Exploring Computational Storage for mixed HPC Simulation and AI/ML Workloads at LANL

LA-UR-xxxxxxx

Transforming Weapons Performance Calculation
and other Mission Workloads
via
Efficiency Mission-Centric Computing Consortium

EMC3

Gary Grider

08/2021

Los Alamos
NATIONAL LABORATORY
EST. 1943

Nine Decades of Production Weapons Computing to Keep the Nation Safe

Maniac

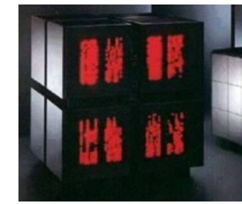
IBM Stretch

CDC

Cray 1

Cray X/Y

CM-2

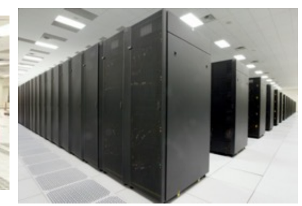


CM-5

SGI Blue Mountain

DEC/HP Q

IBM Cell Roadrunner



Cray XE Cielo

Cray Intel KNL Trinity

Ising DWave

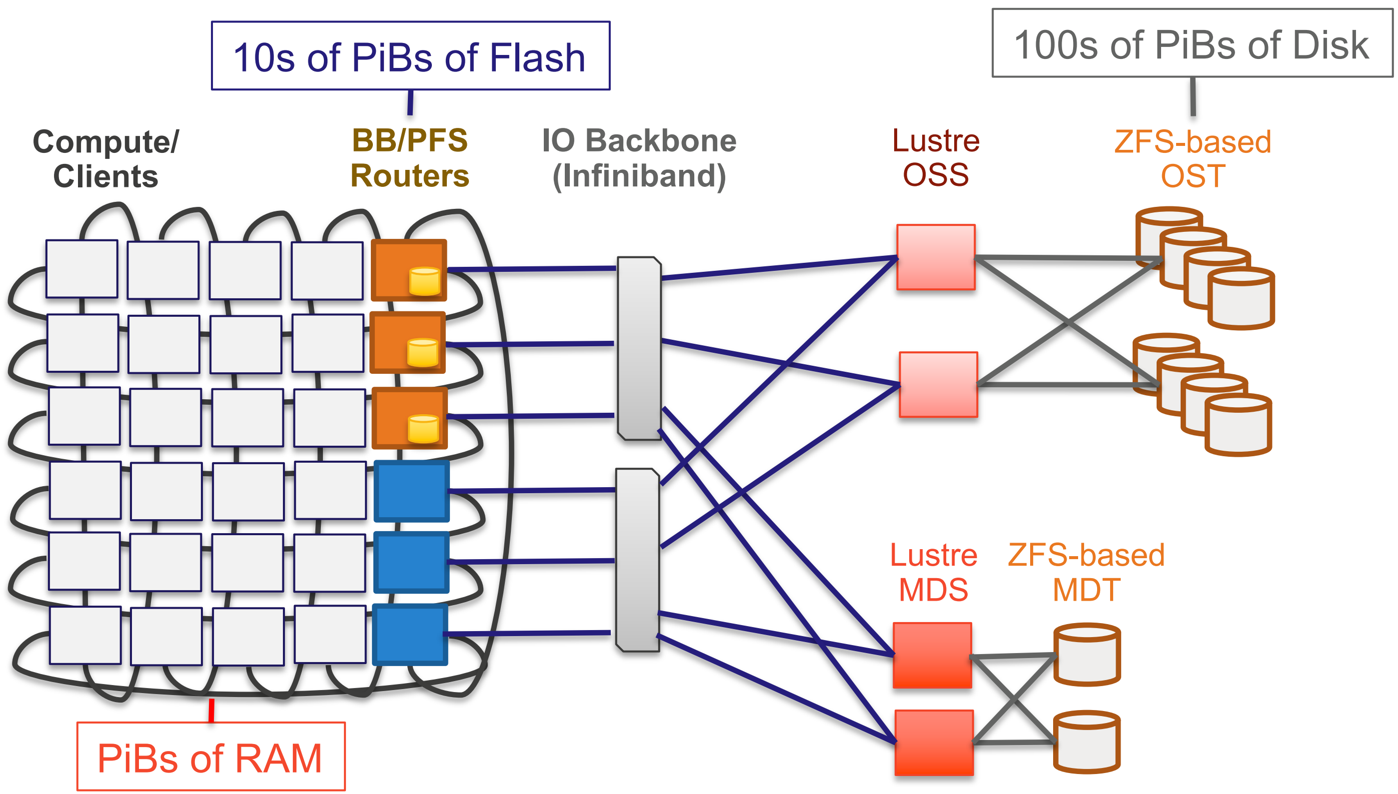
Cross Roads

CROSSROADS
An APEX Collaboration
intel

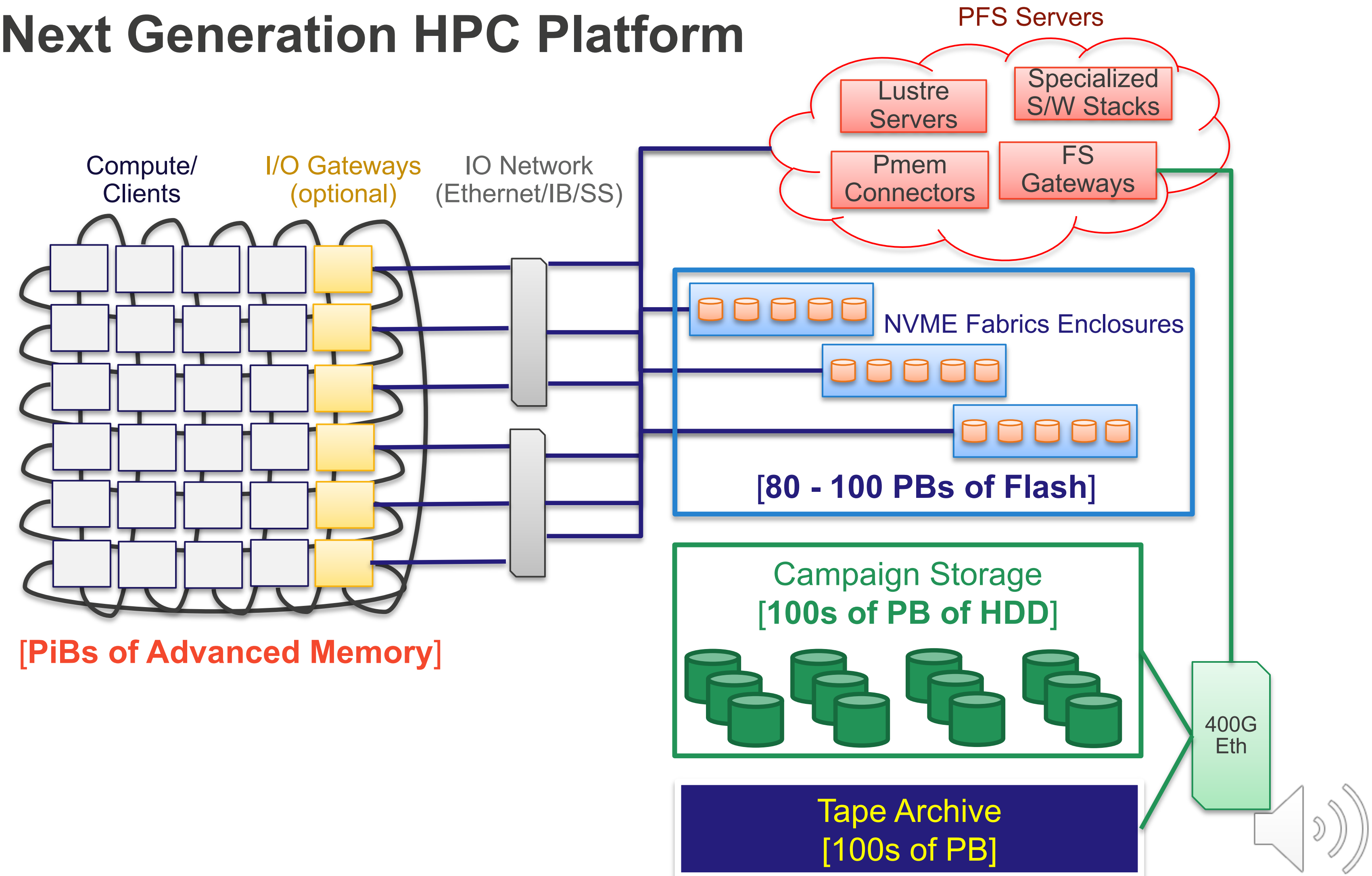
NGP-1

NGP-1
NVIDIA

Current HPC Platforms



A Next Generation HPC Platform



Why computational storage?

LANL mission ~= Weapons Science / $

Lets shrink the $

Economics for leveraging modern storage device trajectory

Considerations

- Cap/bw/iops of devices – flash bw and iops per capacity is orders of magnitude different than disk
- Servers – poor memory bw (just reading from network and writing to device can use ½ memory bw leaving little for erasure/encoding/compression/indexing/etc)
- Kernels/thick IO stacks in compute node client and server make getting IOPS extremely hard
- Network speeds/messaging rates quite astounding
- LANL simulation workload is not friendly with locality, so on compute node or near compute node storage is likely to lead to imbalance/stranding/etc. This is not same for other national labs!
- Leveraging industry trains, Flash, NVME, NVMEoF, RDMA, Smart Nic, Computational Storage, custom SOC etc.

Go after repetitive data agnostic use cases (within byte streams/file systems/etc.)

- Fixed functions like compression, erasure, encoding, dedup
- Fixed functions allow for customized hardware/software/pipelines and take advantage of locality (where the data is (computational storage) (where the data will be (computation in networks))

Find ways of reducing stack thickness to enable extracting performance (IOPS in this case as BW can be extracted with existing thick stacks)

- User space direct access from compute node to storage device (eliminate compute node kernel and server stack)

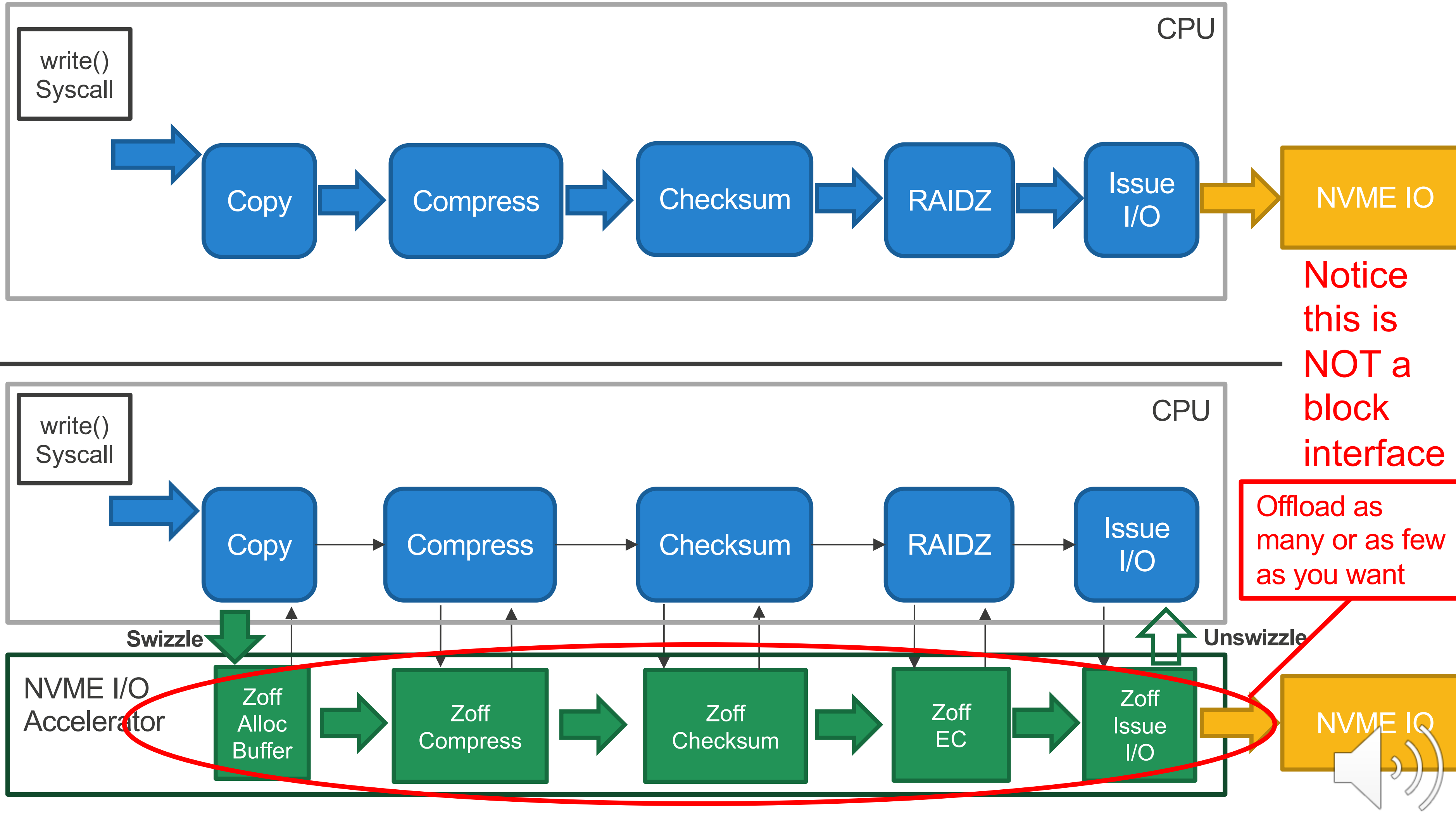
File System Services Offload

Data Agnostic fixed function offloads

- Non-obvious requirements
  - Require transparent data placement for disaster recovery
  - Require parallel file system support
  - Not just Read – in a mixed simulation/AI/ML site - Write-dominant workload for simulation (defensive I/O: write once, read never) still are important
- Computational Storage Benefits/Opportunities
  - Increase compression rates from 1.06:1 -> 1.3:1 for scientific data
  - Enable expensive coding/decoding to protect against correlated failures
  - Achieve higher per-server and per-device bandwidths
  - Lower server costs and quantities

- Overcome poor server memory BW imbalance
- Less expensive file oriented solution
- Use a commonly used HPC file system to leverage offload (ZFS)

Notional fixed function offloads in ZFS



Economic Model Using Measured/Specified Performance/Price for Offloading or Not (does offloading $add up)?

Assumptions, 1) massive parallel N to 1 jobs so getting huge win from placement is not really possible, 2) in/near node storage works well for 3DUQ/AI/ML but not great for full scale complex parallel, 3) min rqmnt is 80% of specitified memory dumped in specified seconds

change what is in yellow ONLY, check the edit on min capacity to get min bw, copy lines below entirely and change only the yellow settings on each line

SAMPLE ONLY

	TB	rawserw GB/s	rawserr GB/s	fswbw ee nc	fswbw ee lc	fswbw ee hc	thru-riop M/s	thru-wiop M/s	nvme slots	$1,000	fsrbw ee nc	fsrbw ee lc	fsrbw ee hc	type	
system memory TB	1000														
nvme device TB	7.68	3.2	3				0.6	0.2		1.88					
srv0 (no servers)	ns	0	0	0	0	0	0	0	0	0	0	0	0	0	no srvs, nvme in ebofs, stg functions live in compute nodes or ebofs
srv1 2-dual400Gbit no stg	rs	45	45	32	28	2	1.1	1.1	0	35	32	28	1	1	srv with no slots
srv2 dual400 Gbit	rs	45	45	18	14	1	0.6	0.6	12	17	18	14	1	2	srv with slots
srv3 dual400 Gbit	rs	45	45	32	28	1	1.1	1.1	24	35	32	28	1	3	srv with slots (alternative schenario)
srv4accel dual400Gbit no stg	as	45	45	32	28	28	1.1	1.1	0	25	32	28	32	4	accel server with no slots
srv5accel	as	45	45	32	28	28	4.4	4.4	24	27	32	28	32	5	accel server with slots
srv6ebofaccel	rs	45	45	32	28	28	4.4	4.4	0	14	32	28	32	6	accel in ebof server with NO slots
ebof0 (no ebofs)	ne	0	0	0	0	0	0	0	0	0	0	0	0	0	no ebofs, nvme in srvs,stg functions in srvs
ebof1 2-dual400 Gbit 8Miop	re	80	80	60	60	60	4.4	4.4	24	10	60	60	60	1	ebof with slots with no offload would really need to use this if you use servers or accel servers or **NO servers**
ebof2accel 2-dual400 Gbit 8Miop	ae	80	80	60	63	80	4.4	4.4	20	15	60	58	60	2	accel ebof with slots with offload you could use this with non accel servers ONLY!
													value above is poor uncompress on accelerators		
serial bw sec dump 80% mem	600														
sys cap type	0	1	2												
sys cap requirement memories	8	16	32		*1*	if 1 you have enough capacity to meet fswb min if 0 then need more mems									
	l	m	h												
comp type	0	1	2												
comp yealding .xx need	1	0.95	0.75												
	n	l	h												

iops background: intel test - so a hot dual socket or 28M iops from devices, running 6vm-servers per server got 13.5 M/iops or 1.15Mi ops/server through the network but the above was with 4 25-gbit nics so mutiply by 4 to get roughly what dualport 400 Gbit (it doesn't look linear) chelsio dual 100 Gbit into a serverr demonstrated 2.8M iops through the server so its not scaling perfectly - but a 400 Gbit pcie-4 maybe 4Miops ebof has 4x iops of server due to double adapters and no kernel

Many Scenarios Considered

goals	minwbw	maxwbw	maxrbw	maxriops
	0	1	2	3

server ebof combo info, srv=0 only makes sense with ebof=1 or 2 means NO SERVERS nvme in ebofs; srv 1,4 only makes sense with ebof=1 or 2 - nvme in ebofs; srv 2,3,5 only makes sense with ebof=0 nvme in server; if you want to add acceleration use srv=4 + ebof=1 (accel in srv nvme in ebof) -or- srv=5 + ebof=0 (accel and nvme in srv) -or- srv=1 + ebof=2 (srv with accel and nvme in ebof)

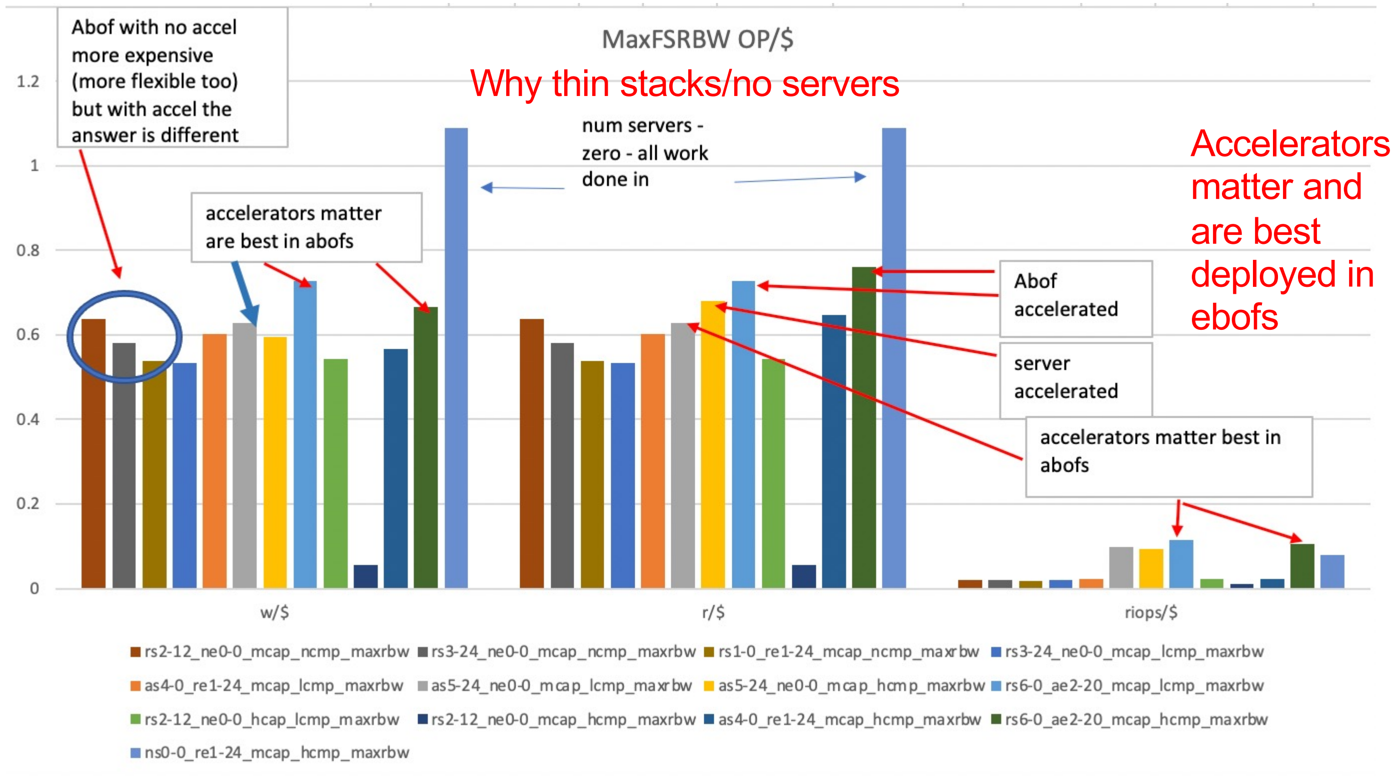
SAMPLE ONLY

scenario	srvtype	eboftype	syscaptype	comptype	goal	totactcap PB	numnvme 1000s	totsrv 100s	totebof 100s	fswbw TB/s	(-) fswbw TB/s	fsrbw TB/s	(-) fsrbw TB/s	riops GIOP/s	(-) riops GIOP/s	srv cost $M	ebof cost $M	nvme cost $M	total cost $M
rs2-24_ne0-0_mcap_ncmp_minwbw	2	0	1	0	0	15.00	7.50	3.13	0.00	9.38	-1.88	9.38	-13.13	1.25	-4.00	4.38	0.00	4.50	8.88
rs3-48_ne0-0_mcap_ncmp_minwbw	3	0	1	0	0	15.00	7.50	1.56	0.00	4.69	-6.56	4.69	-17.81	0.63	-4.63	2.19	0.00	4.50	6.69
rs1-0_re1-24_mcap_ncmp_minwbw	1	1	1	0	0	15.00	7.50	0.44	3.13	1.33	-9.92	1.33	-21.17	0.18	-5.07	0.49	1.25	4.50	6.24
rs3-48_ne0-0_mcap_hcmp_minwbw	3	0	1	2	0	12.00	6.00	2.67	0.00	1.33	-7.67	2.67	-15.33	1.07	-3.13	3.73	0.00	3.60	7.33
as4-0_re1-24_mcap_hcmp_minwbw	4	1	1	2	0	12.00	6.00	0.38	2.50	1.33	-7.67	1.33	-16.67	0.15	-4.05	0.70	1.00	3.60	5.30
as5-18_ne0-0_mcap_hcmp_minwbw	5	0	1	2	0	12.00	6.00	3.33	0.00	11.67	0.00	11.67	-6.33	1.33	-2.87	6.17	0.00	3.60	9.77
rs6-0_ae2-18_mcap_hcmp_minwbw	6	2	1	2	0	12.00	6.00	0.38	3.33	1.33	-7.67	1.33	-16.67	0.30	-3.90	0.42	2.83	3.60	6.85
rs6-0_ae2-18_lcap_ncmp_maxriops	6	2	0	0	3	10.00	5.00	4.38	2.78	15.31	0.00	15.31	0.00	3.50	0.00	4.81	2.36	3.00	10.17
ns0-0_ae2-18_lcap_hcmp_maxriops	0	2	0	2	3	8.00	4.00	0.00	2.22	15.56	0.00	15.56	0.00	3.56	0.00	0.00	1.89	2.40	4.29

scenario naming (auto generated)
Server:
ns – no servers
rs – regular server
as – accelerated server
Ebof:
ne – no ebof
re – regular ebof
ae – accelerated ebof
Capacity:
l low, m med, h high
Compression:
l low, h high
Scenario:
Minwbw, maxwbw, maxrbw, maxriops

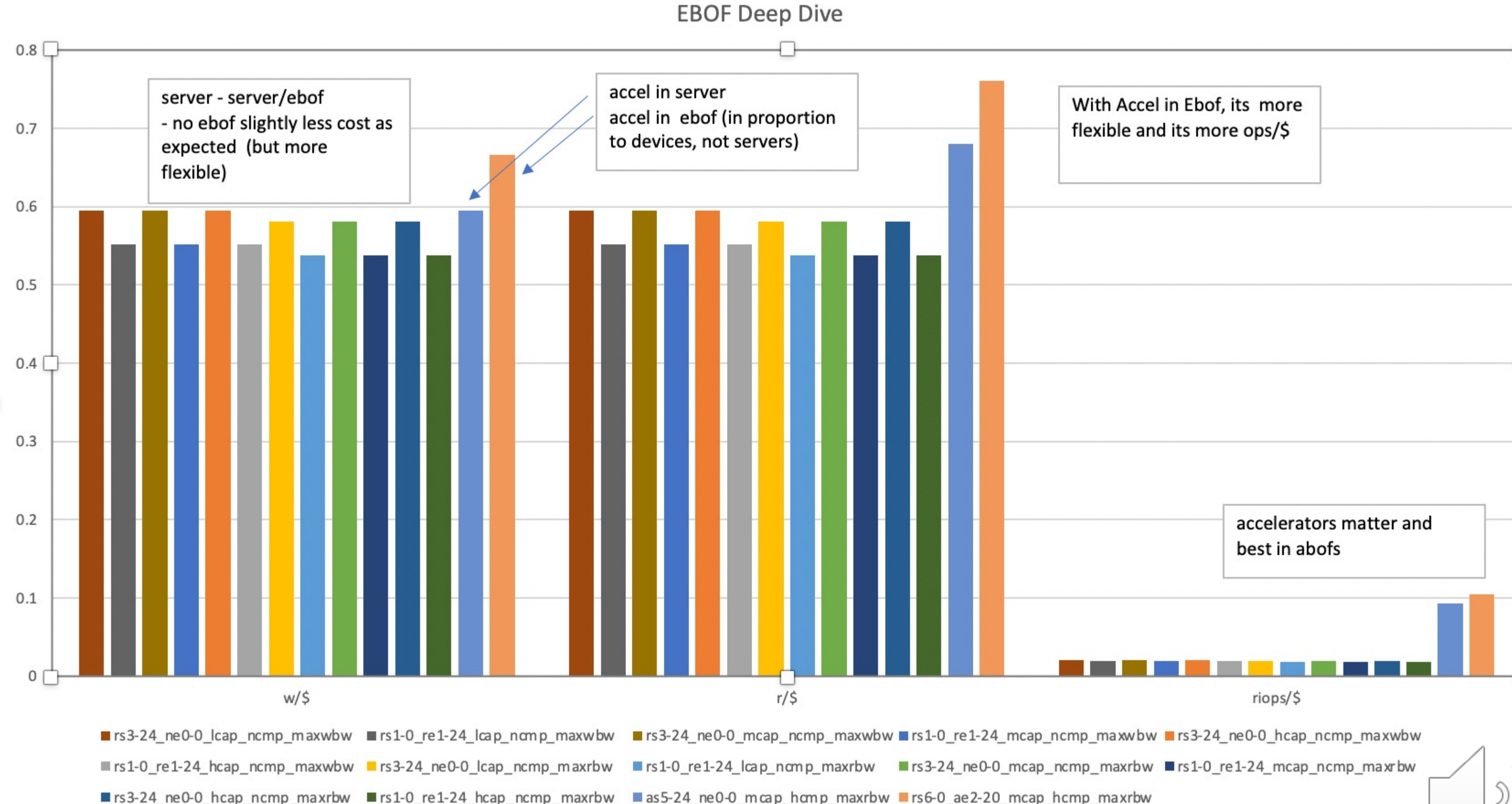
Fewer servers and acceleration gives more fixed ops/$ and using Ebof's gives more flexibility to match capacity/bw/iops and more upside potential (acceleration of fixed functions)

Why Ebof over in server - flexibility (not terribly different in cost)



No servers has huge upside potential, user space compute node service talking directly to storage devices (over network) allows rightsizing capacity, bw, iops needed for task at hand. Servers/kernels make using device potential expensive.

Deeper Dive on No Ebof vs Ebof vs Abof Acceleration Near Data Wins



Why computational storage?

LANL Mission ~= Weapons Science / $

Lets grow the Weapons Science

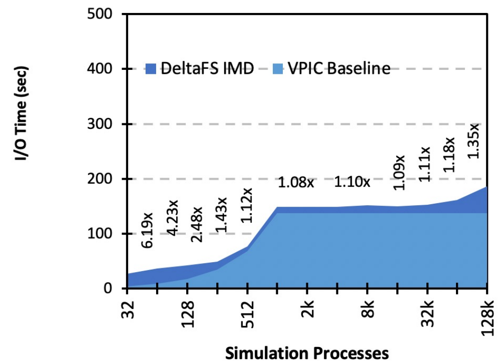
Near-device Indexing and Analytics

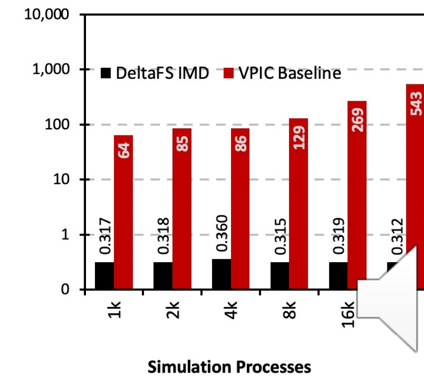
Leverage IOPS we get with our needed Capacity and BW

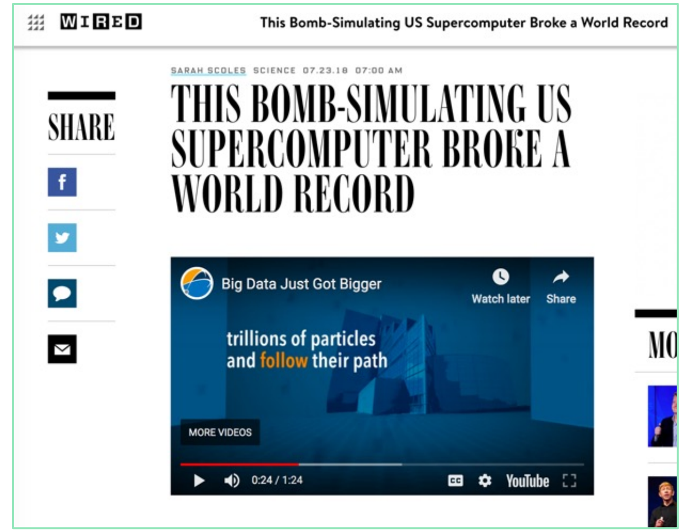
- Non-obvious requirements
  - Simulations run under intense memory pressure (app may use 90%)
  - In-situ indexing runs into scaling limitations
  - Users must only be able to see *their* data (strict security)
- Computational Storage Benefits/Opportunities
  - Speedups for post-hoc analysis (1000x speedup demonstrated)
  - Post-hoc index creation (speculative)
  - Less reliance on massive compute tier as a large merge sort space

Single pass scan vs a single dimension index, our desire is more like 3-5 dimensions of index making the taking 100-1000x to 10,000X

Add a little time indexing on the way out and get 1000X on analysis step (the indexing must scale and be efficient (perfect offload opportunity)







Only Possible Because of Key-Value Storage!

Analytics Application User Space Software Layers

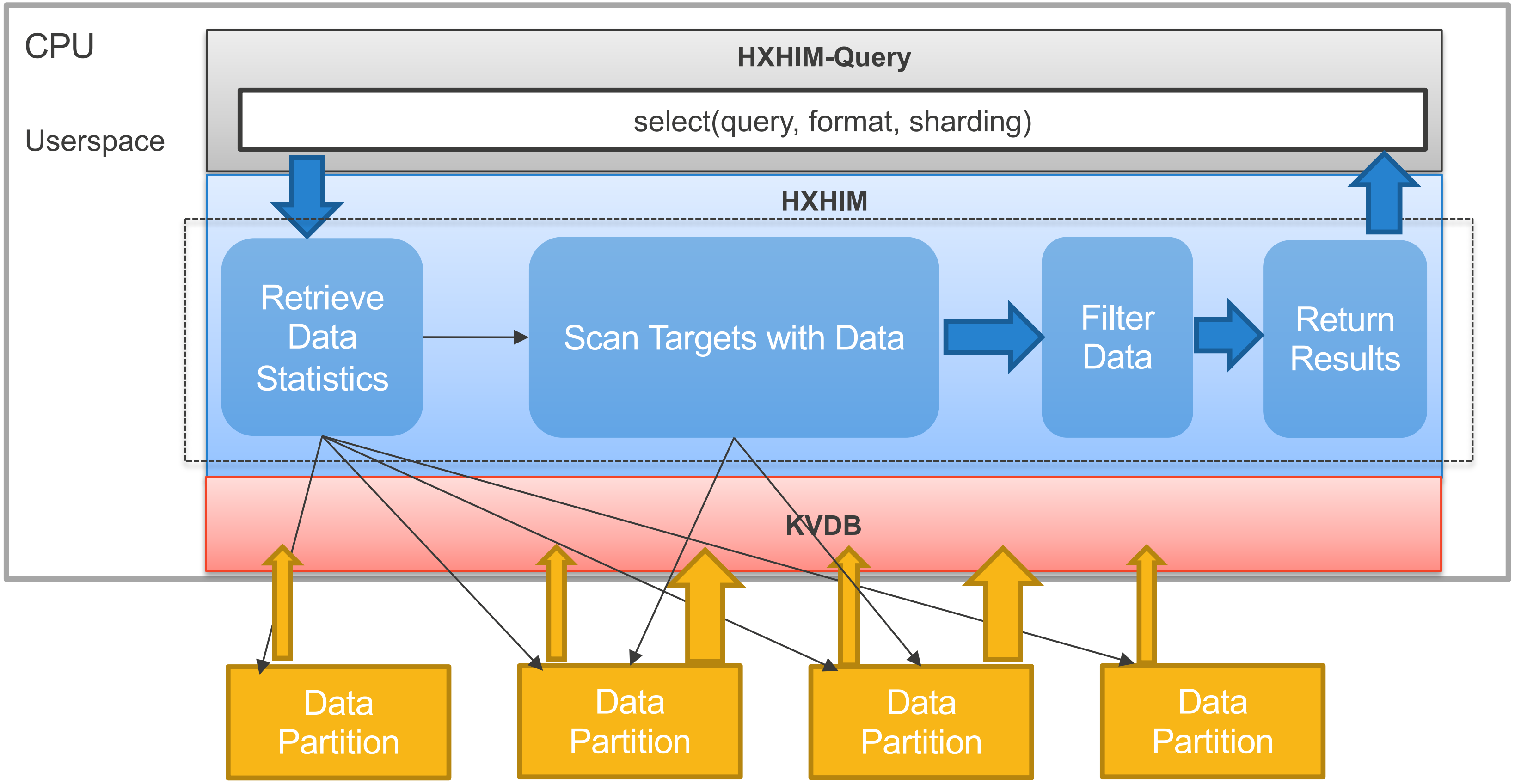
- 2 example scientific queries:
  - **Find the N highest energy particles**
    - Ex: Select 10,000 48B particles from 1 Trillion particles
  - **Return ranges of 10 or more contiguous mesh cells that contain more than N% of material X**
    - Ex: Results in 1-0 1000 cell (1KB cells) ranges from 3 Trillion cells
- Query tool leverages statistics to improve performance
  - Histogram to describe energy distribution
  - Min and max material for an array

Histograms

- Both queries can also leverage data organization for acceleration
  - Sort by energy
  - Sort by cell position

Ordered KVS/indices

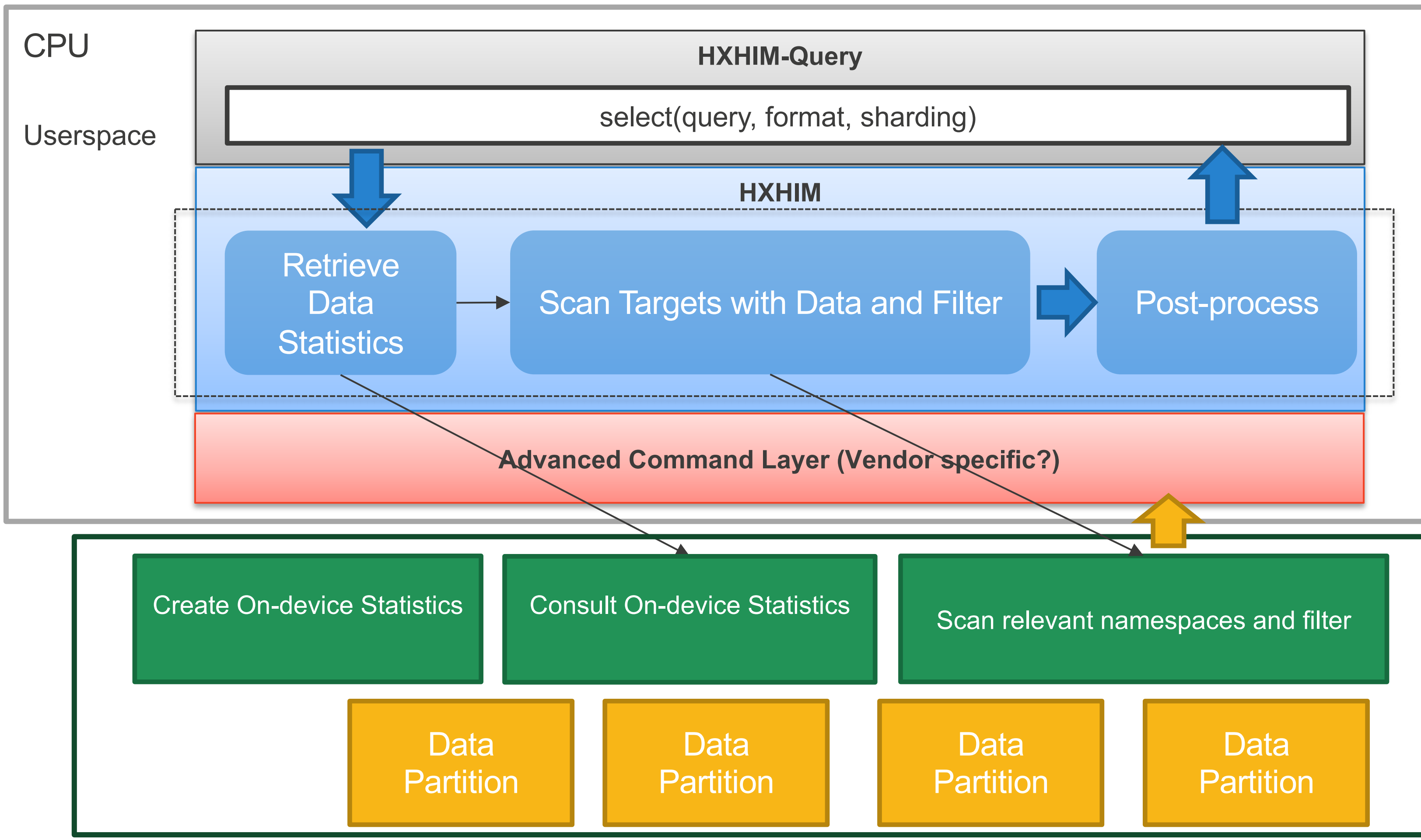
User-space Analytics Application Software Layers using HXHIM (distributed parallel KVS framework)



This is the read path only.

Statistics must be generated during output phase -- which requires additional buffering!

Offloaded Analytics Application Software Layers

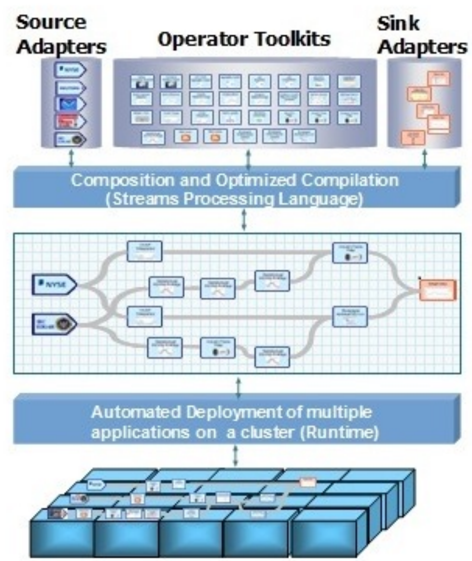


This is the read path only.

Statistics can be generated at the devices

How do you think about using computational storage/offloading functions/programming?

- Middleware: Hxhim (distributed parallel KVS framework)
  Emerging standards? NVME Computational Storage TP4091
  Runtime/Common Api's: Legion, OpenSNAPI
  Different storage paradigm than block? Ordered KVS
  Learn lessons from streams programming paradigm?
  - System S (DOD/IBM)
  - Netsketch (CMU)



Join us in seeking backwards to efficient mission

Ultra-Scale Systems Research Center

The Efficient Mission Centric Computing Consortium