High Availability for Centralized NVMe

Zivan Ori

CEO, E8 Storage

Introduction

- Since the introduction of NVMe drives in 2014, they have been rapidly adopted for use as cache, in-memory DB extension, fast tier of SSDs, fast boot devices, etc.
- NVMe adoption is across the board, starting from laptops through servers/IaaS to HPC
- However, until now, NVMe drives have only been used as local drives in servers
- Redundancy schemes were handled by the applications

Benefits of local NVMe

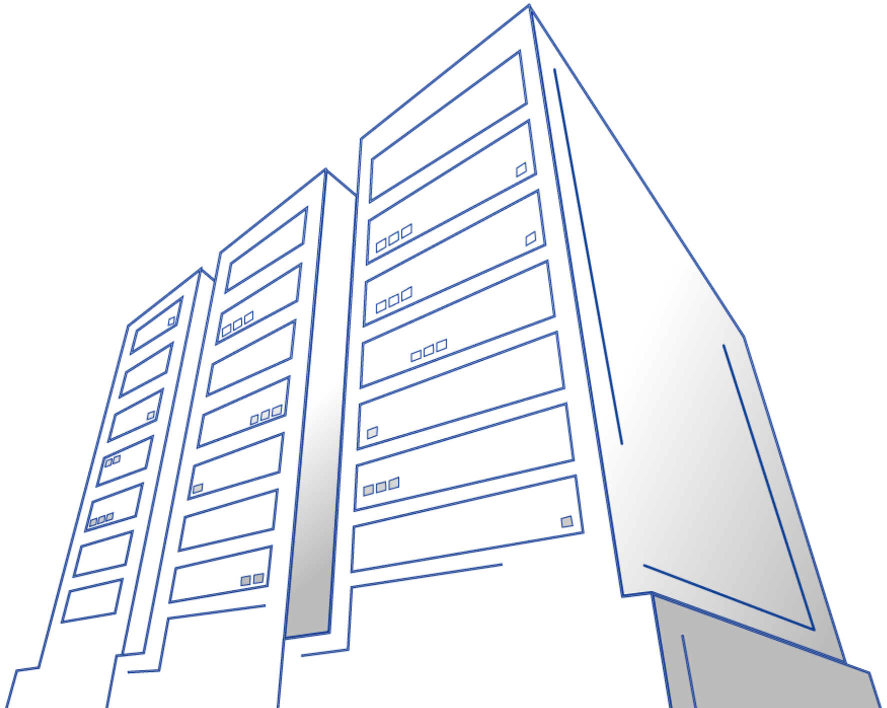
- Latency
- Bandwidth + Throughput: 6x compared to SAS/SATA
- CPU utilization
- Protocol efficiency (no need for SAS/SATA stacks)
- Less hardware needed (SAS/SATA controllers)
- Highly accepted standard, rich roadmap ahead
- Newly available 2.5” form factor allows for easy replacement of drives

Problems with local NVMe

- Redundancy / Resiliency
  - What happens to my data if an NVMe drive fails?
- Provisioning
  - I need to decide, per server, how much NVMe capacity to provision it with
  - What if my servers run out of capacity?
- “Islands of Storage”
  - Efficiency of NVMe allocation: customers surveyed indicate that on average, 30% of the SSDs capacity is utilized
- Coupling of storage and compute
  - Need to buy storage now, that I will be using for the next 4-5 years
  - Marriage of life-cycle of storage and compute

So… what about NVMe sharing?

- If one could share NVMe drives between multiple servers, there would be…
  - Redundancy / Resiliency
  - No more “islands of storage”
  - Storage-compute de-coupling (aka flash dis-aggregation)
  - Dynamic provisioning
    - Increase NVMe capacity per server – dynamically
  - LUN sharing
    - Many concurrent reads on large data-sets
    - LUN sharing for application clusters



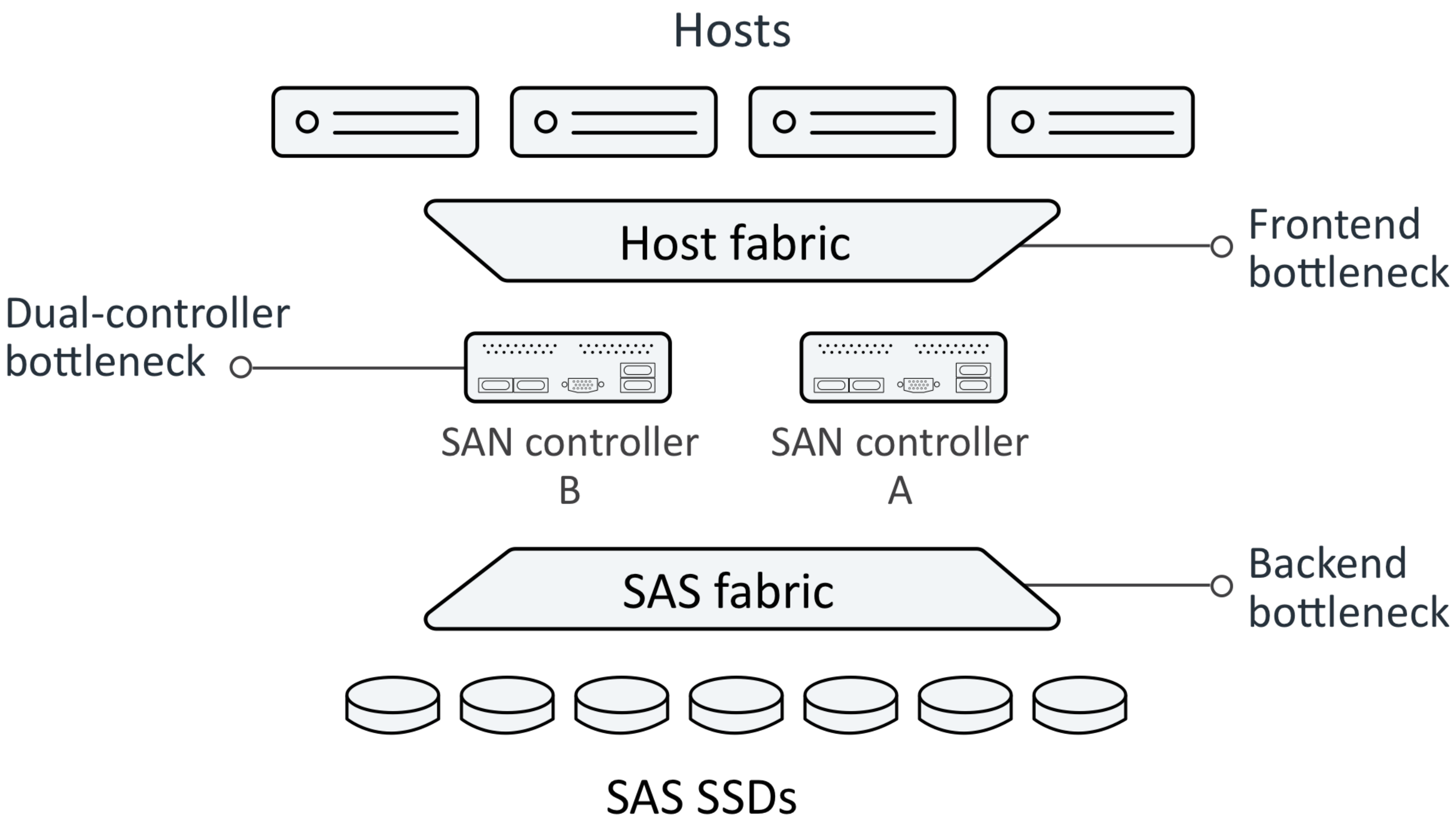
But what about performance?

- Remoting the NVMe SSDs from the server will impact latency
- **Challenge #1: retain remote latency on par with local latency**
  - (*or as close as possible)
- Bunching many SSDs in a box creates a throughput/bandwidth explosion
- **Challenge #2: get all the available bandwidth/throughput of remote NVMe**

So let’s use NVMe in our SAN

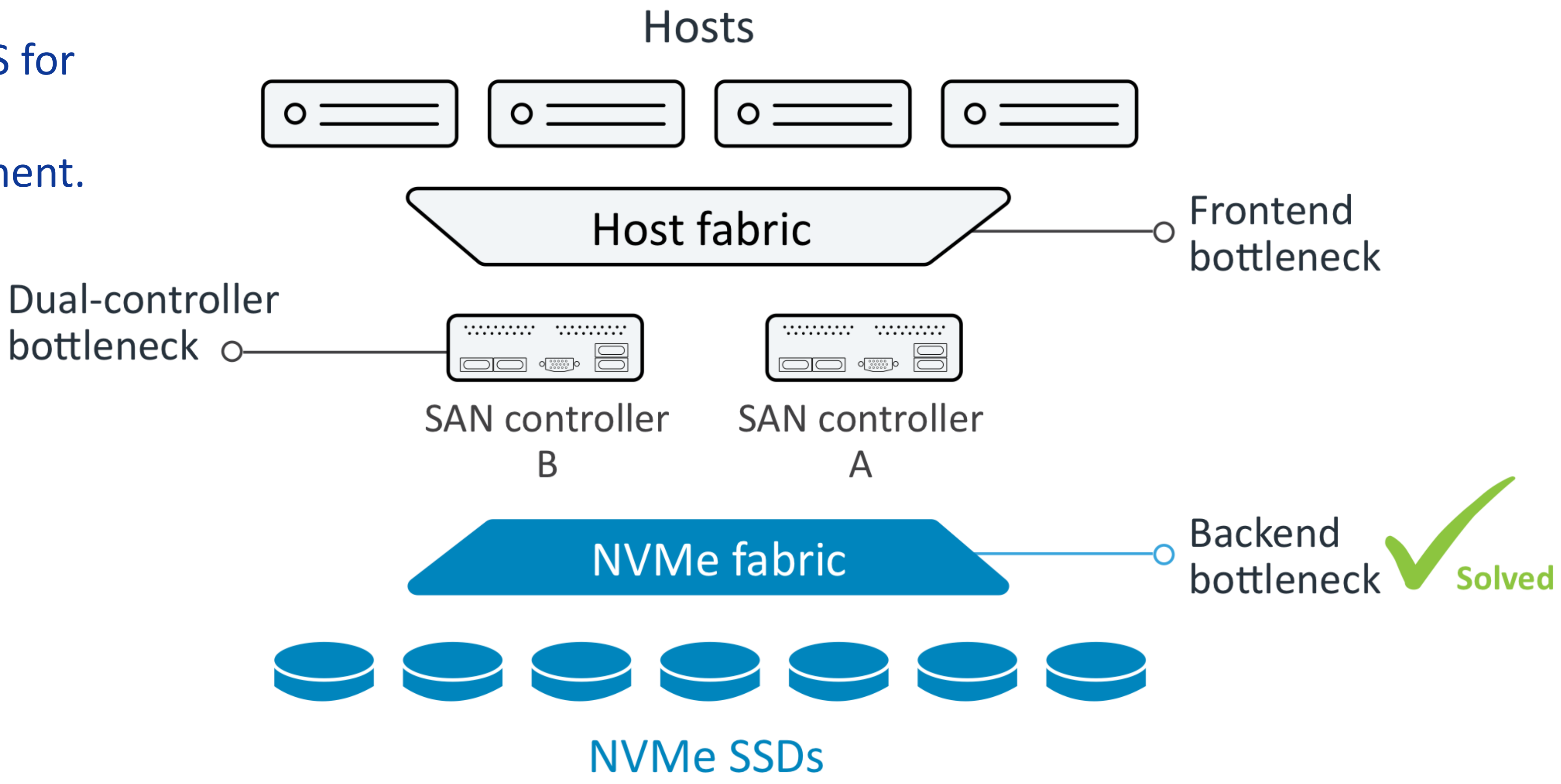
- Why not use NVMe as a SAS or SATA replacement in our SAN?
- In traditional and even in modern SAN arrays, even AFAs, **SSD connectivity is not the bottleneck**
- The I/O stack of the storage controller is the bottleneck
- Replacing SAS/SATA SSDs with NVMe SSDs will lead to little performance improvement if any at all
- **SAN with NVMe ~= SAN without NVMe**

Traditional SAN design



SAN design + NVMe

Merely replacing SAS for NVMe SSDs will not obtain the improvement.



Is the SAN adequate for NVMe?

- Single NVMe SSD = Up to 4GB/s (32 Gbps)
- Typical FC deployment = 16Gbps
- A 2U24 NVMe box would require 48 (!) FC16 ports

- **Remote NVMe requires a low-latency high-bandwidth network**

Enter NVMf

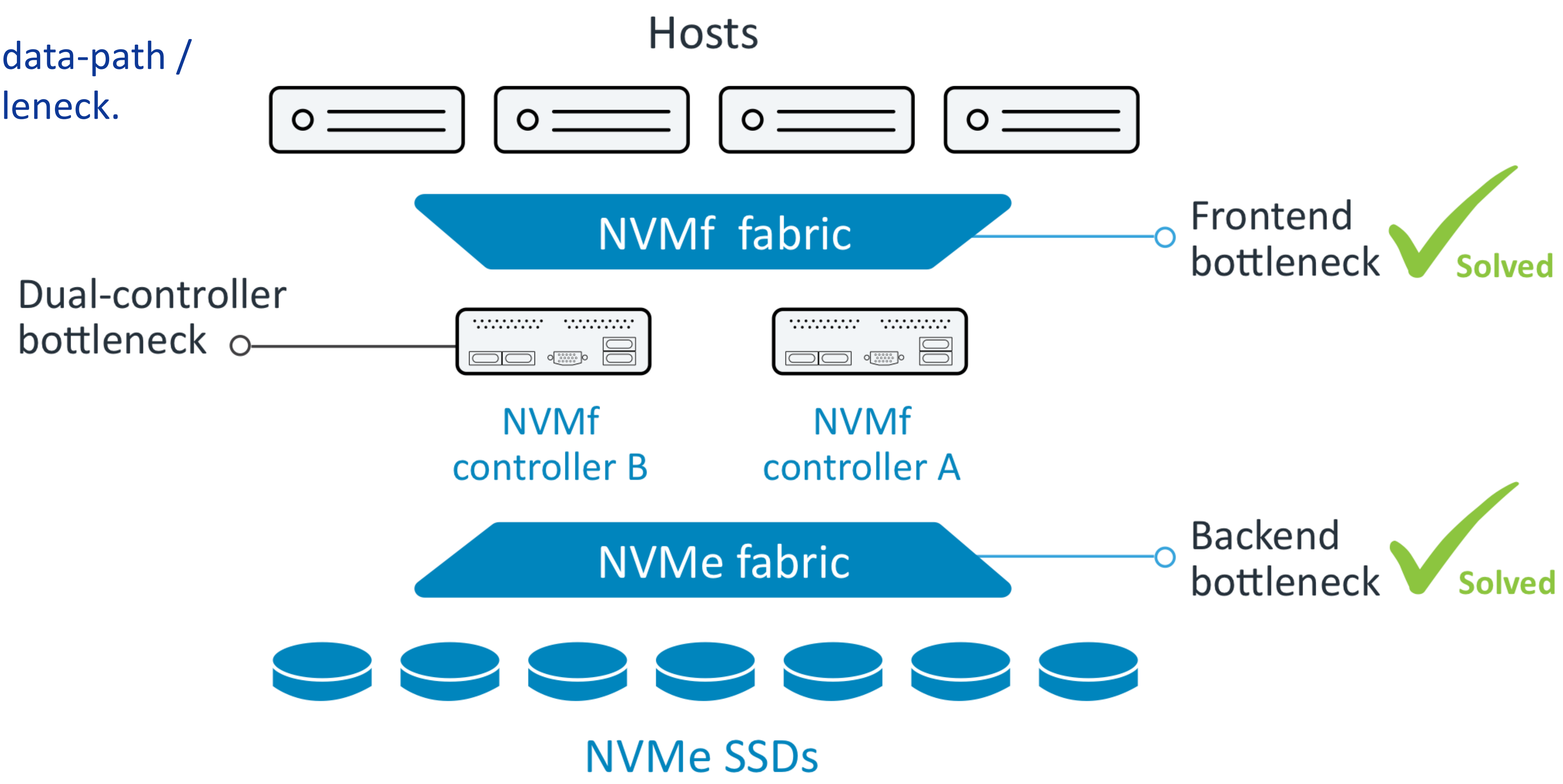
- NVMe over Fabrics standard addresses the challenge of accessing remote NVMe subsystems
- NVMf v1.0 standard was ratified in June 2016

NVMf solves data-path performance

- NVMf fabrics include:
  - High-bandwidth Ethernet (e.g. 40GE, 100GE)
  - Infiniband (e.g. 56G IB, 100G IB)
  - FC (e.g. FC32, FC128)
- Corresponds with transition of the data center from 10GE/40GE DL/UL to 25GE/100GE DL/UL
  - Server ports: 10GE → 25GE
  - Storage ports: 40GE → 100GE
  - Instead of 48 16G FC ports, use 8 * 100GE/FC128 ports
- NVMf guideline: no more than 10us of additional latency
- Such networks have the needed bandwidth/throughput

SAN design + NVMf

NVMf solves the data-path / connectivity bottleneck.


Hosts

NVMf fabric

Frontend bottleneck ✓ Solved

Dual-controller bottleneck

NVMf controller B

NVMf controller A

NVMe fabric

Backend bottleneck ✓ Solved

NVMe SSDs

Importance of RDMA for NVMf

- RDMA allows one machine to access the memory of another machine without going through its CPU
- Why is this important?
  - Lower CPU utilization on the hosts
  - Allows for “direct placement” approach in the targets
  - Lower end-to-end latency
- RDMA
  - A native property of InfiniBand fabrics
  - Available on Ethernet: RoCE
  - Available on TCP/IP: iWARP

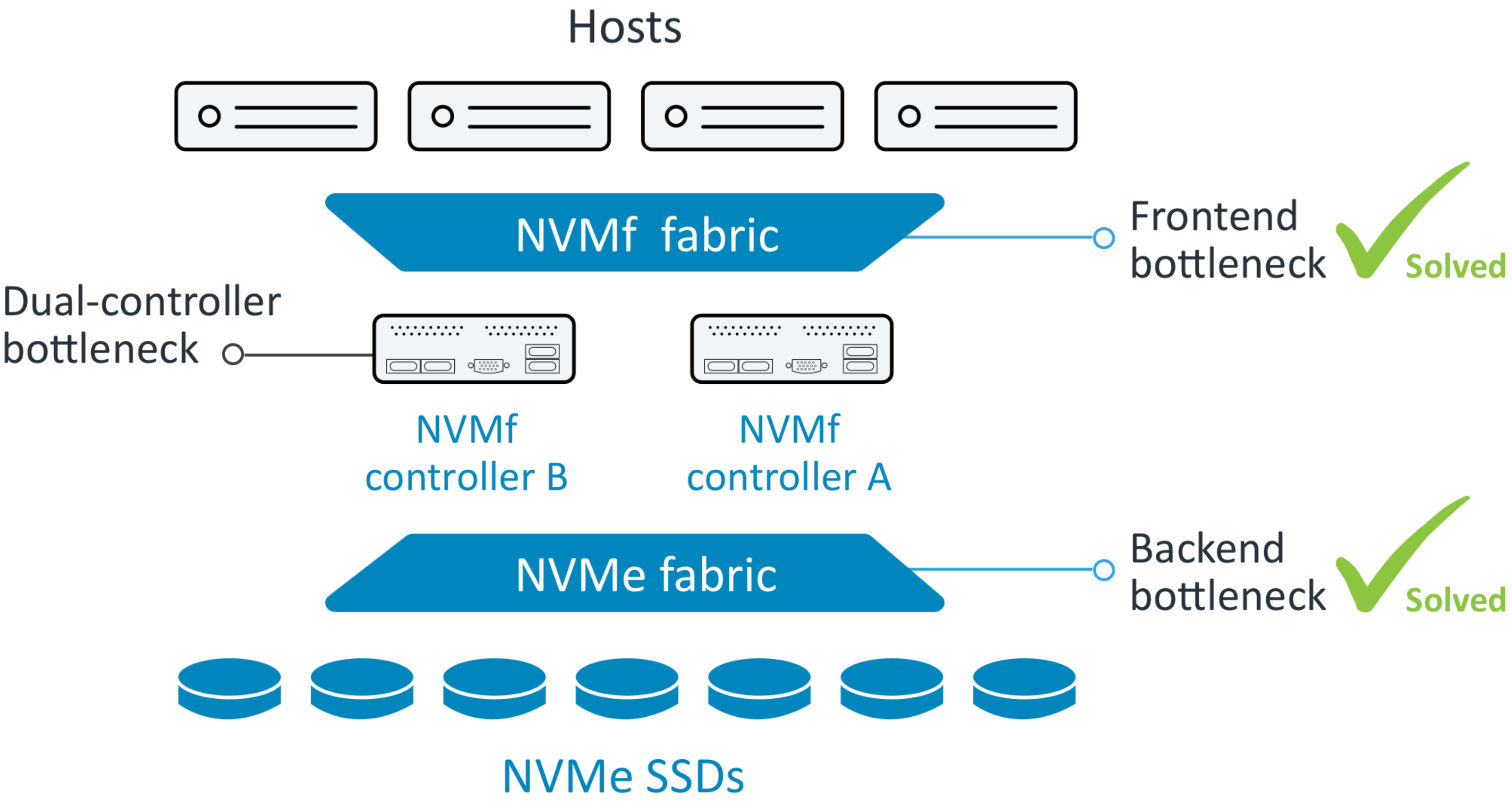
NVMf and the I/O stack

- NVMf allows a client to access a remote box of NVMe with low latency and high bandwidth/throughput
- But… it doesn’t solve the I/O stack challenges
- How is resiliency implemented in NVMf?
- Is there RAID in NVMf?
- Snapshots?
- Replication?
- Thin provisioning?
- **NVMf only formalizes the access protocol (like SCSI).**
- **NVMf doesn’t explain how to implement features.**

SAN design + NVMf

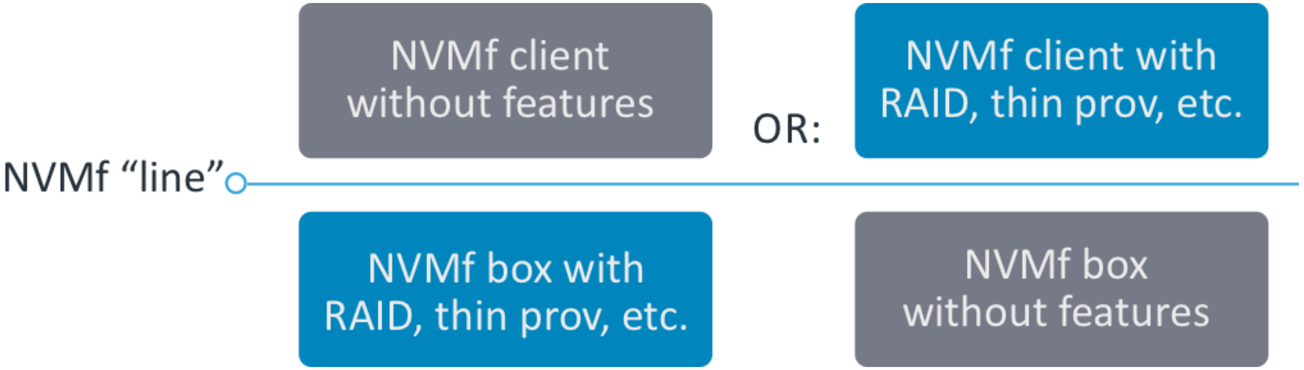
Q: Does the controller **do** anything?

If YES: dual-controller is the bottleneck still.
If NO: **where are features implemented?**



The dividing line

- NVMf is the dividing line:
  - Features like RAID or thin provisioning can be implemented either **below** the line (in the NVMf target) or **above** the line (in the NVMf host)
- Implement features below the line = scale-up solutions
- Implement features above the line = scale-out solutions

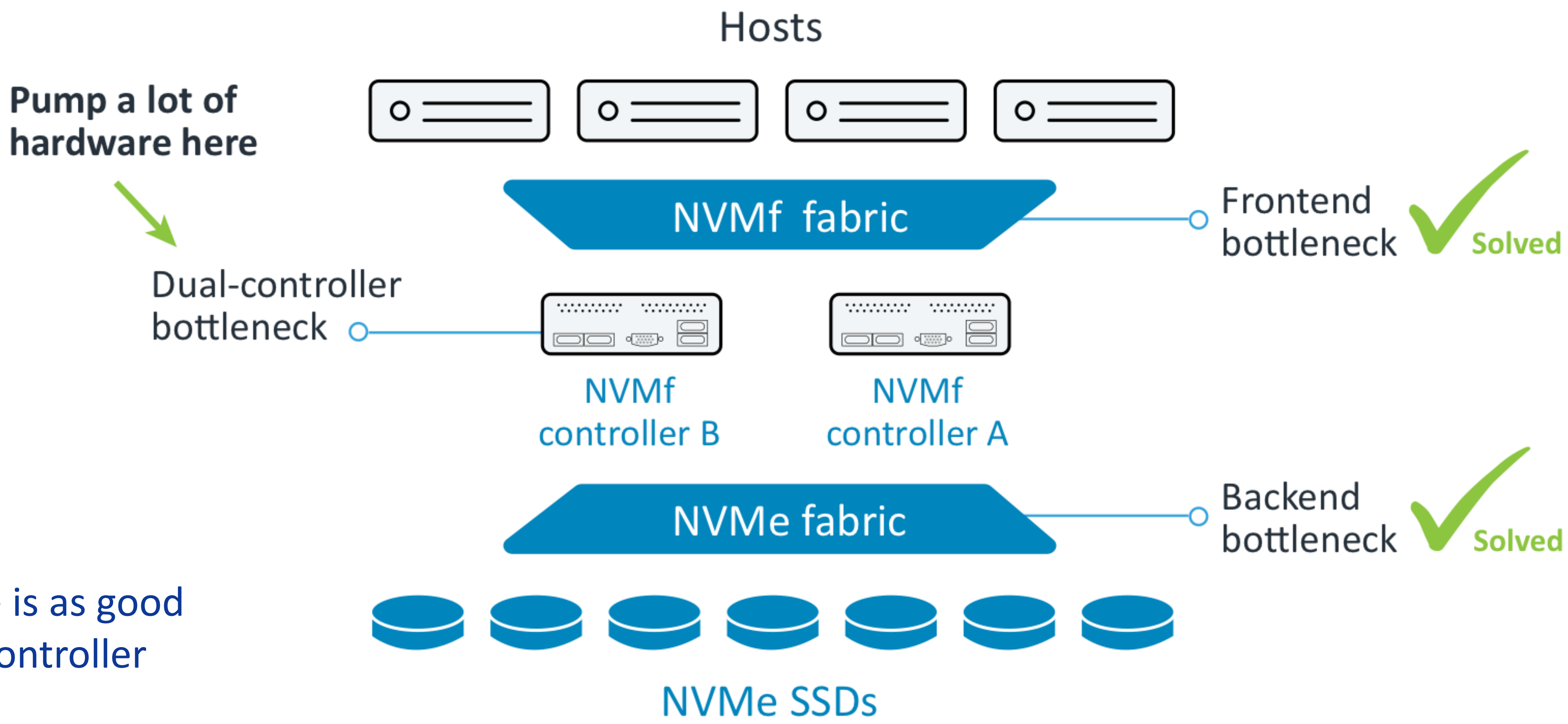

NVMf client without features
OR:
NVMf client with RAID, thin prov, etc.
NVMf "line"
NVMf box with RAID, thin prov, etc.
NVMf box without features

Scale-up solutions

- “Below the NVMf line” solutions implement RAID, thin provisioning, etc. within the NVMe/NVMf appliance
- Performance is as good as the hardware performing these operations – classic scale-up
- Can lead to hardware-centric solution
  - Hardware RAID implementation
  - Hardware-optimized data structure

- **Hardware-defined NVMf approach implements storage features in the appliance.**

NVMf: Scale-up

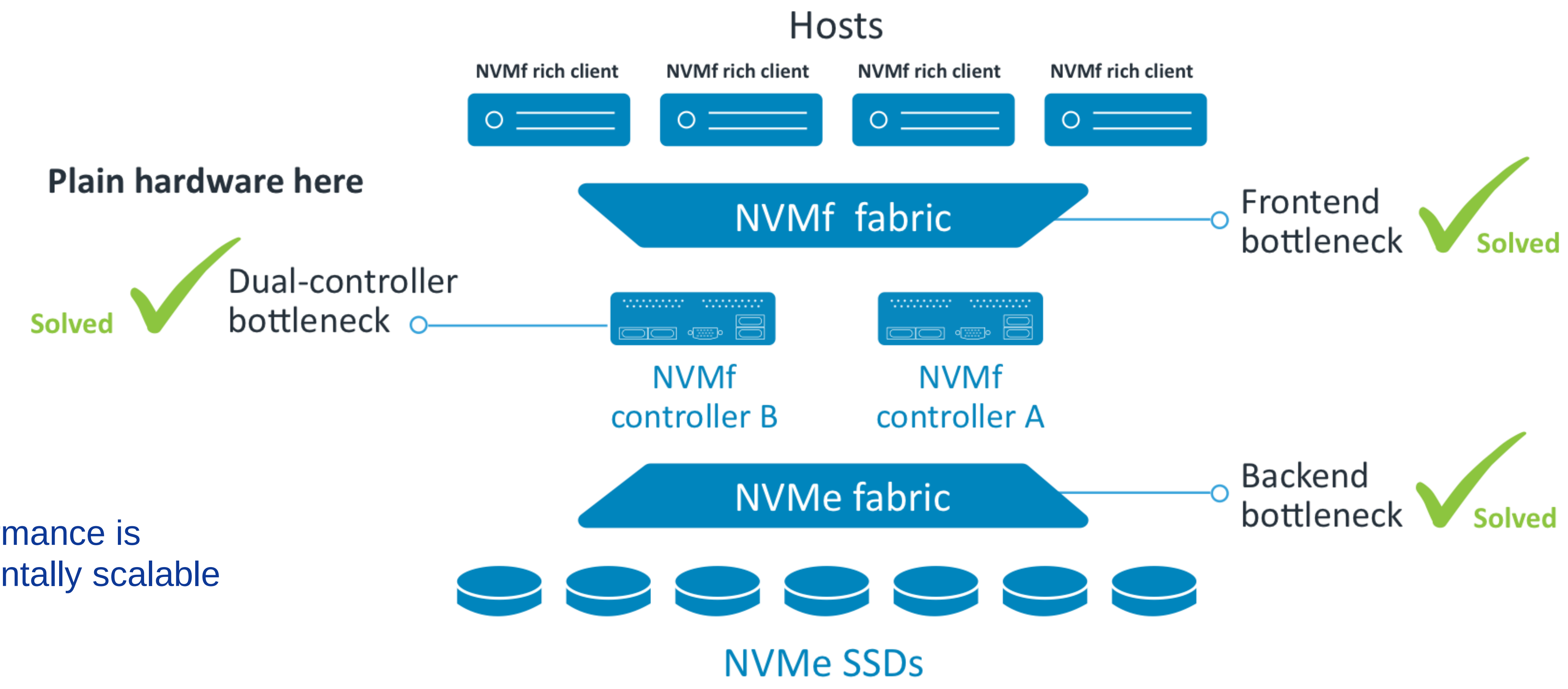


Performance is as good as my dual-controller

Scale-out solutions

- “Above the NVMf line” solutions implement RAID, thin provisioning, etc. **outside** the NVMe/NVMf appliance
- **Scale-out** the implementation to many small entities
- Requires a fast-enough network
  - NVMf requires that anyway
- Ideally a switchable (L2) network
  - FC, Ethernet & IB are switchable
- Would benefit from a routable (L3) network
  - ROCEv2 and iWARP are routable
- **Software-defined NVMf approach implements storage features outside the appliance.**

NVMf: scale-out



Performance is horizontally scalable

Challenges with scale-out NVMf

- How are the different entities coordinated?
- What happens when an entity fails?
- How are LUNs shared in this approach?

- Requirements:
  - Network resiliency, multi-pathing
  - Host failure resiliency
  - Efficient SSD redundancy scheme, replicas provide no improvement over local NVMe usage
  - Enclosure redundancy scheme
  - Enclosure high availability

Benefits of scale-out NVMf

- Scalable in number of SSDs
  - Able to extract the entire bandwidth & throughput of the SSDs
- Scalable in number of network ports
  - Add more ports → get more bandwidth
- Scalable in number of NVMe enclosures
  - Add more enclosures to get more capacity and performance
- Scale to hundreds of servers

Comparison of the 2 approaches

Scale-up	Scale-out
NVMf target (“Below the line”)	NVMf host (“Above the line”)
Hardware-centric	Software-centric
Performance is as good as the scale-up controller	Get the entire performance of all SSDs – no single controller
Add more SSDs? → HW change	Add SSDs easily
Add more networking ports? → HW change	Increase number of network ports and network rate easily
Aggregate enclosures? → HW change	Stack SSD enclosures easily
Add more hosts? → HW change	Scale to hundreds of servers
Segregated network	Converged network
Future = more hardware revisions	Future-proof

Summary & Conclusion

- NVMe is radically changing data centers
- NVMe in local usage is well understood
- Opportunity for NVMe in shared usage has tremendous value and benefits for customers
- Shared NVMe introduces performance and scalability challenges
- Newly available dual-ported NVMe SSDs and the NVMe over Fabrics standard are enablers for shared NVMe
- While the problem lends itself to hardware solutions, a scale-out software-centric NVMf solution will provide greater robustness and scalability