

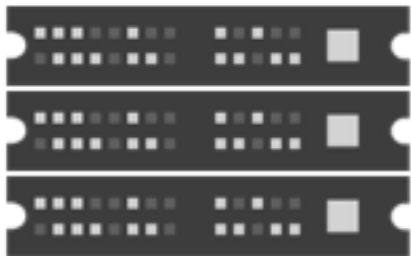
Best practices for using flash in hyperscale software storage architectures

Brandon Hoang
Solutions Architect





Software-Defined Storage (SDS)



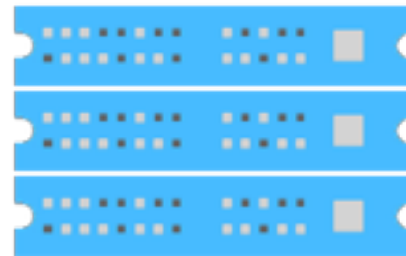
Commodity Servers

+



Software

=



Software-Defined
Storage

Double-digit growth

A \$2.8 billion market by 2017

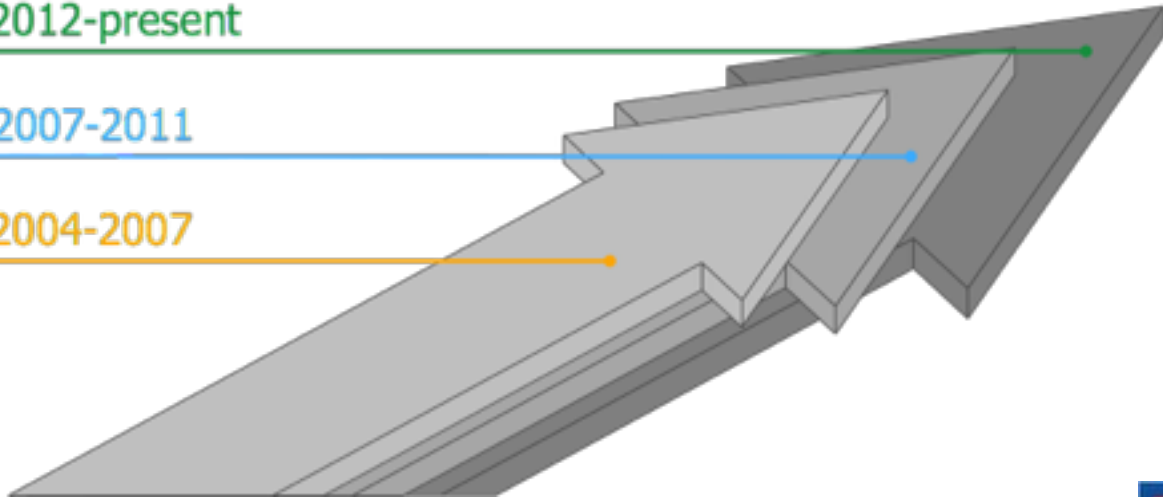
IDC 2015

Who is Hedvig?

2012-present

2007-2011

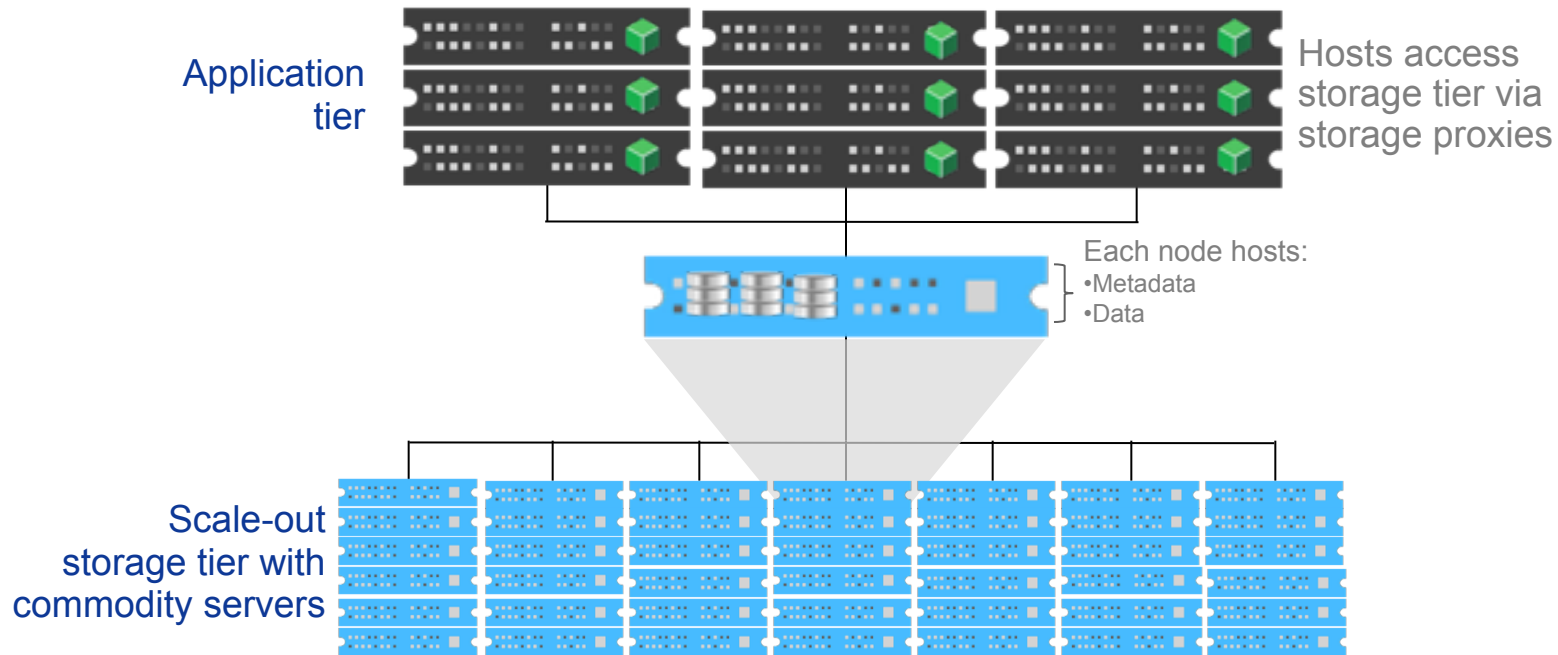
2004-2007



- Founded in 2012 by Avinash Lakshman
 - Co-inventor of Amazon Dynamo and inventor of Apache Cassandra
- Develop the Hedvig Distributed Storage Platform
 - A software-defined storage solution



Anatomy of a distributed, hyperscale storage system



Taking advantage of flash w/ SDS



- At the storage server node
 - Store metadata on SSD: fast lookups and tracking
 - Write-optimization: sequentialize random I/O
 - Auto-tier and cache active data on SSDs: speed access to hot data and buffer HDD capacity tier
 - Provision volumes on “all-flash” persistent storage (aka “pin to flash”): dedicated, consistent performance for latency sensitive apps

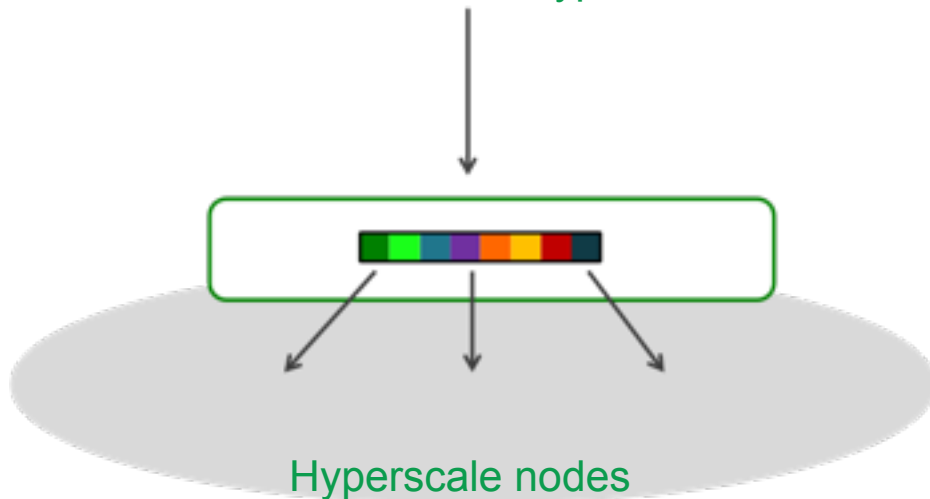


- At the application host
 - Cache hot data on local SSD or PCIe flash to accelerate access and avoid network latencies

Biggest flash benefit to hyperscale: Sequentializing random I/O

Application Server

Hyperscale client



1

Application writes data in random blocks, and gets immediate ack from cluster

2

Storage cluster sequentializes incoming blocks (in RAM+SSD) into larger chunks

3

Larger sequentialized data chunks written to underlying disks according to policy



Three ways flash is used in hyperscale systems

Read/write cache on storage nodes

“Pin to flash” dedicated primary storage volume

Client side read cache



Option #1: Node OS storage

Type of flash:

SLC/MLC SSDs or PCIe Flash

Use of flash:

- Store metadata for fast operations – dedupe, compression, snaps, clones
- Autotiering to ensure hot data is migrated to flash
- Write logs for metadata and data

Typical configuration:

2x 300GB MLC SAS/SATA SSDs

Read/write cache on storage nodes

ent side read cache



Option #2: Node volume storage

Type of flash:
SLC/MLC SSDs or PCIe Flash

Use of flash:
-- All-flash virtual volumes
for dedicated, consistent
performance on a per-app
basis
-- Flash performance for read
and write operations

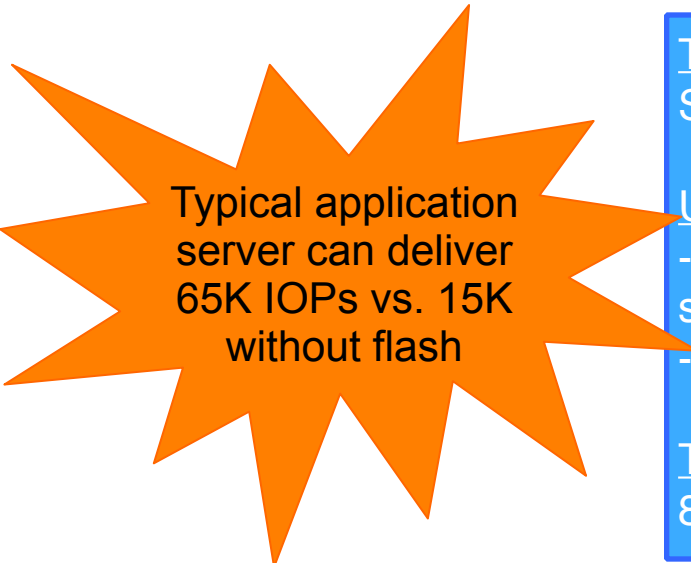
Typical configuration:
2x 800GB MLC SAS/SATA
SSDs

Read/write cache on
storage nodes

“Pin to flash” dedicated
primary storage volume

Client side read cache

Option #3: Client-side cache



Typical application server can deliver 65K IOPs vs. 15K without flash

Type of flash:
SLC/MLC SSDs or PCIe Flash

Use of flash:
-- Write-through cache to store hot blocks
-- Local metadata storage

Typical configuration:
800GB MLC SAS/SATA SSDs

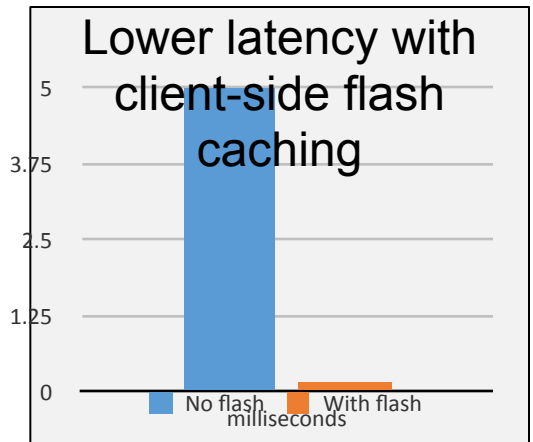
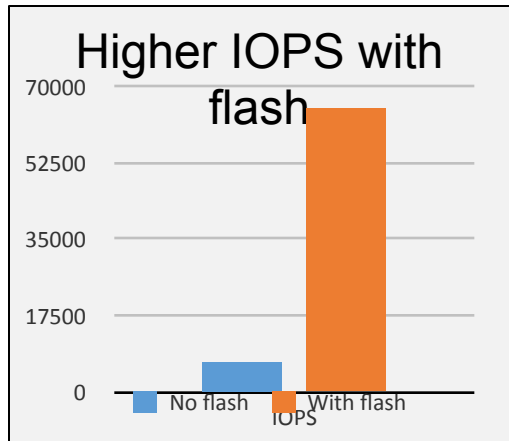
Read/write cache on storage nodes

“Pin to flash” dedicated primary storage volume

Client side read cache

Results: Law Firm

- Challenge:
 - Needed quick, reliable indexing and lookups of massive 100 million active client legal docs
 - Traditional NAS underperformed required access time
 - Standalone servers with flash performed well, but predictably ran out of space
- Solution/Result:
 - Hedvig software-defined storage with SSD/HDD and client-side flash caching
 - ~9x faster performance with flash
 - Scale-out architecture simplifies growth and expansion





Thank you!

For more on Hedvig visit:

- Web: hedviginc.com
- Twitter: [@hedviginc](https://twitter.com/hedviginc)

