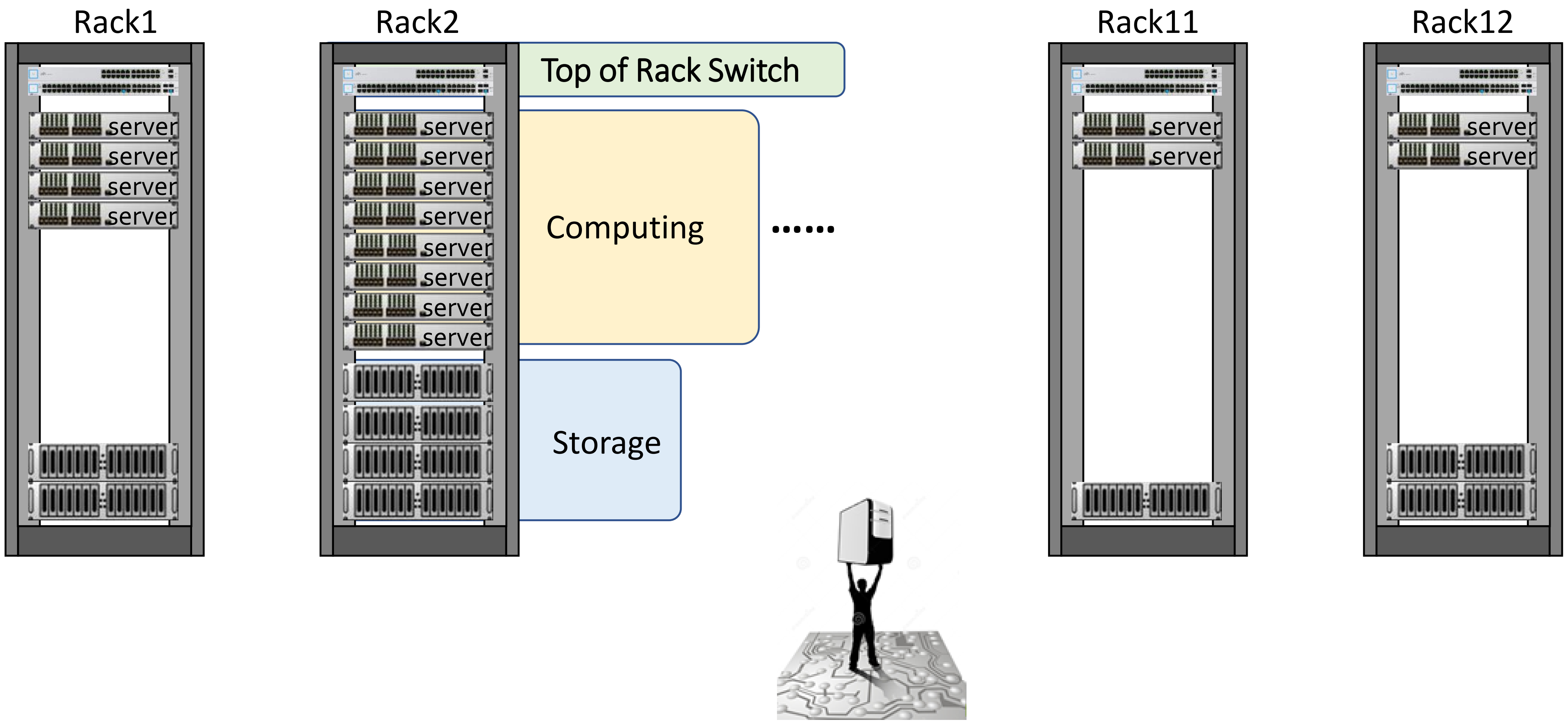
Rethinking Cloud Storage System Software under Multi-Tenancy

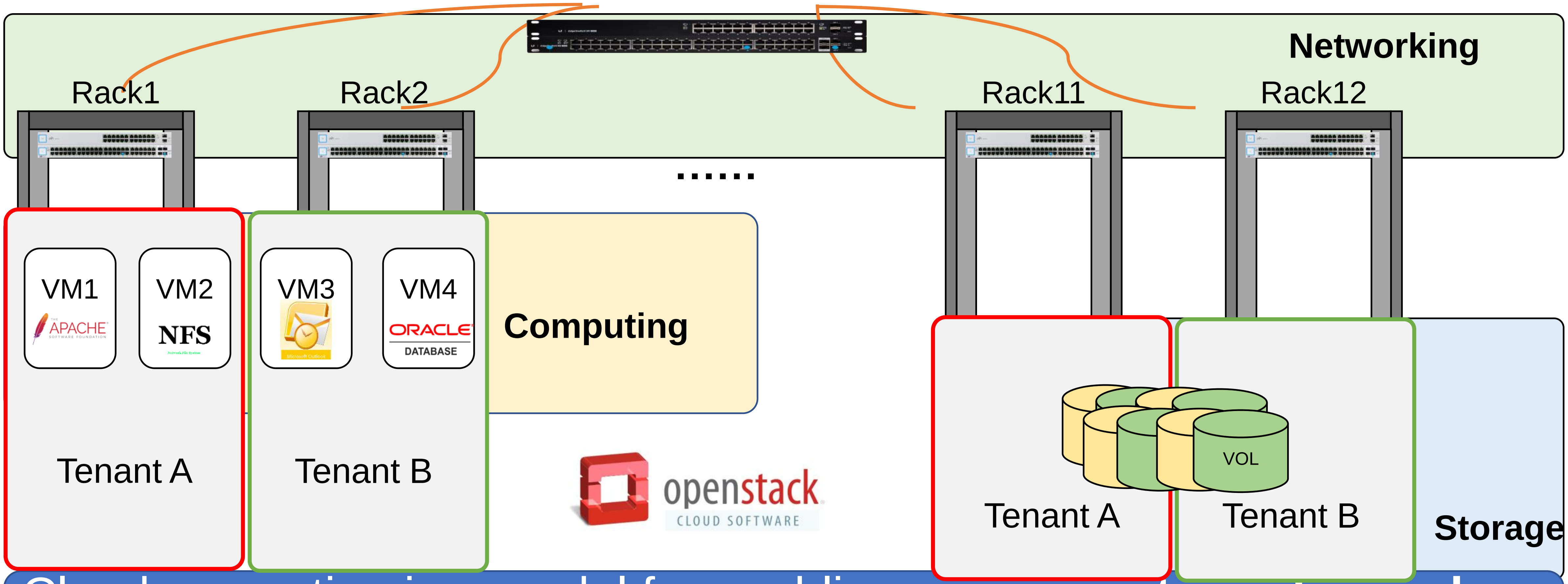
Hui Lu
Purdue University

PURDUE
UNIVERSITY

A R&D Lab Infrastructure

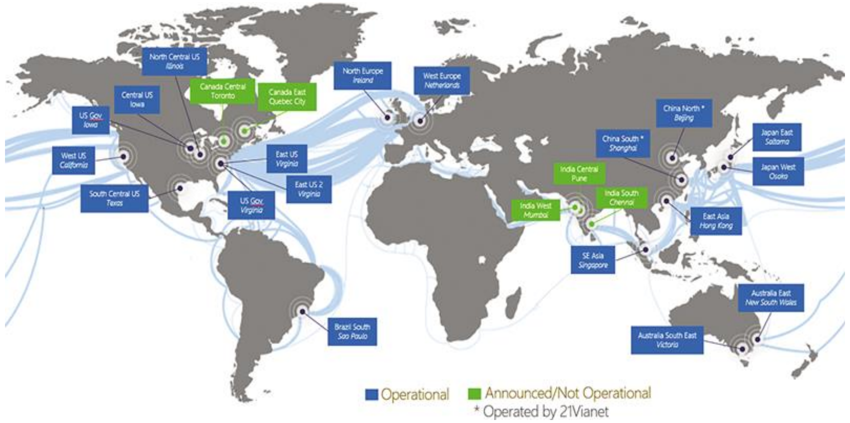


A R&D Lab Infrastructure with **Cloud Computing**



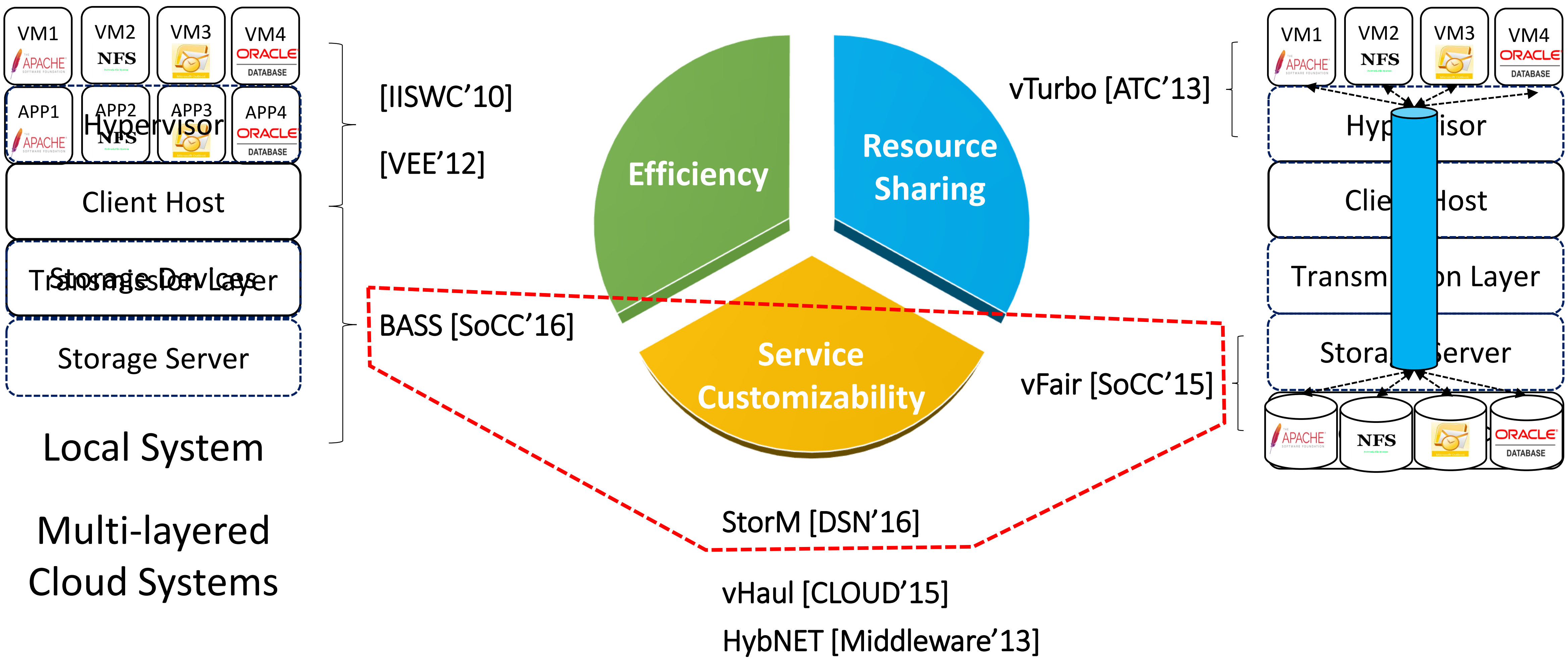
Cloud computing is a model for enabling **convenient, on-demand** network access to a shared pool of configurable resources under **Multi-tenancy**.

Datacenters with Cloud Computing

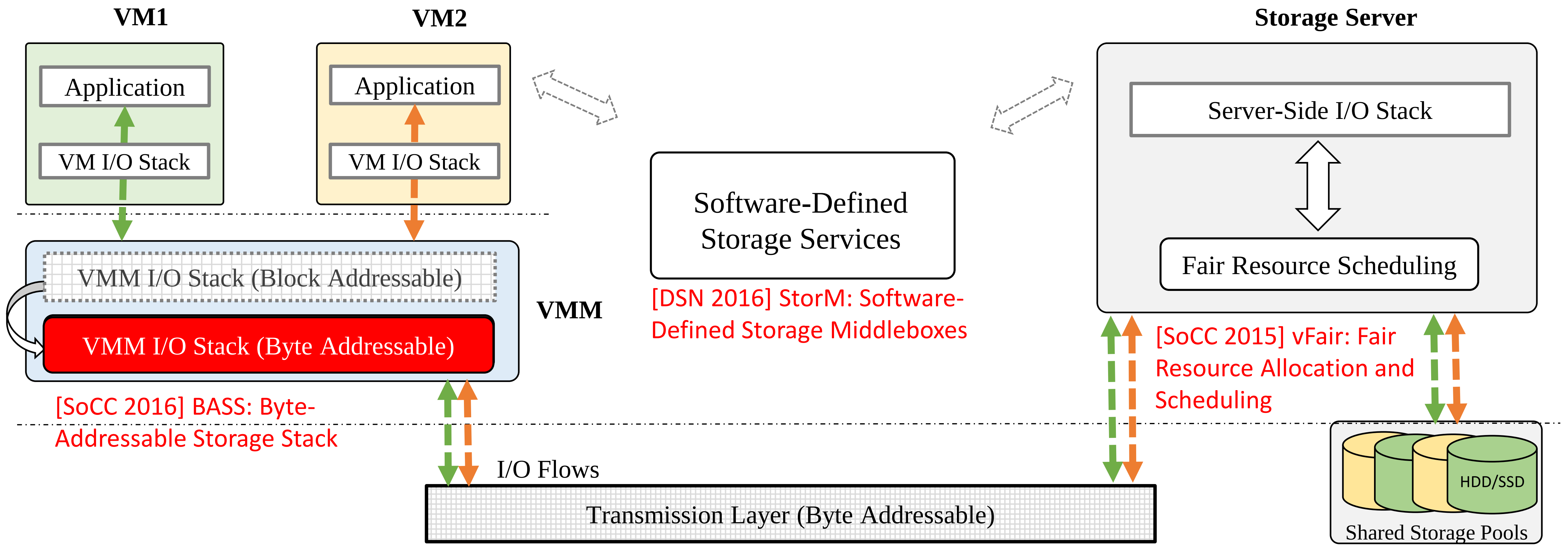


Cloud computing has transformed from a promising model to a commercial reality

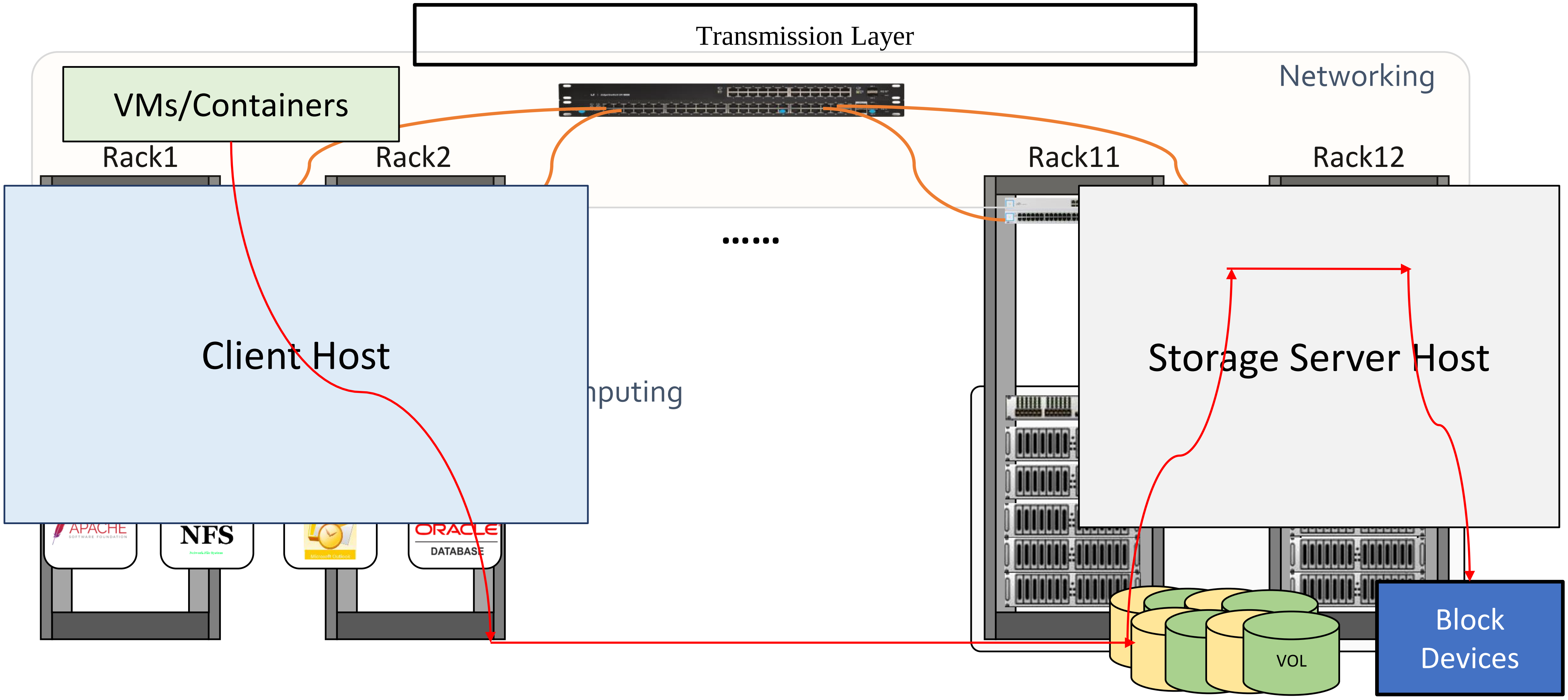
Cloud Computing Challenges



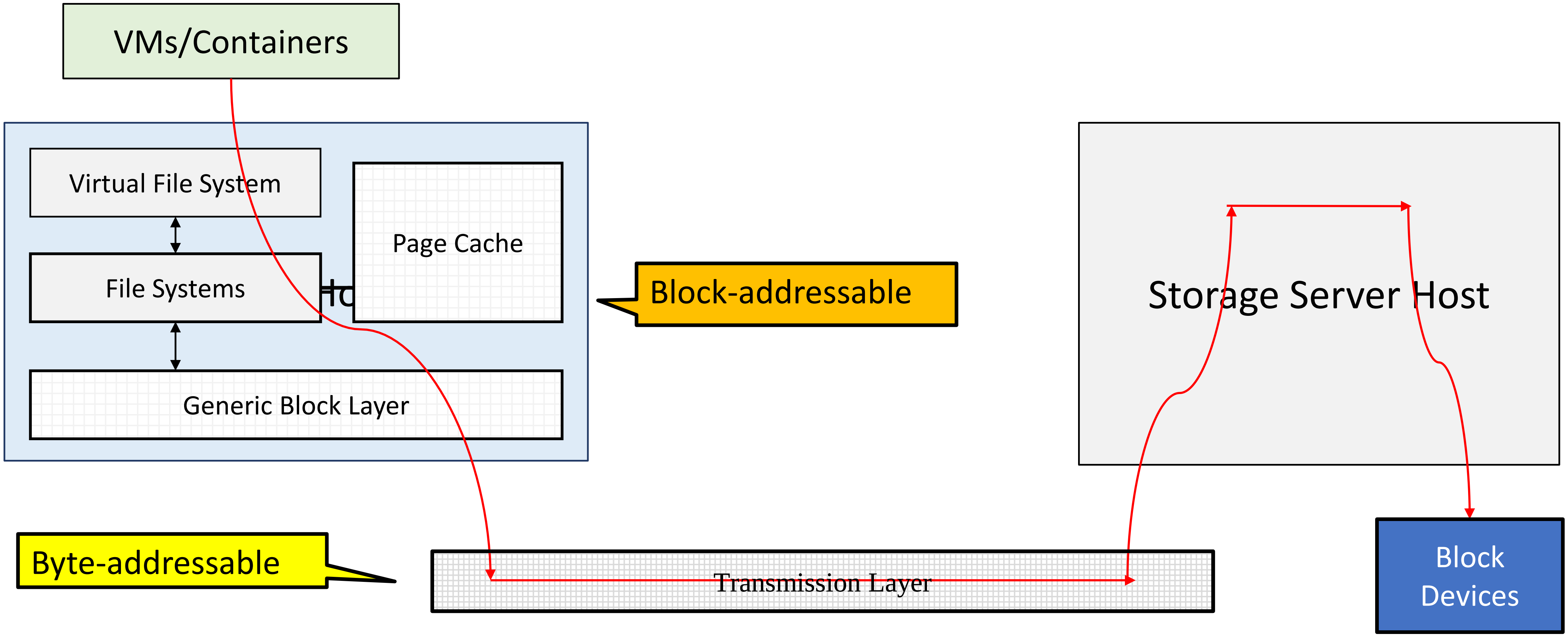
Research Overview: Cloud Storage Systems



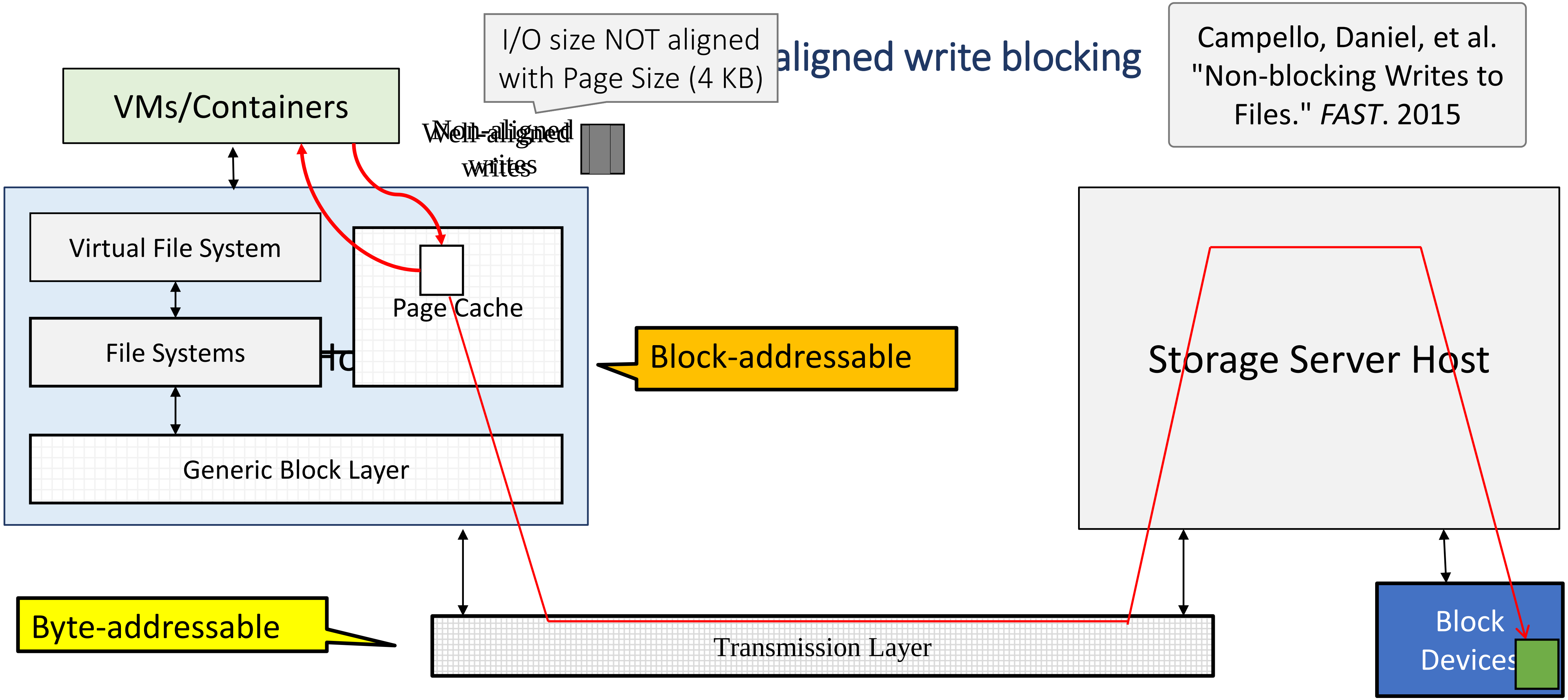
Cloud Block Storage System



Data Granularity Mismatch



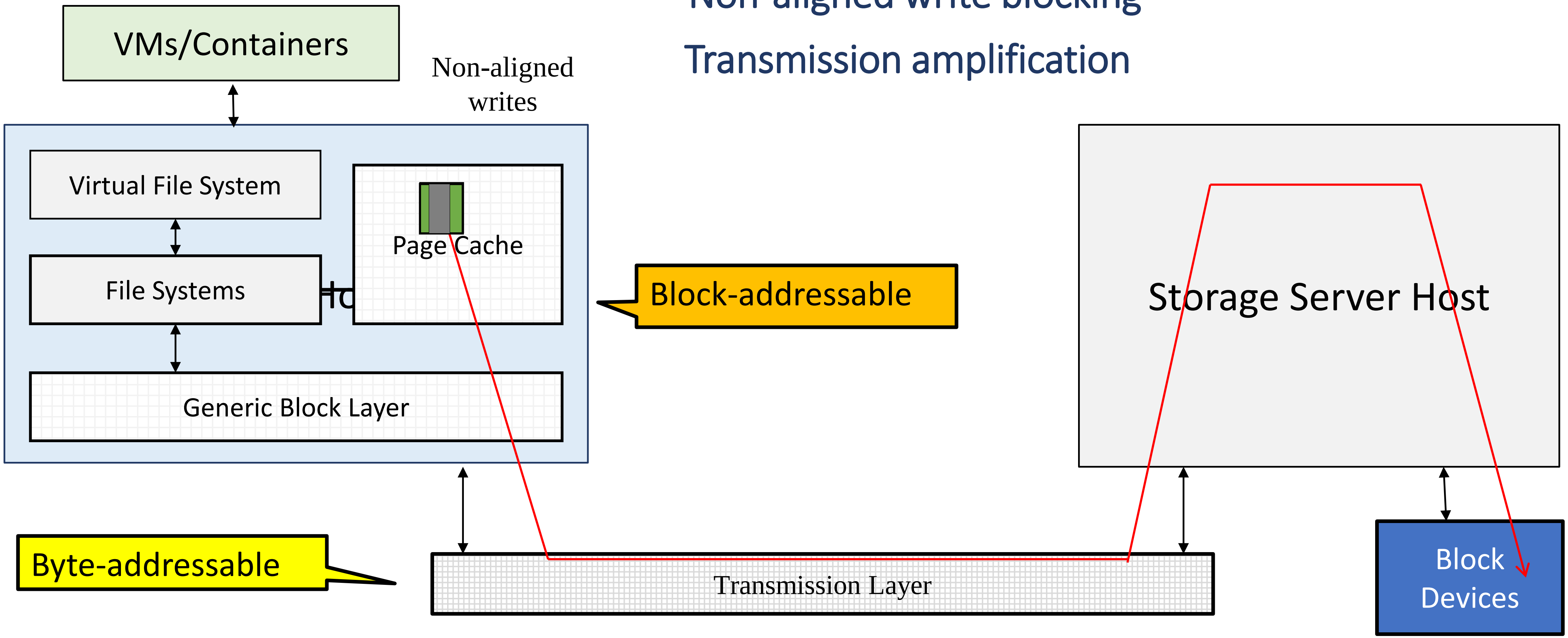
Non-aligned Write Blocking



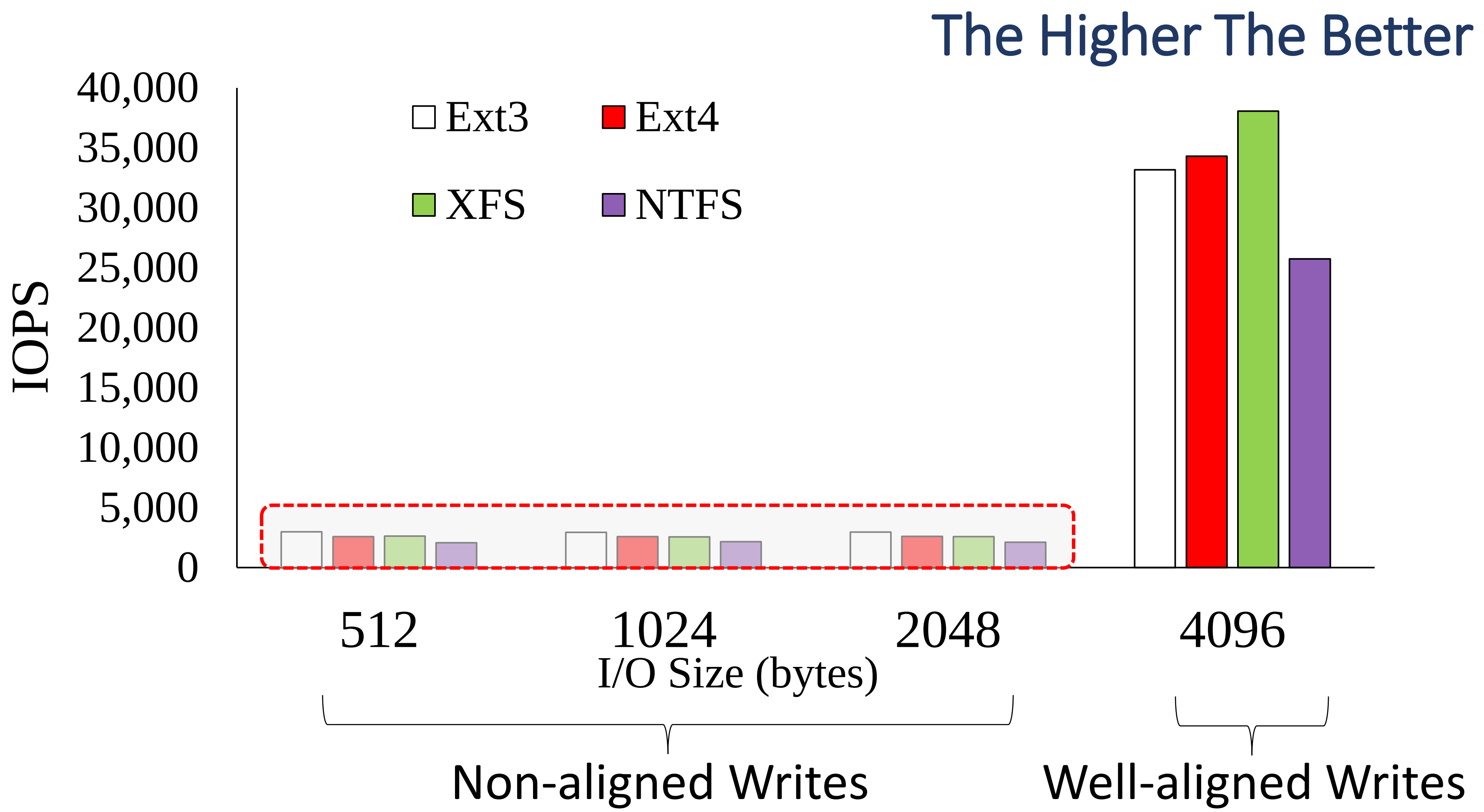
Transmission Amplification

Non-aligned write blocking

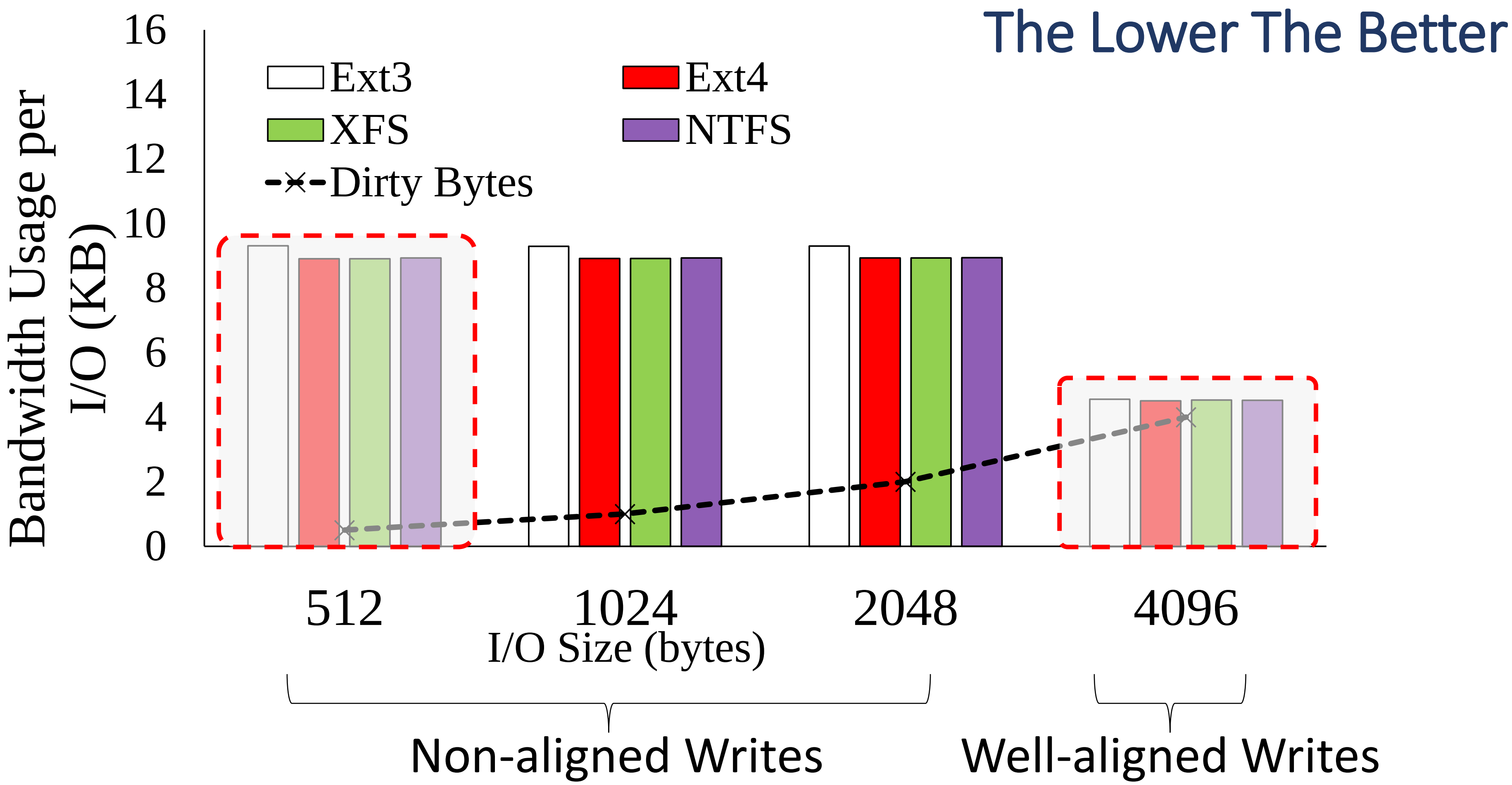
Transmission amplification



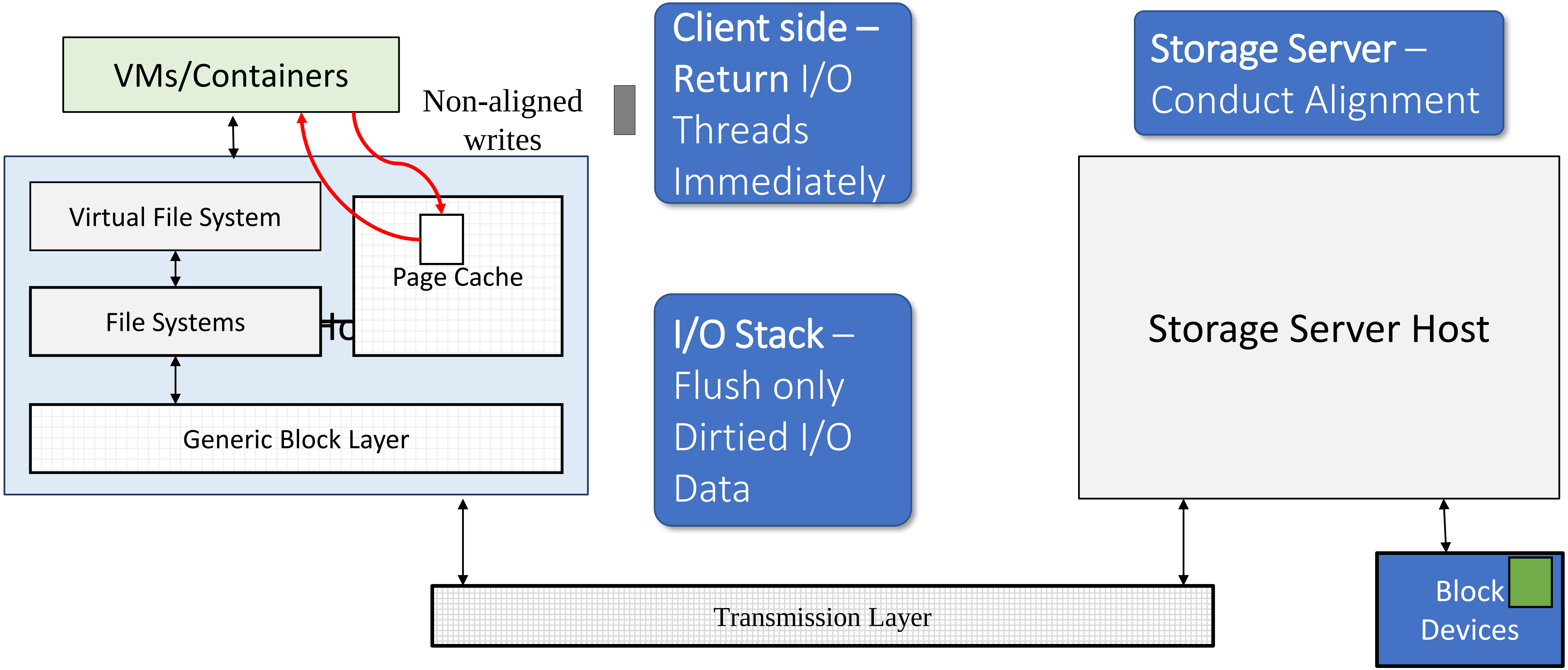
Low End-to-End I/O Throughput



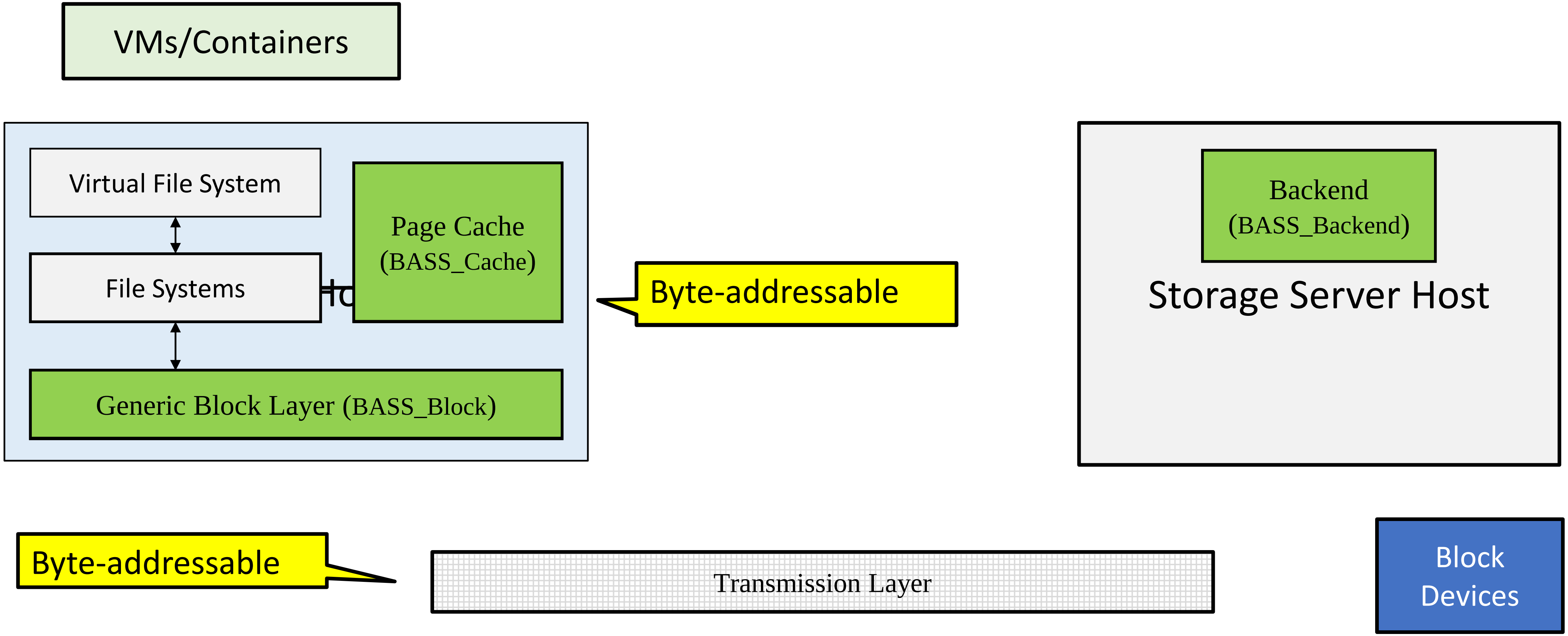
High Network Bandwidth Consumption



A Highly-Efficient Write Framework



Byte-Addressable Storage Stack

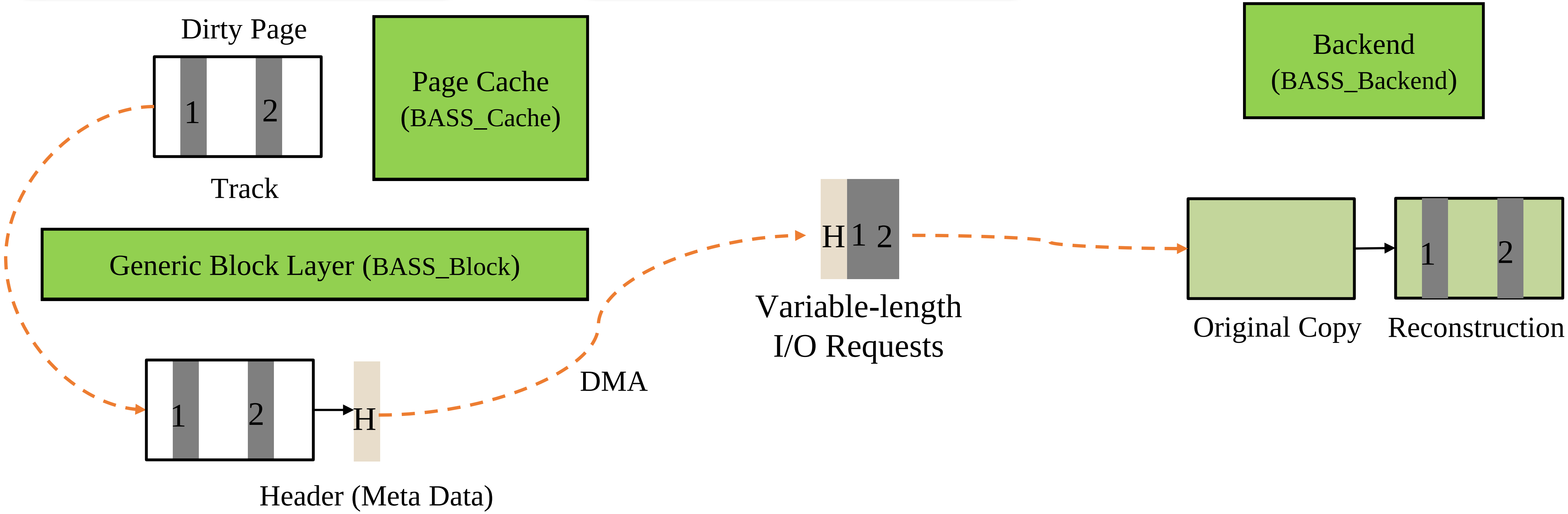


Byte-Addressable Storage Stack

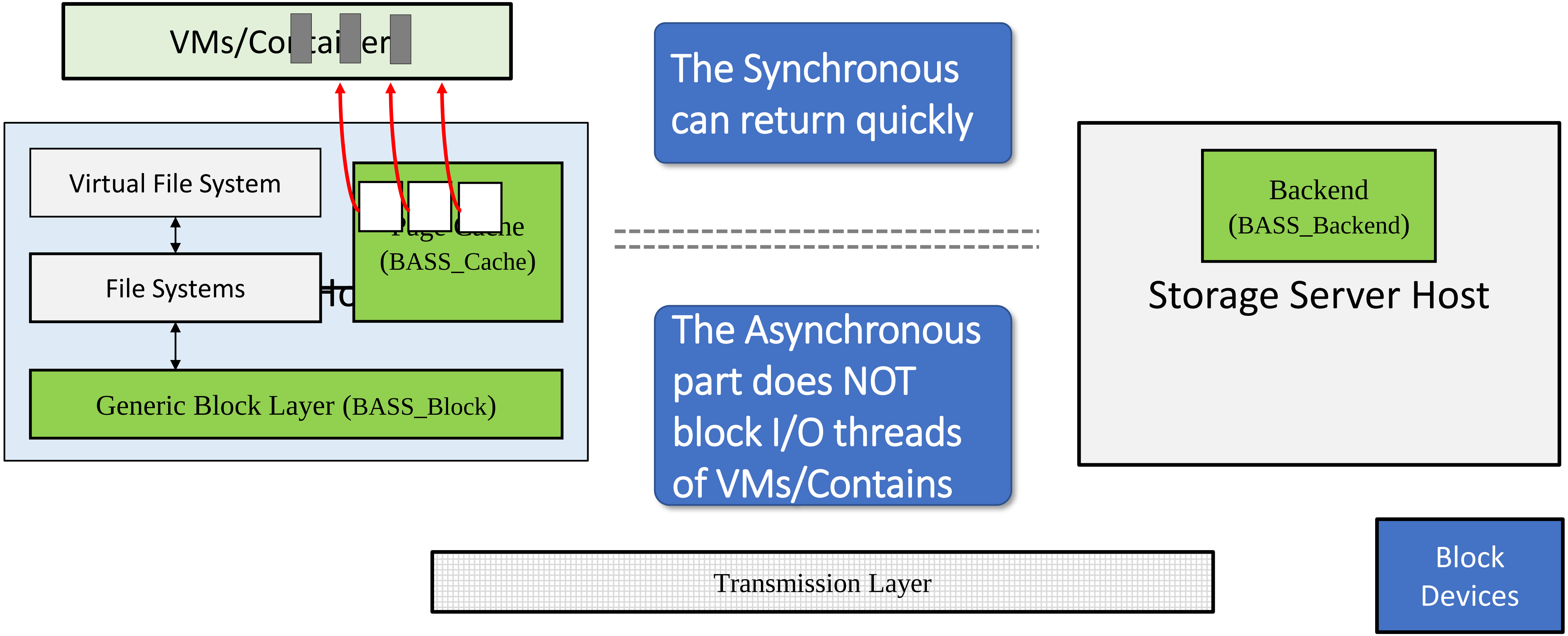
BASS Cache: Keep track of application-level accesses at a byte granularity

BASS Block Layer: Generate "Arbitrary-Length" I/O requests

BASS Backend: Conduct "block-level" operations to align with storage devices

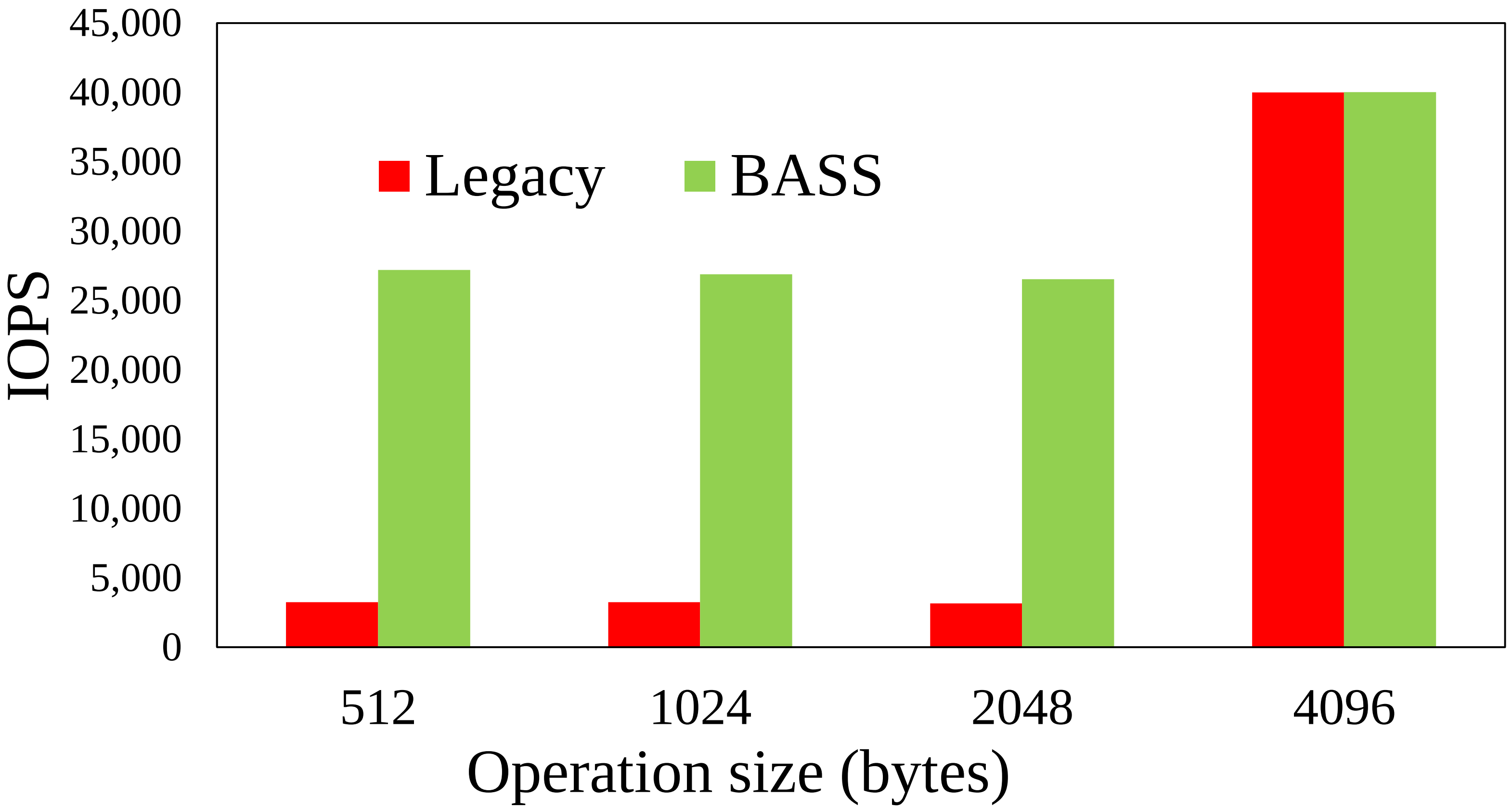


Non-blocking Writes



Performance Gains

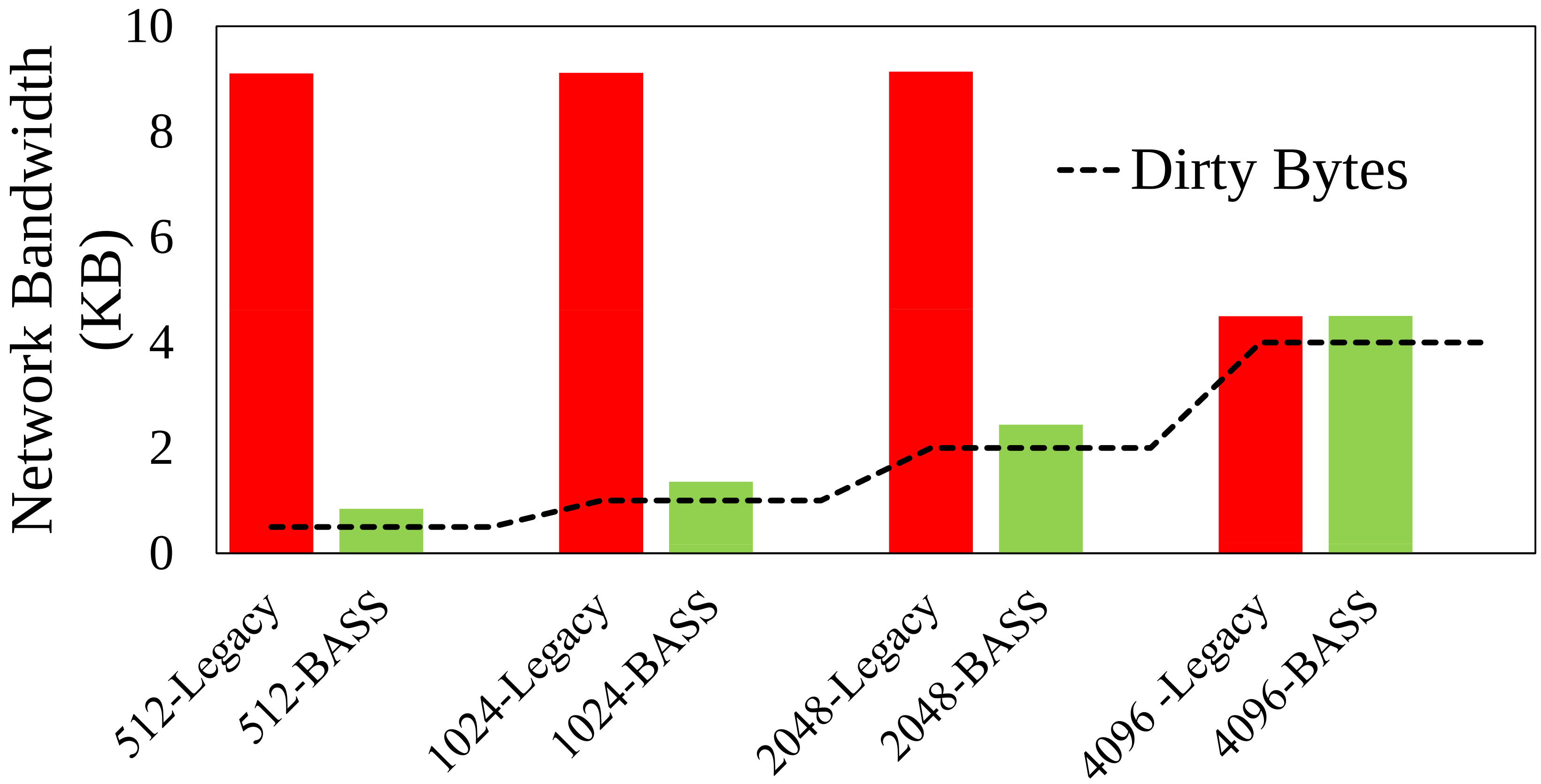
The Higher The Better



BASS increases the performance of non-aligned writes by **8X**, compared to Legacy

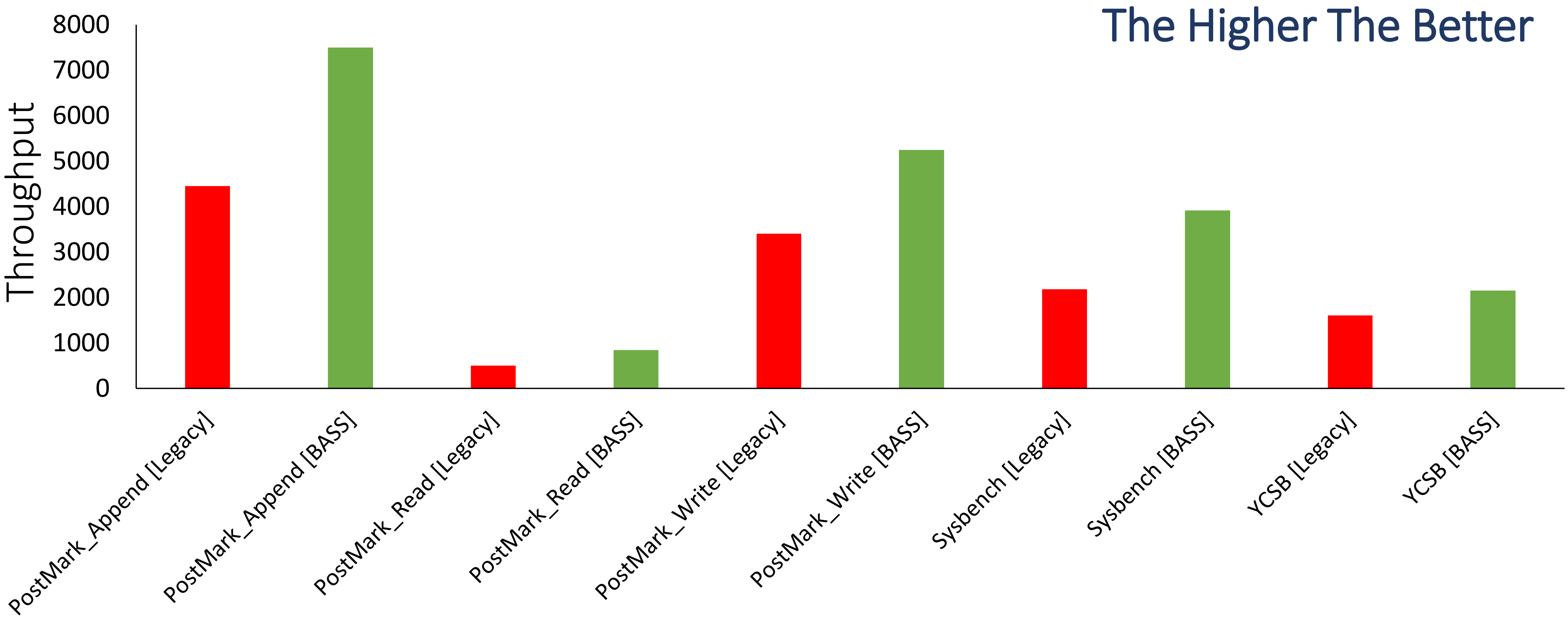
Bandwidth Savings

The Lower the Better



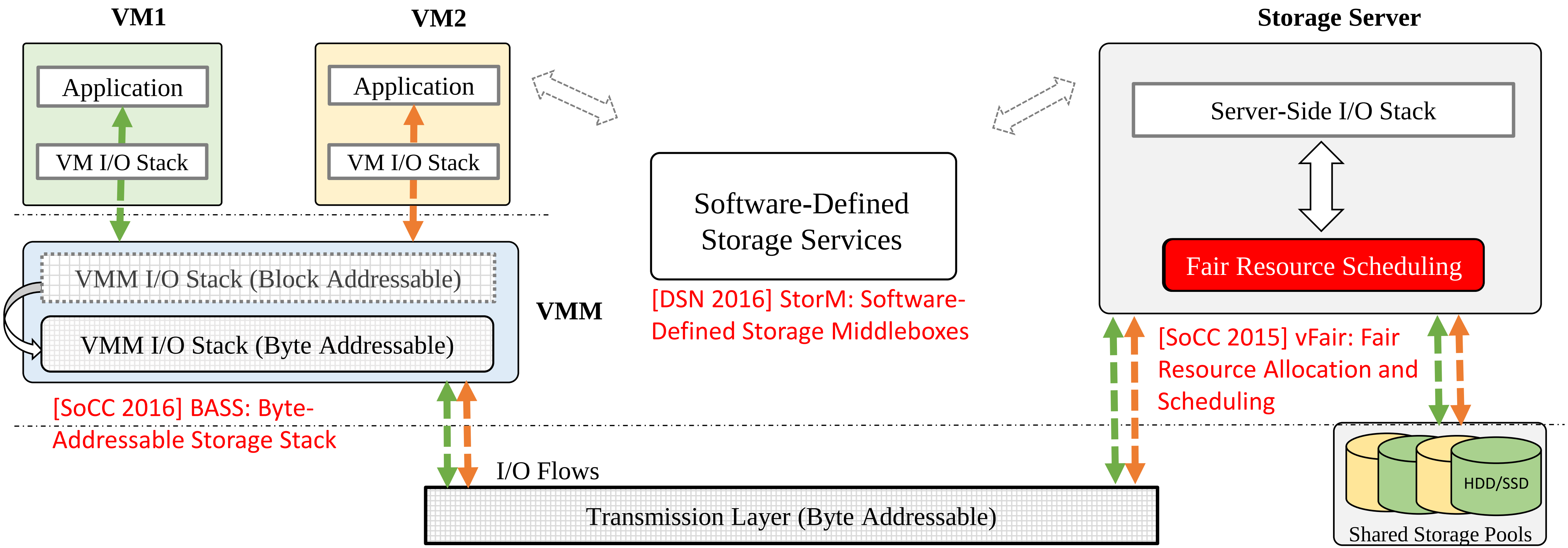
BASS greatly reduces the network usage of non-aligned writes by up to **90%**

Application-level Performance Gains

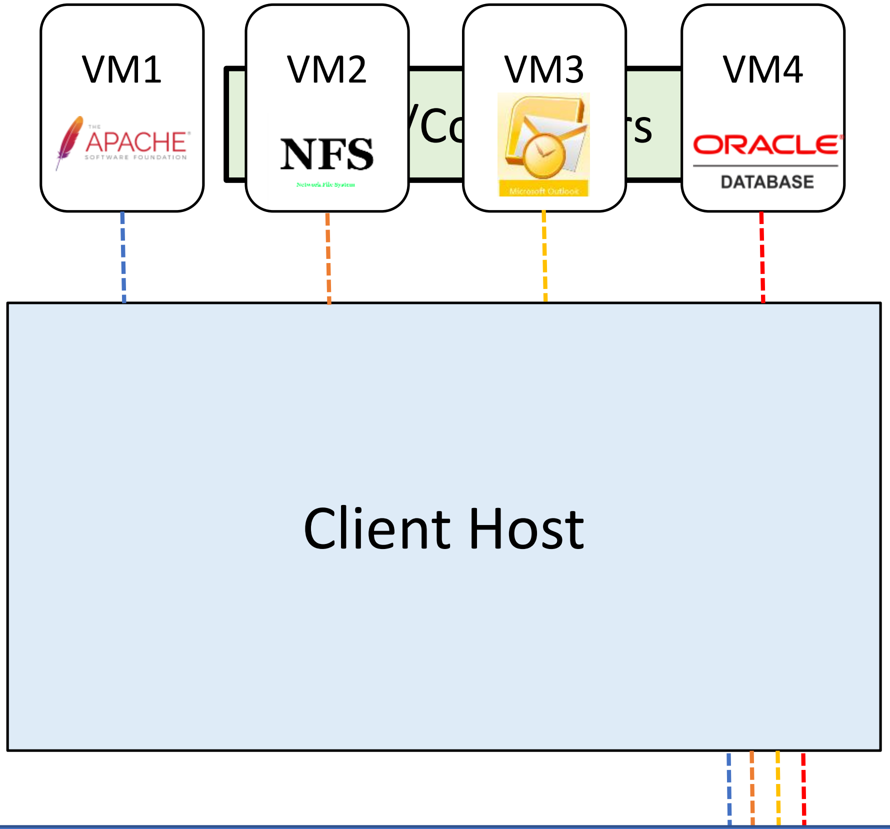


BASS increases the throughput of the write-intensive applications (e.g., **63%** for PostMark), and saves network bandwidth (e.g., **60%** for YCSB)

Research Overview: Cloud Storage Systems

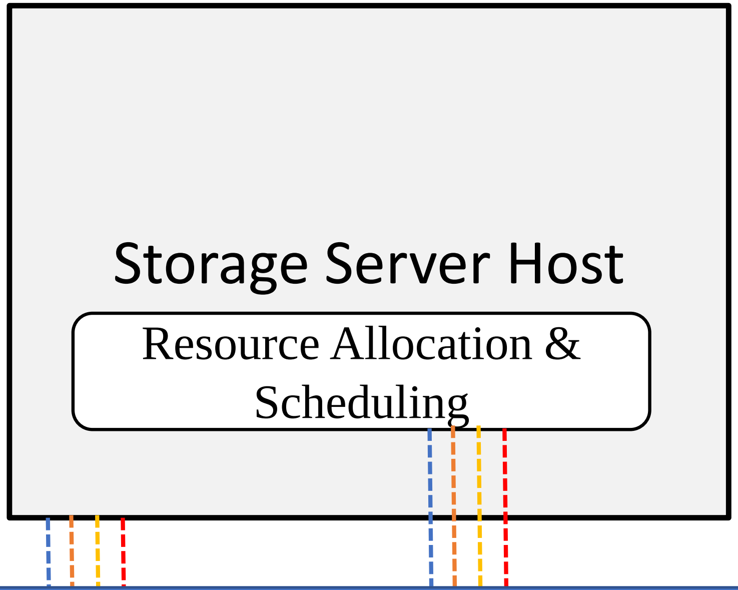


I/O Resource Allocation and Scheduling



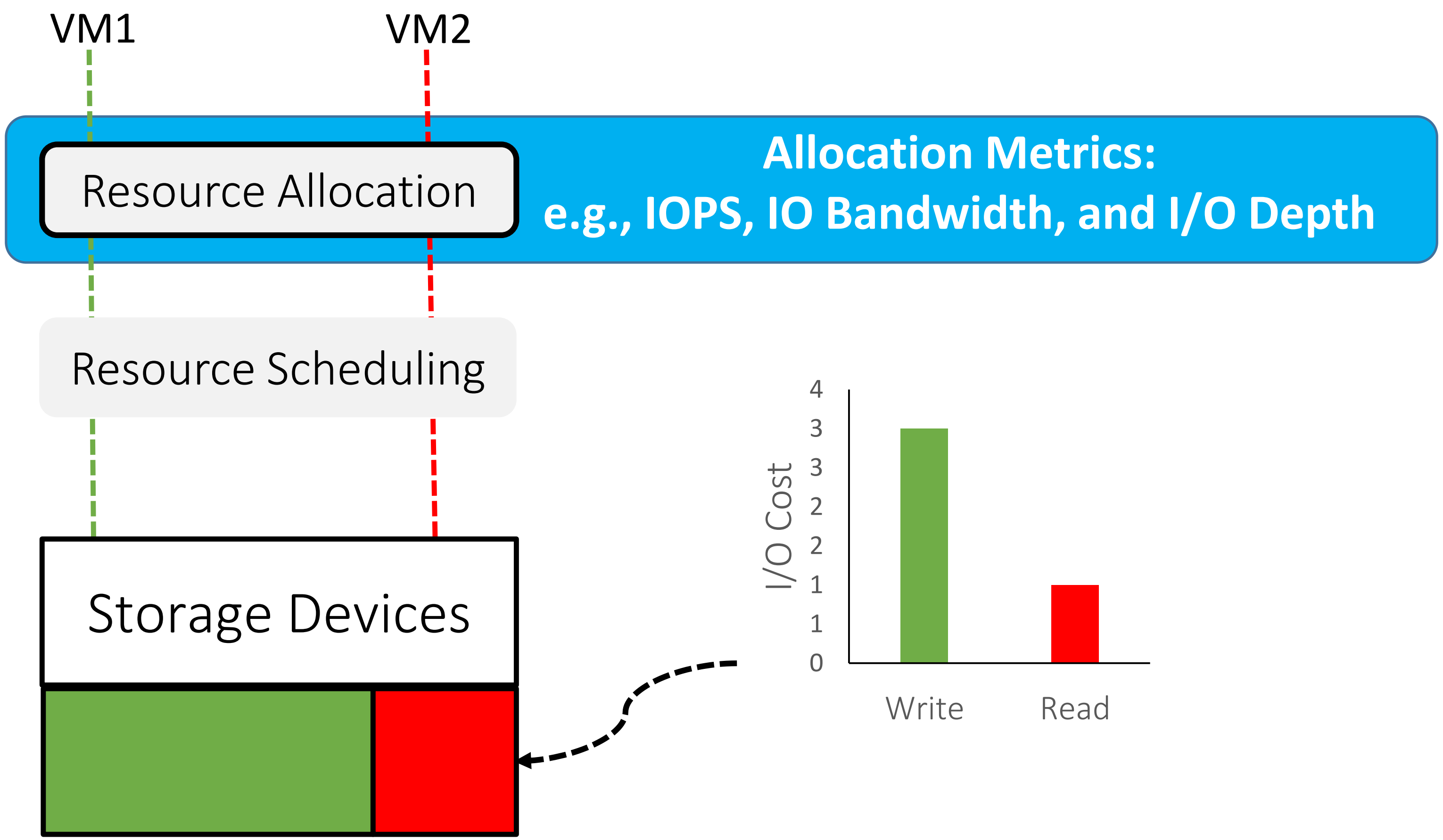
VMs exhibit various IO patterns!

- Read / Write
- Sequential / Random
- Synchronous / Asynchronous
- Various I/O sizes



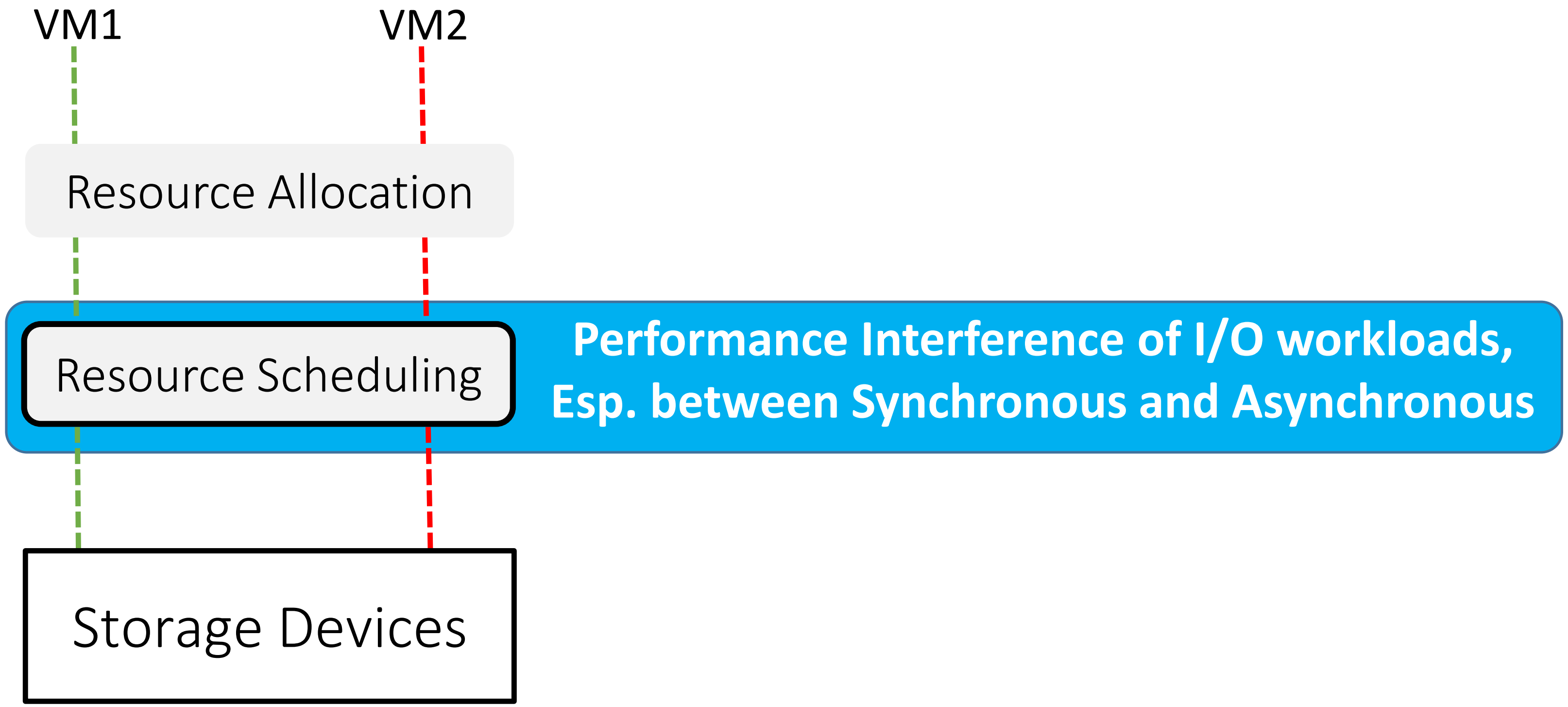
It's hard for I/O schedulers to guarantee fair I/O resource allocation and scheduling

Unfairness of I/O Resource Allocation

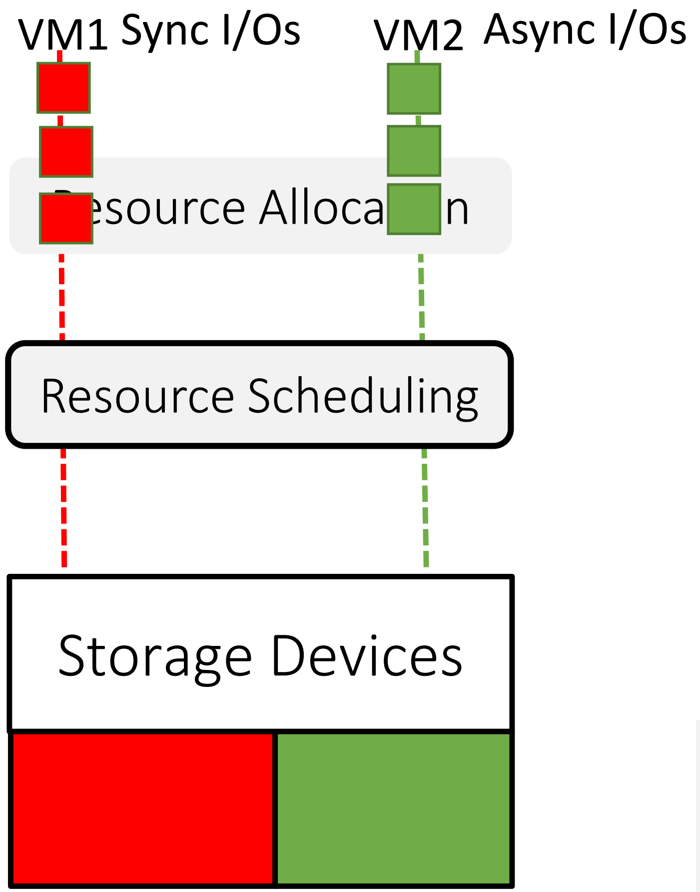


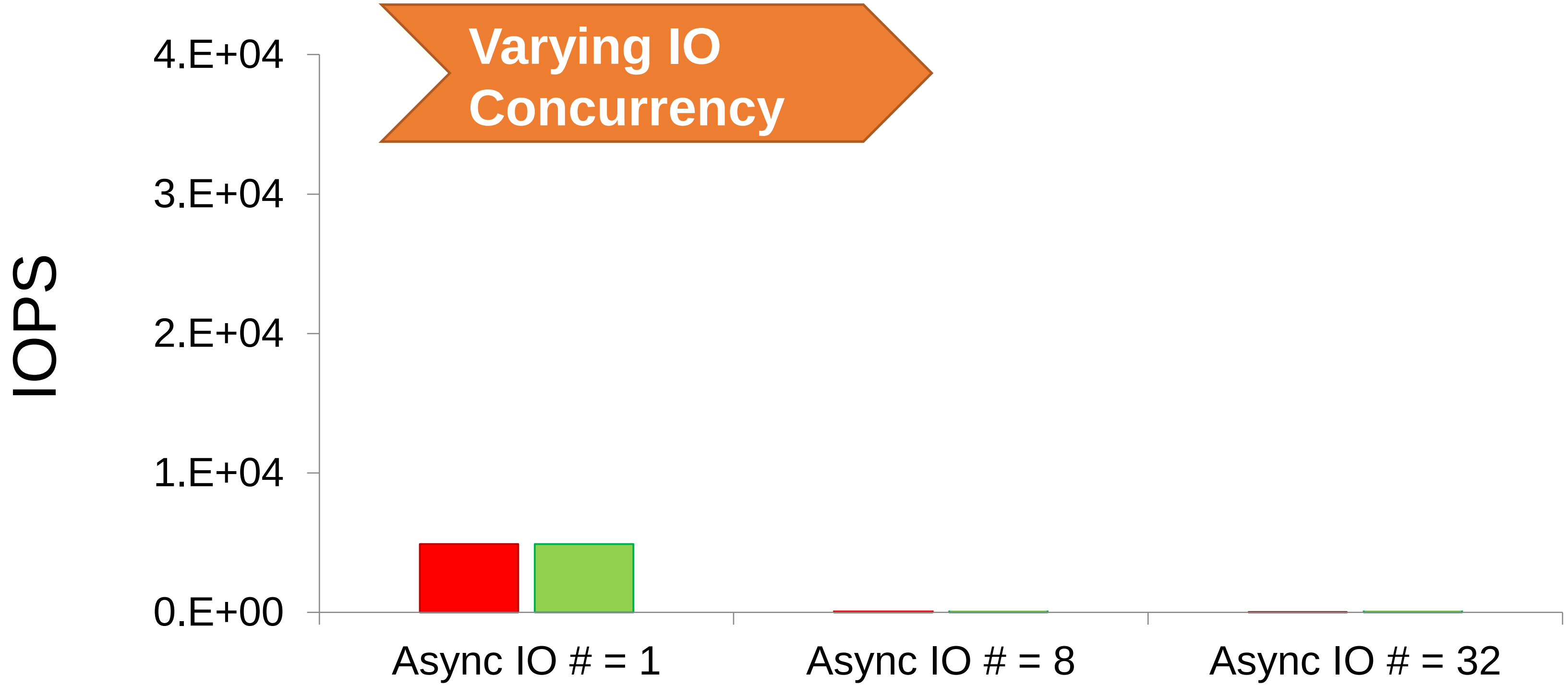
Using simple I/O metrics for I/O resources allocation leads to unfairness

Unfairness of I/O Resource Scheduling



Unfairness of I/O Resource Scheduling

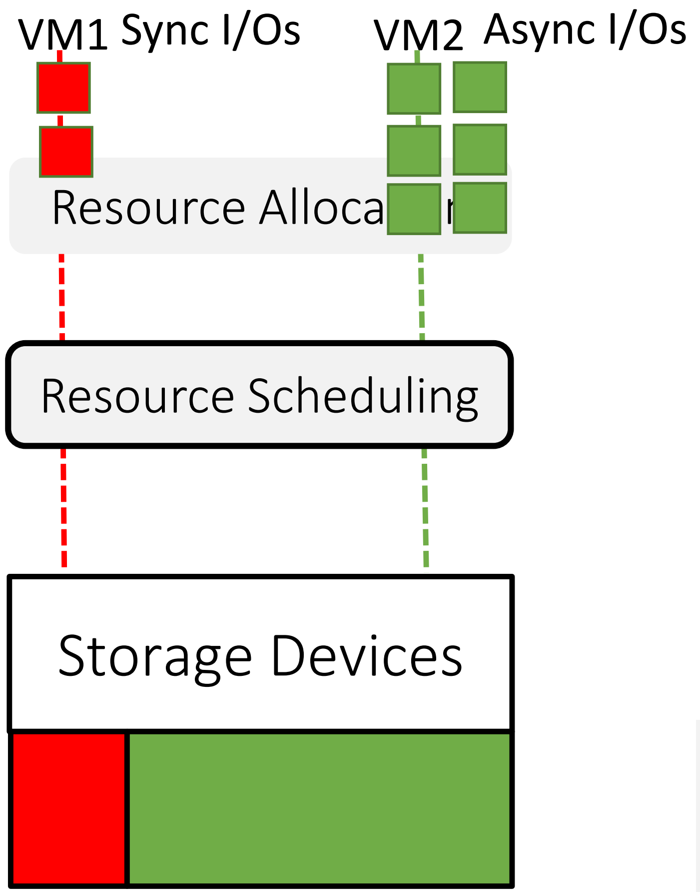


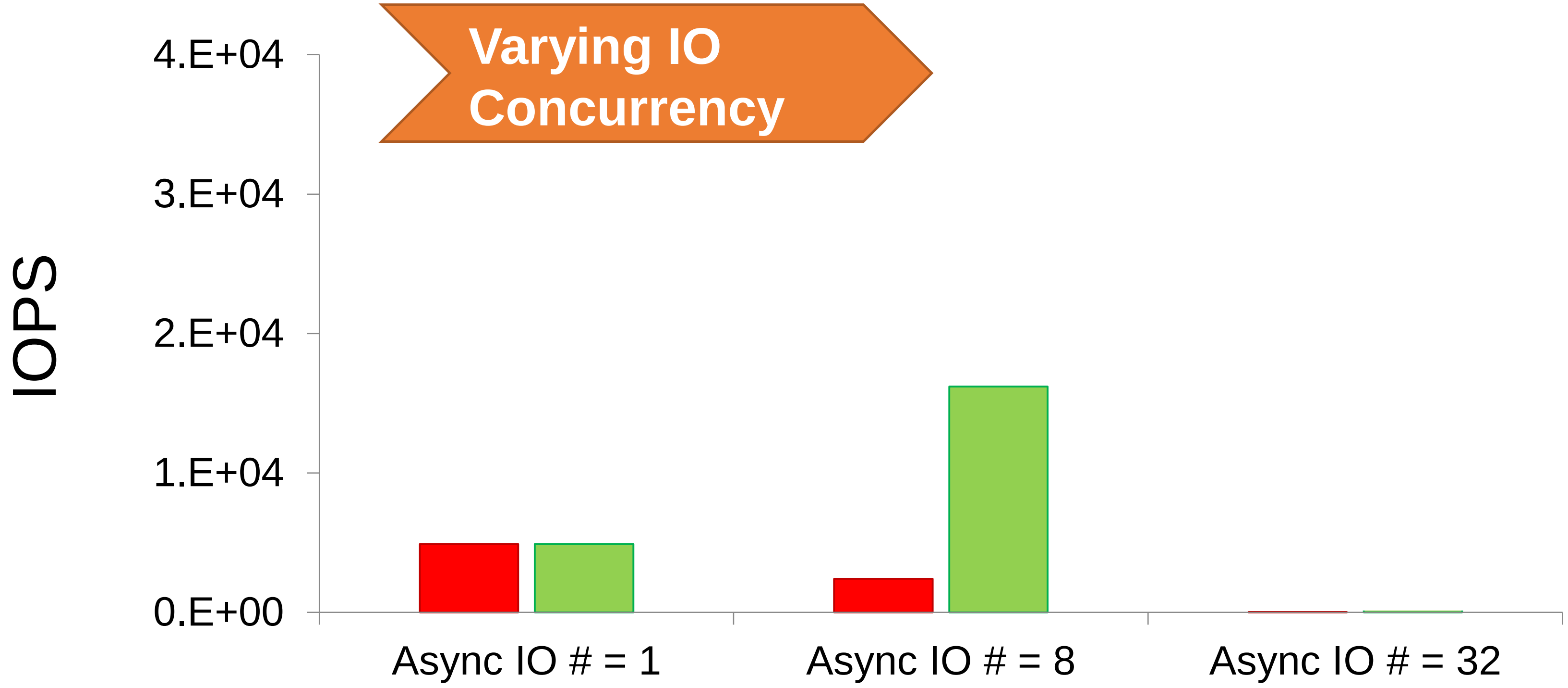


VM1 ■ 4 KB Synchronous Sequential Read (IO # = 1)

VM2 ■ 4 KB Asynchronous Random Read

Unfairness of I/O Resource Scheduling

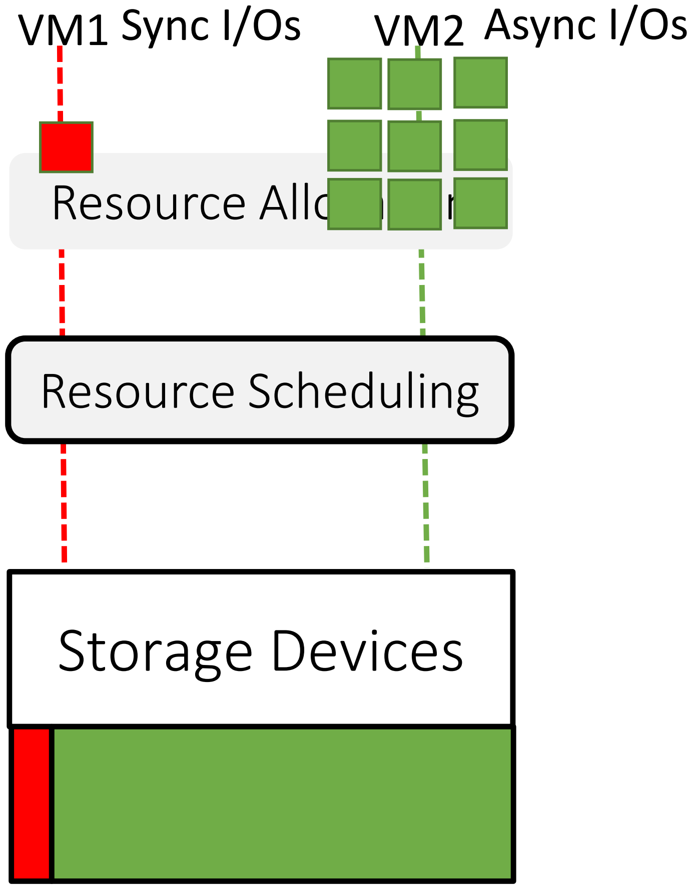


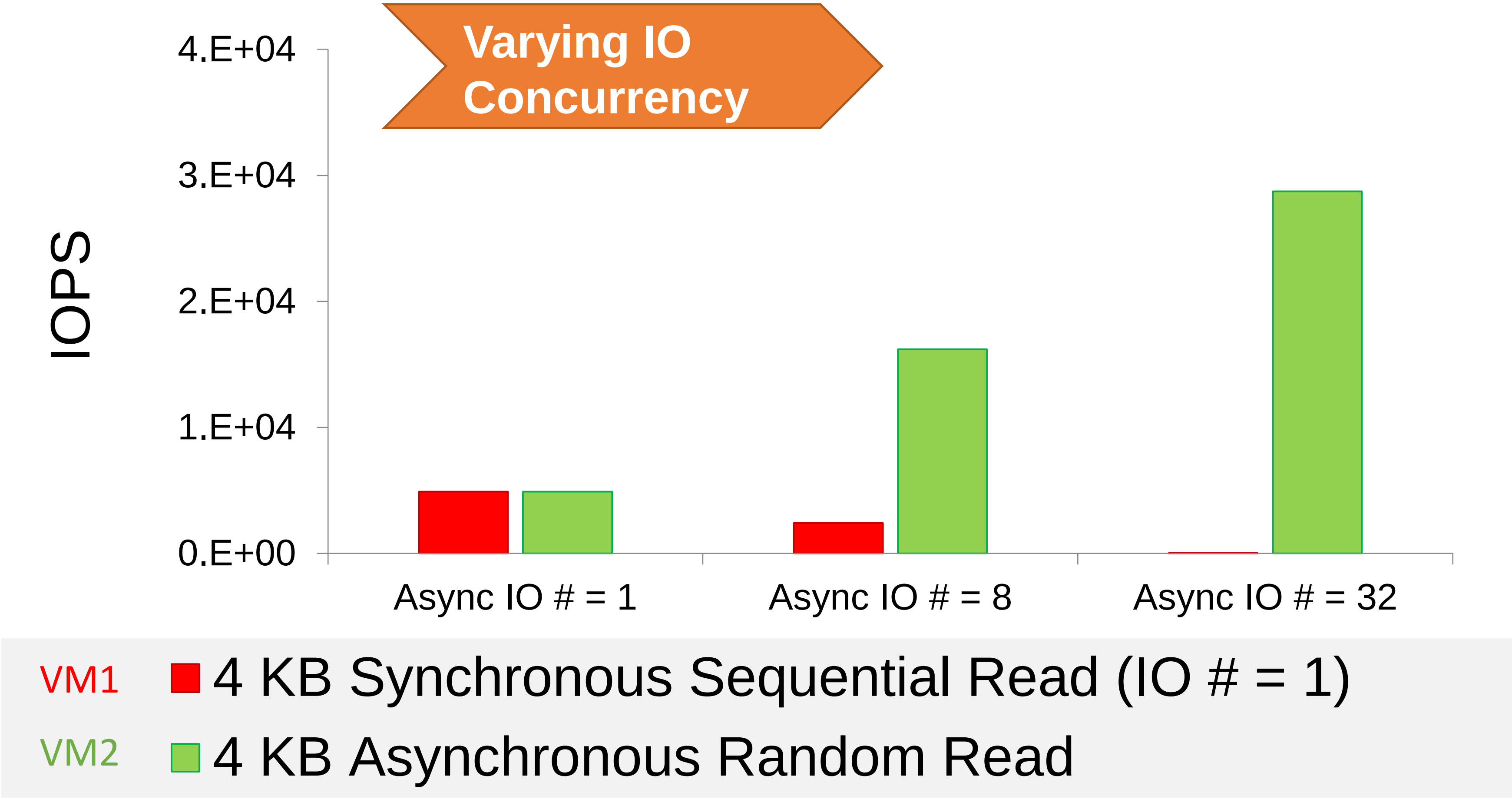


VM1 4 KB Synchronous Sequential Read (IO # = 1)

VM2 4 KB Asynchronous Random Read

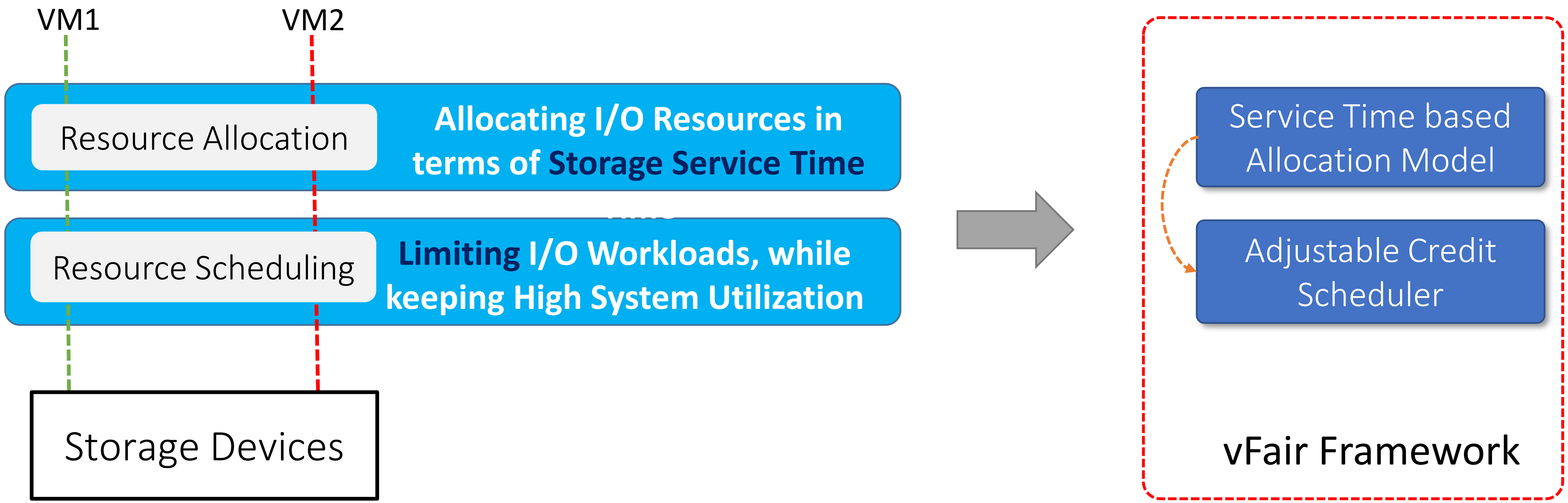
Unfairness of I/O Resource Scheduling



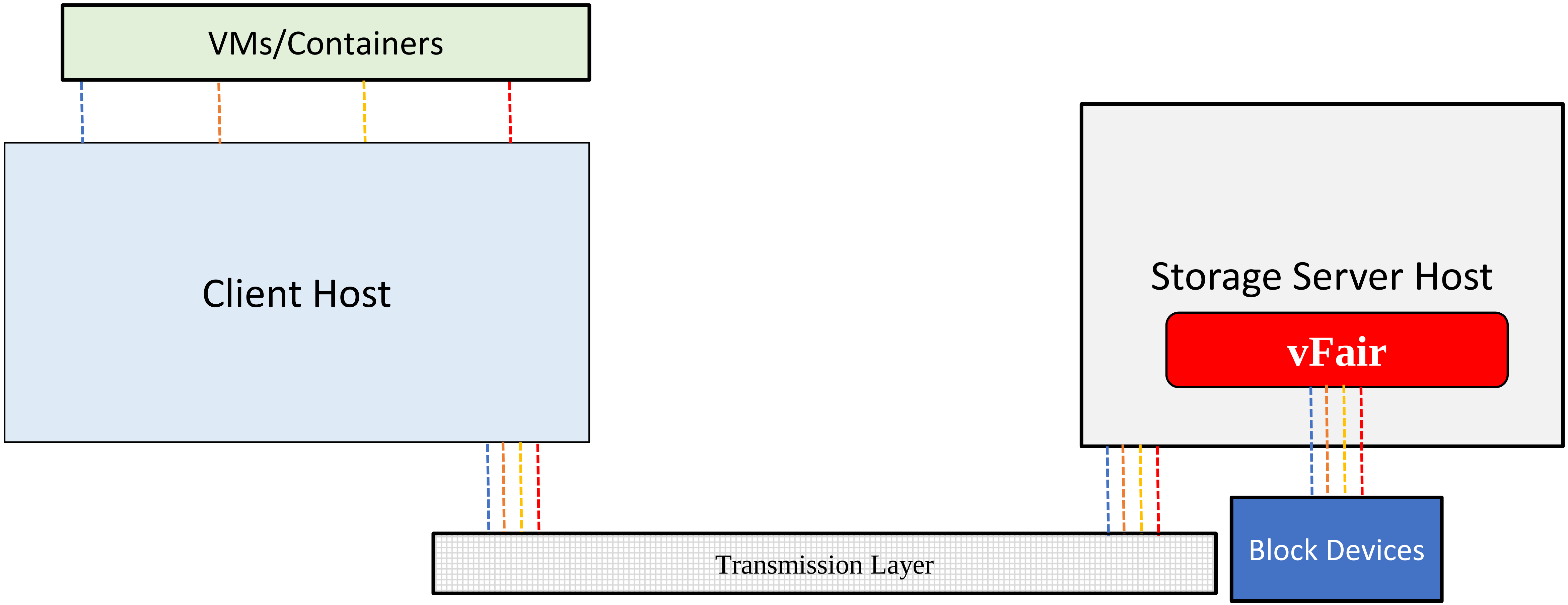


VMs with synchronous, low I/O-concurrency receive unfair storage resources due to interference of I/O workloads

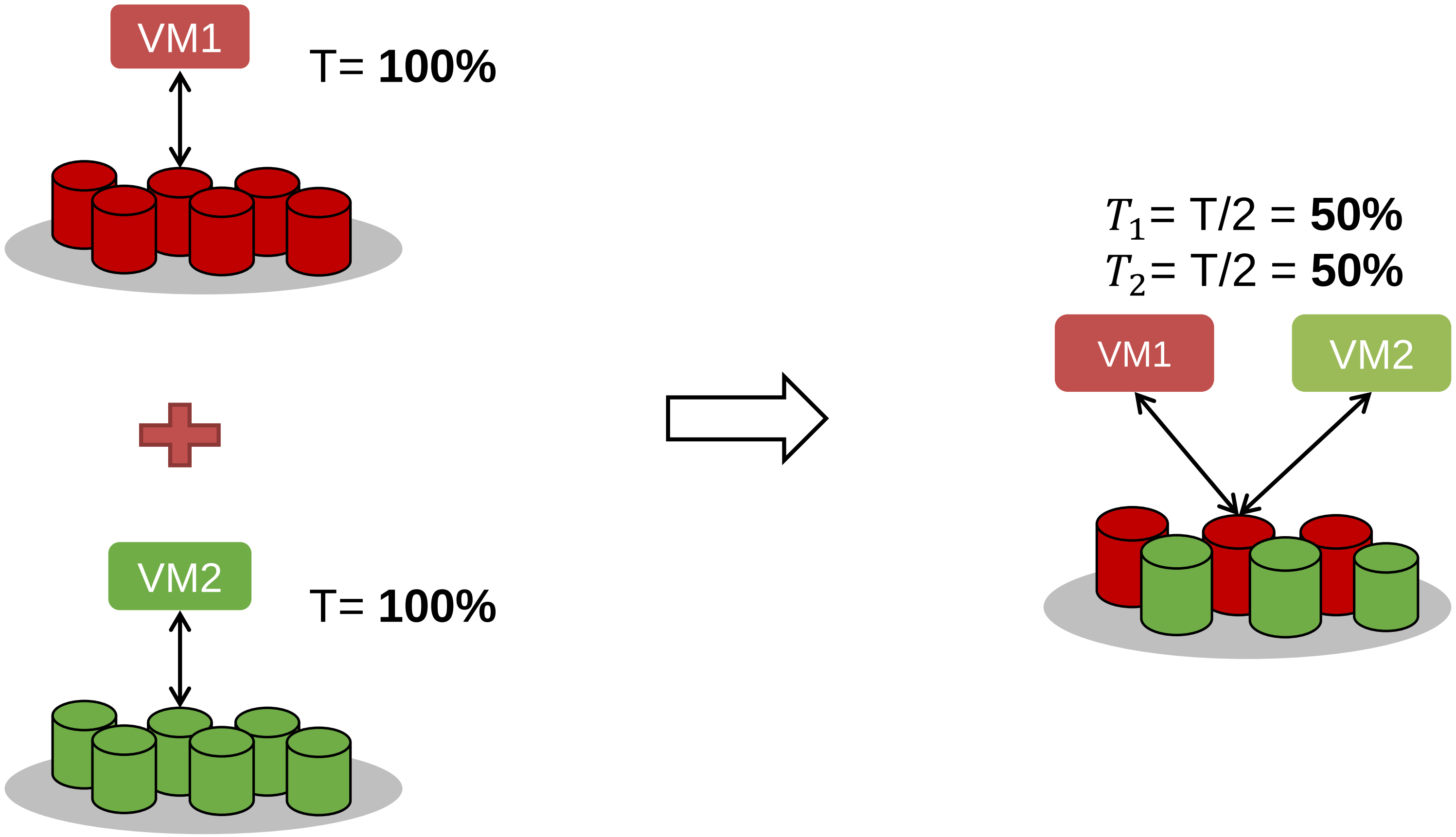
A Fair I/O Resource Scheduling Framework



Fair Resource Allocation and Scheduling

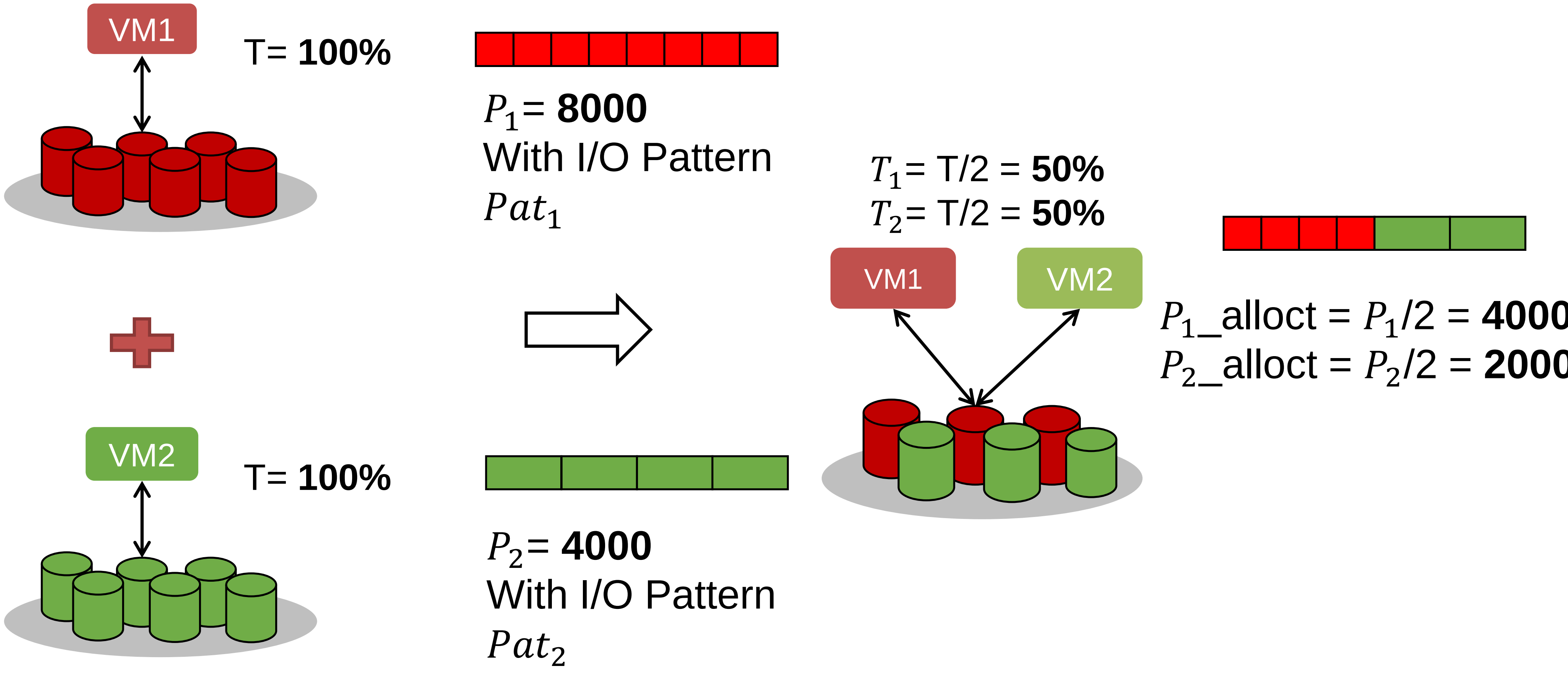


Service Time Based I/O Resource Allocation



In practice, it is **NOT** possible to know the processing time of each I/O in advance

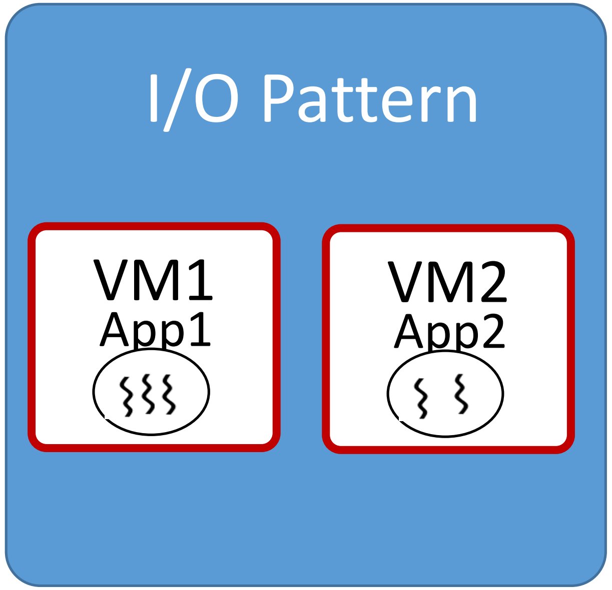
Service Time Based I/O Resource Allocation

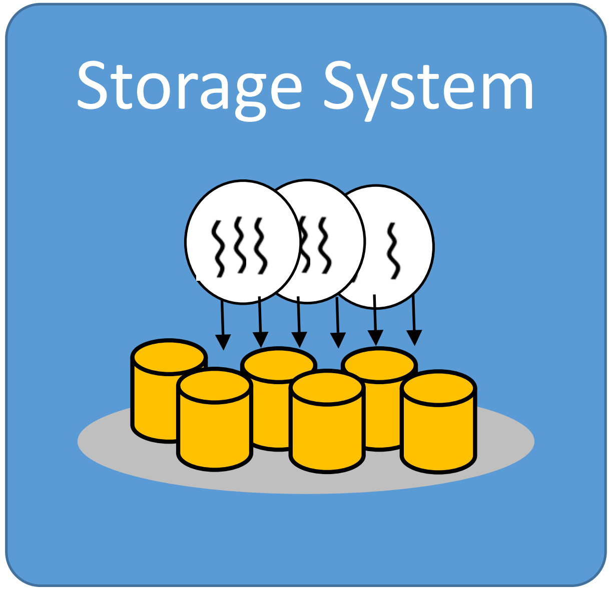


Saturation performance – the maximal I/O throughput that a VM could receive *in isolation* with a certain I/O pattern

How to Calculate Saturation Performance?

Saturation performance depends on two factors





Online Sampling

Offline Profiling

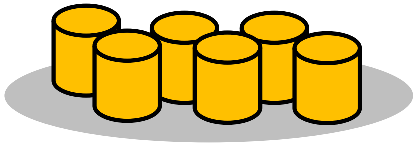
Approximating Saturation Performance

VM1

Sequential Read * $a\%$
Sequential Write * $b\%$
Random Read * $c\%$
Random Write * $d\%$
… …

With IO Pattern Pat_1

Online Sampling

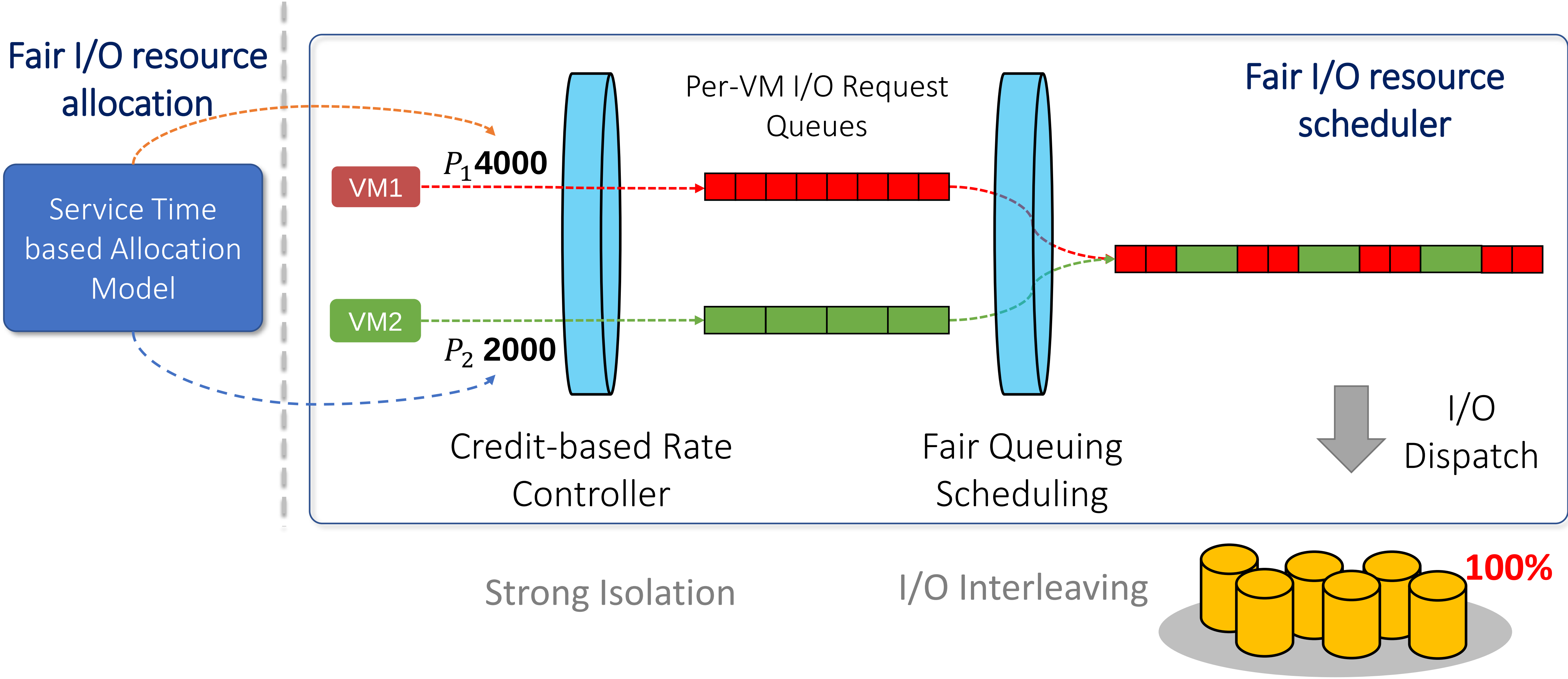


Offline Profiling

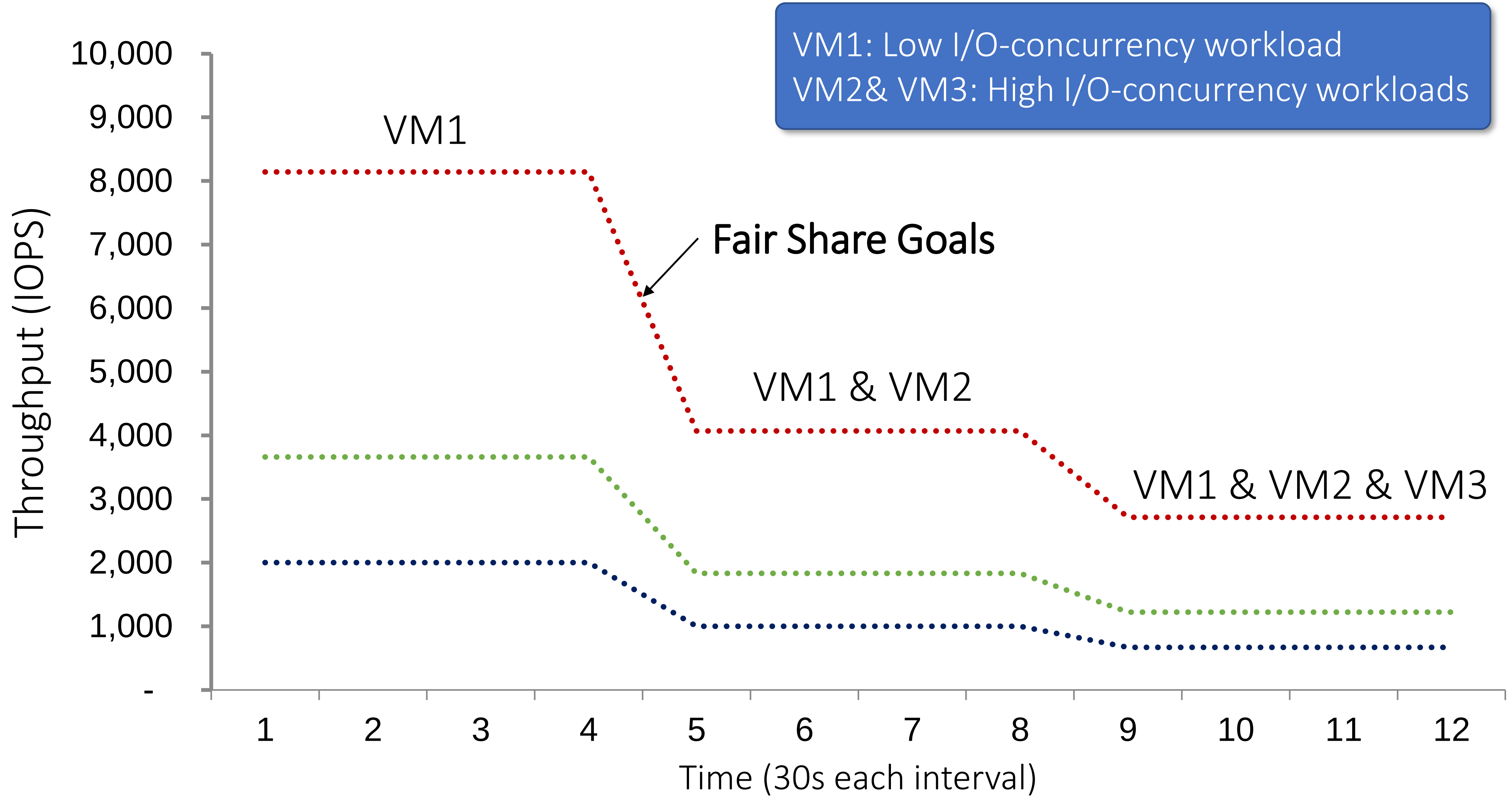
IO Pattern	Saturation Performance
Pat_1	8000
Pat_2	3000
Pat_3	5000
Pat_4	3000
Pat_5	1000
…	…

P_1= **8000**
With IO Pattern Pat_1

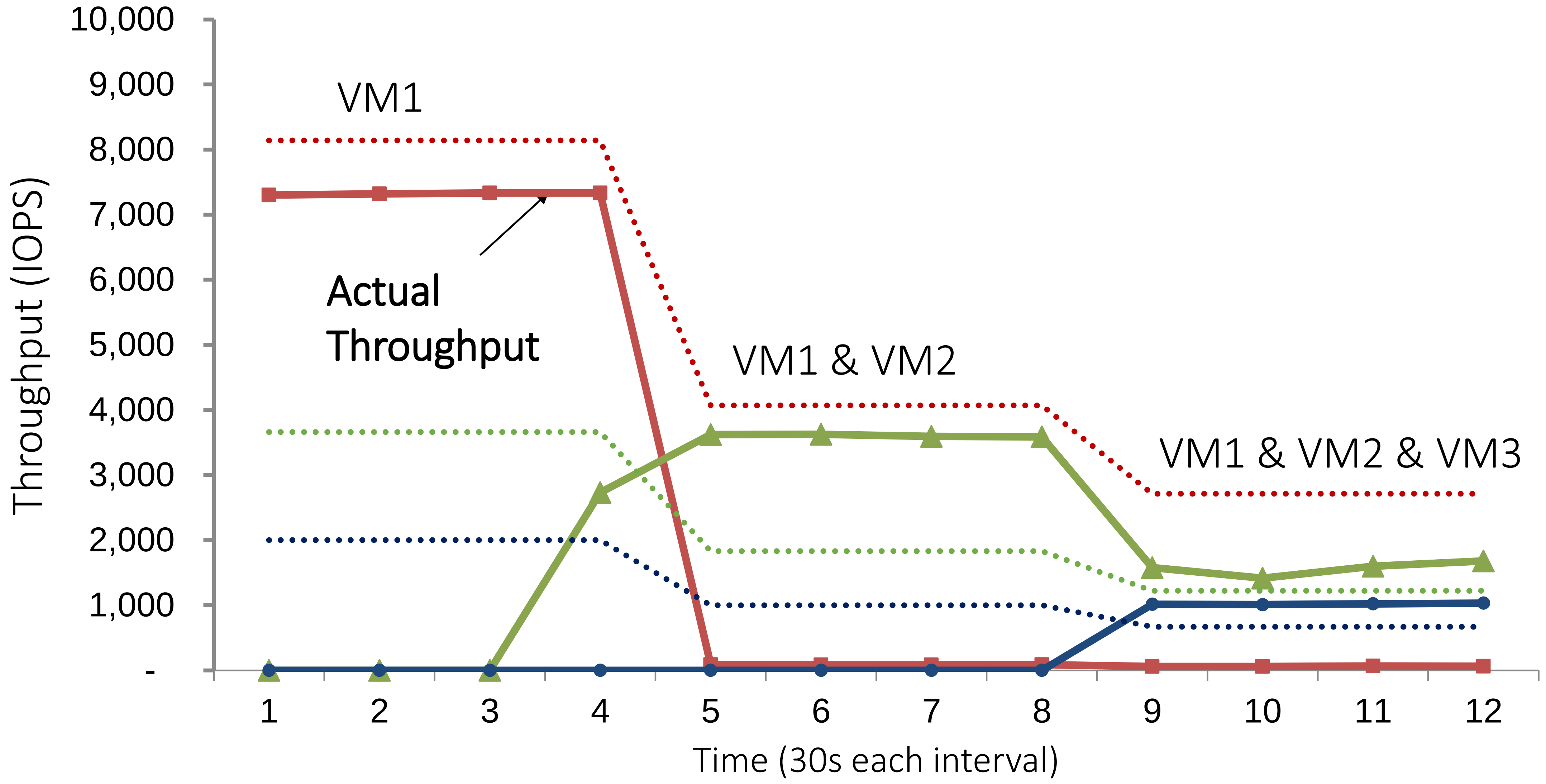
vFair Scheduling Framework



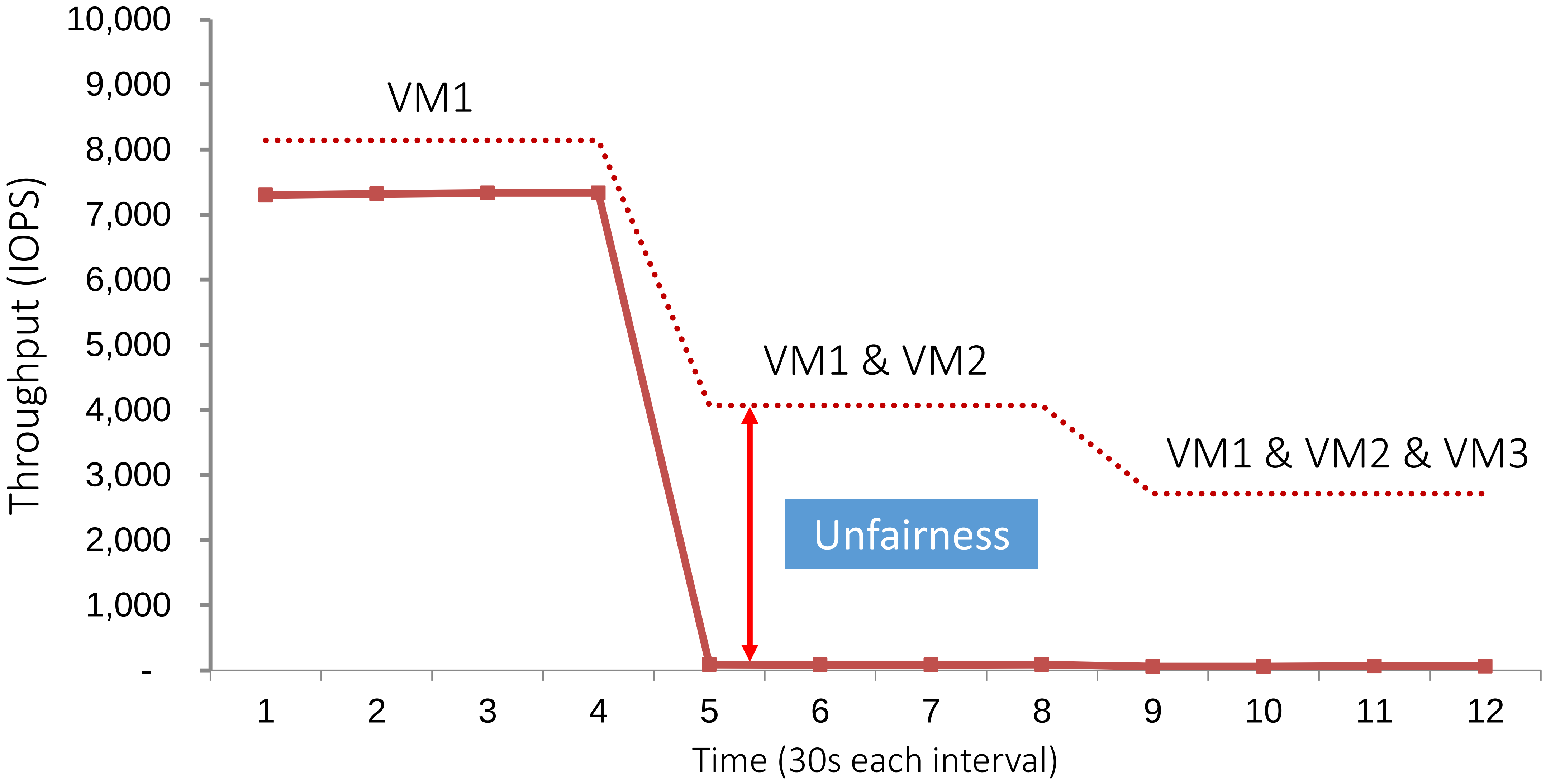
Fairness Evaluation



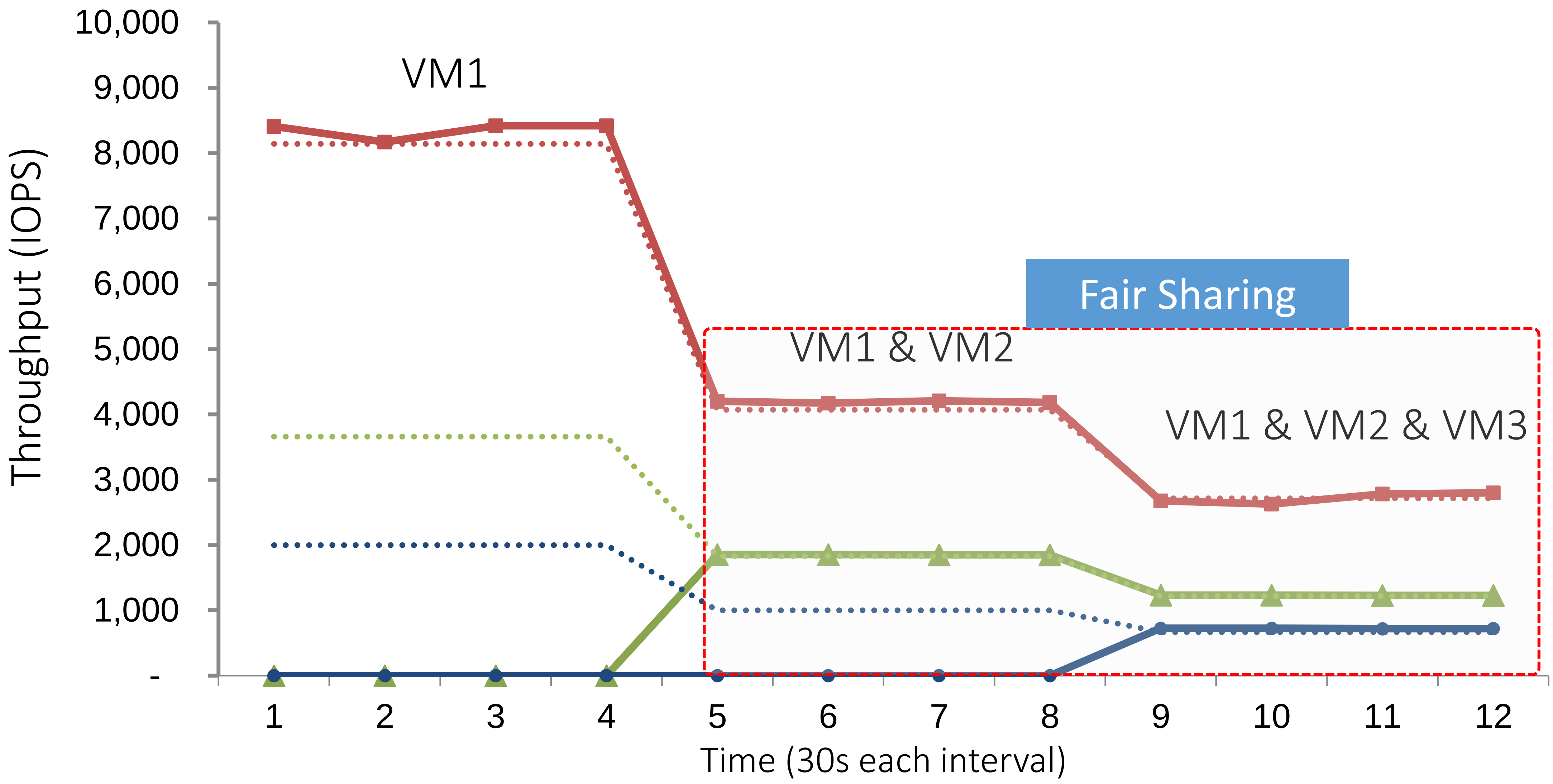
Fairness Evaluation – CFQ (Linux)



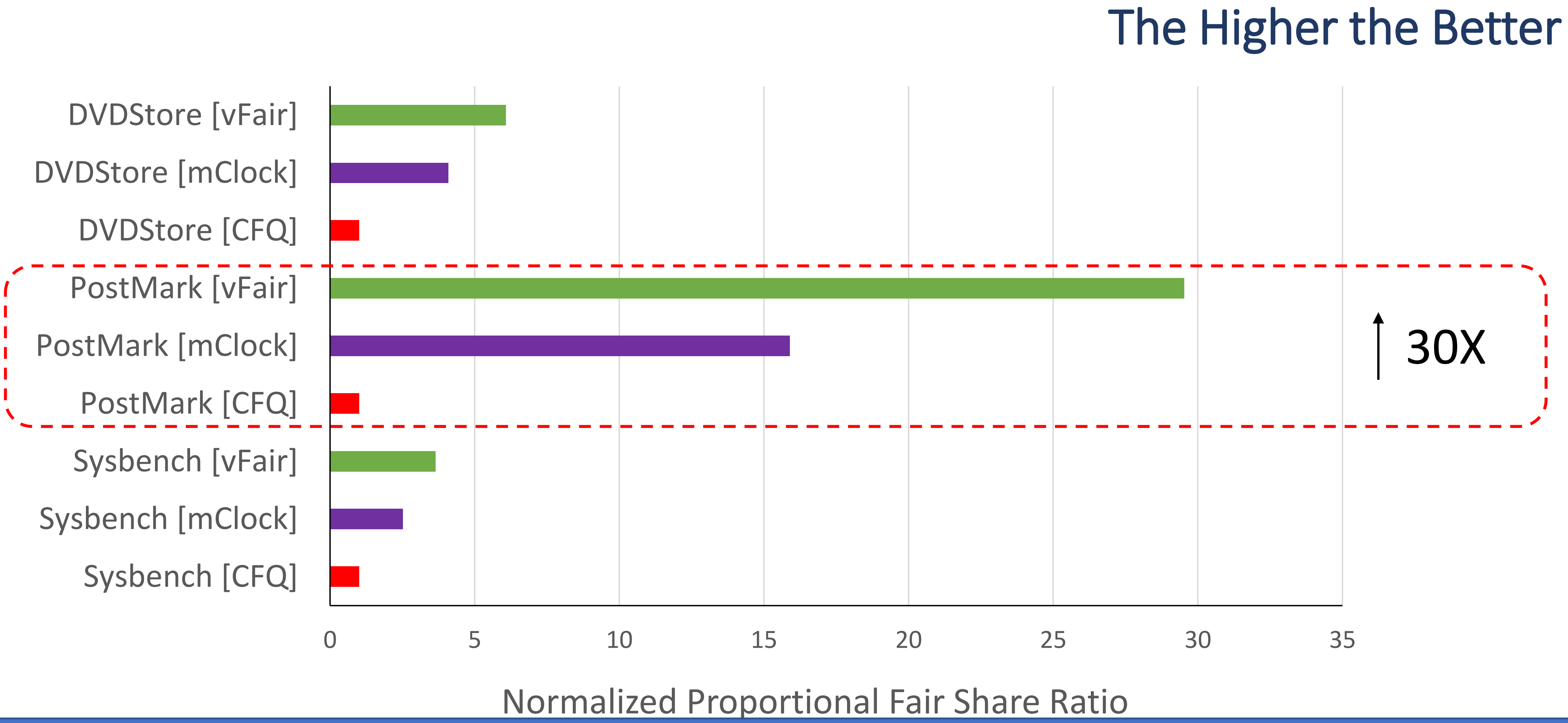
Fairness Evaluation – CFQ (Linux)



Fairness Evaluation – vFair

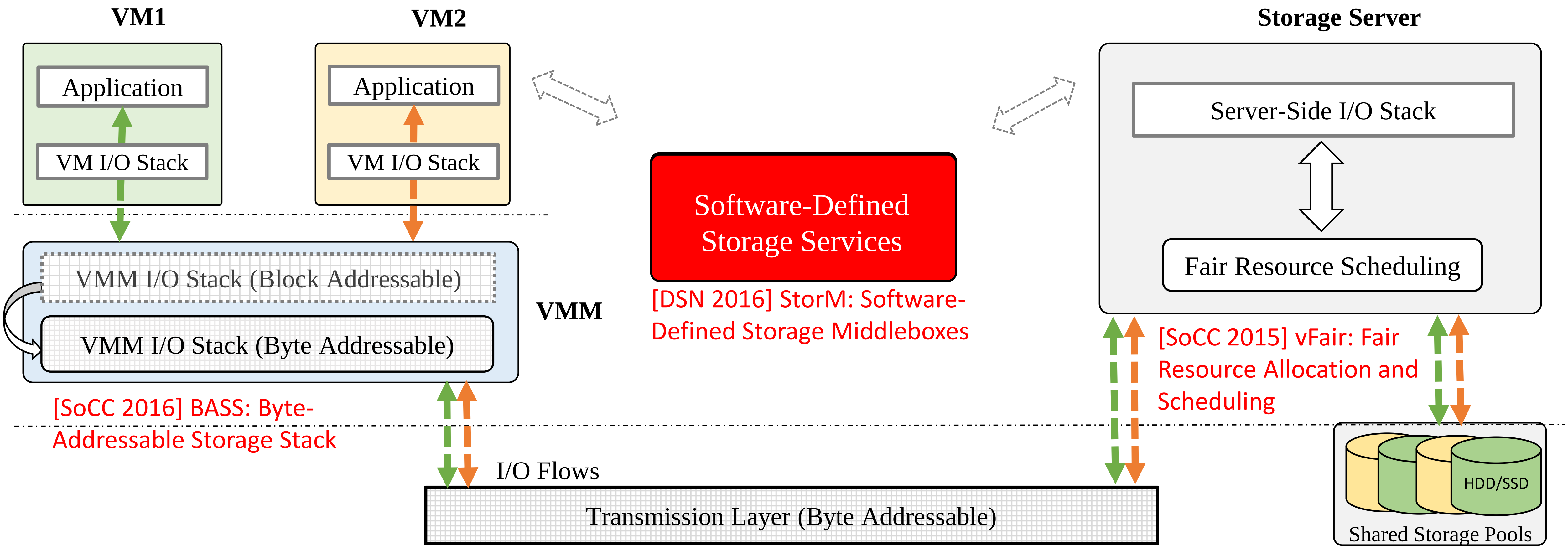


Application Workloads

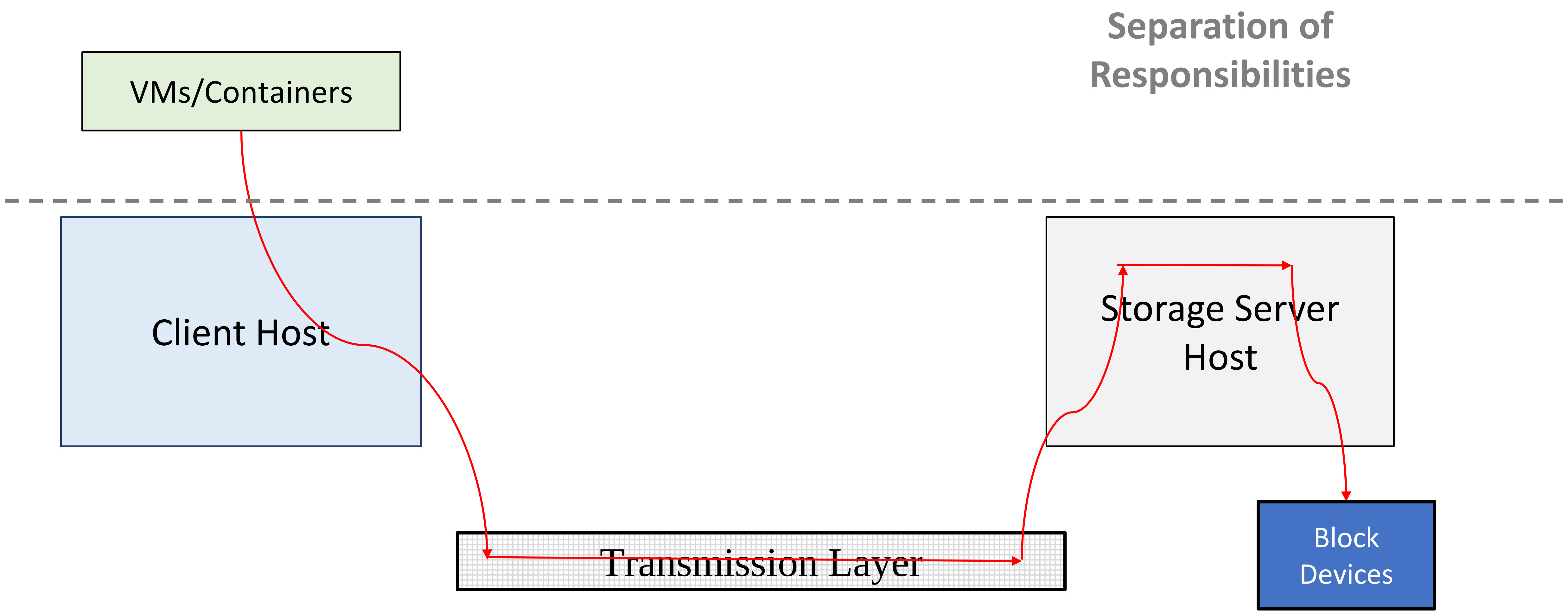


vFair increases the performance of the Low I/O-concurrency applications, e.g., **30X** for PostMark

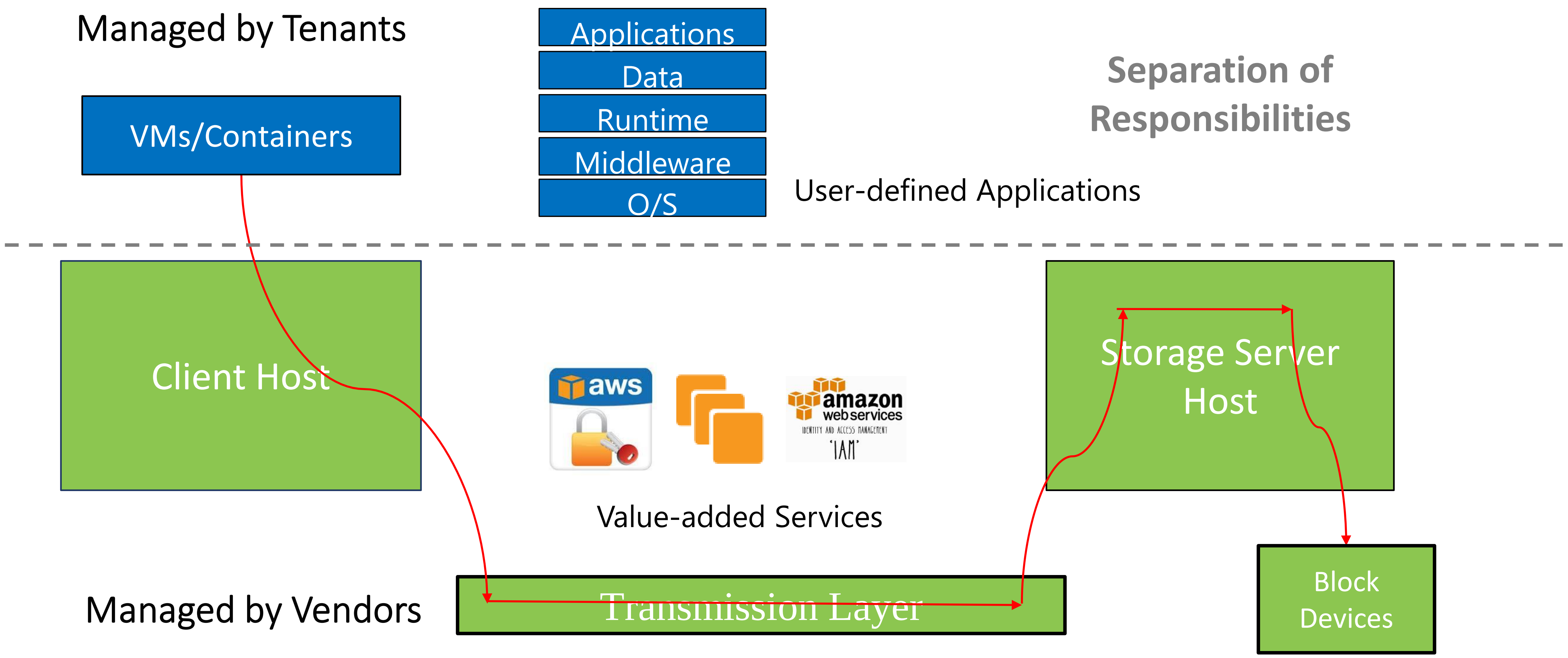
Research Overview: Cloud Storage Systems



Infrastructure as a Service (IaaS)



Infrastructure as a Service (IaaS)

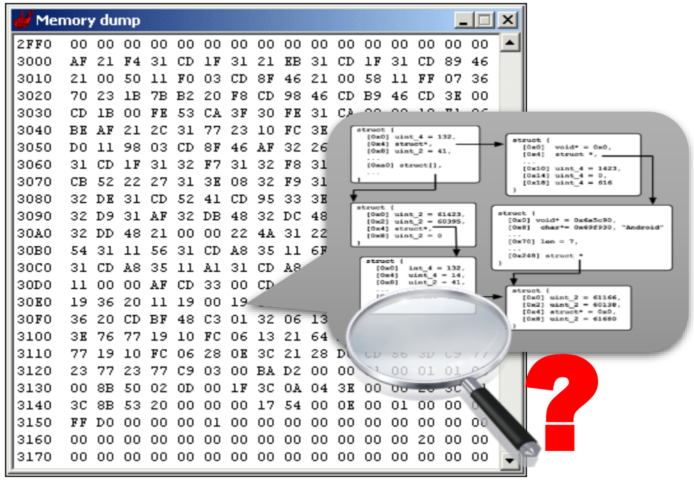


Problems of Today's IaaS

Provider-controlled service model cannot b… …eet individual tenants' needs

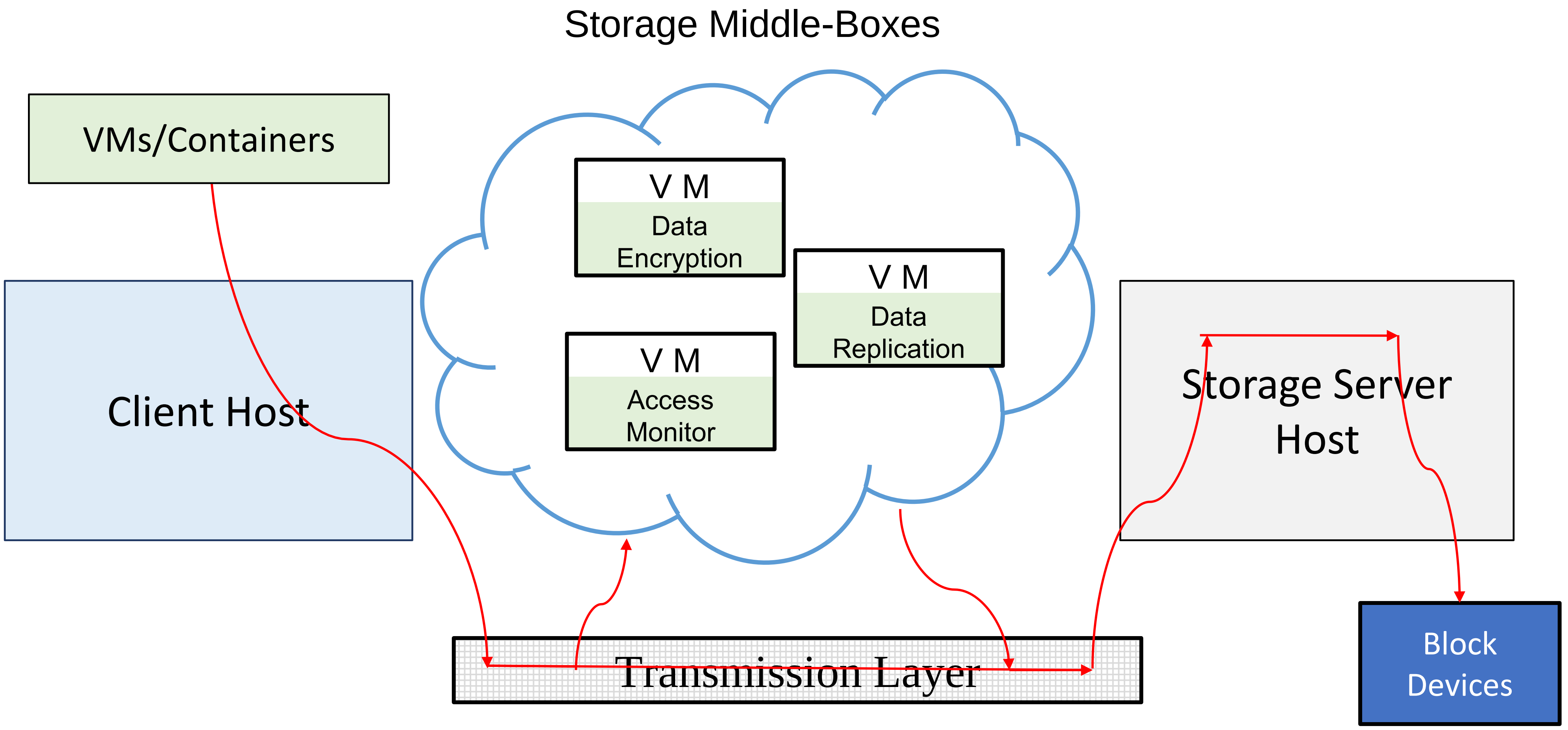
Proprietary Algorithms

Inter-Cloud Replication

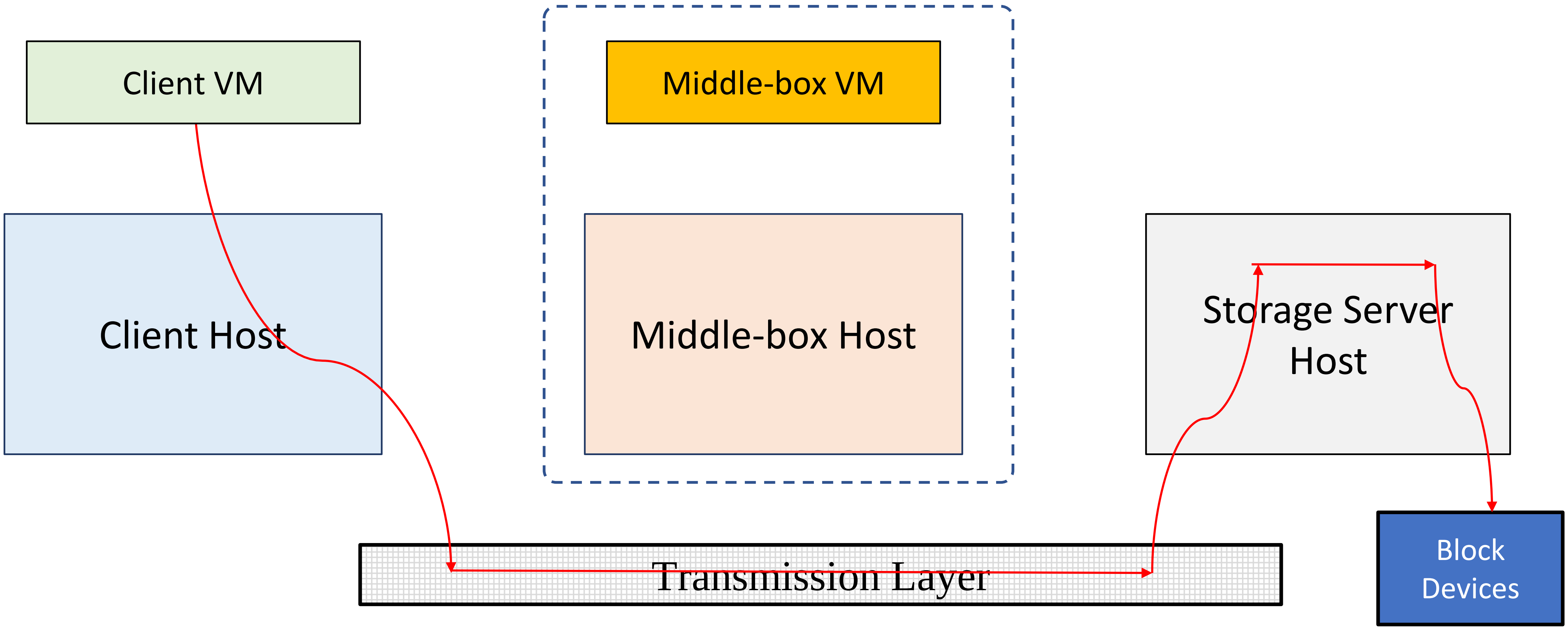


Sensitive File Monitor

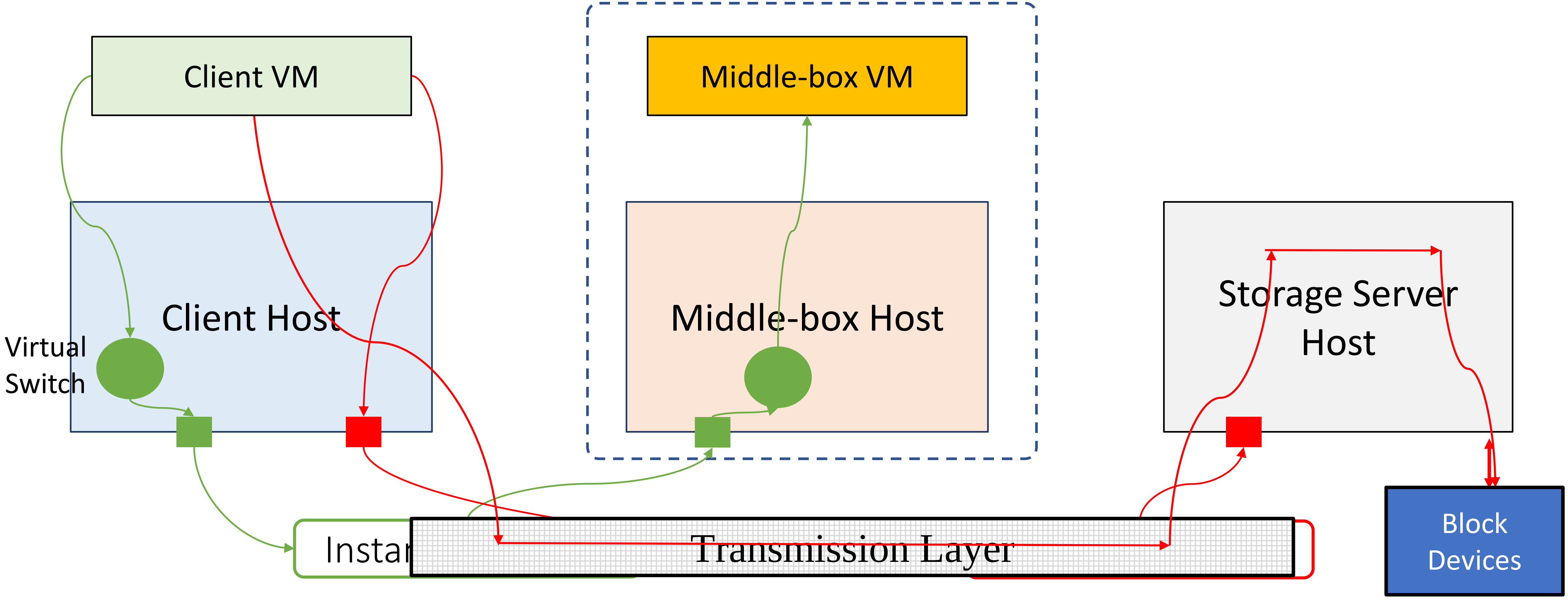
Storage Middle-Box Platform Overview



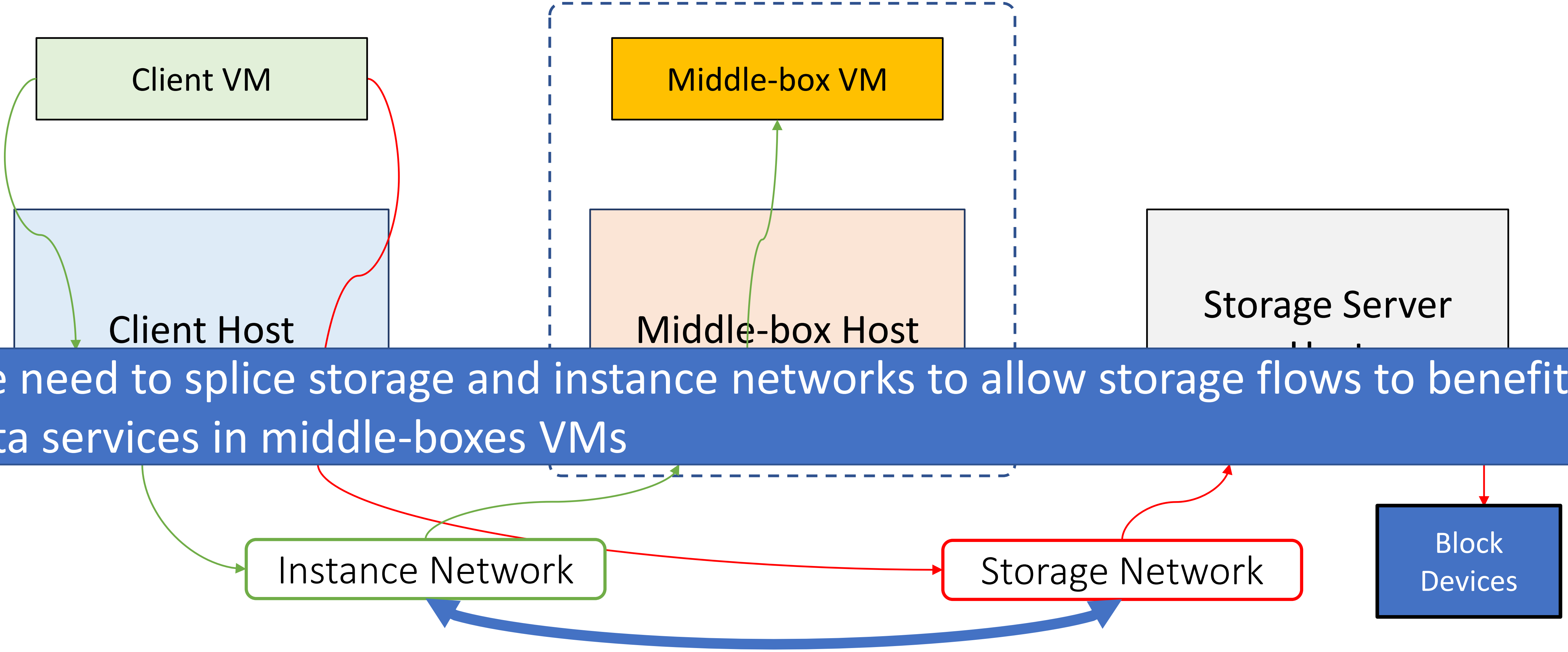
How to Enable the Middle-Box Platform?



Network Architecture in Cloud

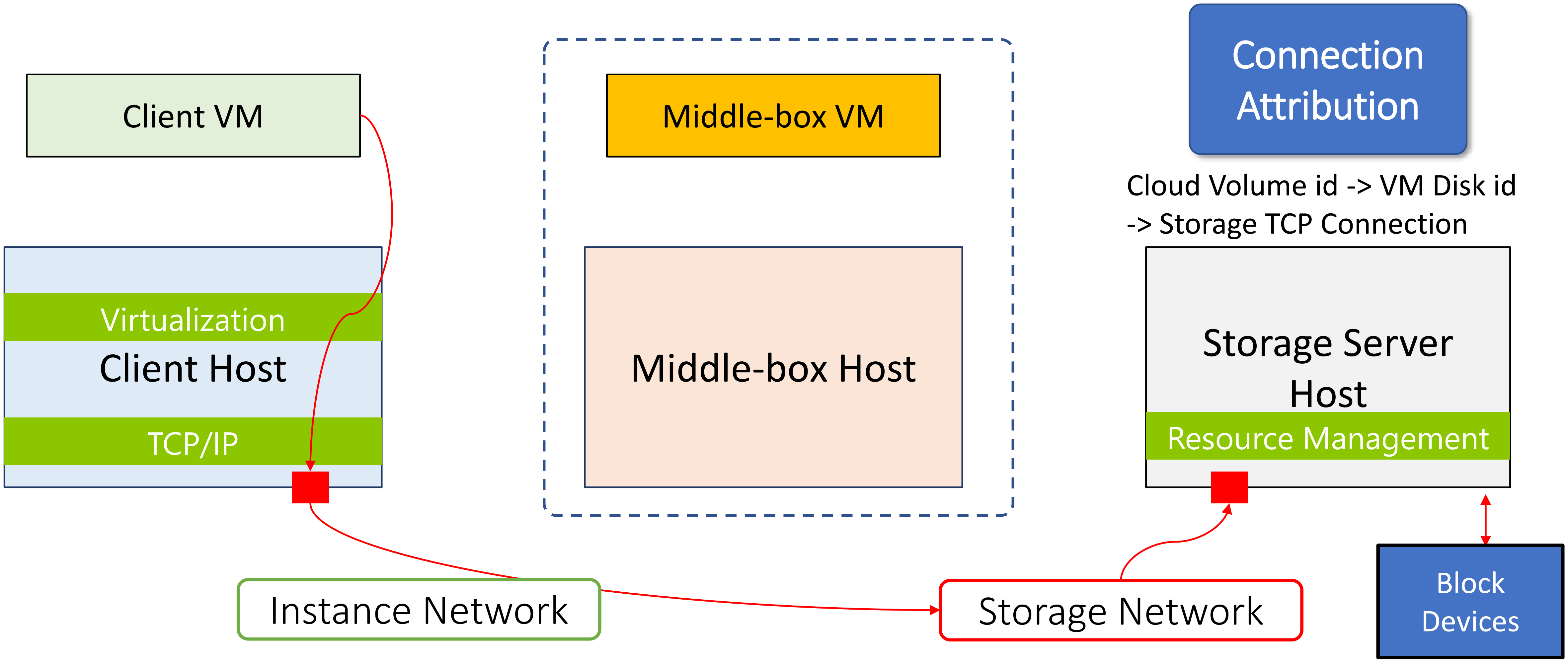


Challenge 1: Splicing Storage and Instance Networks

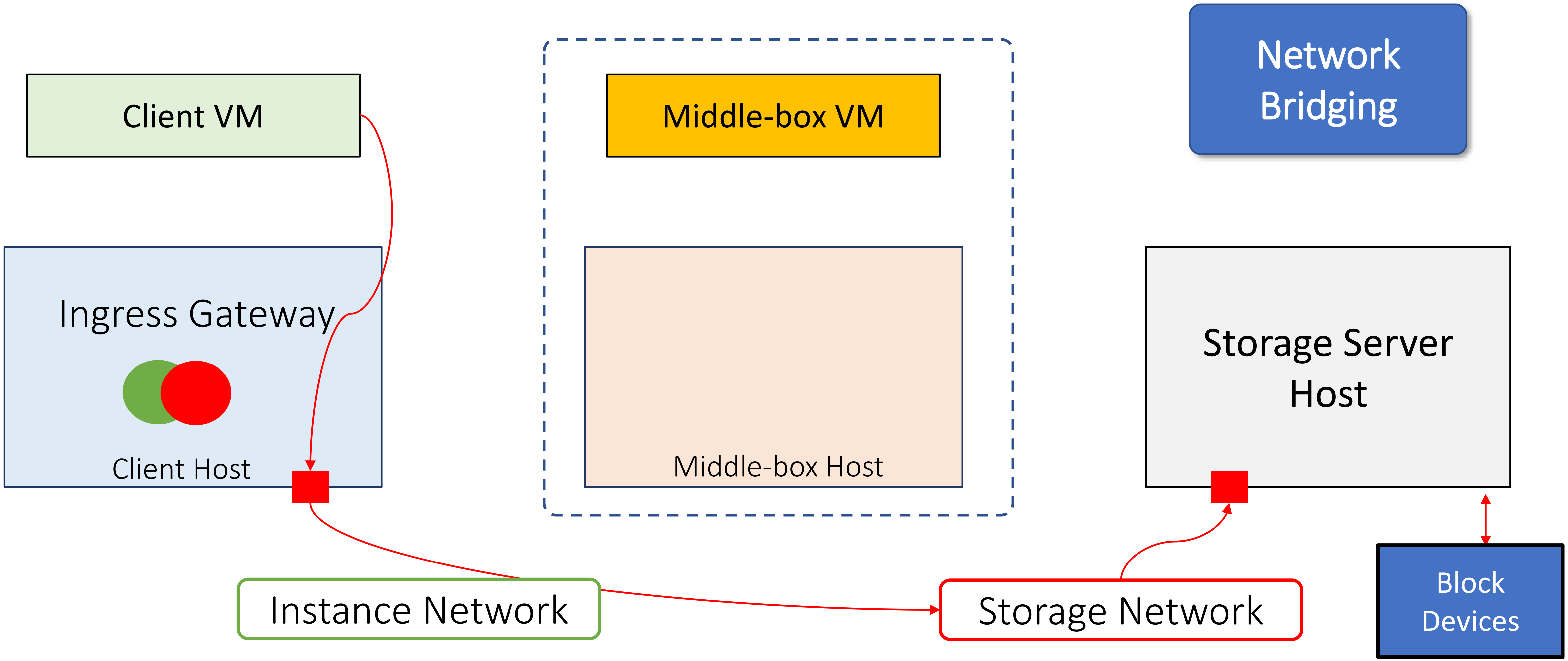


We need to splice storage and instance networks to allow storage flows to benefits data services in middle-boxes VMs

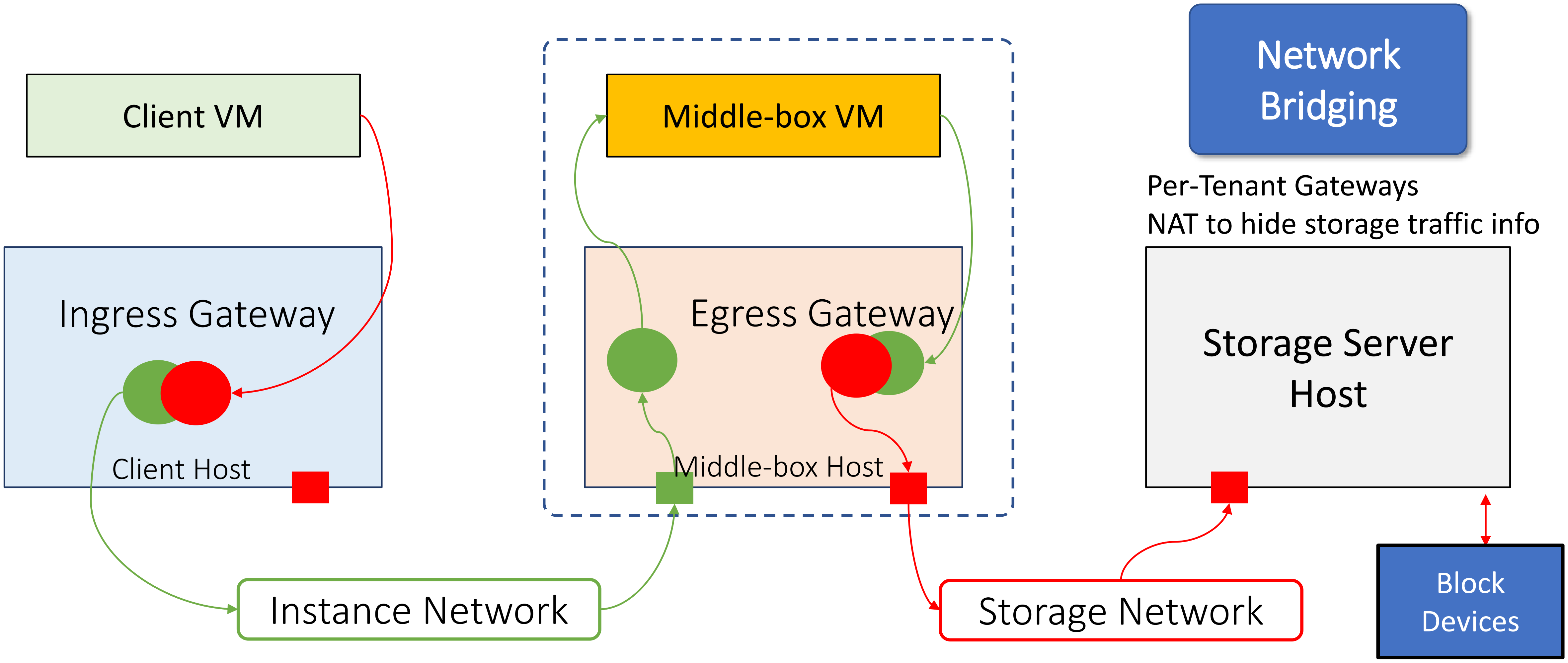
Step 1: Identifying Storage Connection



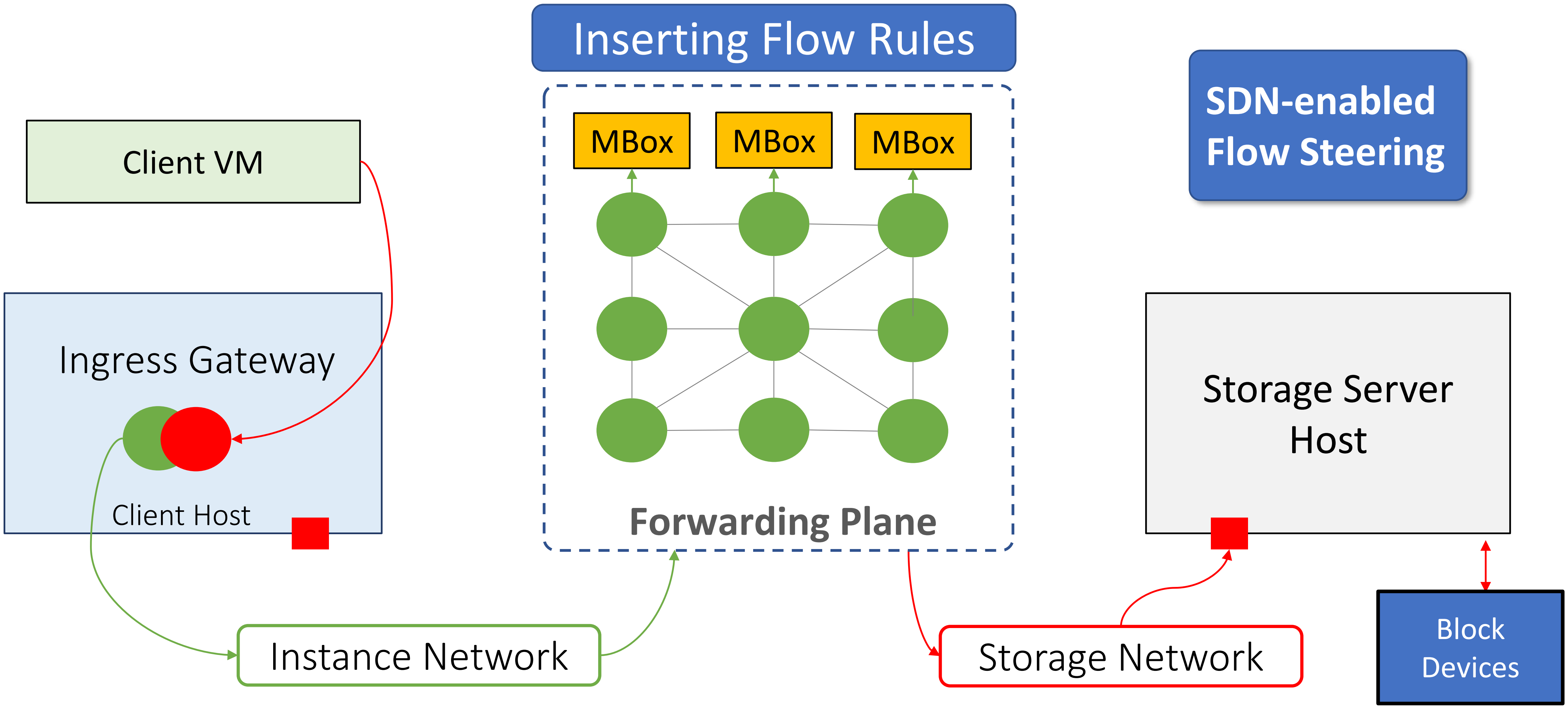
Step 2: Bridging Two Networks



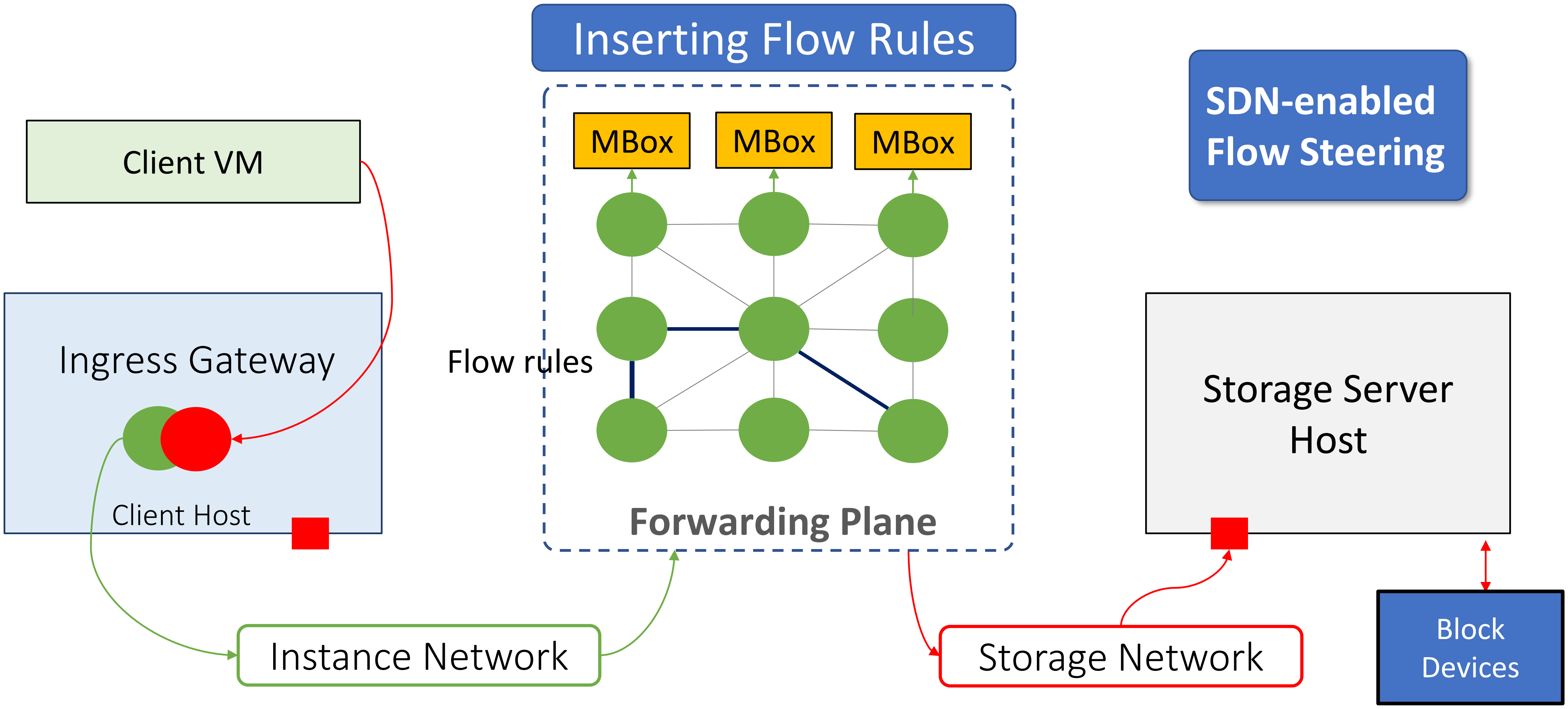
Step 2: Bridging Two Networks



Step 3: Steering Flows via Middle-Boxes



Step 3: Steering Flows via Middle-Boxes

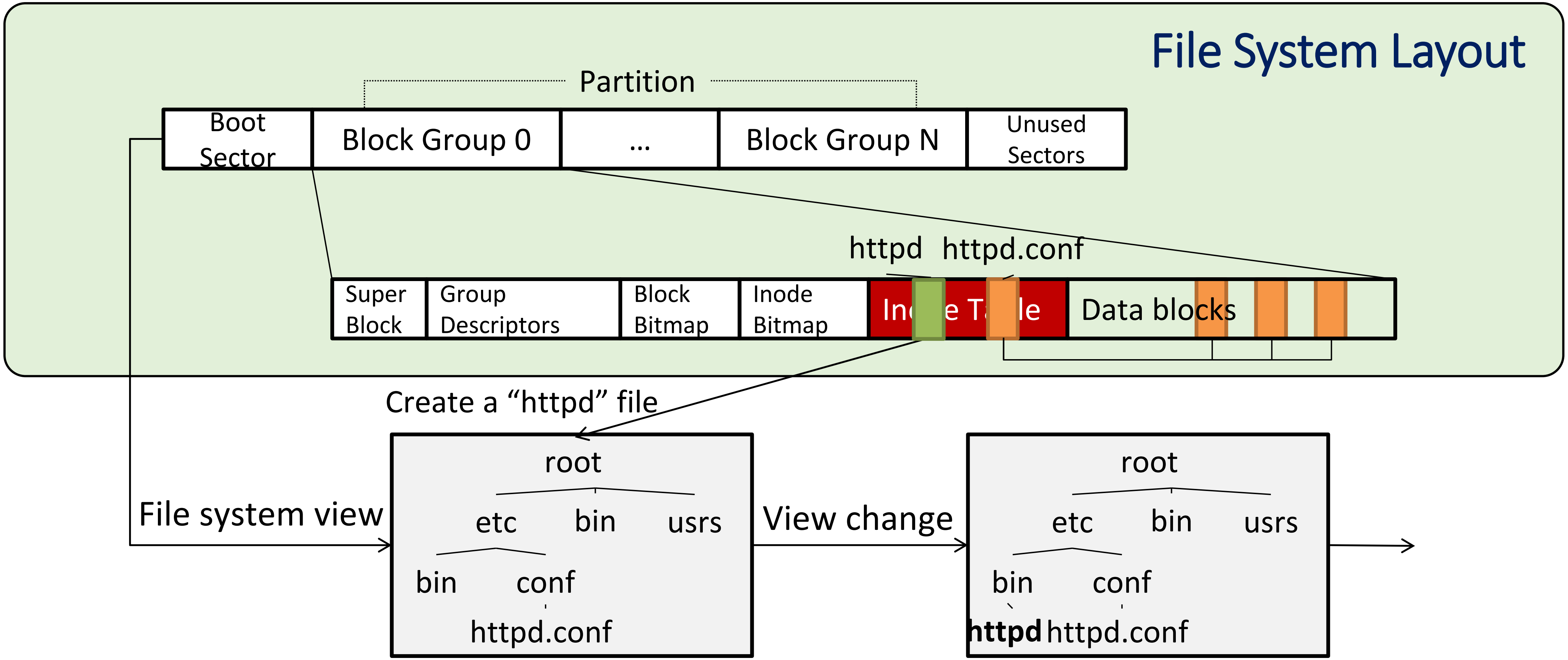


Challenge 2: Semantic Gap

Storage services usually operate at a file and directory granularity

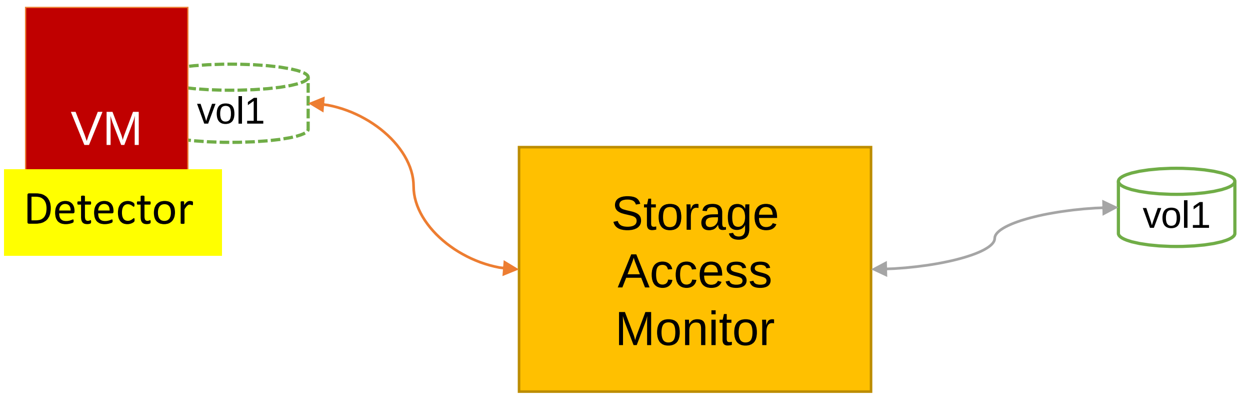
However, storage network packets only carry low-level storage system information

Semantics Reconstruction



StorM reconstructs high-level file structures on-the-fly using metadata accesses

Case Study: Storage Access Monitor



File Access ID	Operation	File	Size
1	write	/mnt/box/name1/1.img	4096
2	read	/mnt/box/name9/7.img	4096
3	read	/mnt/box/name4/3.img	4096
4	read	/mnt/box/name7/6.img	4096
5	read	/mnt/box/name6/1.img	4096
6	read	/mnt/box/name8/7.img	4096
7	read	/mnt/box/name3/8.img	4096
8	write	/mnt/box/name0/5.img	4096
9	read	/mnt/box/name9/9.img	4096
10	write	/mnt/box/name7/8.img	4096

Original file operations

I/O Access ID	Operation	File	size	Comment
1	read	/mnt/box/.	4096	Read root directory block
2 ~ 34	read	META: inode_group_90	4096	Read group 90 inode tables
35	read	/mnt/box/name1/.	4096	Read name1 directory inode
36	read	META: bitmap	4096	Read bitmap of group 18
37	read	META: bitmap	4096	Read bitmap of group 1
38 ~ 70	read	META: inode_group_106	4096	Read group 106 inode tables
71	read	/mnt/box/name9/.	4096	Read name9 directory inode
72	read	/mnt/box/name9/7.img	16384	Read file 7.img
73 ~ 105	read	META: inode_group_94	4096	Read group 94 inode tables
106	read	/mnt/box/name4/.	4096	Read name4 directory inode
107	read	/mnt/box/name4/3.img	16384	Read file 3.img
281	read	/mnt/box/name0/.	4096	Read name0 directory inode
282	read	/mnt/box/name0/5.img	16384	Read file 5.img
283	read	/mnt/box/name9/9.img	16384	Read file 9.img
284	read	/mnt/box/name7/8.img	4096	Read file 8.img
285	write	META: bitmap	4096	Update group 0 bitmap
286	write	META: bitmap	4096	Update group 0 bitmap
287	write	/mnt/box/name1/1.img	32768	Write data to file 1.img
288	write	/mnt/box/name0/5.img	4096	Write data to file 5.img
289	write	META: bitmap	4096	Update group 18 bitmap
290	write	META: inode_group_18	4096	Update group 18 inode table
291	write	/mnt/box/name7/8.img	4096	Write data to 8.img
292	write	/mnt/box/name7/8.img	57344	Write data to 8.img
293	write	/mnt/box/name0/5.img	57344	Write data to 5.img
294	write	/mnt/box/name1/1.img	4096	Write data to file 1.img
295	write	/mnt/box/name1/1.img	24576	Write data to file 1.img
296	write	inode_group_90	4096	Update group 90 inode table
297	write	inode_group_152	4096	Update group 152 inode table

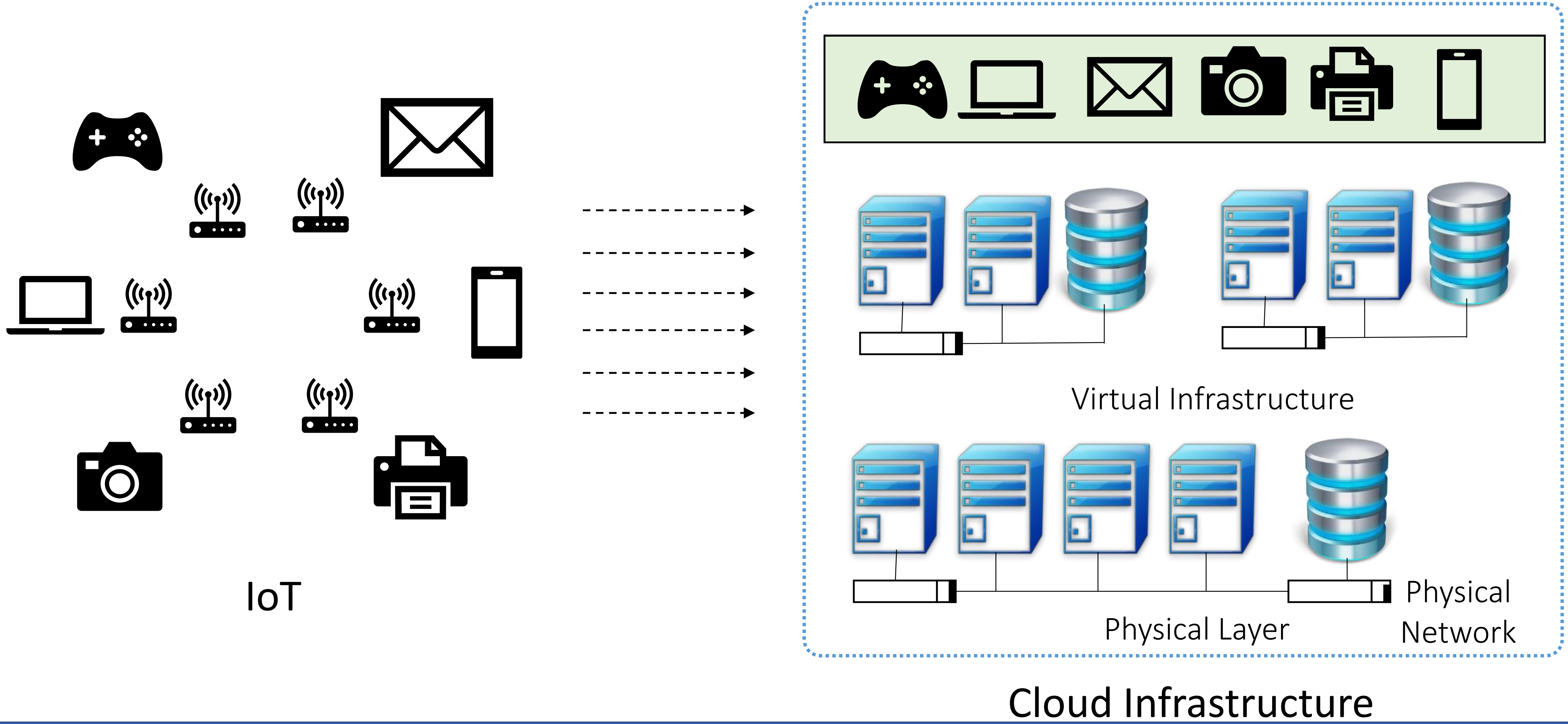
Reconstructed accesses

Case Study: Storage Access Monitor

Real-world Malware Study: A Linux backdoor Trojan detected by Kaspersky in 2015

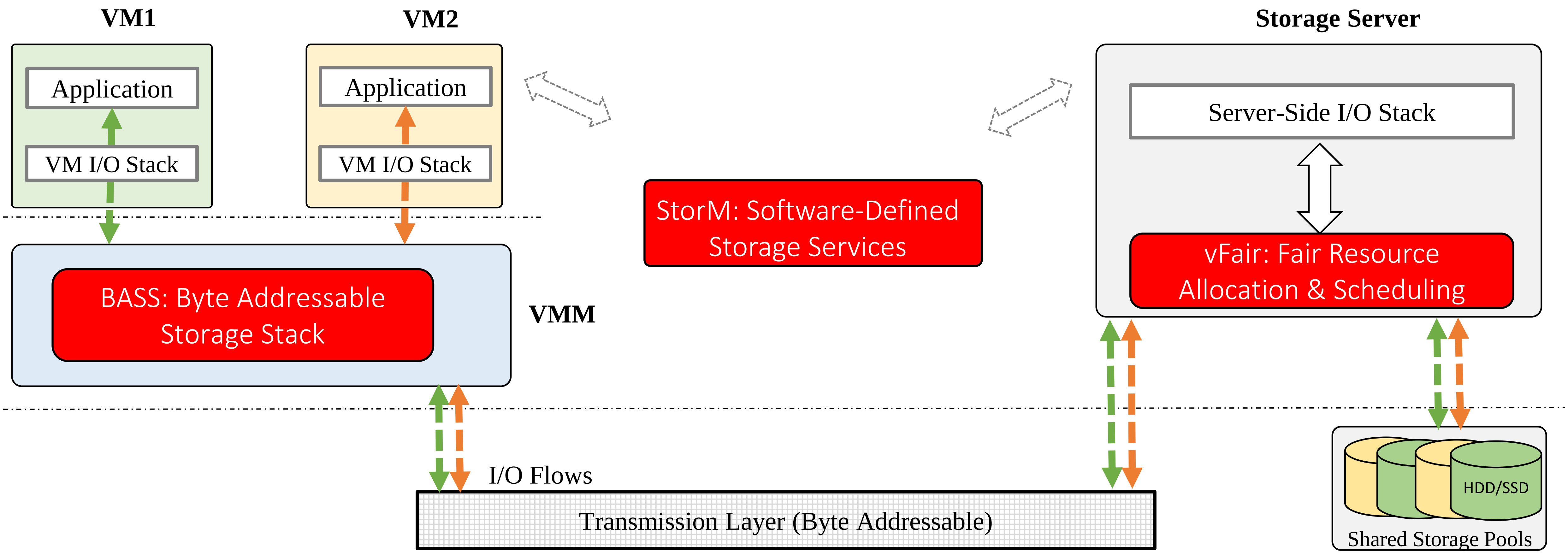
Step 1	cp “#!/bin/bash\n <path_to_malware>" /etc/init.d/DbSecuritySpt
Step 2	ln -s /etc/init.d/DbSecuritySpt /etc/rc[1-5].d/S97DbSecuritySpt
Step 3	cp <path_to_malware> /usr/bin/bsd-port/getty
Step 4	cp “#!/bin/bash\n/usr/bin/bsd-port/getty” /etc/init.d/selinux
Step 5	ln -s /etc/init.d/selinux /etc/rc[1-5].d/S99selinux
Step 6	cp <path_to_malware> /bin/netstat cp <path_to_malware> /usr/bin/lsof cp <path_to_malware> /bin/ps cp <path_to_malware> /bin/ss

Vision: “Cloud of Things”



Cloud and IoT are the complementary aspects of the future Internet, creating Cloud of Things

Conclusions



Future Work: System supports in clouds in response to emerging cloud applications and high-performance hardware