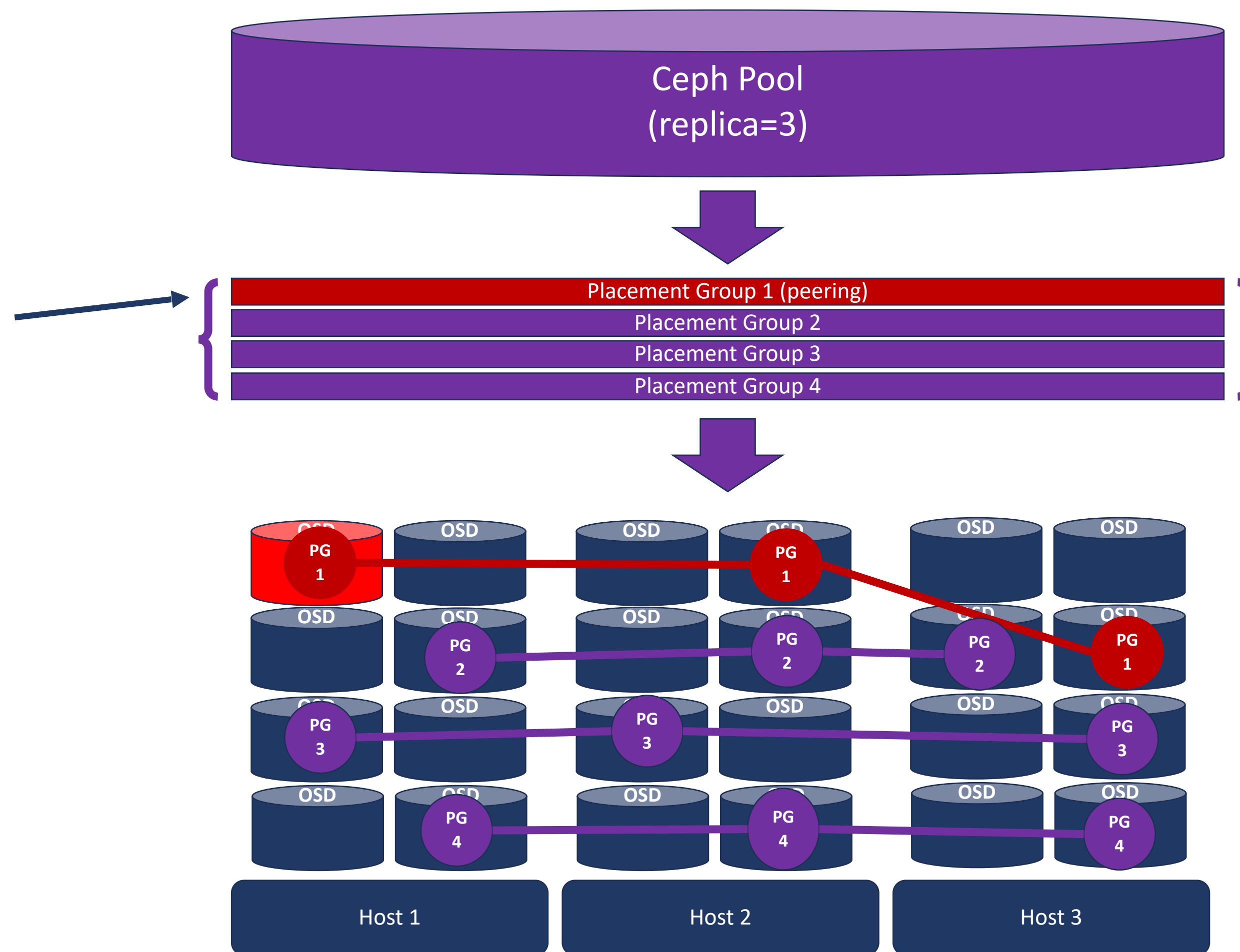
# Understanding Durability

1. When the media underlying an OSD fails it's initially marked **'down'**.
2. There's a brief window where the affected PGs will go into a **'peering'** state, usually just a few seconds but during this time I/O is paused.
3. After 10 minutes the OSD is marked **'out'**. This causes the degraded PGs to be remapped to use the remaining good OSDs. The affected PGs are now **'active+degraded'**
4. PGs will transition to a composite state like **'active+undersized+degraded+remapped+backfilling'** as the data is being replicated to other OSDs to bring the cluster health back to 100%
5. If there are not enough OSDs remaining in a PG (**min\_size param**) the pool will stop.



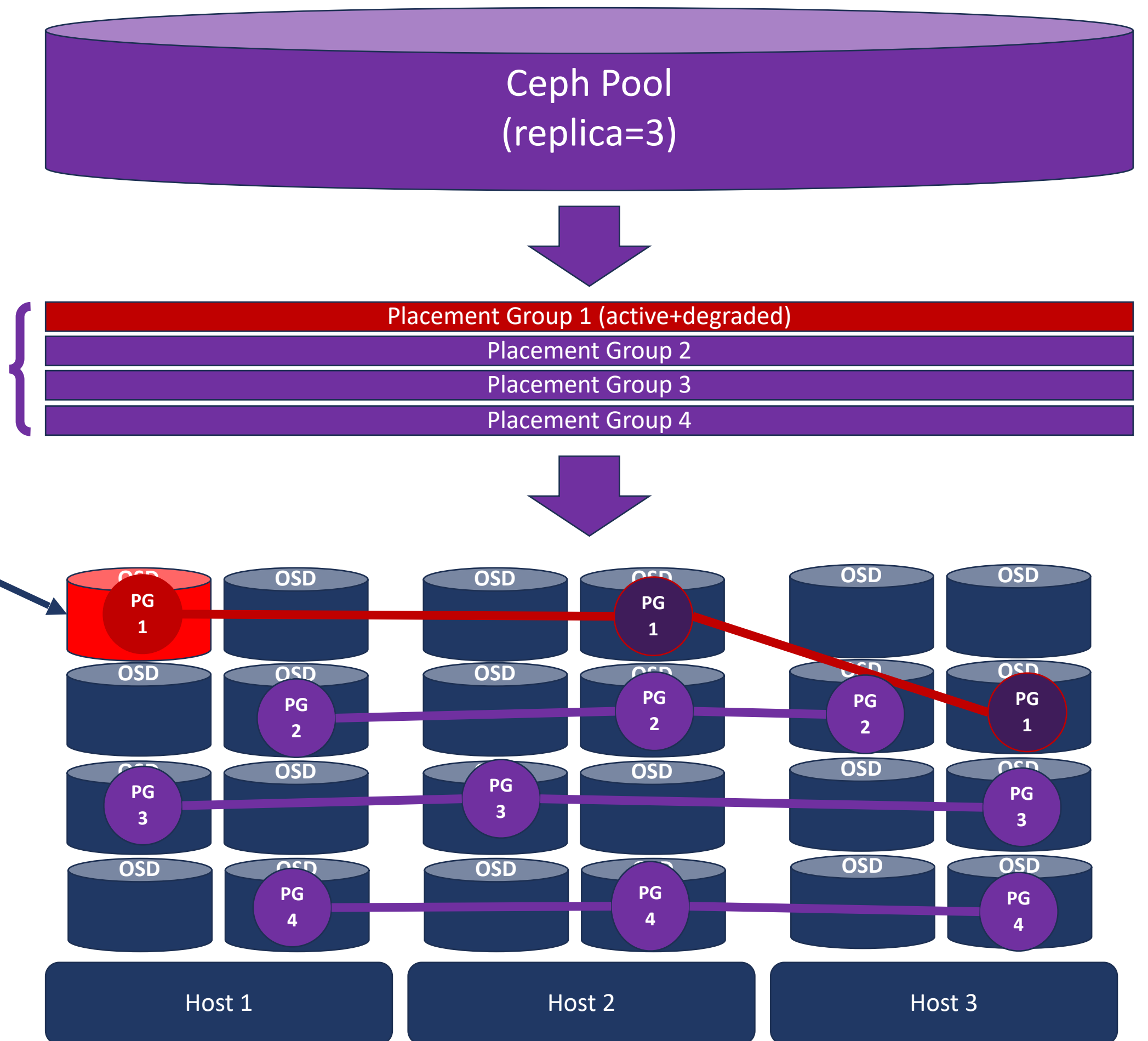
# Understanding Durability

1. When the media underlying an OSD fails it's initially marked **'down'**.
2. There's a brief window where the affected PGs go into a **'peering'** state, usually just a few seconds but during this time I/O is paused.
3. After 10 minutes the OSD is marked **'out'**. This causes the degraded PGs to be remapped to use the remaining good OSDs. The affected PGs are now **'active+degraded'**
4. PGs will transition to a composite state like **'active+undersized+degraded+remapped+backfilling'** as the data is being replicated to other OSDs to bring the cluster health back to 100%
5. If there are not enough OSDs remaining in a PG (**min\_size param**) the pool will stop.



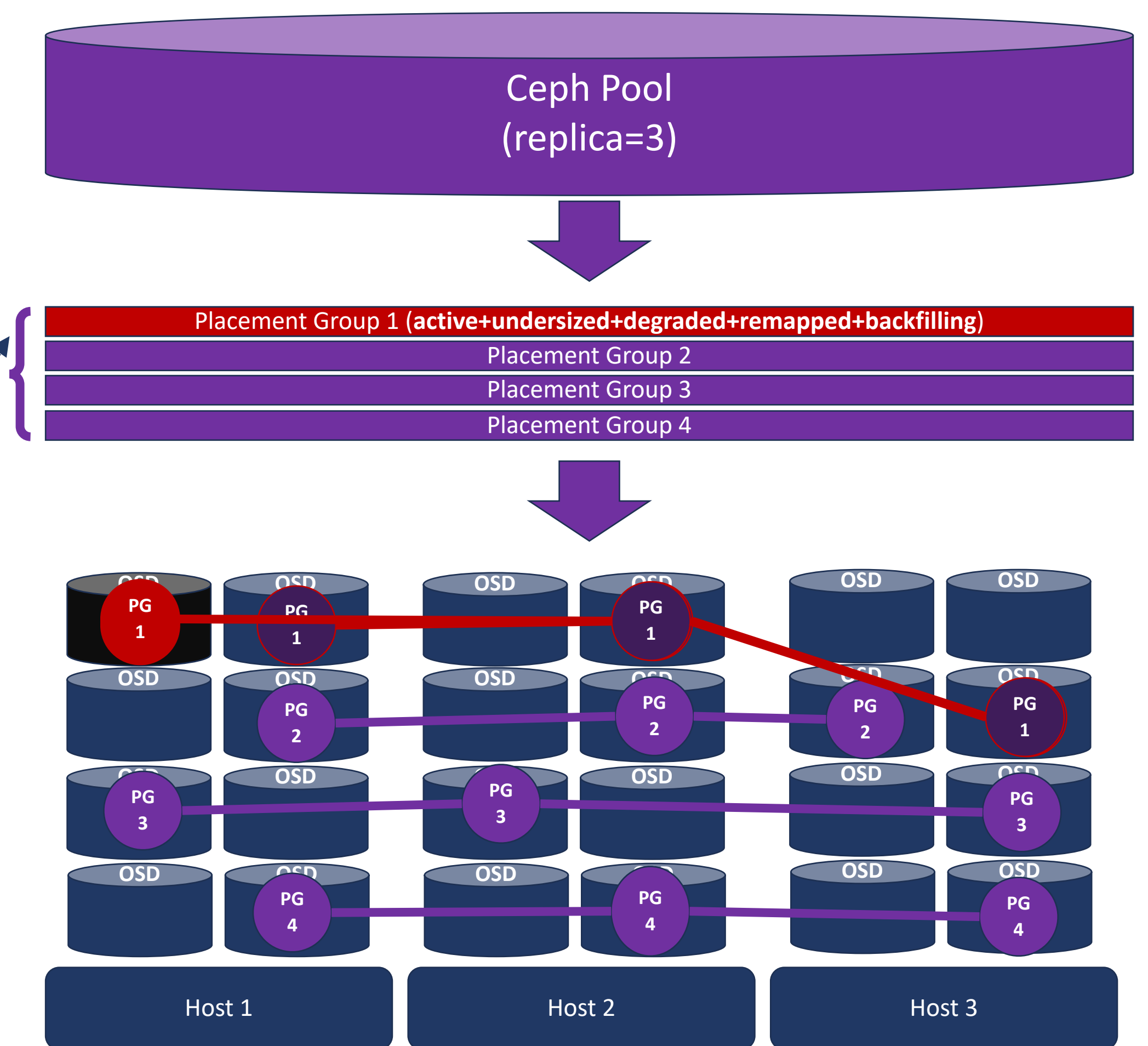
# Understanding Durability

1. When the media underlying an OSD fails it's initially marked **'down'**.
2. There's a brief window where the affected PGs will go into a **'peering'** state, usually just a few seconds but during this time I/O is paused.
3. After 10 minutes the OSD is marked **'out'**. This causes the degraded PGs to be remapped to use the remaining good OSDs. The affected PGs are now **'active+degraded'**
4. PGs will transition to a composite state like **'active+undersized+degraded+remapped+backfilling'** as the data is being replicated to other OSDs to bring the cluster health back to 100%
5. If there are not enough OSDs remaining in a PG (**min\_size param**) the pool will stop.



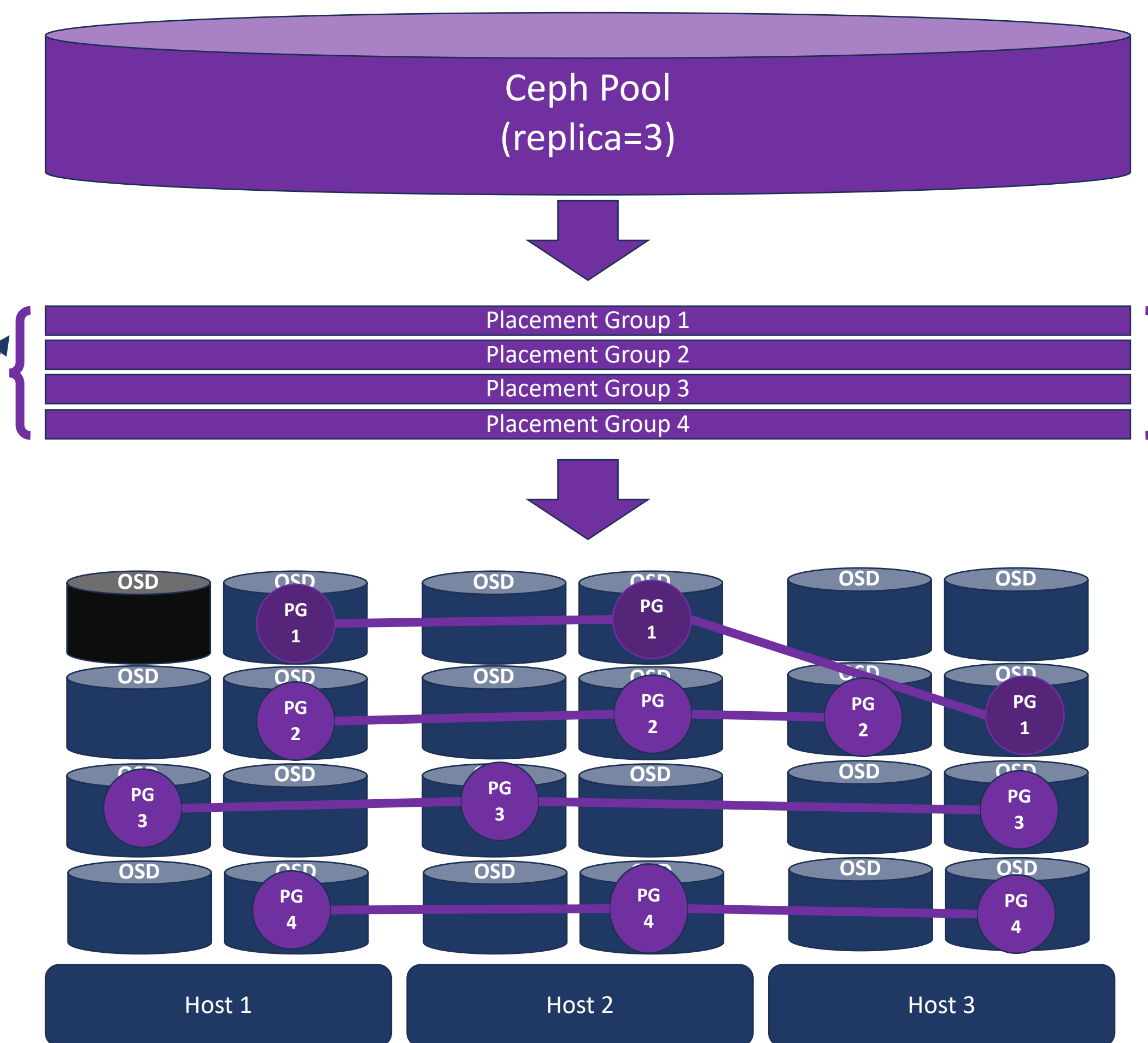
# Understanding Durability

1. When the media underlying an OSD fails it's initially marked **'down'**.
2. There's a brief window where the affected PGs will go into a **'peering'** state, usually just a few seconds but during this time I/O is paused.
3. After 10 minutes the OSD is marked **'out'**. This causes the degraded PGs to be remapped to use the remaining good OSDs. The affected PGs are now **'active+degraded'**
4. PGs will transition to a composite state like **'active+undersized+degraded+remapped+backfilling'** as the data is being replicated to other OSDs to bring the cluster health back to 100%
5. If there are not enough OSDs remaining in a PG (**min\_size param**) the pool will stop.



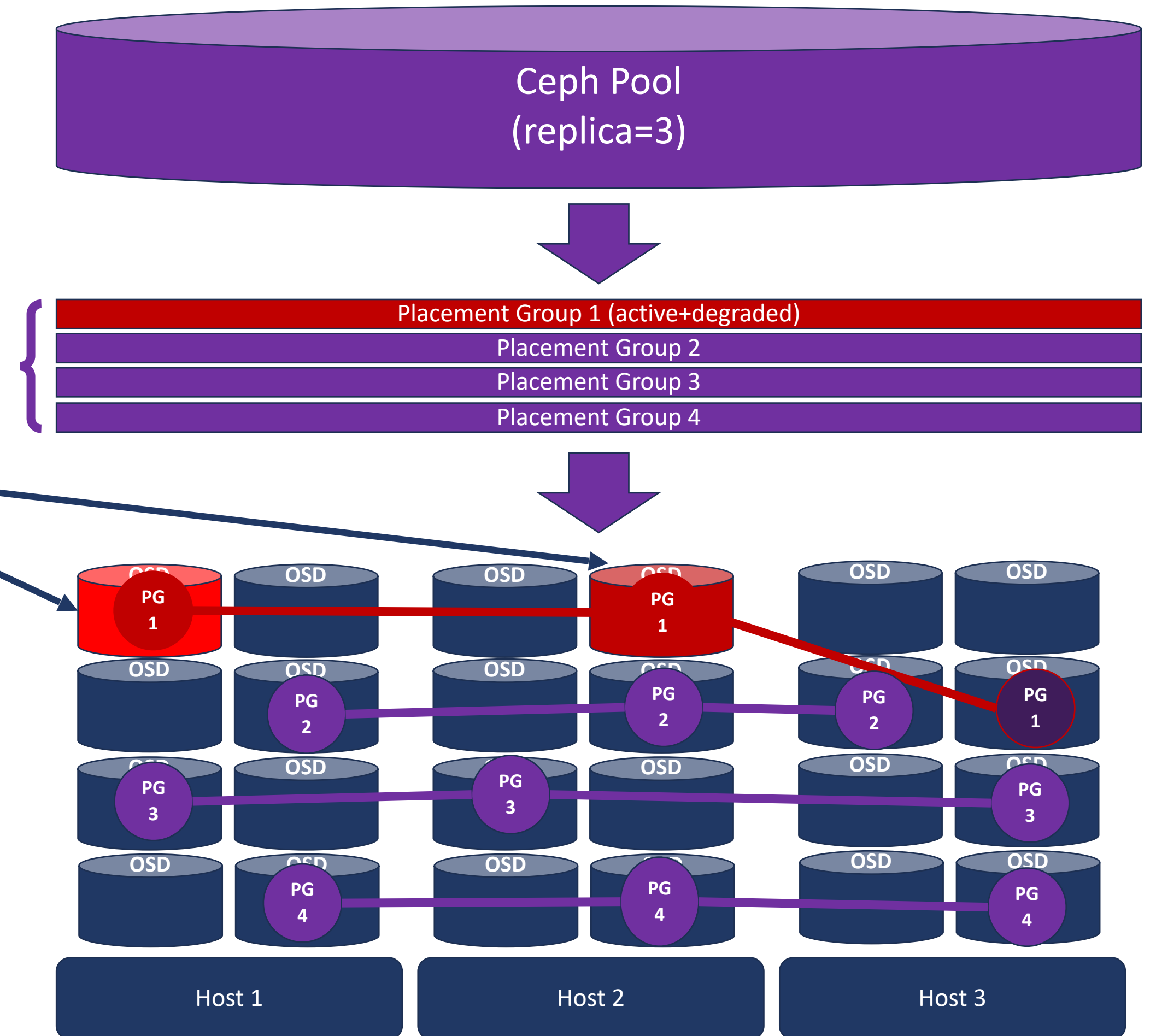
# Understanding Durability

1. When the media underlying an OSD fails it's initially marked **'down'**.
2. There's a brief window where the affected PGs will go into a **'peering'** state, usually just a few seconds but during this time I/O is paused.
3. After 10 minutes the OSD is marked **'out'**. This causes the degraded PGs to be remapped to use the remaining good OSDs. The affected PGs are now **'active+degraded'**
4. PGs will transition to a composite state like **'active+undersized+degraded+remapped+backfilling'** as the data is being replicated to other OSDs to bring the cluster health back to 100%
5. If there are not enough OSDs remaining in a PG (**min\_size param**) the pool will stop.

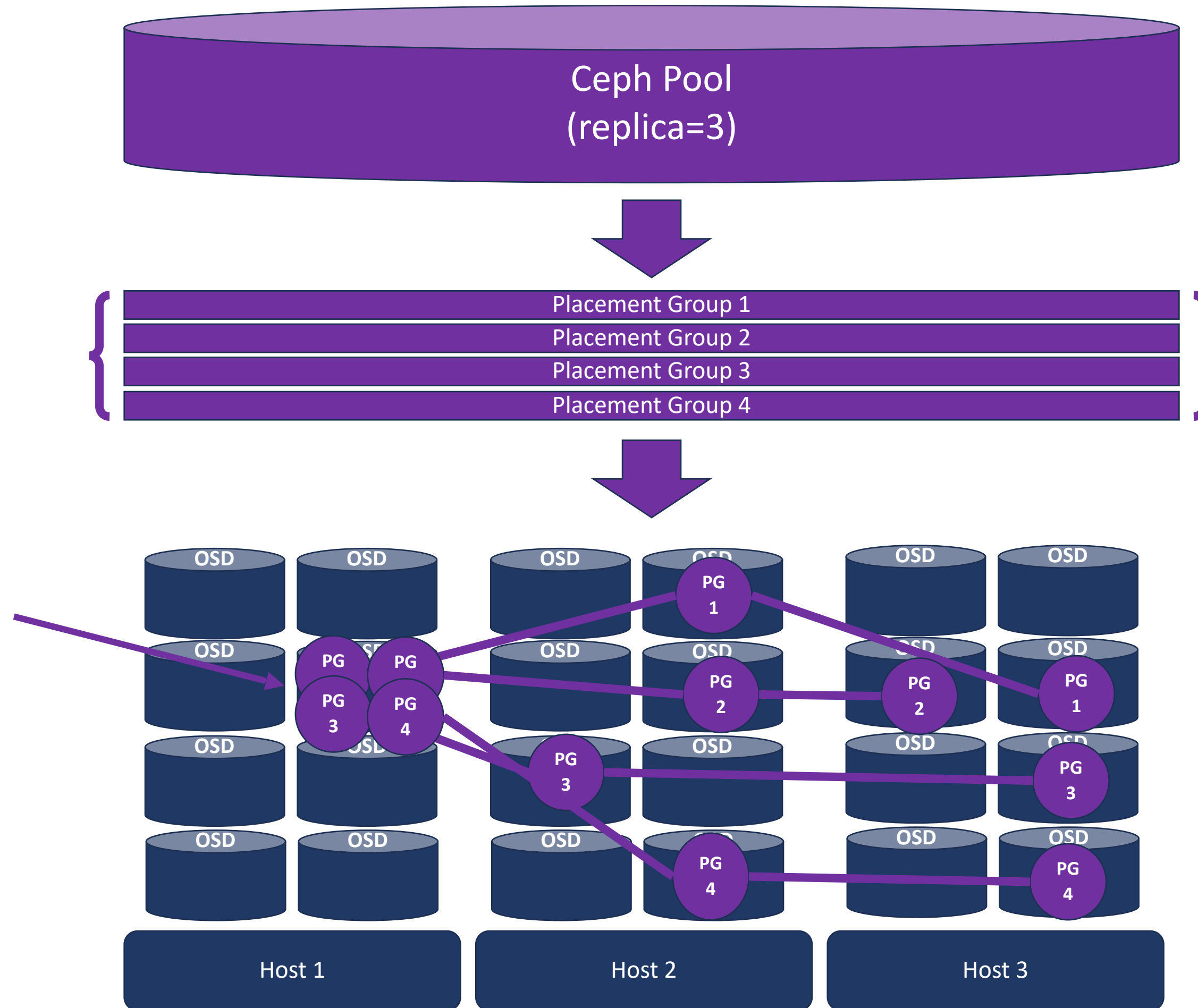


# Understanding Durability

1. In this example two OSDs have failed. A first OSD and then a second OSD within the window where the cluster was backfilling to recover from the first OSD failure.
2. If there are not enough OSDs remaining in a PG (**min\_size param**) the pool will stop. **min\_size** is a pool level param and it should always be set by default so that there's remaining durability. For a replica=3 pool with three copies of the data that means we want a minimum of replica=2 for all PGs at all times, so we'd set **min\_size=2** (default). With two copies of the data if one copy is bad we can detect that via a checksum and still continue and recover.
3. In this example we've lost two (2) OSDs out of three. With only one OSD remaining in PG1 we're not meeting the **min\_size=2** requirement. The PG will stop accepting IO, writes will hang until one of the OSDs recovers or the **min\_size** is reduced to 1.



# Understanding Durability – Teamwork!



More realistic example showing the multiple PGs per OSD. By our calculation for this cluster there would be **64** PGs per OSD but we're showing just 4x PGs for clarity.

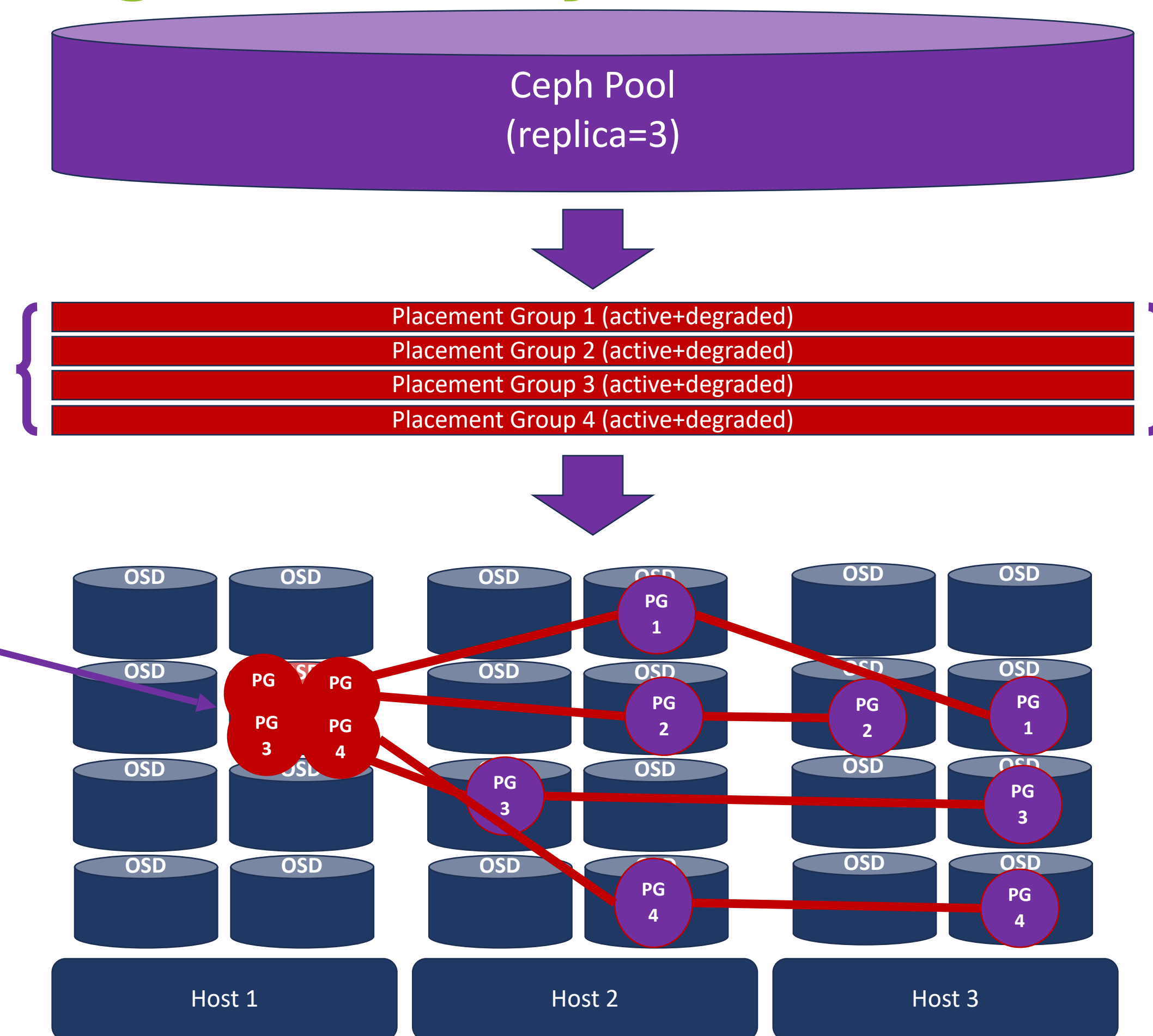
## The Core Formula

- PG Length = Replica count or K+M for EC pools
- Base PG count ( $PC_{base}$ ) =  $(100 * OSDs) / (PG_{length})$
- Scaled PG count ( $PC_{scaled}$ ) =  $PC_{base} * 50\%$
- Total PG count =  $PC_{scaled}$  rounded up (nearest base 2)
- PGs per OSD =  $PC_{total} / (OSDs / PG_{length})$

## Example

- PG Length = 3
- Base PG count ( $PC_{base}$ ) =  $(100 * 24) / 3 = 800$
- Scaled PG count ( $PC_{scaled}$ ) =  $PC_{base} * 50\% = 400$
- Total PG count =  $PC_{scaled}$  rounded up (nearest base 2) = 512
- PGs per OSD =  $512 / (24 / 3) = 512 / 8 = 64$

# Understanding Durability – Teamwork!



When an OSD fails, many PGs are affected but many OSDs come to the rescue.

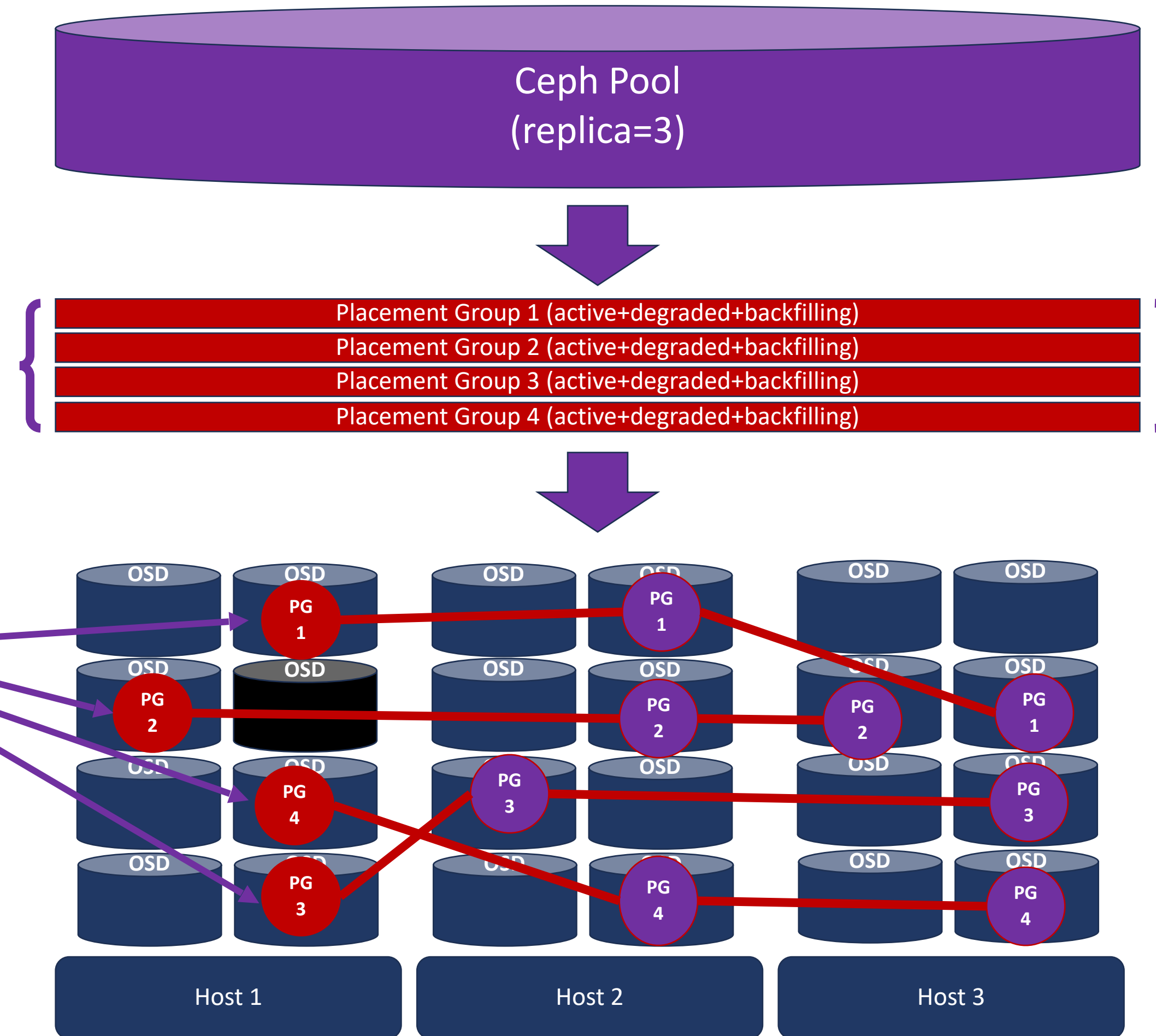
## The Core Formula

- PG Length = Replica count or K+M for EC pools
- Base PG count ( $PC_{base}$ ) =  $(100 * OSDs) / (PG_{length})$
- Scaled PG count ( $PC_{scaled}$ ) =  $PC_{base} * 50\%$
- Total PG count =  $PC_{scaled}$  rounded up (nearest base 2)
- PGs per OSD =  $PC_{total} / (OSDs / PG_{length})$

## Example

- PG Length = 3
- Base PG count ( $PC_{base}$ ) =  $(100 * 24) / 3 = 800$
- Scaled PG count ( $PC_{scaled}$ ) =  $PC_{base} * 50\% = 400$
- Total PG count =  $PC_{scaled}$  rounded up (nearest base 2) = 512
- PGs per OSD =  $512 / (24 / 3) = 512 / 8 = 64$

# Understanding Durability – Teamwork!



PGs on the effected OSD have been remapped to many other OSDs with the help of many OSDs on other nodes. Each PG is responsible for 1/64<sup>th</sup> of the data on a given OSD so backfilling it's data to other OSDs goes quickly and the process can be highly parallelized unlike traditional RAID.

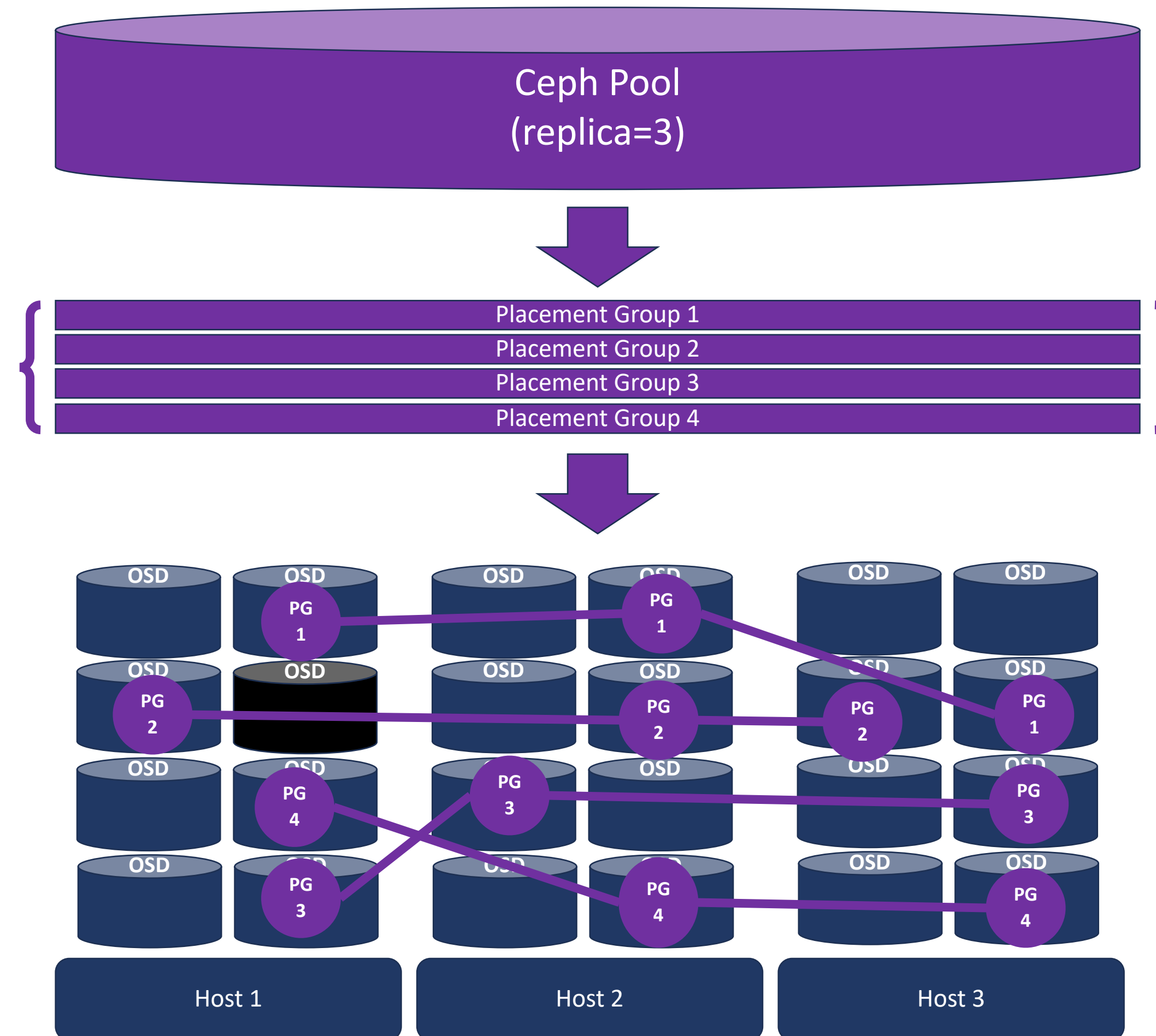
## The Core Formula

- PG Length = Replica count or K+M for EC pools
- Base PG count ( $PC_{base}$ ) =  $(100 * OSDs) / (PG_{length})$
- Scaled PG count ( $PC_{scaled}$ ) =  $PC_{base} * 50\%$
- Total PG count =  $PC_{scaled}$  rounded up (nearest base 2)
- PGs per OSD =  $PC_{total} / (OSDs / PG_{length})$

## Example

- PG Length = 3
- Base PG count ( $PC_{base}$ ) =  $(100 * 24) / 3 = 800$
- Scaled PG count ( $PC_{scaled}$ ) =  $PC_{base} * 50\% = 400$
- Total PG count =  $PC_{scaled}$  rounded up (nearest base 2) = 512
- PGs per OSD =  $512 / (24 / 3) = 512 / 8 = 64$

# Understanding Durability – Teamwork!



1. Our cluster is now back to 100% full health without a hot-spare.
2. Recovery was many times faster than it would take to return to full health using traditional RAID with a hot-spare.

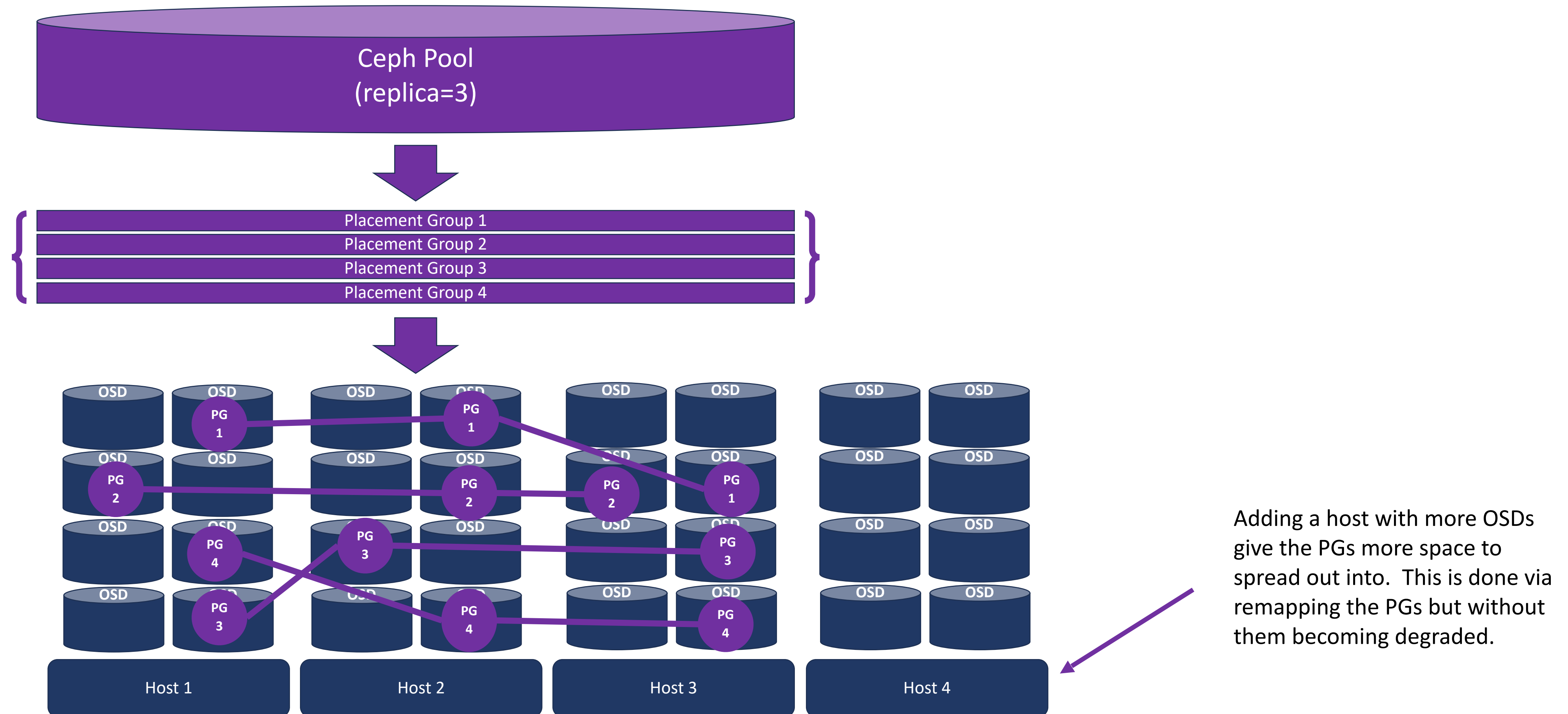
## The Core Formula

- PG Length = Replica count or K+M for EC pools
- Base PG count ( $PC_{base}$ ) =  $(100 * OSDs) / (PG_{length})$
- Scaled PG count ( $PC_{scaled}$ ) =  $PC_{base} * 50\%$
- Total PG count =  $PC_{scaled}$  rounded up (nearest base 2)
- PGs per OSD =  $PC_{total} / (OSDs / PG_{length})$

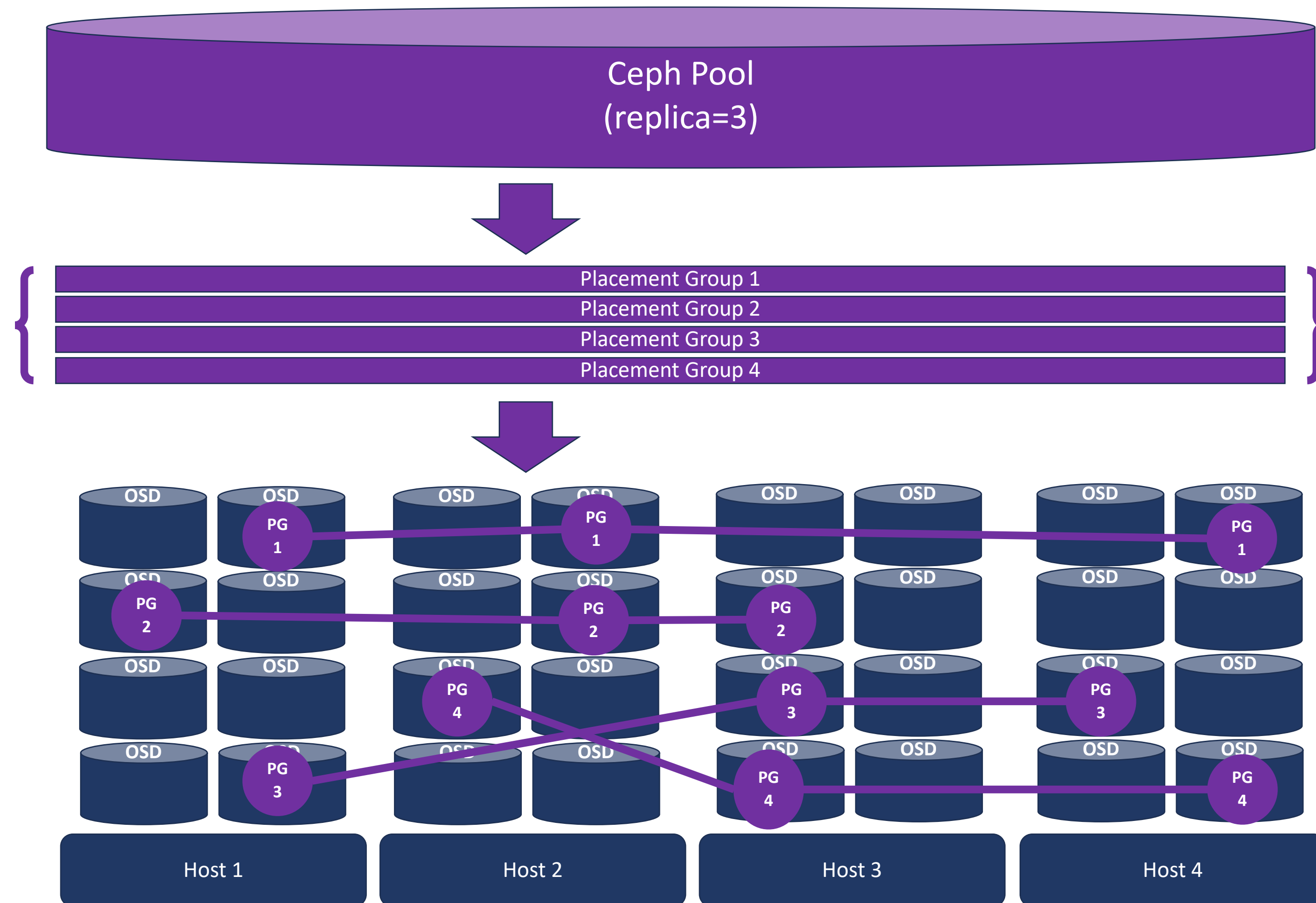
## Example

- PG Length = 3
- Base PG count ( $PC_{base}$ ) =  $(100 * 24) / 3 = 800$
- Scaled PG count ( $PC_{scaled}$ ) =  $PC_{base} * 50\% = 400$
- Total PG count =  $PC_{scaled}$  rounded up (nearest base 2) = 512
- PGs per OSD =  $512 / (24 / 3) = 512 / 8 = 64$

# Understanding Durability

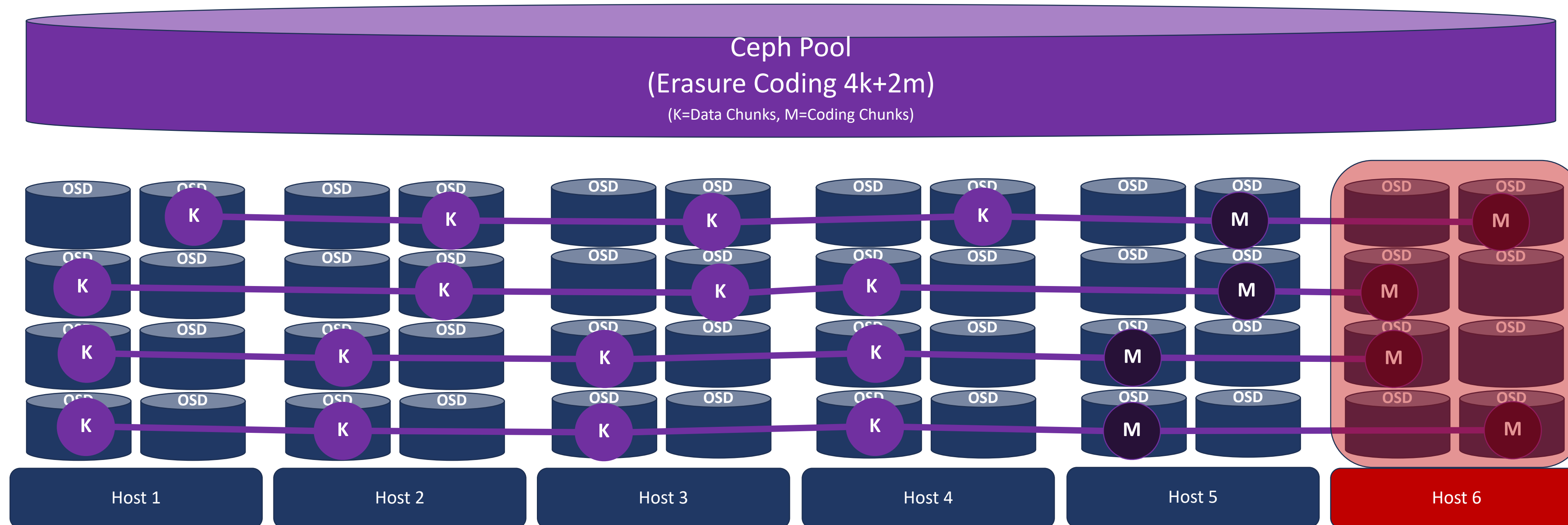


# Understanding Durability



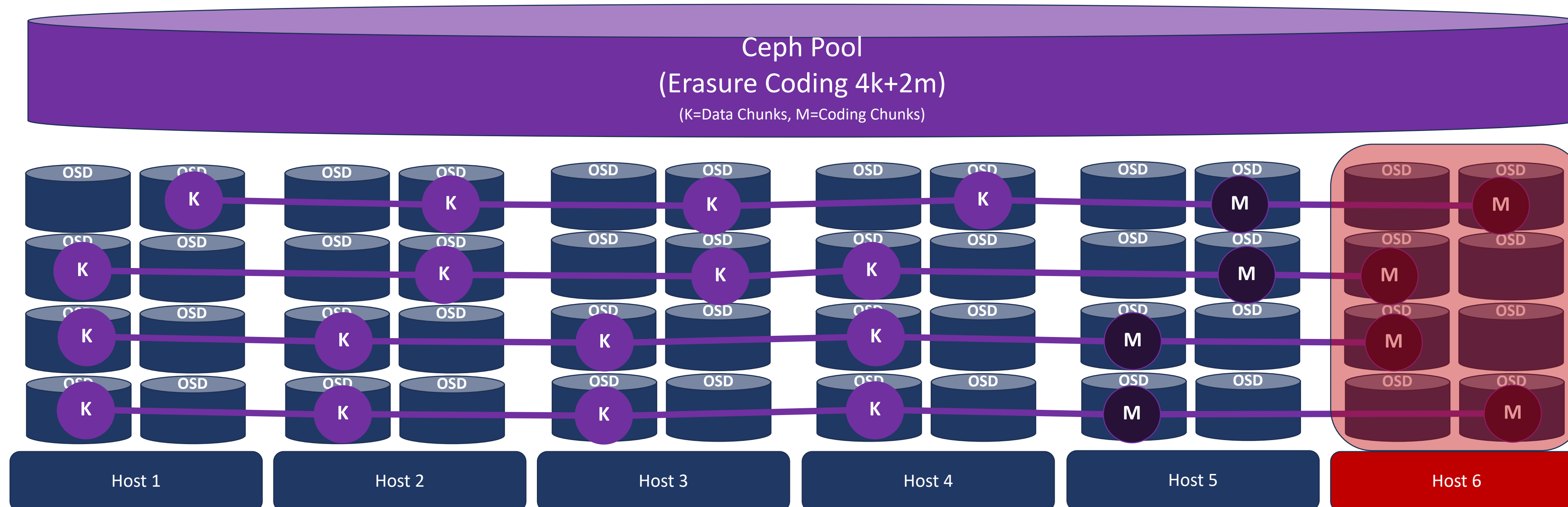
Adding a host with more OSDs give the PGs more space to spread out into. This is done via remapping the PGs but without them becoming degraded.

# Understanding Durability



With EC pools we need a `min_size` set to  $K+1$  to ensure we always have a coding block available to recover from.  
 With a 4K+2M pool ( $K=4$ ,  $M=2$ ) this means we need a `min_size` of 5 ( $K = 4 + 1$  coding block = 5 `min_size`).  
 With a 8K+3M pool ( $K=8$ ,  $M=3$ ) this means we need a `min_size` of 9 ( $K = 8 + 1$  coding block = 9 `min_size`).  
 With a 16K+4M pool ( $K=16$ ,  $M=4$ ) this means we need a `min_size` of 17 ( $K = 16 + 1$  coding block = 17 `min_size`).

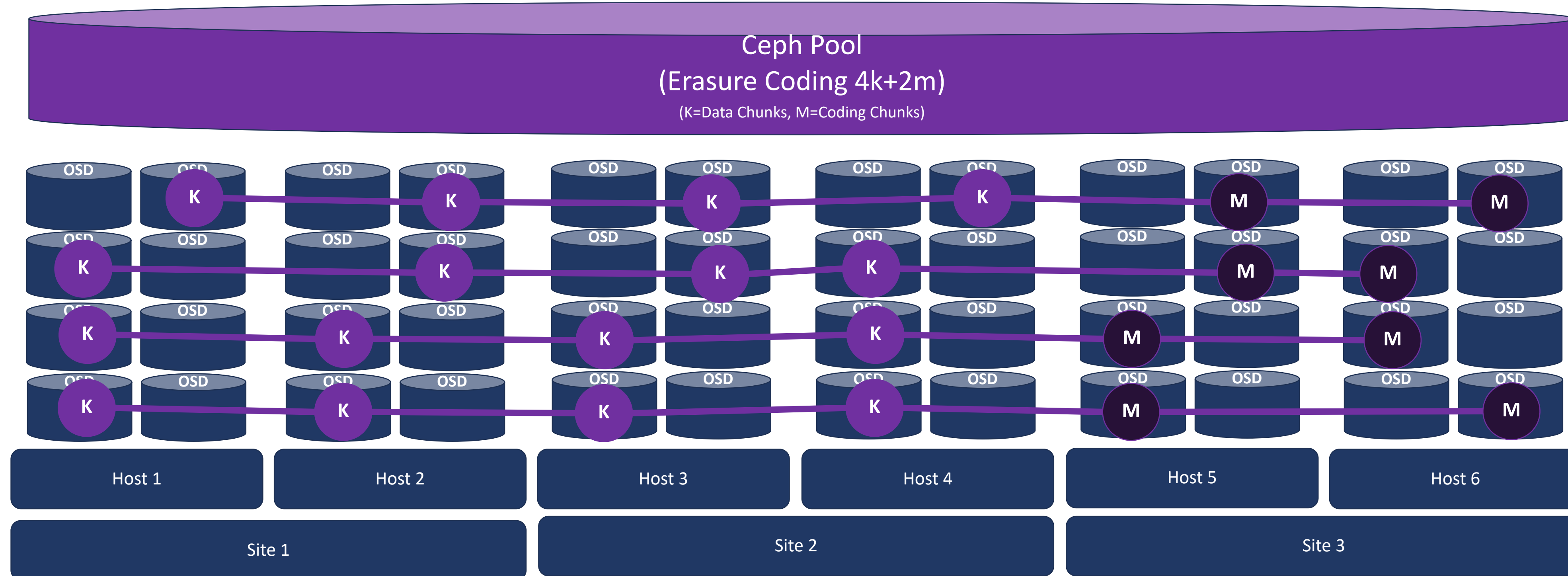
# Trivia time, is it safe?



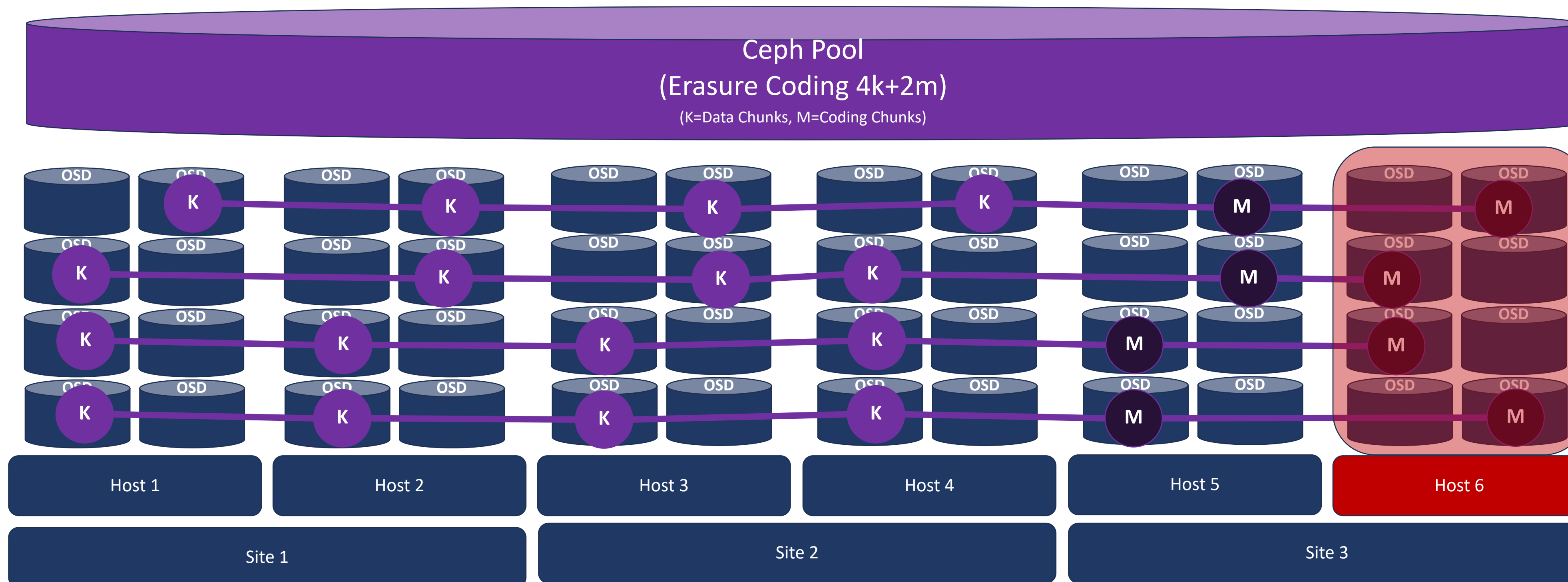
**Yes!** With 4K+2M we need to maintain K+1 (5) and with one host offline we've 5 out of 6 chunks.

```
ceph config set mon mon_osd_down_out_subtree_limit host
```

# Trivia time, is it safe?

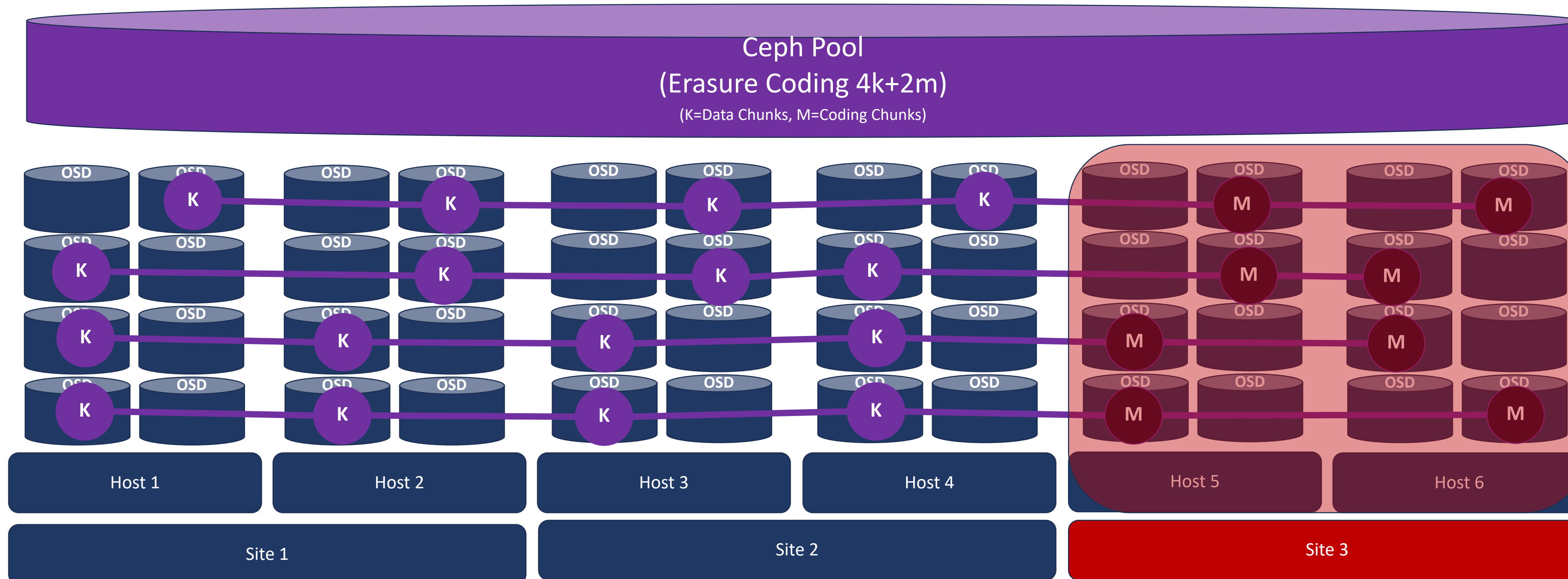


# Trivia time, is it safe?



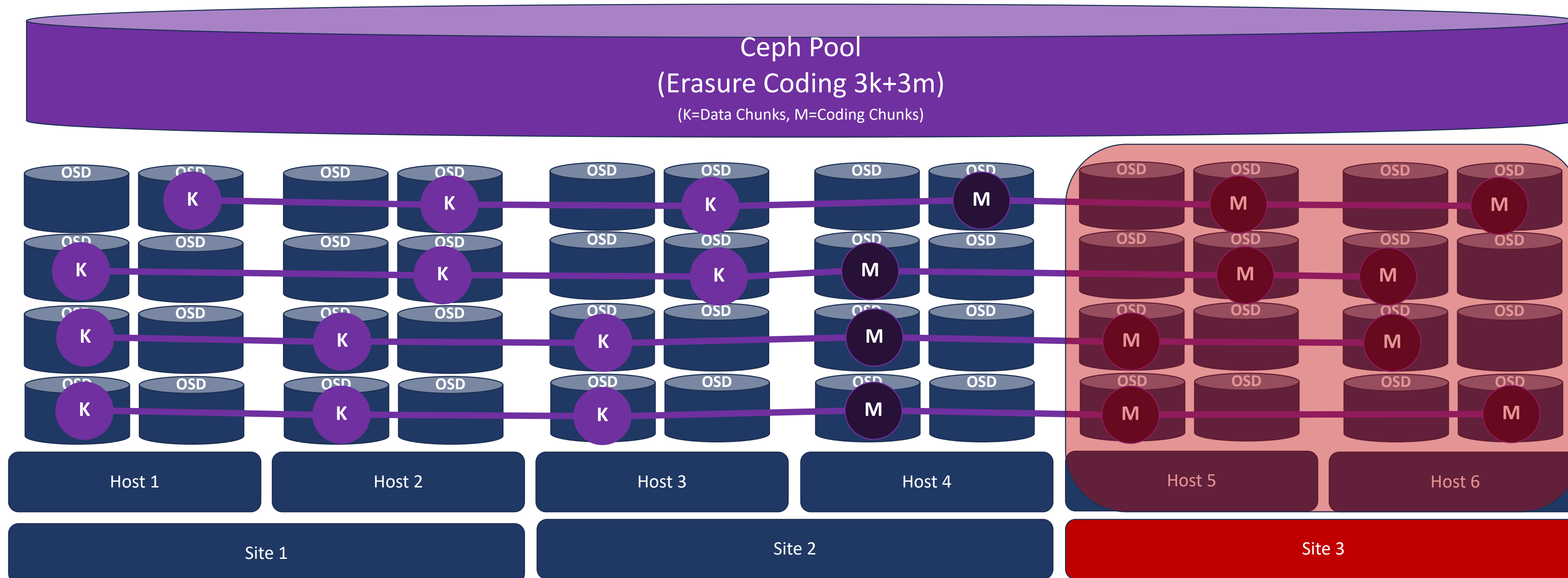
**Looks safe, but not safe!** Yes with 4K+2M we maintain K+1 (5) and with one host offline but this is multi-site!

# Trivia time, is it safe?



**Not Safe!** With a site outage we lose two hosts at once and we'll no longer have  $K+1$  chunks available per PG.

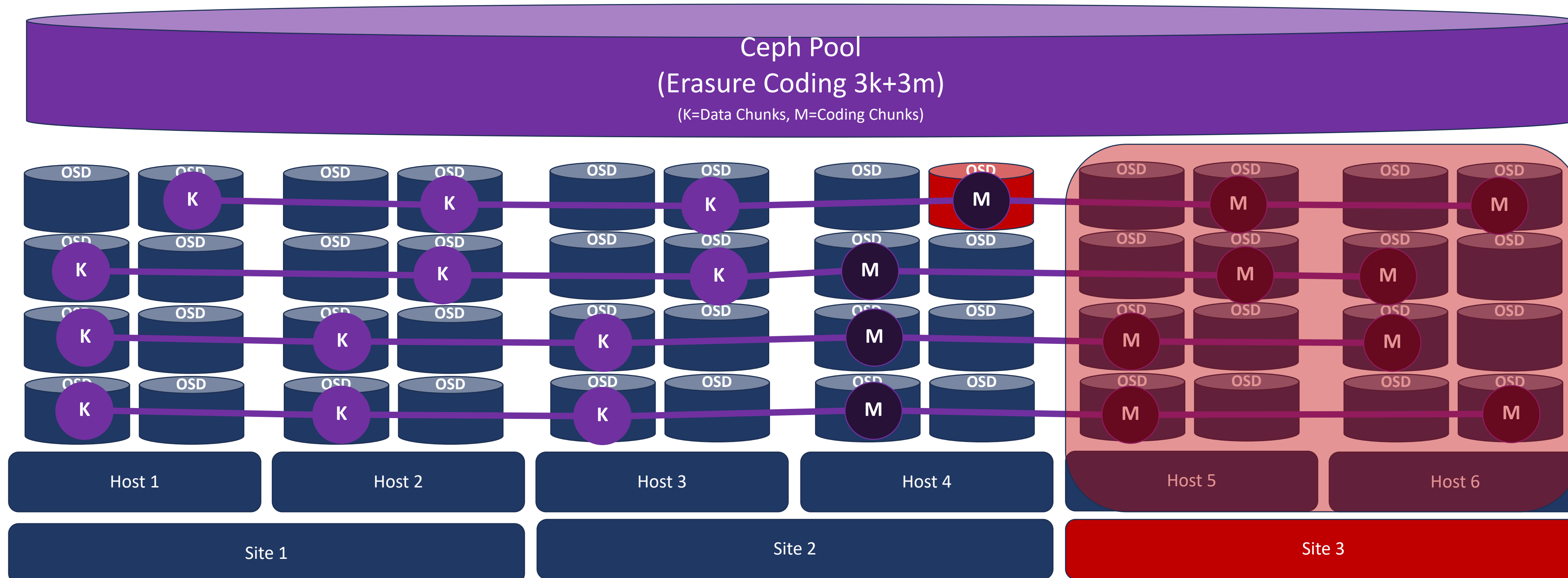
# Trivia time, is it safe?



We've now changed our design to use EC 3K+3M so our K+1 is now 4. Is it safe?

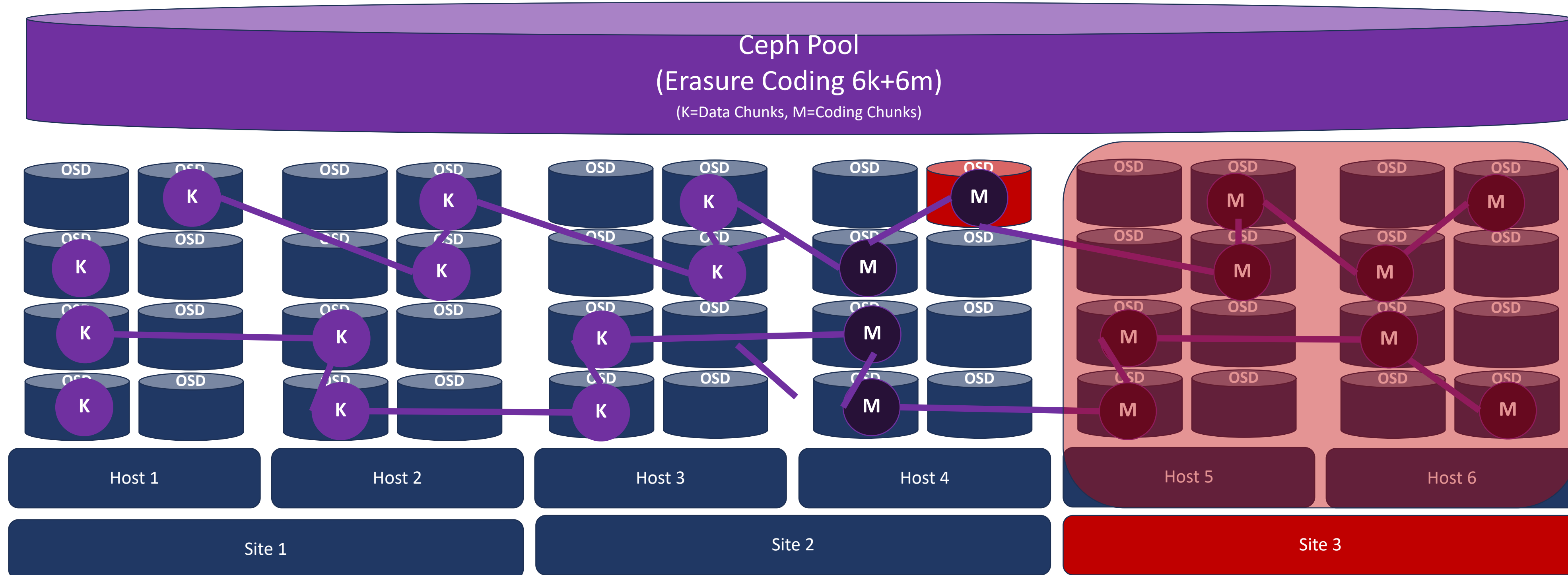
**Yes!** With 3K+3M we need to maintain K+1 (4) and with one site offline we've 4 out of 6 chunks.

# Trivia time, is it safe?



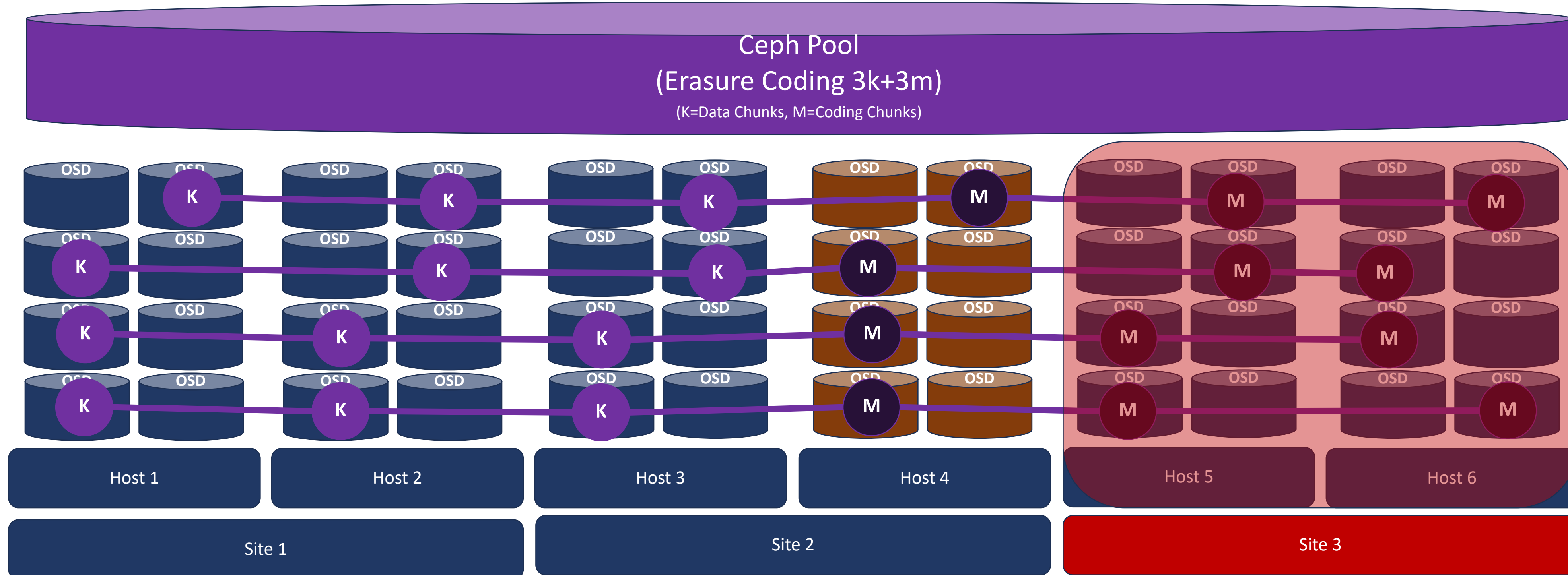
What happens if we have an OSD fail in another site just before or after a site outage?

# Trivia time, is it safe?



Doubling the PG length from 3K+3M to 6K+6M retains our 50% usable and gives us an additional coding block remaining after a combination site outage and device failure.

# Double-Layer Durability



Having storage virtualization be it hardware or software RAID underneath the OSDs makes sense at large scale and in certain multi-site scenarios. With the OSDs produced from fault-tolerant block devices from an underlying SAN (eg. Seagate EP) or NVMe RAID stack (eg. Xinnor, Graid Tech) the entire cluster is insulated from individual device failures. This typically comes at a cost of 12% to 20% overhead but localizes the rebuilds, eliminates the WAN load due to media failures, and can reduce CPU and RAM costs.

# Q&A

Solution Design Review: [sdr@osnexus.com](mailto:sdr@osnexus.com)

Design Utilities: <https://osnexus.com/ceph-designer>



## QuantaStor Scale-out Storage Solution Designer

QuantaStor solutions provide scale-out file (NFS/SMB/CephFS), block (iSCSI/FC/CephRBD) and S3 compatible object storage. Storage is made highly-available using 3x or more servers and a storage layout support includes erasure-coding and replicas. QuantaStor bare-metal installs onto servers and this utility enables designing solutions around reference hardware from Supermicro, Seagate, Dell, HPE, Western Digital, Intel, and Cisco servers and JBODs. QuantaStor integrates with Ceph open storage technology.

Usable Capacity:  10000 TB

Use Cases:

### Server & Cluster Specification

Server Model:  \$/Server

Expansion Chassis:  \$/JBOD

Data Device (OSD):  \$/Device

Metadata (OSD) / Node:  \$/Device

Offload (WAL+MDB) / Node:  \$/Device

RAM/Device Ratio:  Metadata/Data Ratio:

Ethernet Ports:

Fibre Channel Ports:

Processors:

Encryption:

### Storage Configuration

Storage Layout:  Layout Usable: 66%

Data Compression (%):

Backfill Reserved Space (%):

Reserved Drive Slots (%):

### License Summary

License Duration:

Support Level:

License Capacity:

### Contact Information

\*All fields required

\*First Name:

\*Last Name:

\*E-mail:

\*Company:

### Rack Summary

### Solution Summary

Servers (nodes)	8
Data Pool Devices	672 (84/node)
Metadata Pool Devices	16 (2/node)
Write-Log/MDB Devices	32 (4/node)
Rack Space Required	48 RU
Usable Capacity	10726 TB
Usable w/ Compression	11918 TB
Raw Capacity	16865 TB
Estimated Power Req	17984 W
Estimated Power Cost/mo	\$2201
Estimated HW Cost	\$0
HW Price/TB (raw)	\$0.00
Estimated Cluster Write Throughput*	18.9 GB/sec
Estimated Cluster Read Throughput*	28.3 GB/sec
Estimated Cluster Write IOPS	N/A
Estimated Cluster Read IOPS	N/A
Min Recommended Networking	Dual-port 50GbE (QSFP28) NIC
Recommended Networking	Dual-port 100GbE (QSFP28) NIC
Physical-to-Virtual Ratio	4.0 P : 1.0 V

### Server Specification

8x Supermicro SYS-121C-TN10R server containing:

- 2x Intel Xeon Gold 4516Y + 2.2GHz (24 core) processors
- 512GB DDR5 ECC RAM
- 2x 400GB M.2 SSDs (boot)
- Dual-port 100GbE (QSFP28) NIC
- Dual redundant power supplies
- Software Encryption

### Exp. Chassis Specification

8x Seagate Exos EP 5U84 bay RBOD system

- 4x HD 12Gb mini-SAS cables
- Dual redundant hot-swap controllers
- Dual redundant hot-swap power supplies
- 4.0 P : 1.0 V Physical-to-Virtual Ratio
- [Click here for SAS cabling diagrams](#)

### Device Specification

672x 24TB SATA/SAS HDD (Data)

16x 15.36TB NVMe SSD (Metadata)

32x 15.36TB NVMe SSD (WAL+MDB Offload)

### Solution Design URL