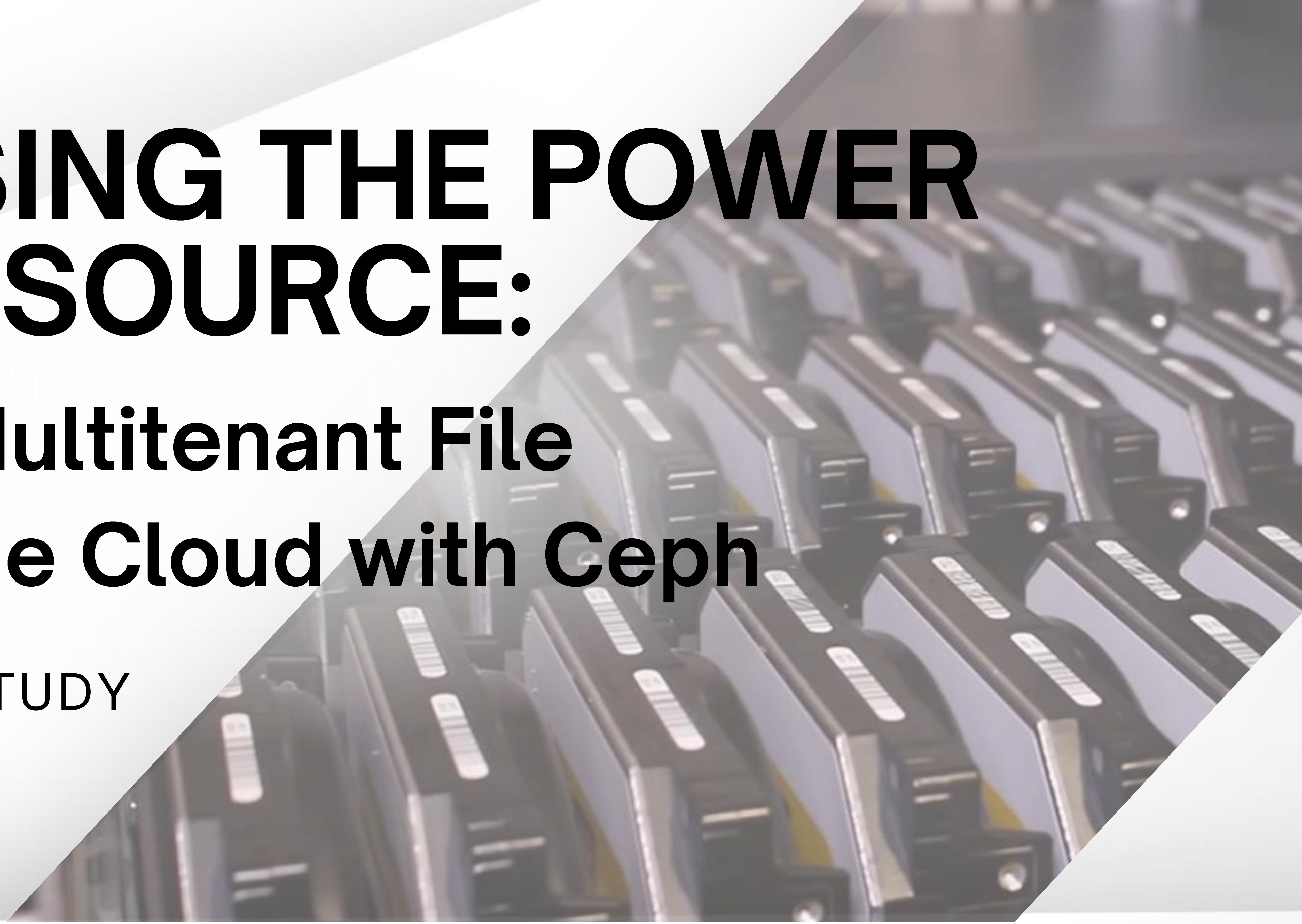
HARNESSING THE POWER OF OPEN SOURCE:

Designing a Multitenant File System for The Cloud with Ceph

A 45DRIVES CASE STUDY



45Drives

TABLE OF CONTENTS

+ + +

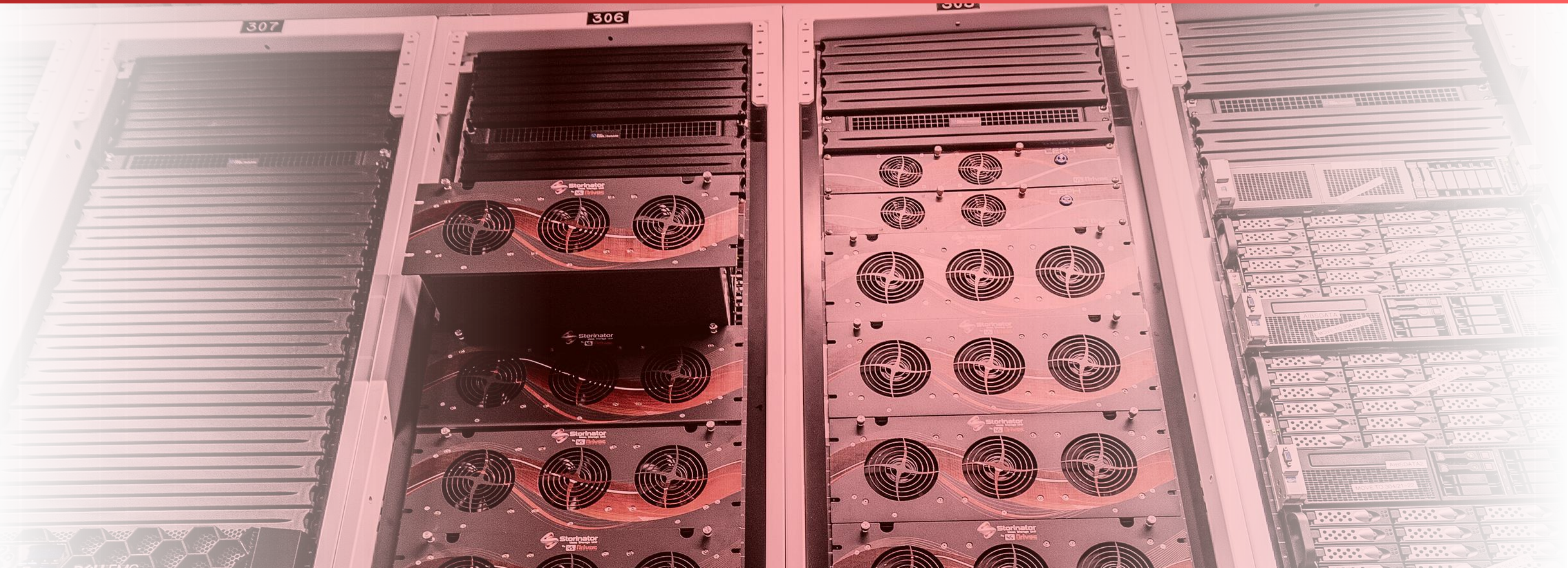
A STORAGE ARCHITECT'S DREAM

45Drives was approached by a Fortune 100 company about a new cloud facing solution they were designing for the M&E industry.

This opportunity came with many unique constraints which brought on many fun challenges to sink our teeth into.

PROJECT REQUIREMENTS

45Drives



THE DESIGN PHASE

- Multi-tenant solution – Storage for use in their larger cloud product
- Storage tiers – HDD and SSD
- Start small then expand to massive sizes
- Encryption at Rest
- Automated deployment of storage for new tenants
- Filesystem workflow – No Object (yet)

FIVE MAIN CHALLENGES

Scalability Challenge

The solution must be scalable to accommodate upwards of hundreds of multi-tenant workstations.

Multitenancy Challenge

The solution must be fully multi-tenant. No tenant data shall ever be visible to any other tenant.

Encryption-at-Rest Challenge

The solution must encrypt all data stored on the system at rest to ensure the theft of any drive would not compromise data integrity.

Automation Challenge

The solution must have automation to do things such as configure, decommission and provide maintenance functions.

File System Challenge

The solution must natively support ubiquitous file system network protocols such as SMB/NFS and have support for things such as quota management and snapshots.

SCALABILITY - HARDWARE CHOICES

- 5X S45H32
- 3X Proxmox VE Compute nodes

Each node has:

- 30 HDD slots
- 32 SSD slots

- 10X SSD slots reserved for RocksDB/WAL for HDD
- 22X SSD slots for dedicated SSD tier 0

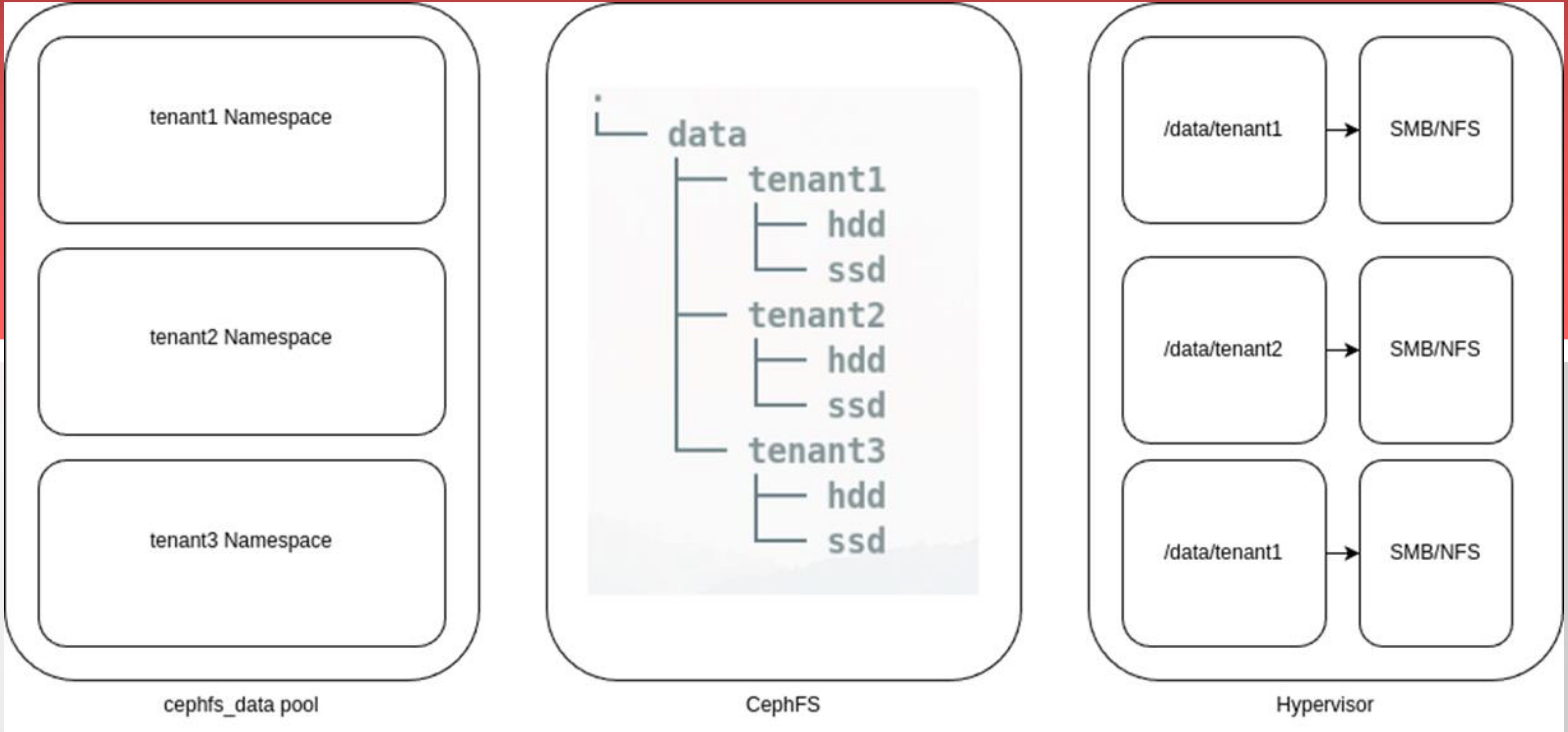
SCALABILITY – SOFTWARE CHOICES

45Drives is a firm believer and sponsor of Rocky Linux – and it is our #1 OS choice for our solutions – Rocky Linux 8.7

Ceph Octopus 15.7.2 was chosen as it was our highest supported version of Ceph at the time

Proxmox VE 7.1

MULTITENANCY – NAMESPACES



MULTITENANCY - NAMESPACES

- We can control which RADOS namespace data in certain directory by using CephFS File Layouts
- Here is a snippet from our ansible code that runs during initial deployment of new tenant

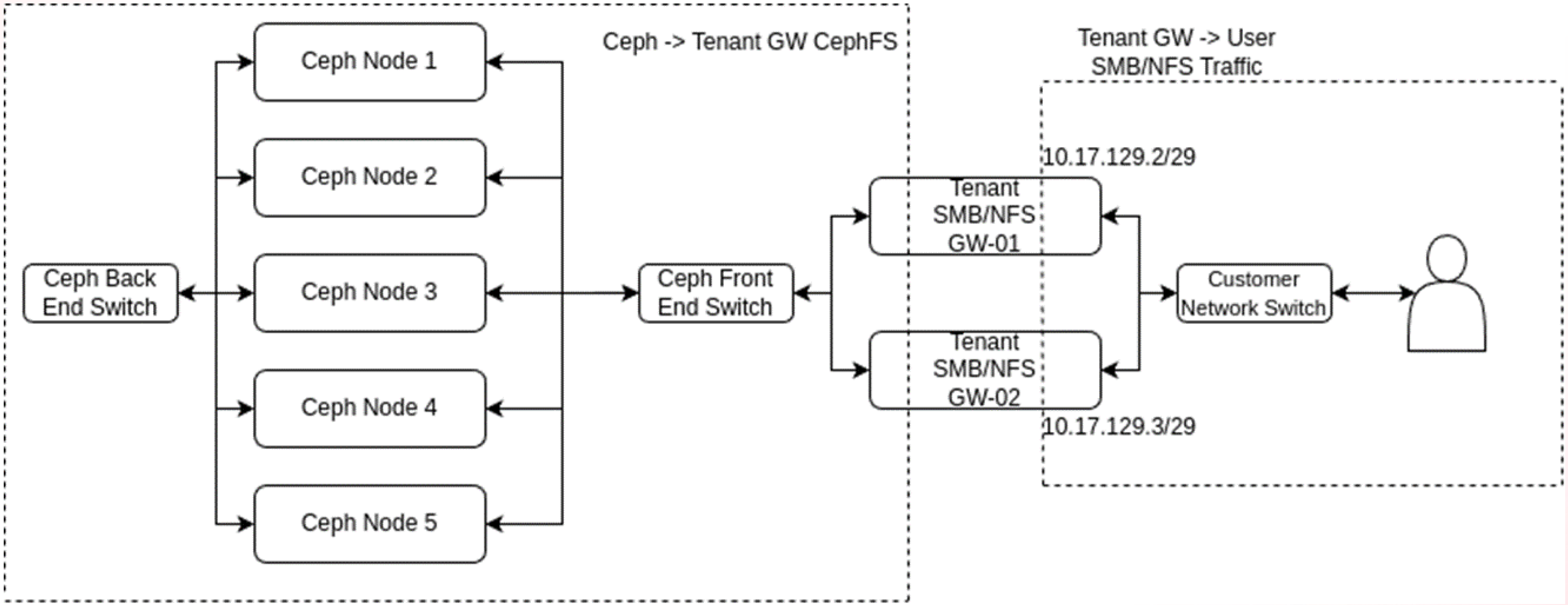
```
- name: configure_cephfs_backend | Set namespace on {{ item.uuid }} data path on cephfs
  command: |
  /usr/bin/cephfs-shell "setxattr /data/cu{{item.uuid}} ceph.dir.layout.pool_namespace {{item.uuid}}"
```

MULTITENANCY - KEYRINGS

- Example of a tenant keyring:

```
[client.tenant1]
    key = AQA48ddeHSZsEhAA/BHREi/Z/YceNAqBe0HpAg==
    caps mon = "allow r"
    caps mds = "allow r path=/data/tenant1, allow rws path=/data/tenant1/hdd, allow rws path=/data
/tenant1/ssd"
    caps osd = "allow rw namespace=tenant1 tag cephfs data=cephfs, allow rwx pool=ctdb"
```

MULTITENANCY - NETWORKING



ENCRYPTION AT REST - DMCrypt

Easy! Use dmcrypt.

ENCRYPTION AT REST - SED DRIVES

- Problem #0 – Hard requirement from customer to use SED drives
- Problem #1 – No existing tools for unlocking drives automatically
- Problem #2 – Open source tool for locking/unlocking drives “sedutil” only managed up to 26 drives
- Problem #3 – Unlocking drives in time before ceph-volume wanted access to the drives

ENCRYPTION AT REST - SED DRIVES

- Problem #1 – No existing tools for unlocking drives automatically
- Solution – Combination of udev rules and Python script

```
#
# /usr/lib/udev/rules.d/90-sed-unlock
KERNEL=="sd*[!0-9]", ENV{ID_MODEL}=="ST8000NM004A-2KE", ACTION=="add", RUN+="/usr/lib/udev/sed_init %k"
```

1

ENCRYPTION AT REST - SED DRIVES

- Problem #2 – Open source tool for locking/unlocking drives “sedutil” only managed up to 26 drives
- Solution – Fork sedutil repo and update code

5 linux/DtaDevOS.cpp

```
@@ -135,7 +135,10 @@ int  DtaDevOS::diskScan()
        if(dir!=NULL)
        {
            while((dirent=readdir(dir))!=NULL) {
-               if((!fnmatch("sd[a-z]",dirent->d_name,0)) ||
+               if((!fnmatch("sd[a-z]",dirent->d_name,0)) ||
+                                           (!fnmatch("sda[a-z]",dirent->d_name,0)) ||
+                                           (!fnmatch("sdb[a-z]",dirent->d_name,0)) ||
+                                           (!fnmatch("sdc[a-z]",dirent->d_name,0)) ||
                        (!fnmatch("nvme[0-9]",dirent->d_name,0)) ||
                        (!fnmatch("nvme[0-9][0-9]",dirent->d_name,0))
                        ) {
```

ENCRYPTION AT REST - SED DRIVES

- Problem #3 – Unlocking drives in time before ceph-volume wanted access to the drives
- Solution – custom override for ceph-volume service to wait for sedunlocker service

3

```
[root@bk ~]# systemd-analyze plot > /tmp/bootup.svg
```

Automation – Software Choices

Required tools to create/destroy/expand/shrink a tenant's access into the cluster

PROXMOX

ANSIBLE

HashiCorp Terraform

cloud-init

FILE SYSTEM – SMB + NFS INTEGRATION

- Re-used much of our standard Ansible code to deploy HA SMB/NFS Clusters
- For SMB we use a CTDB + SAMBA Cluster with CTDB lock file stored on RADOS pool
- For NFS we use NFS Kernel server with PCS handling VIP and NFS service fail-over

```
[legacy]
  #realtime scheduling = true will cause ctdb to fail when docker containers are running
  realtime scheduling = false

[cluster]
  recovery lock = !/usr/libexec/ctdb/ctdb_mutex_ceph_rados_helper ceph client.tenant1 ctdb lock_tenant1
```

FILE SYSTEM – SNAPSHOTS & QUOTAS

- CephFS quotas are used in conjunction with CephFS Kernel mounts to deliver quota visibility straight to SMB/NFS clients
- CephFS snapshots are used with 45Drives code that prunes snapshots based on scheduling

SCALED TESTS IN 45DRIVES LAB

Ceph cluster environment

- 4X OSD nodes (Xeon Gold 6230R / 128GB DDR4 / 2X 40Gb LACP bond)
- 12X Micron 3.8TB 5300 Pro per node
- 3 Replica CephFS 1024 PGs
- Rocky Linux 8.7 Ceph Octopus (v15.2.6)

Proxmox VE cluster environment

- 3X Proxmox VE nodes 7.1 (Xeon Silver 4216 / 128GB DDR4 / 2X 40Gb LACP bond)

SAMBA-GW VM

- 4VCPU
- 12GB DDR4
- Rocky Linux 8.7

45DRIVES CONFIGURATION CHOICES

Ceph cluster environment

- Each tenant results in a new MDS being deployed

Proxmox VE cluster environment

- All tenant gateway pairs will be deployed in HA groups to ensure balancing gateways across compute nodes, and automatic failover in the event of node outage
- All VM's will use librbd RBDs as OS disks

Gateway environment

- Each share will be an individual CephFS kernel mount
- Kernel clients offer higher performance, and if each share maps back to an individual CephFS mount, clients share have full visibility of quotas set on share

BENCHMARKING IN OUR SLEEP

In order to get a full picture on the scalability of our solution, we had to design and build a robust test environment and load generator to simulate workloads.

Samba-multitenancy playbooks adapted

- Re-designed multitenancy playbooks to spin up client VMs for load generation
- Settled on 6x different fio benchmarks + small file read/write tests for load testing to simulate many workloads
- Began with control tests using 1X dedicated bare-metal SMB server to identify where it began to get overwhelmed

THE DEVIL IS IN THE DATA

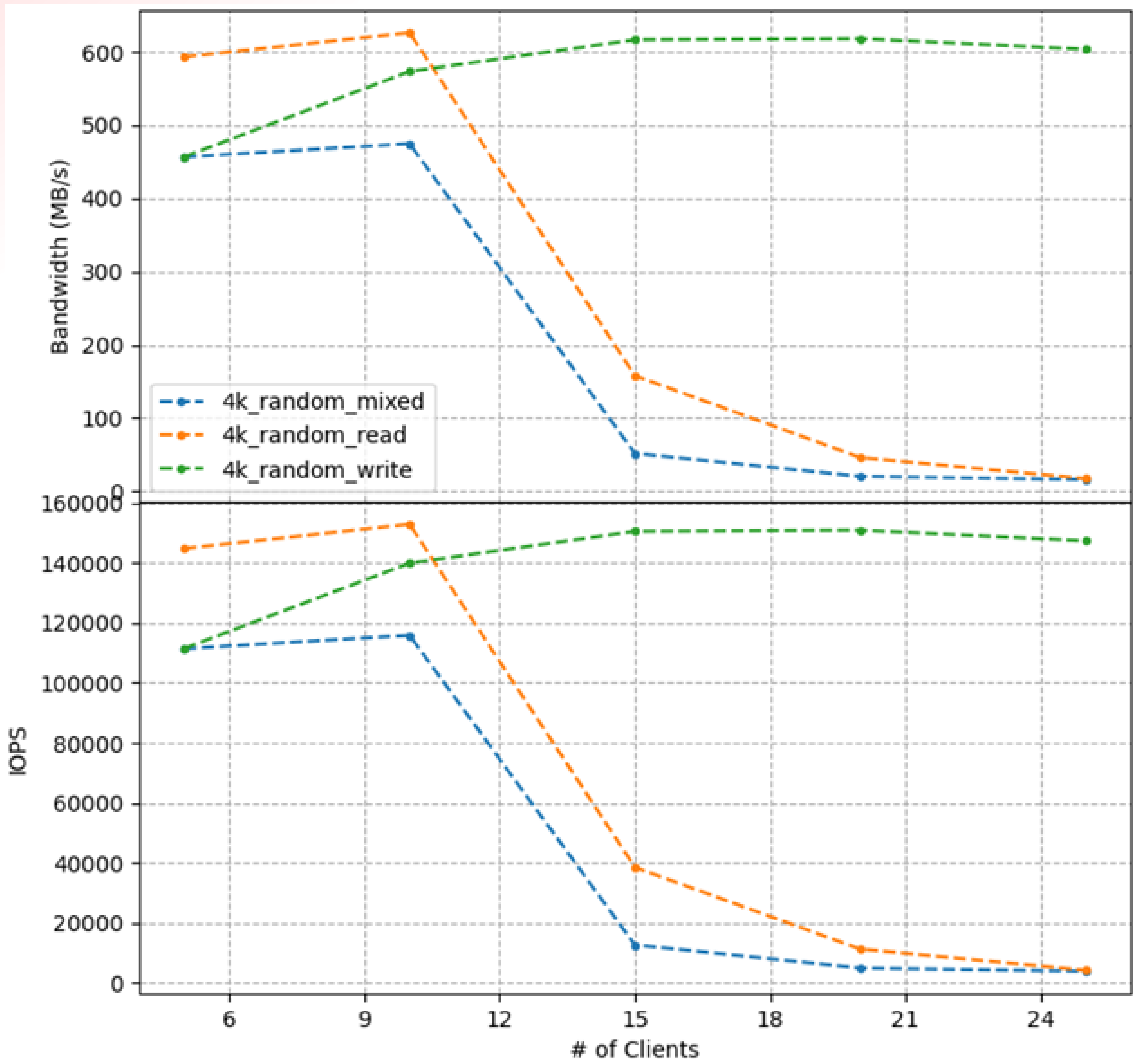
Summary of most interesting findings:

Single Bare Metal SMB GW becomes overwhelmed
When moving from 10 to 15 simultaneous clients hitting SMB hard for sequential and random read/write workloads, SAMBA + Single CephFS mount become large bottleneck

Virtualized gateways fare much better
Per client performance improves considerably even with 25 clients when scaling out SMB GWs

For latency sensitive workloads (video editing), scaling out is key
How video editing feels is key – and keeping latency down when scrubbing timelines ensures this

Bare Metal SMB GW Becomes Overwhelmed

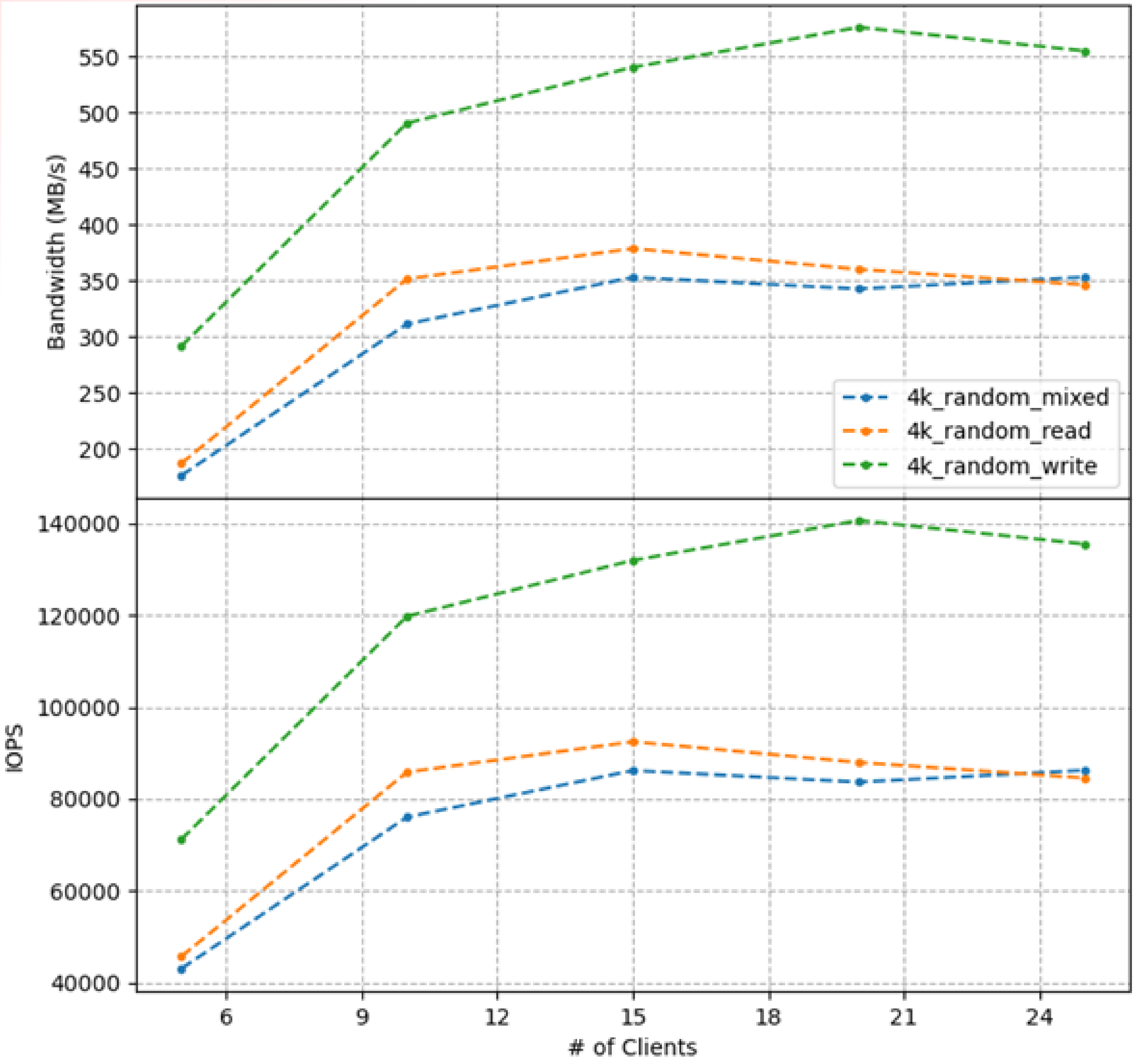


Random IO hits its peak

If there are a very small number of concurrent clients hitting the solution, bare metal is the clear winner.

Performance begins to degrade severely after 10 clients – overwhelming the single host.

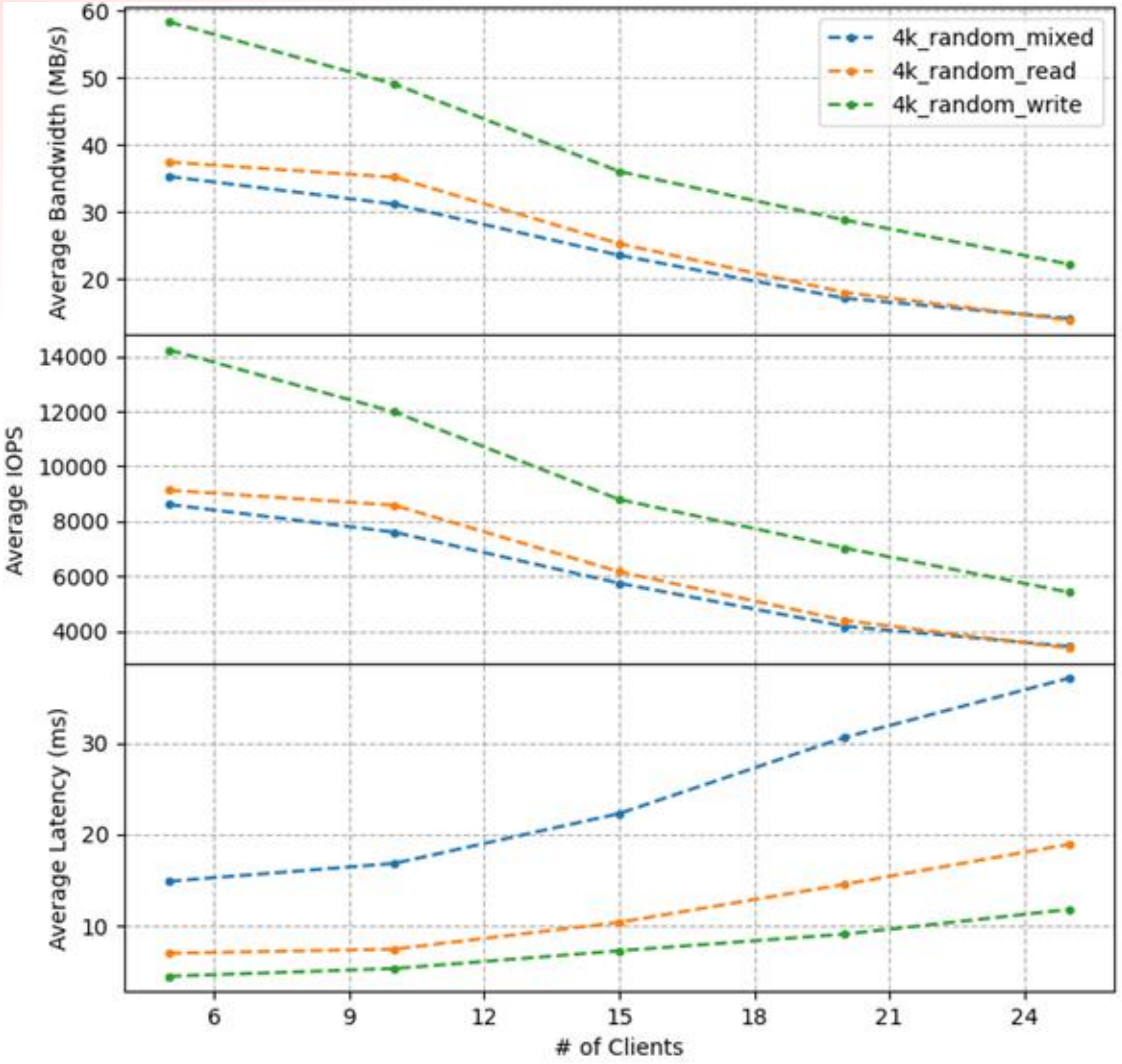
Virtualized gateways fare much better



Per-client average performance improved

Performance continues to scale well past the 10 clients and as a result, latency remains low as more clients continue to come online.

For latency sensitive workloads (video editing), scaling out is key



Keeping latency low is key

Even with 25 clients hitting the solution simultaneously, the latency remains overall under 40 ms.

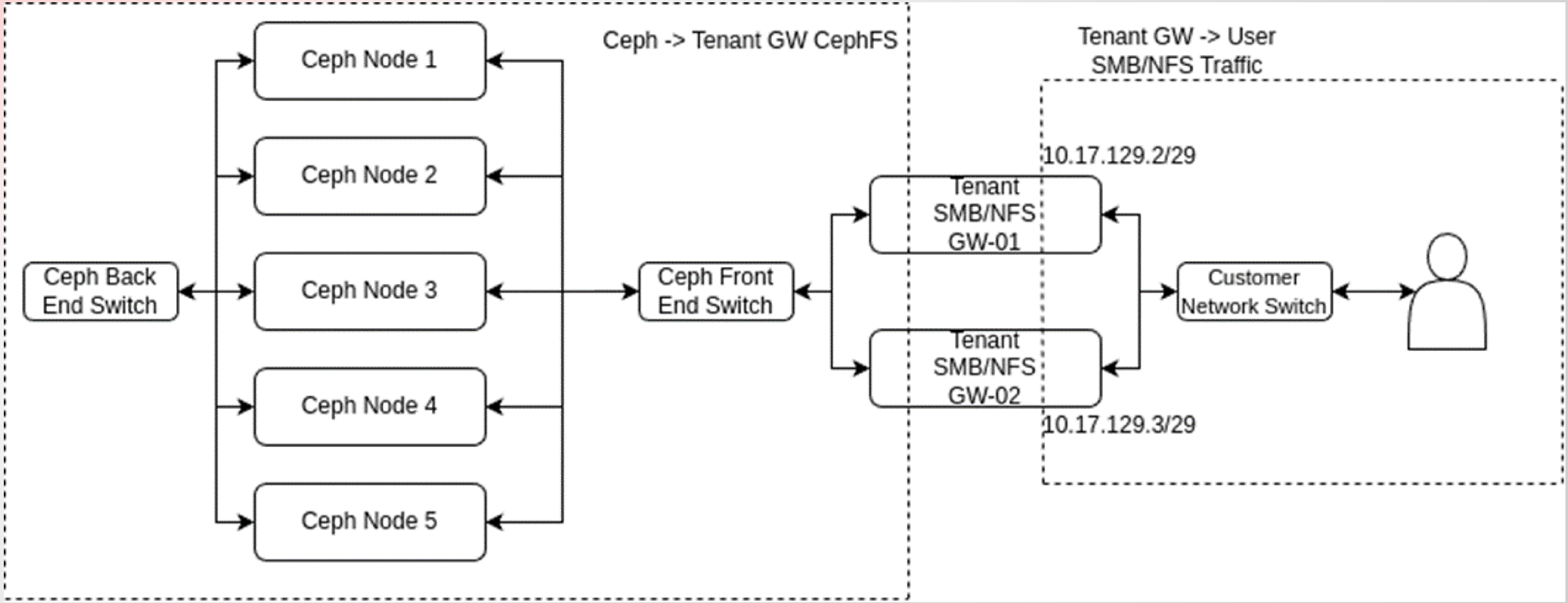
TIME TO DEPLOY!

45Drives

BIG. STRONG. FAST.

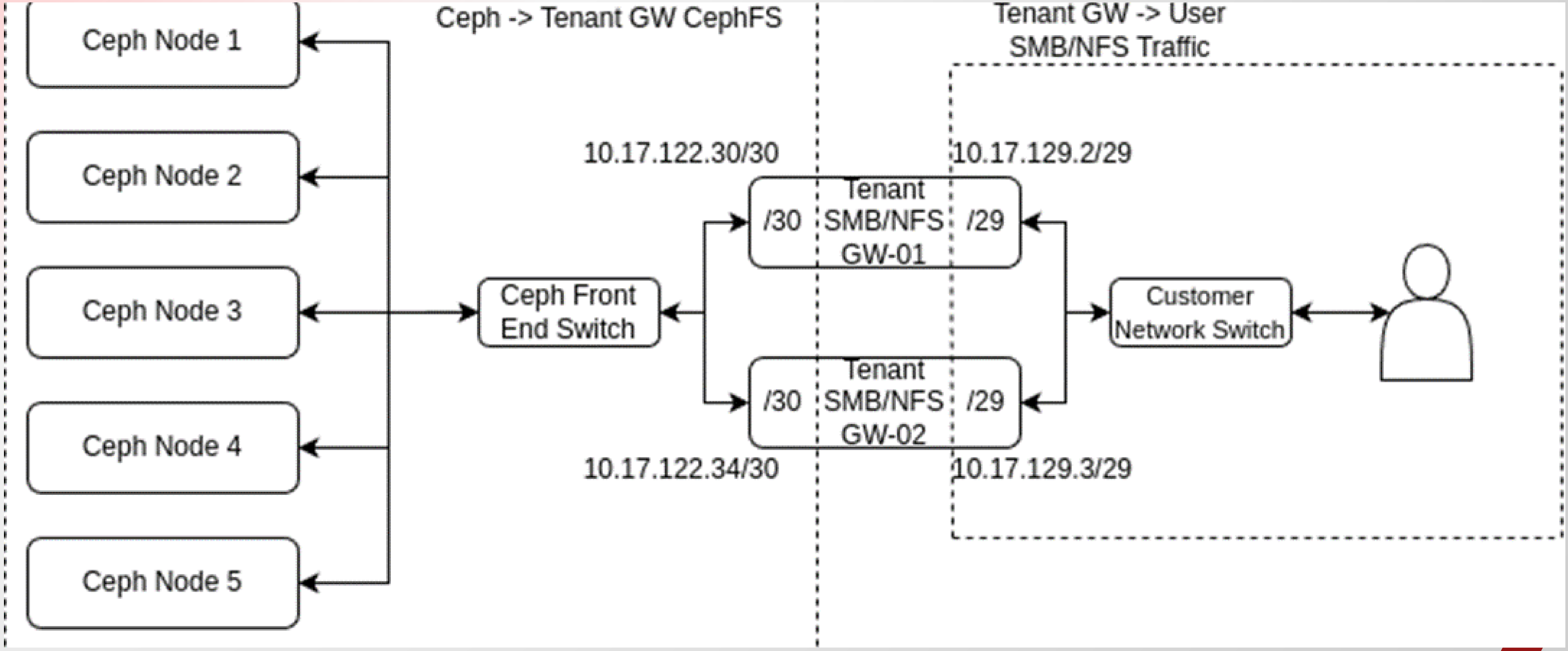
WRENCH IN THE WORKS

Expectation vs.



WRENCH IN THE WORKS

The Reality:



UNDER THE GUN IN THE DATA CENTER

```yaml
- name: eth0 configuration - ceph communication network
  delegate_to: "{{ node.ceph_network.ip_address }}"
  when: (cust_network_ping.rc !=0) or (gateway.state == "update")
  block:
    - name: Add a new network interface - Primary gateway interface
      command: nmcli con add con-name auto_eth0 ifname eth0 type ethernet ip4 {{ node.ceph_network.ip_address }}/{{ node.ceph_network.ip_cidr }} gw4 {{ node.ceph_network.ip_gateway }}
      ignore_errors: yes  # Ignore errors if the interface already exists

    - name: Add static routes - Primary gateway interface
      command: nmcli con modify auto_eth0 +ipv4.routes "{{ node.ceph_network.route }} {{ node.ceph_network.ip_gateway }} 0"

    - name: Apply the new configuration - Primary gateway interface
      command: nmcli con up auto_eth0

- name: eth1 configuration- customer communication network
  delegate_to: "{{ node.ceph_network.ip_address }}"
  when: cust_network_ping.rc !=0
  block:
    - name: Add a new network interface - customer network
      command: nmcli con add con-name auto_eth1 ifname eth1 type ethernet ip4 {{ node.cust_network.ip_address }}/{{ node.cust_network.ip_cidr }} gw4 {{ node.cust_network.ip_gateway }}
      ignore_errors: yes  # Ignore errors if the interface already exists

    # - name: Add static routes - SMB gateway
    #   command: nmcli con modify auto_eth1 +ipv4.routes "{{ node.cust_network.route }} {{ node.cust_network.ip_gateway }} 0"

    - name: Apply the new configuration - SMB gateway
      command: nmcli con up auto_eth1

- name: eth2 configuration
  delegate_to: "{{ node.ceph_network.ip_address }}"
  when:
    - (cust_network_ping.rc !=0) or (gateway.state == "update")
    - dev_mode | bool
  block:
    - name: Add a new network interface
      command: nmcli con add con-name auto_eth2 ifname eth2 type ethernet
      register: nmcli_output
      ignore_errors: yes  # Ignore errors if the interface already exists

    - name: Add static routes
      command: nmcli con modify auto_eth2 +ipv4.routes "0.0.0.0/0 192.168.0.1 0"

    - name: Apply the new configuration - dev interface
      command: nmcli con up auto_eth2
```

```yaml
vm_config:
  memory: 12288
  cores: 4
  disk_size: "30G"
  disk_storage: "pve_rbd"
  username: "rocky"
  password: "password"
  agent: false
  os_type: "rocky"

pve_nodes:
  - gw01
  - gw02

pve_credentials:
  api_endpoint: ""
  api_token_id: ""
  api_secret: ""

ssh_private_key_path: "/opt/vm-config/id_rsa"

ssd_dir_name: "ssd"
hdd_dir_name: "hdd"

cephfs_hdd_pool: "cephfs_data_hdd"
cephfs_ssd_pool: "cephfs_data_ssd"

gateways:
    - uuid: 1001
      state: present
      datacenter_id: "ld71"
      hypervisor_id: "pm1"
      comments: "45Drives Ltd."
      share_type: smb
      ssd_quota: "1TB"
      hdd_quota: "2TB"
      nodes:
        - ceph_network:
            ip_address: 10.17.122.26
            ip_gateway: 10.17.122.25
            ip_cidr: 30
            bridge: vmbr1 # optional, defaults to vmbr0
            #tag: 1111 # optional, defaults to -1
            #rate: 1000  # optional, default to -1 units are in MiB/s
            route: "10.17.122.0/24"
          cust_network:
            ip_address: 10.17.124.10
            ip_gateway: 10.17.124.9
```

7,380

Number of individual benchmark runs completed in lead up to validation.

1,015

Estimated number of cups of coffee consumed during the design phase of this project.

9,458

Number of KM's travelled deploying and configuring the solution for customer.

FUTURE PLANS FOR THE PROJECT

- Moving from virtualized SMB to a containerized solution with Kubernetes as orchestration

- Deploying a more robust automation solution for multi-MDS