

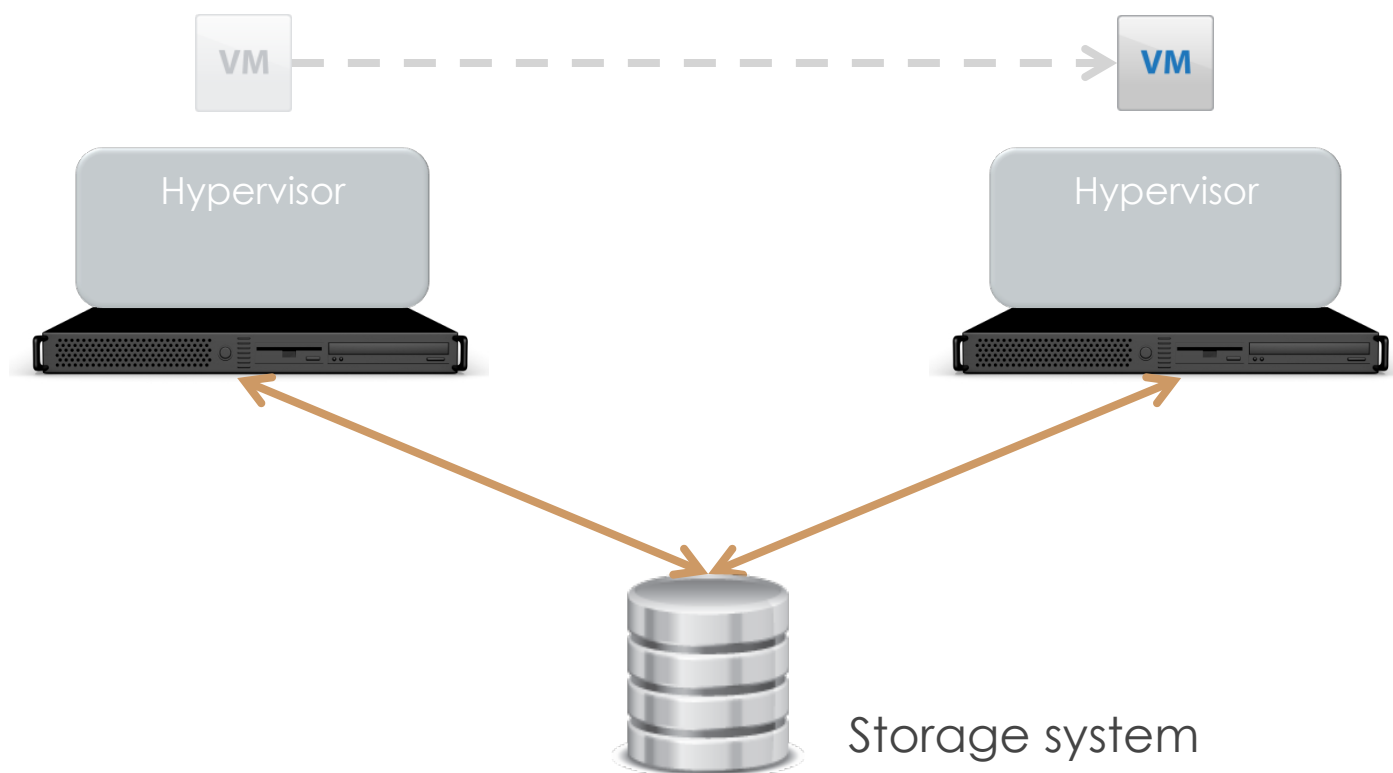
# Hypervisors and Server Flash

Satyam Vaghani  
pernixdata

# Agenda

- Case for server flash in SDDC
- Technology implications of marrying hypervisors with server flash
  - Empirical data
- Conclusion and open issues

# Reference setup



# Calculator or watch?

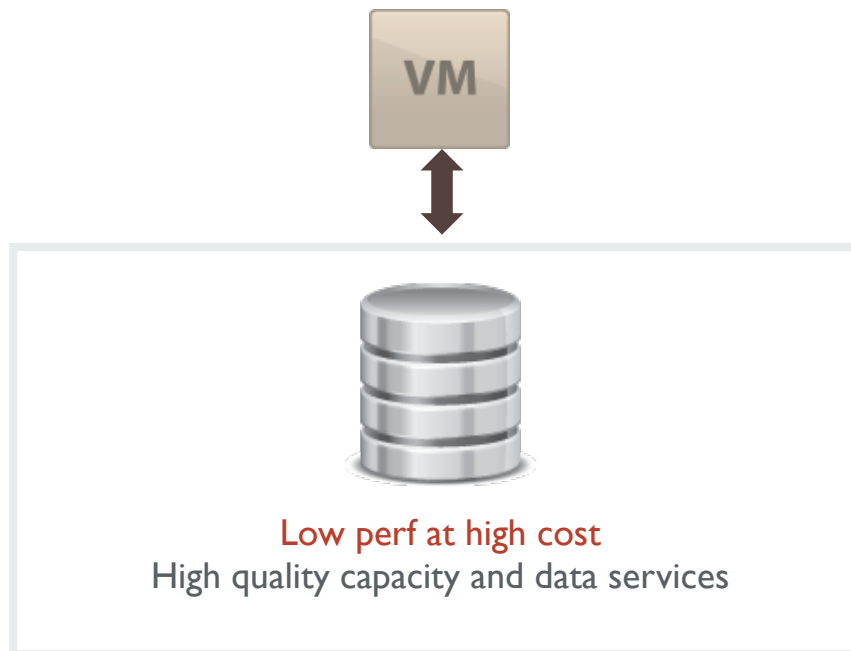


# Performance or capacity?

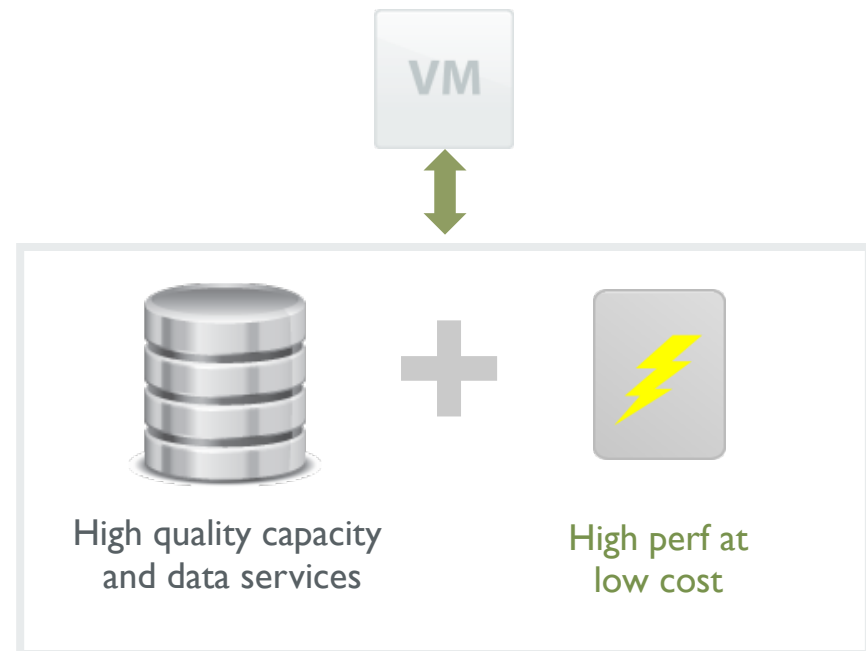


# Virtualization ♥ Flash

## Current state



## Desired state



But, how?

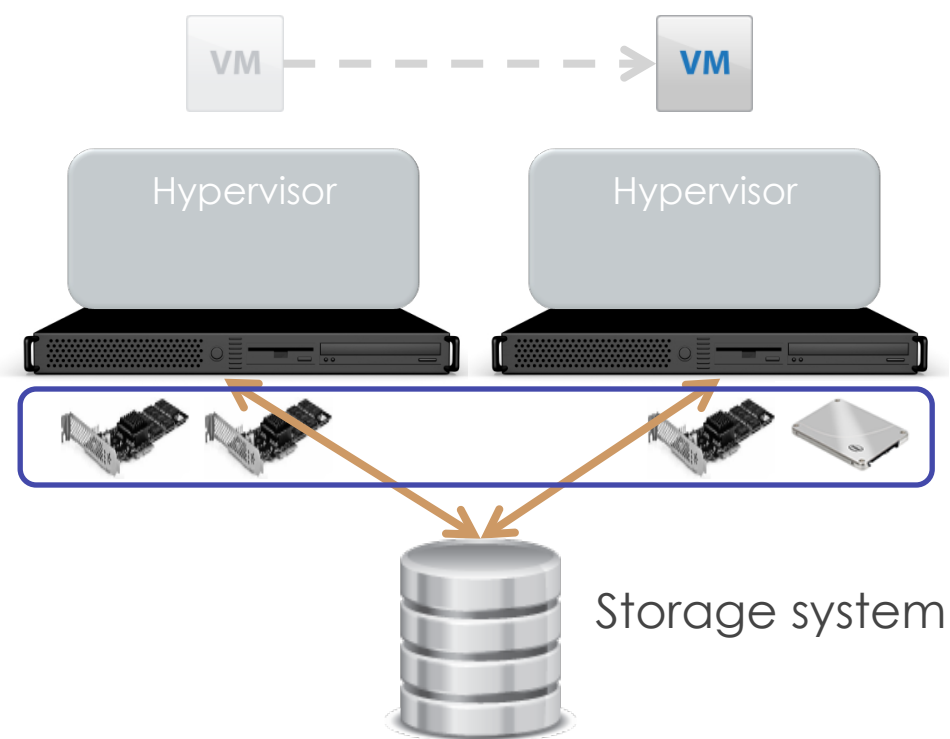
# Storage system flash has limits

## □ Using storage system flash

- Many are rip-and-replace solutions
- Higher latency: SAN Flash is “too far” across the network
- Loss of application/VM identity
- Higher cost, lower scalability, depends on intermediate moving parts

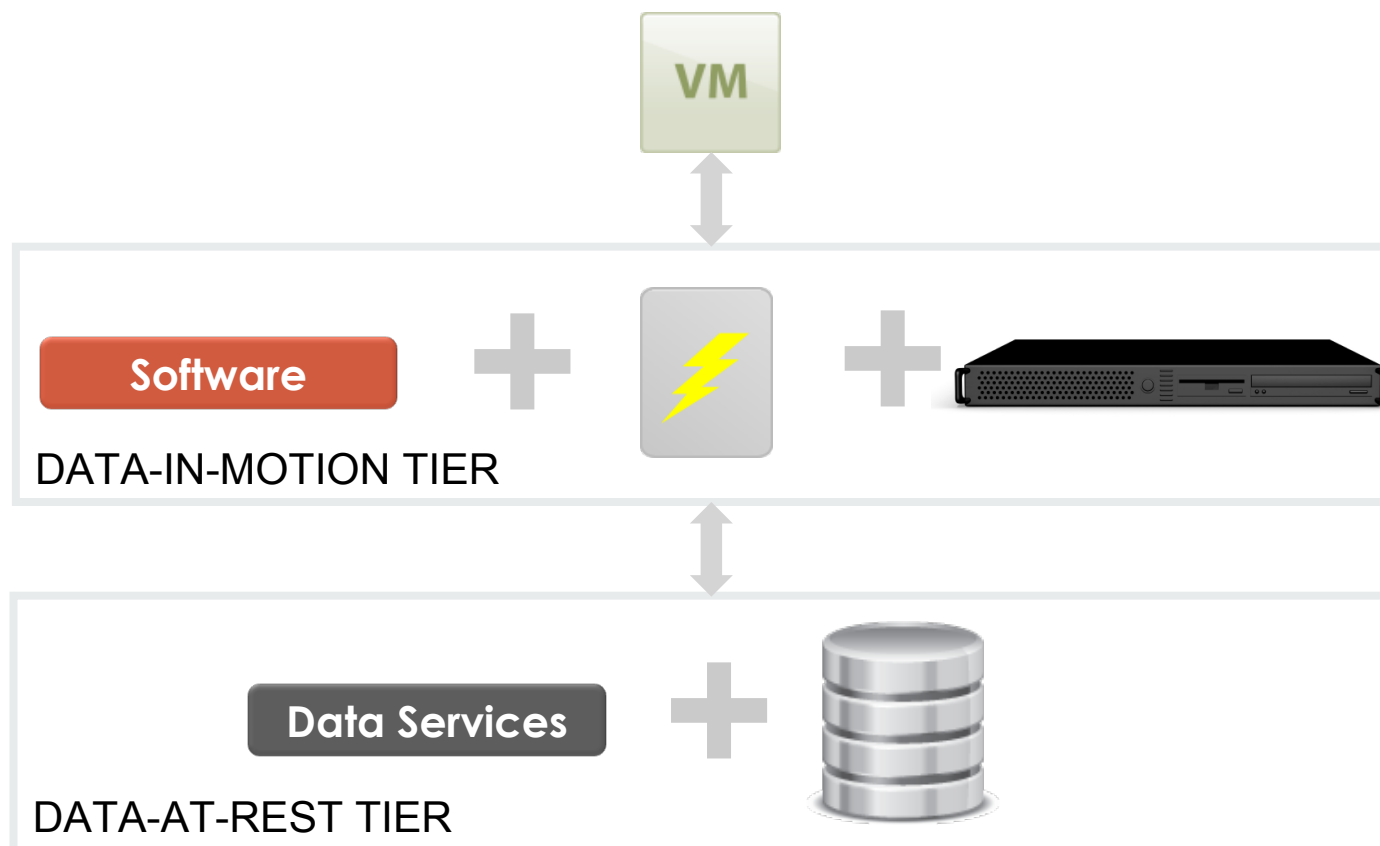
# Server flash with hypervisors

- ❑ Limited to niche use cases with
  - ❑ Local datastores, or
  - ❑ Single host read only cache
- ❑ Not clustered, breaks cross-host hypervisor functions
- ❑ Need custom applications or in-VM changes
- ❑ Low performance, less available virtual appliances





# Time for a brand new data tier?

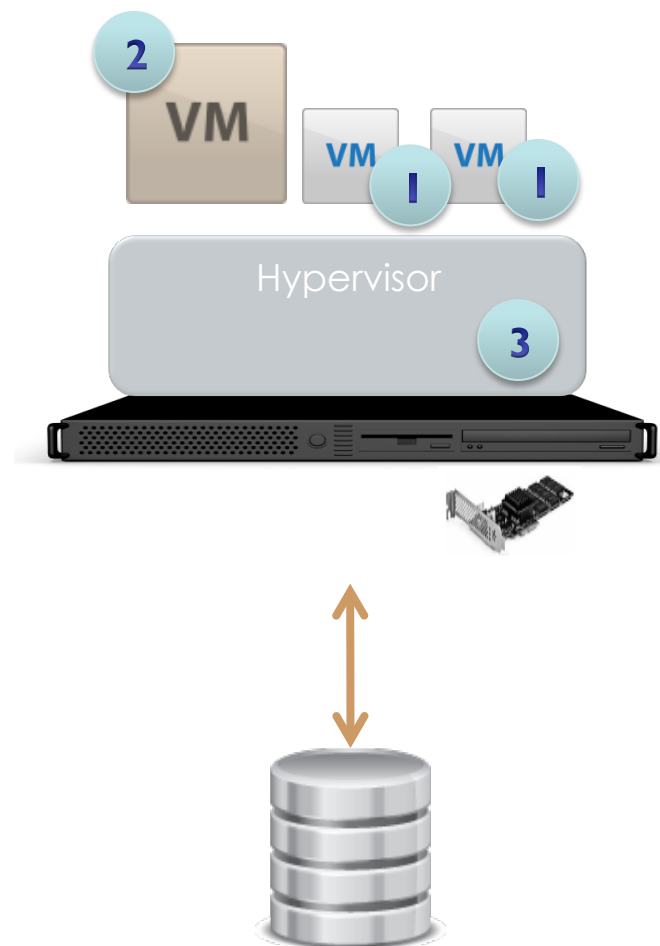


# Scope of the problem

- ❑ Form factor and placement of software
- ❑ Optimal flash format
- ❑ Clustering of flash resources
- ❑ Expanding beyond niche use cases

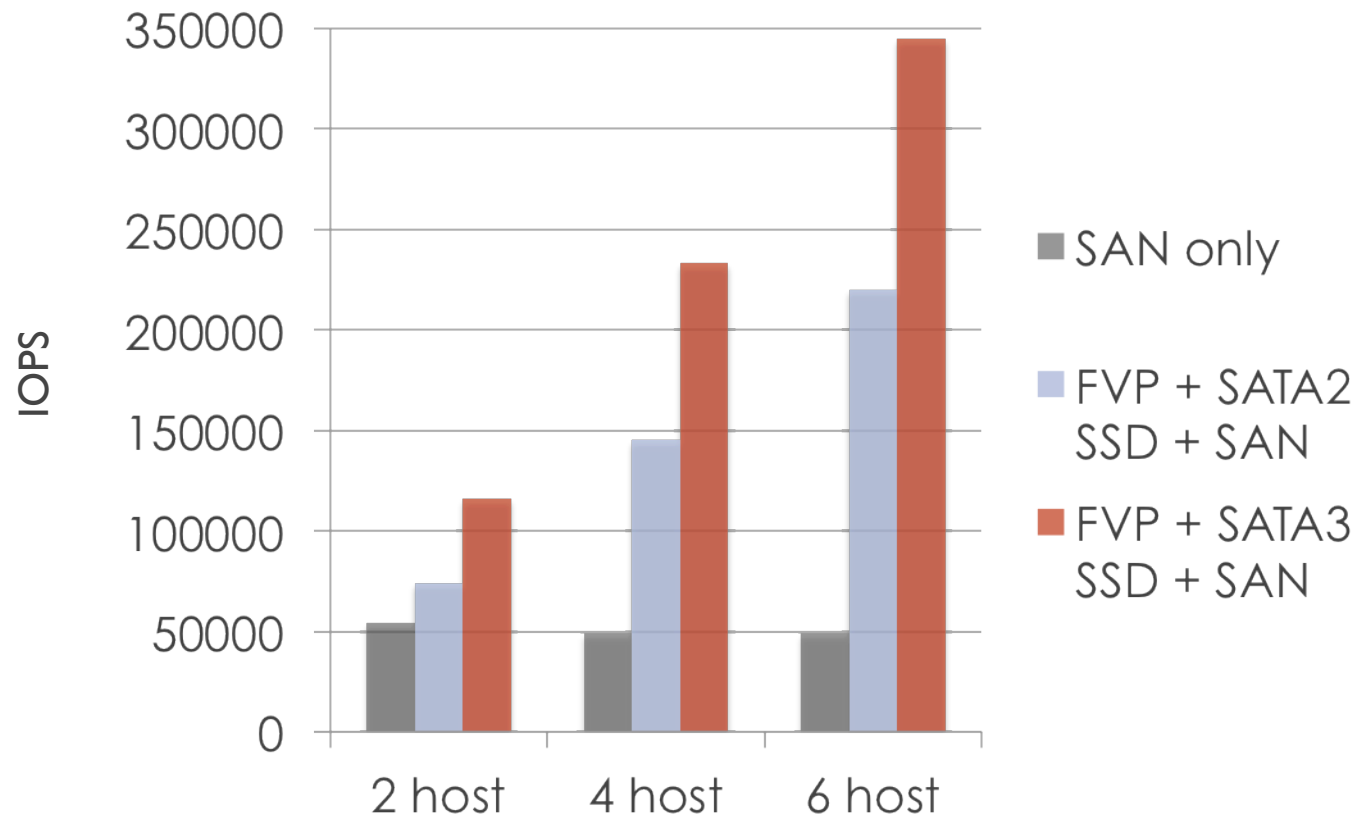
# How to leverage server flash

- Form factor
  - Local datastore, or
  - Acceleration tier
- Placement
  1. Inside guest OS
  2. Virtual appliance
  3. Inside hypervisor



# Scale out

## Server flash beats SAN IOPS even in small clusters!

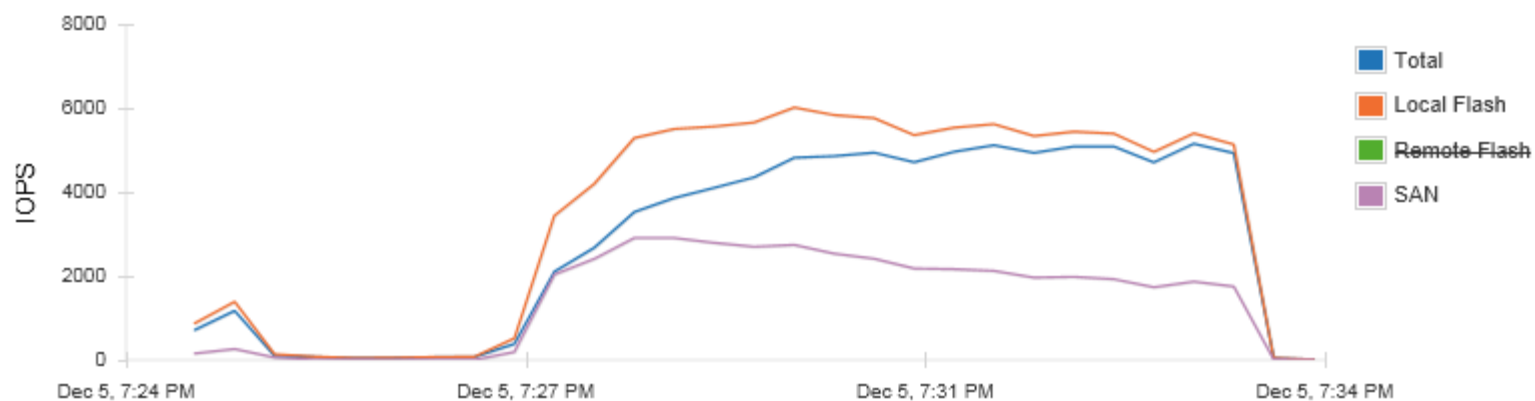


# Read heavy (VDI) results

Pernix WT | [Back to performance home](#)

Summary VMs Flash Devices

Avg VM IOPS



10 writethrough Win 7 Linked Clones running 80% Read, 80% Random for 6 minutes  
[including boot up and shut down]

# Flash format considerations

- ❑ Hypervisors need something else from flash
  - ❑ VM QoS mgmt required, not capacity mgmt
  - ❑ Performance isolation between VMs
  - ❑ Write intensive: even read intensive workloads cause (false) writes. Eg...

**4KB workload READ latency sensitive to OIO, presence of WRITES!**

**300us**

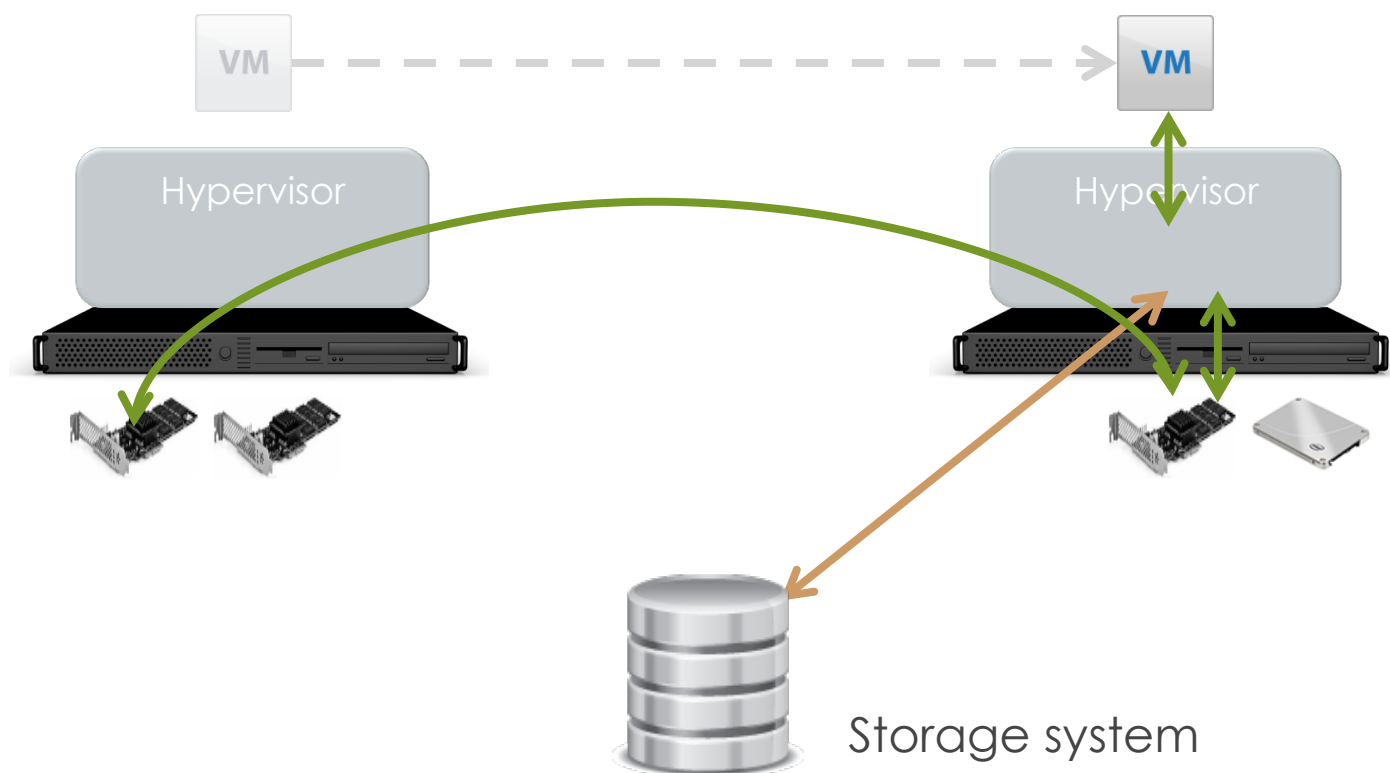
100% read, 80% random, 16 OIOs

**3200us**

67% read, 80% random, 64 OIOs

# Clustering considerations

- What: Transparent remote flash footprint access on behalf of VMs

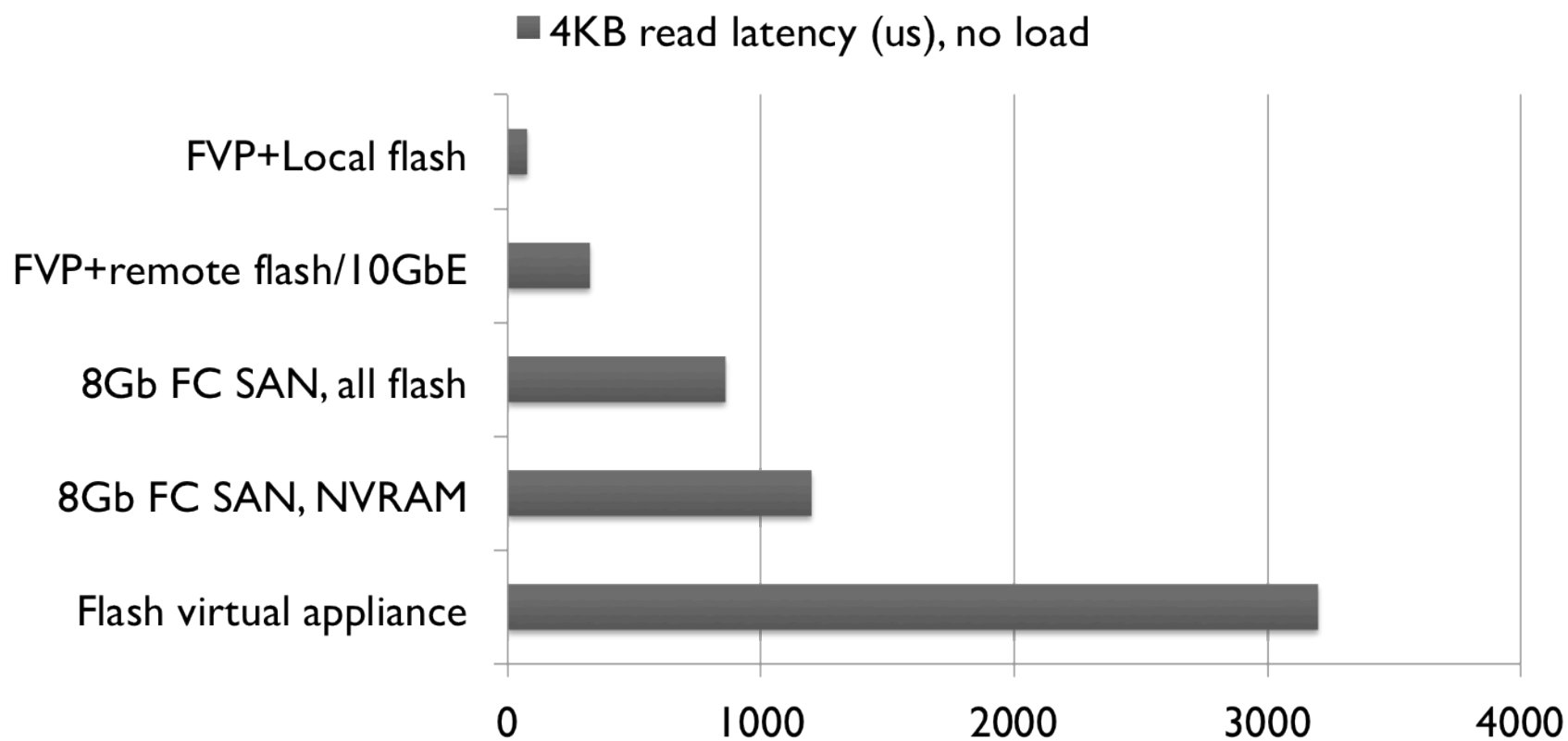


# Clustering considerations - 2

- Why
  - More efficient use of precious flash
  - Server-2-server network more scalable
  - Compatibility with clustered hypervisor features
- Challenges
  - Network cache coherency performance and scale undoes flash performance

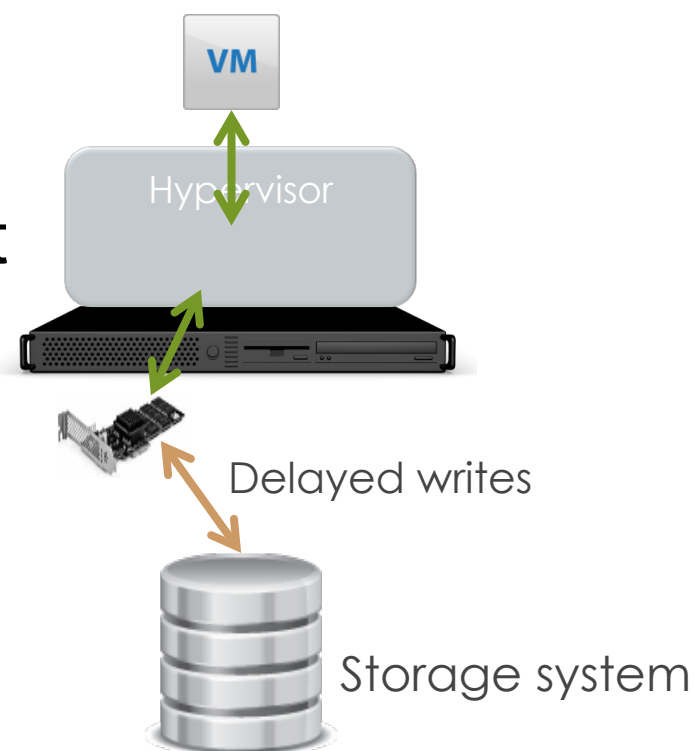


# Latency advantage



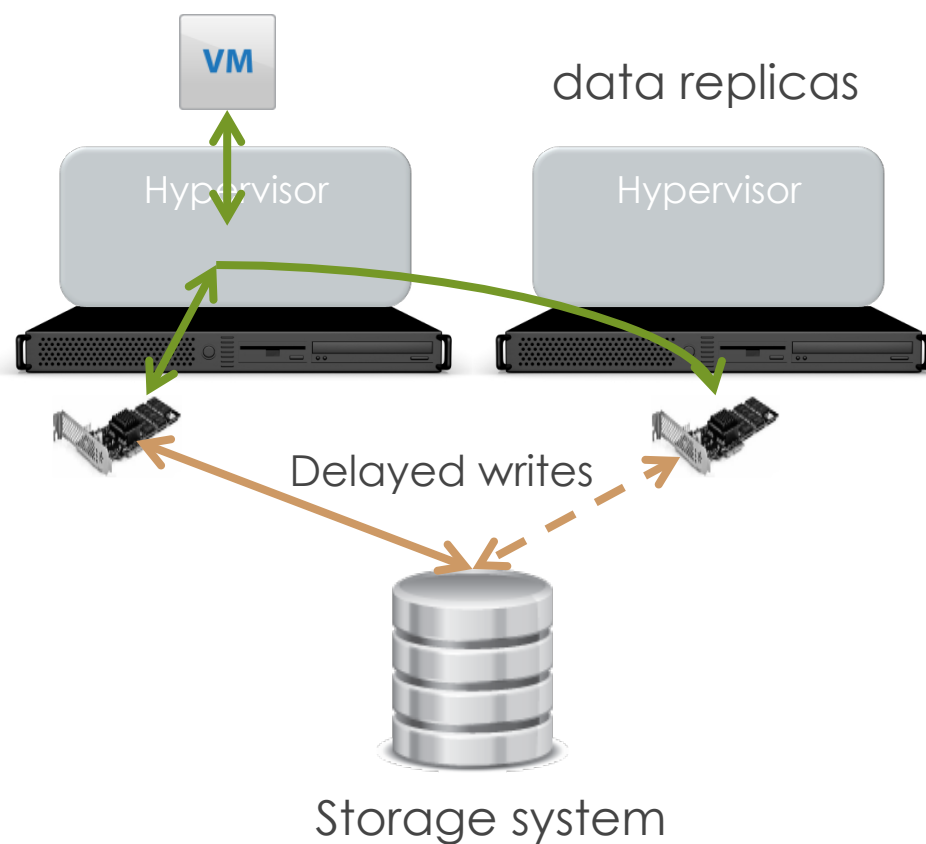
# Write problem - 1

- ❑ What: Use server flash to accelerate VM writes just like reads
- ❑ Why
  - ❑ Make this applicable to most workloads
  - ❑ Fundamentally change storage infra design
    - ❑ Deploy storage systems for new “worst case”



# Write problem - 2

- ❑ Challenges
  - ❑ Failure handling
  - ❑ Crash consistency
  - ❑ SAN replication
- ❑ Solution
  - ❑ Fault tolerant write back acceleration



# Effect of writeback on databases

win7-sqliosim-WB

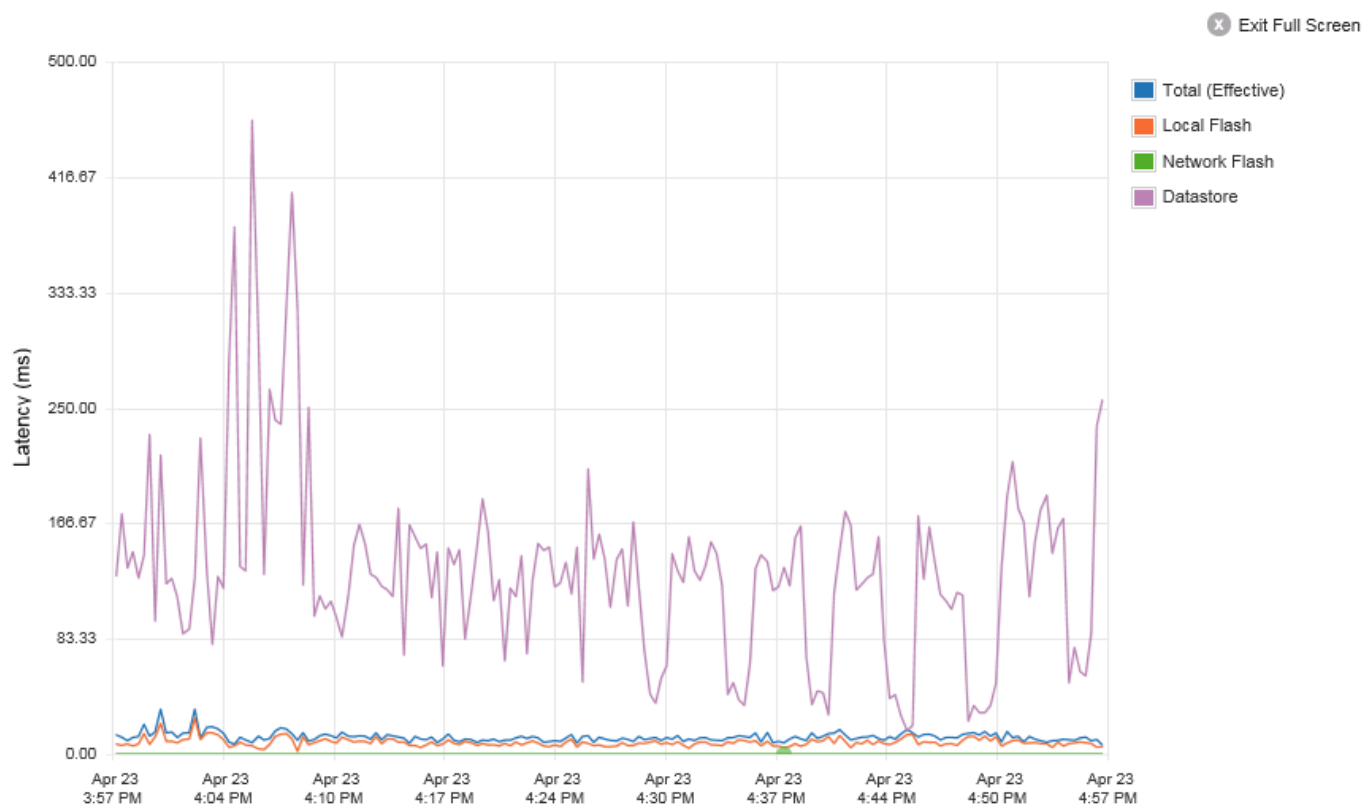
IOPS

Latency

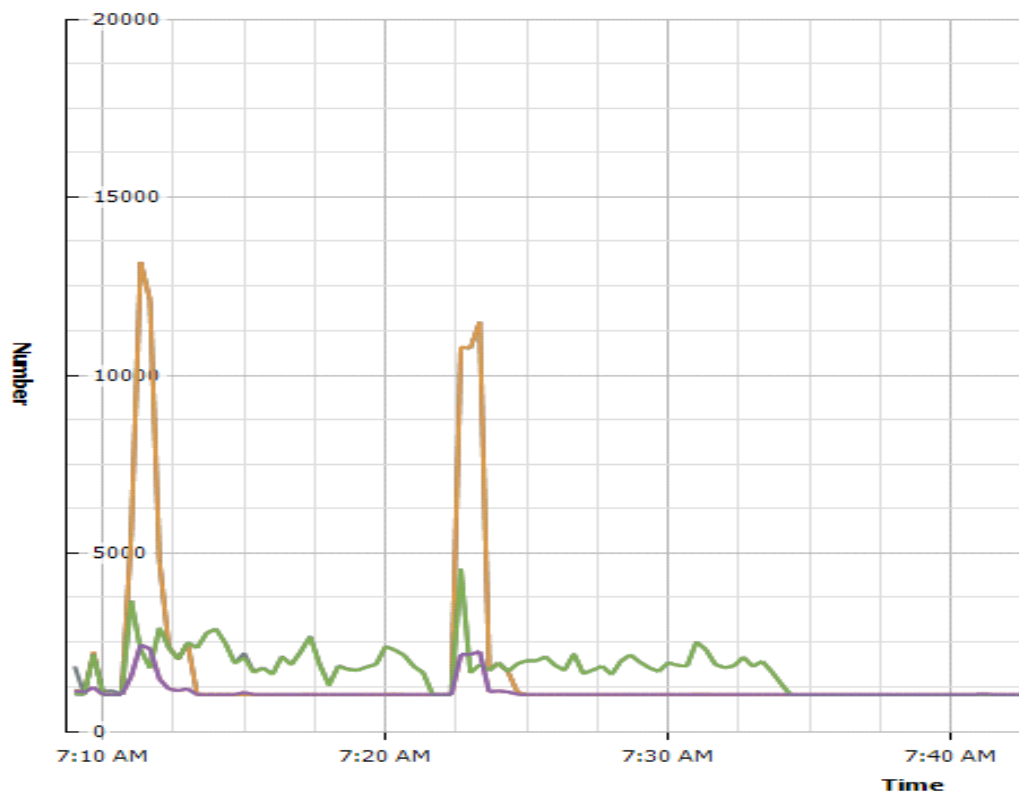
Throughput

Hit Rate & Eviction Rate

[View Usage](#)



# The big deal with write acceleration



## Performance Chart Legend

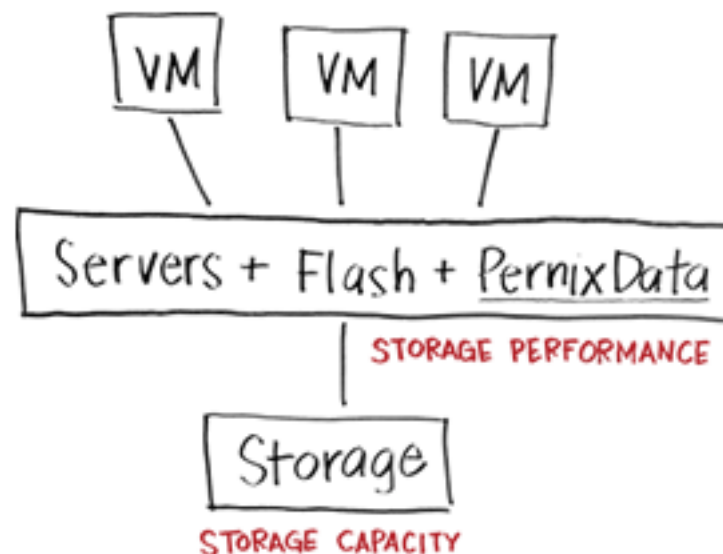
Key	Object	Measurement	Rollup	Units
■	p SCSI.vmhba38-p SCSI.0:0-naa.6006016044a0e01e07e56f3219e21e	average write requests per second	average	Number
■	p SCSI.vmhba38-p SCSI.0:0-naa.6006016044a0e01e07e56f3219e21e	average read requests per second	average	Number
■	block.fio/c0d0:0-block.0:0-mpx.vmhba3:0	average write requests per second	average	Number
■	block.fio/c0d0:0-block.0:0-mpx.vmhba3:0	average read requests per second	average	Number
■	fc.20000000c9dd5c45:10000000c9dd5c45-fc.5006016044a0e01e07e56f3219e21e	average write requests per second	average	Number
■	fc.20000000c9dd5c45:10000000c9dd5c45-fc.5006016044a0e01e07e56f3219e21e	average read requests per second	average	Number

# Summary

- ❑ These are the beginnings of a data-in-motion tier via a Flash Hypervisor
- ❑ What is that?
  - ❑ **Scale-out acceleration:** No disruption to existing VMs and storage systems
  - ❑ **Hypervisor-resident resource manager:** Virtualizes flash just like hypervisors virtualize CPU, memory, network
  - ❑ **Clustered:** Allows cross-server access of hot data, and support for clustered hypervisor features (eg vMotion)
  - ❑ **Fault Tolerant:** Solves write acceleration problem for wide (infrastructure-level) applicability

# Example: PernixData FVP

- ❑ FVP is software to aggregate server-side flash into a scale out data acceleration tier
  - ❑ Non-intrusive acceleration
  - ❑ Transparent remote flash access
  - ❑ Fault tolerant write acceleration



# Other open issues

- ❑ Storage protocol between data-in-motion tier and data-at-rest tier
- ❑ Effect on server hardware architecture
- ❑ Storage policy interface for flash hypervisors
- ❑ Storage stack changes to accommodate flash
- ❑ ...and many more



---

# Thank you

 @SatyamVaghani

More resources: [www.pernixdata.com](http://www.pernixdata.com)