Is PostgreSQL On My Raspberry Pi Faster Than Your Cloud PostgreSQL Instance

PGDay Lowlands - 2026

chris@nexteam.co.uk

https://nexteam.co.uk

Hello!

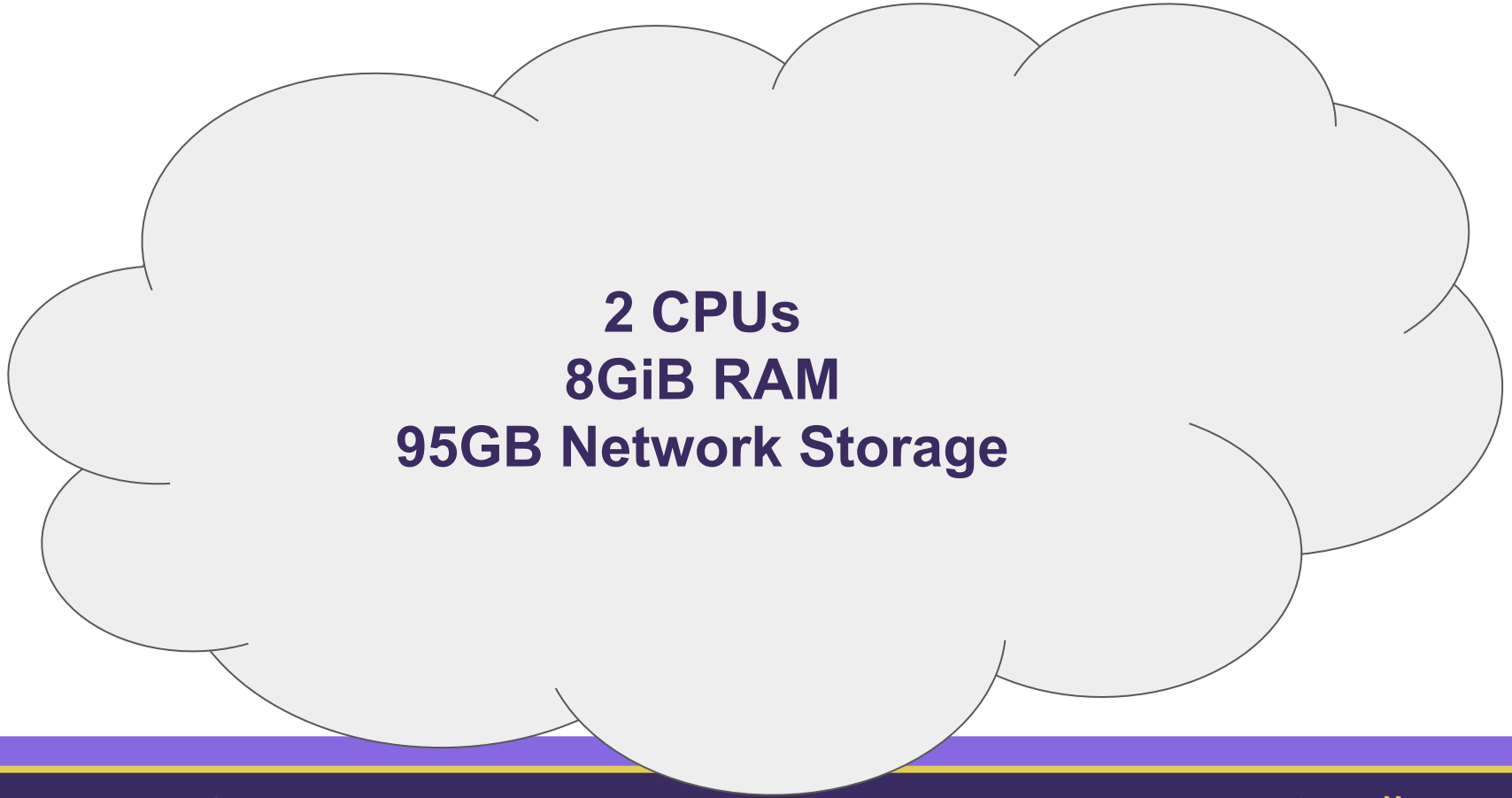
- I’m Chris, I run Nexteam, we solve tech problems for people
  - A passion for electronics (EE is my degree)
  - IT jack of all trades, an Open Source specialist
- Been using PostgreSQL ~20 years, lots of projects:
  - Website search engine, postal address search, GIS / mapping, service directory, smart energy analytics, IOT, TV & VoD catalogues, booking / subscriptions
- I’ve built stuff in the cloud
- I’ve built stuff ‘on prem’ (underrated)
- Oddly, I’ve migrated stuff from the cloud and to the cloud

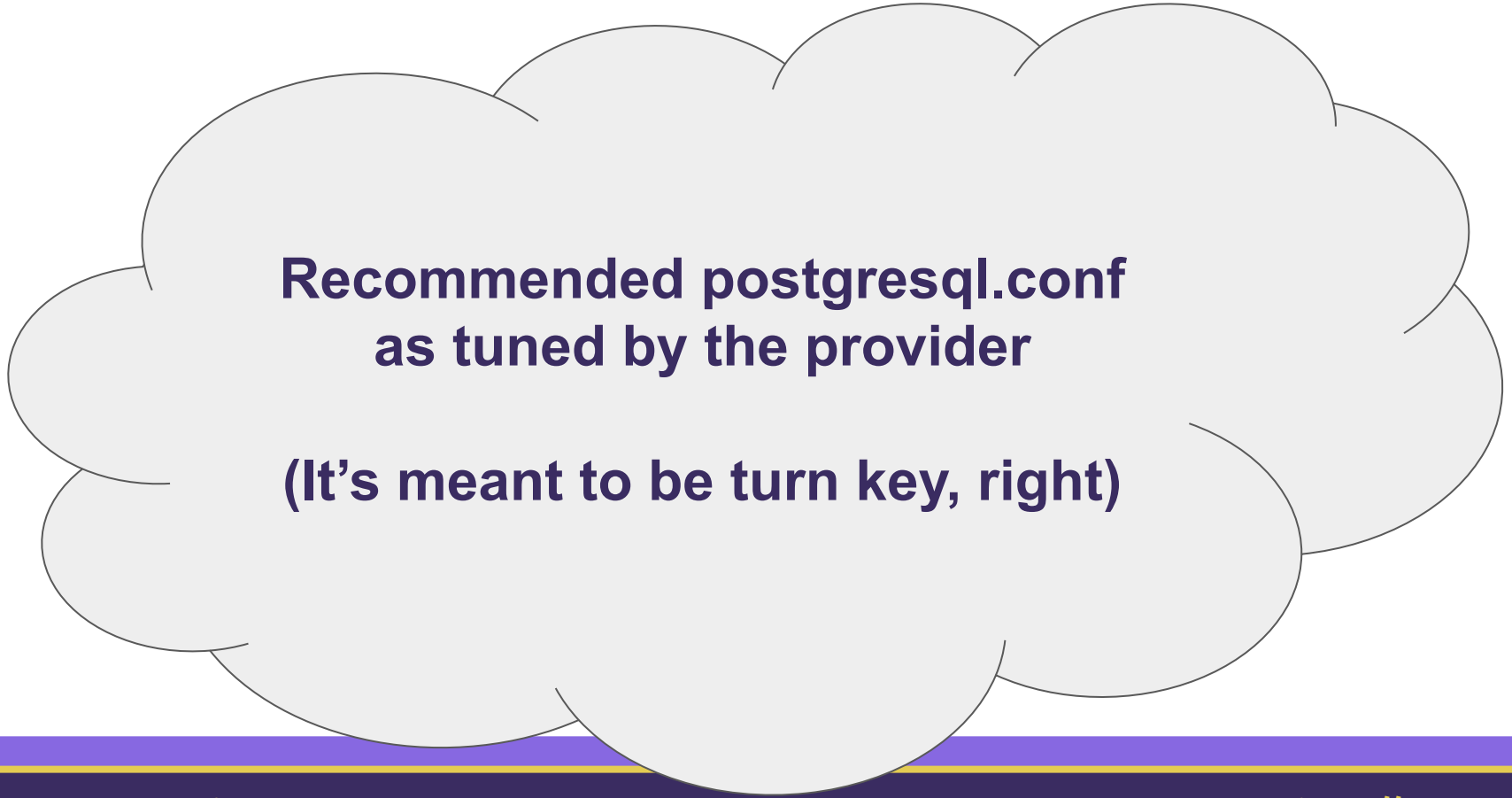
“This is your last chance. You've got to fight for your goddamn life if you want to see Trinity again.”

The Contenders!

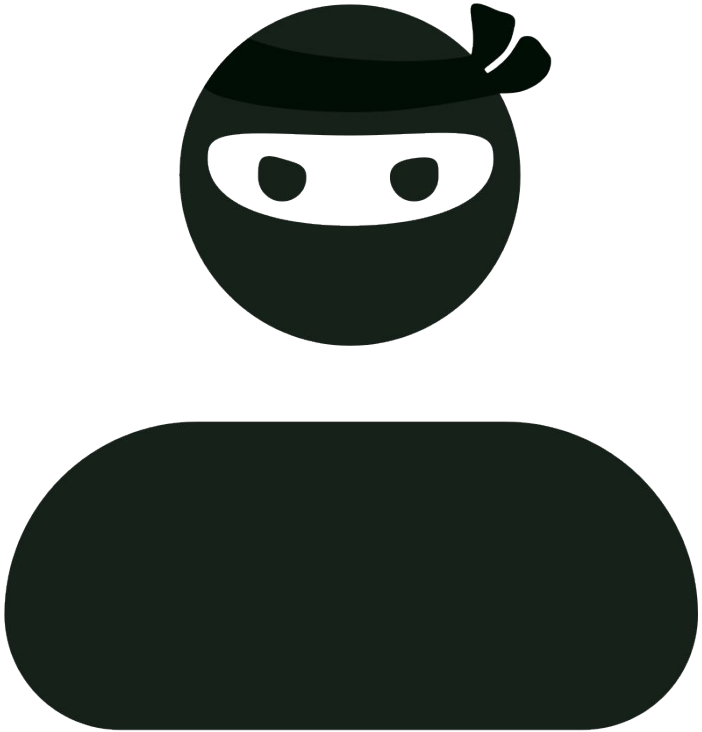
Corner A


Managed
PostgreSQL
Instance
(public cloud)

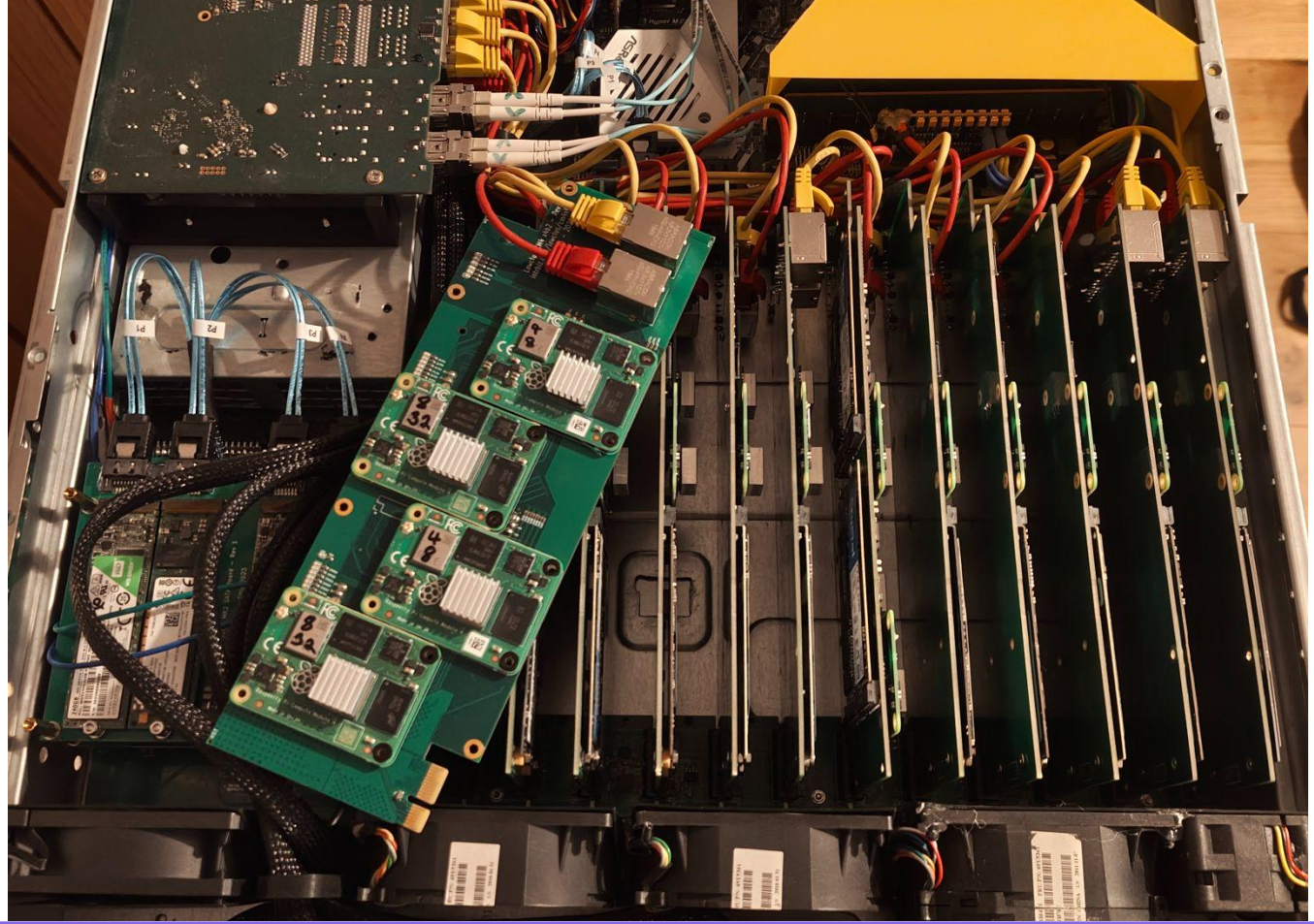

2 CPUs
8GiB RAM
95GB Network Storage

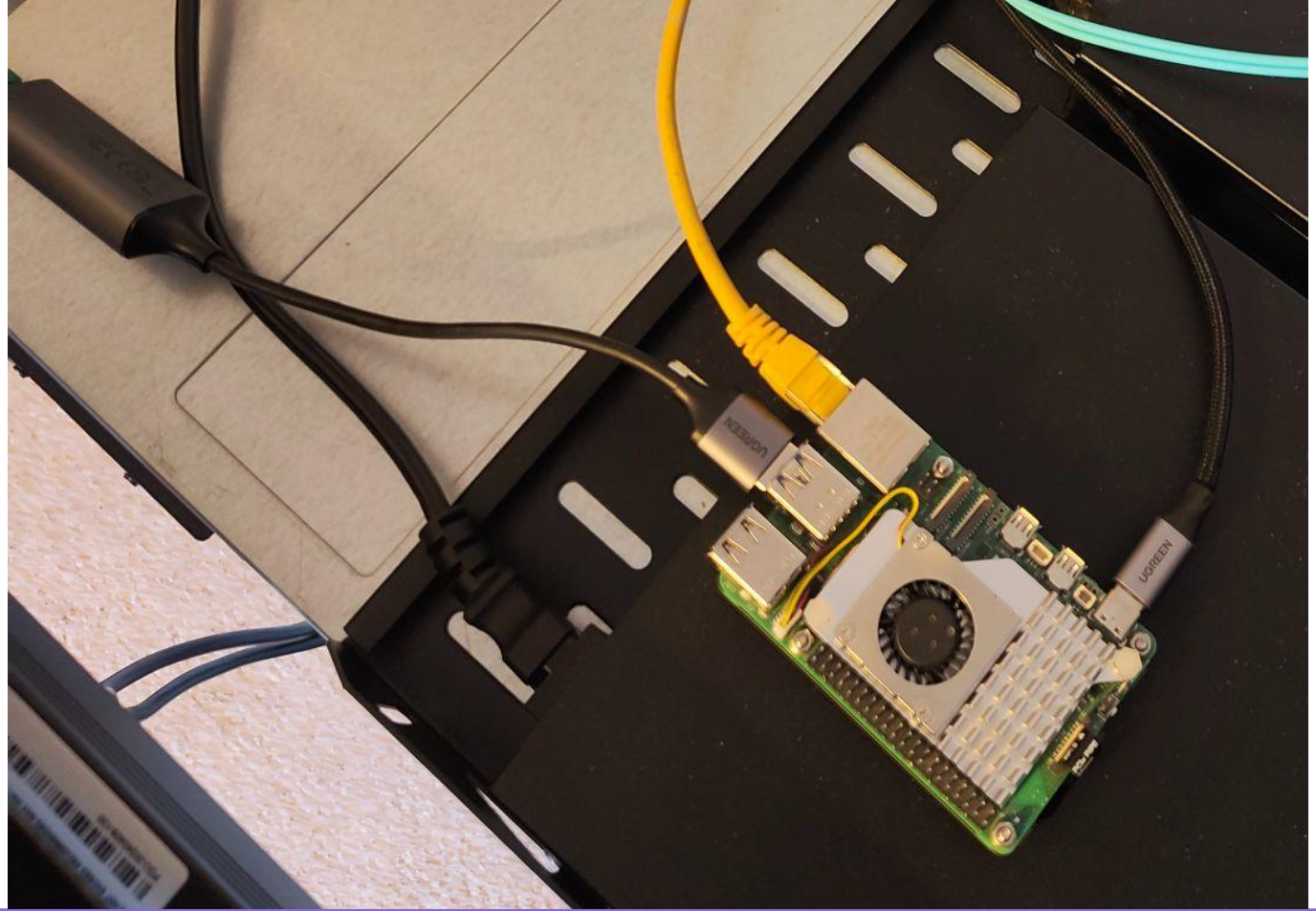

Recommended postgresql.conf
as tuned by the provider
(It’s meant to be turn key, right)

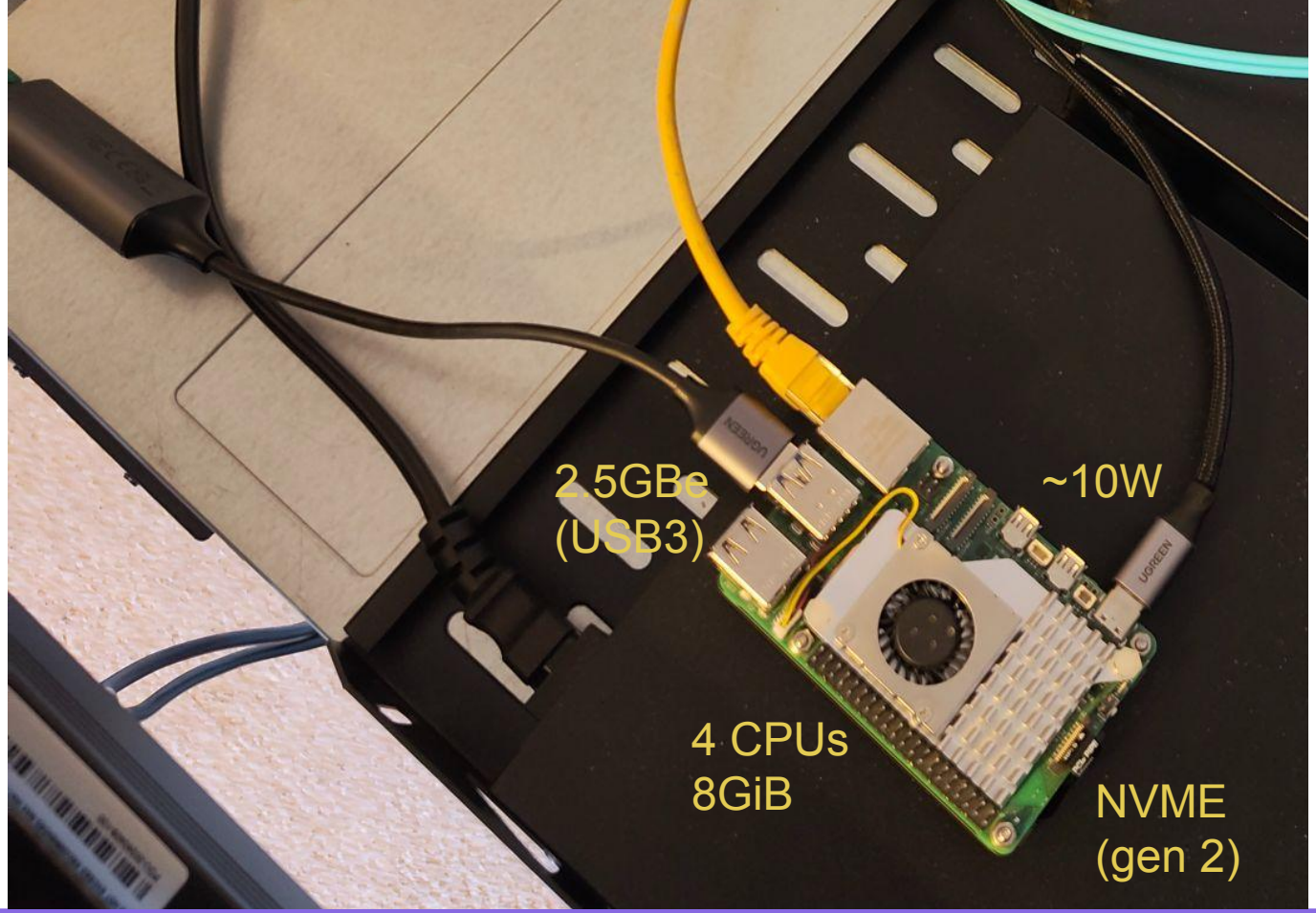
Corner B



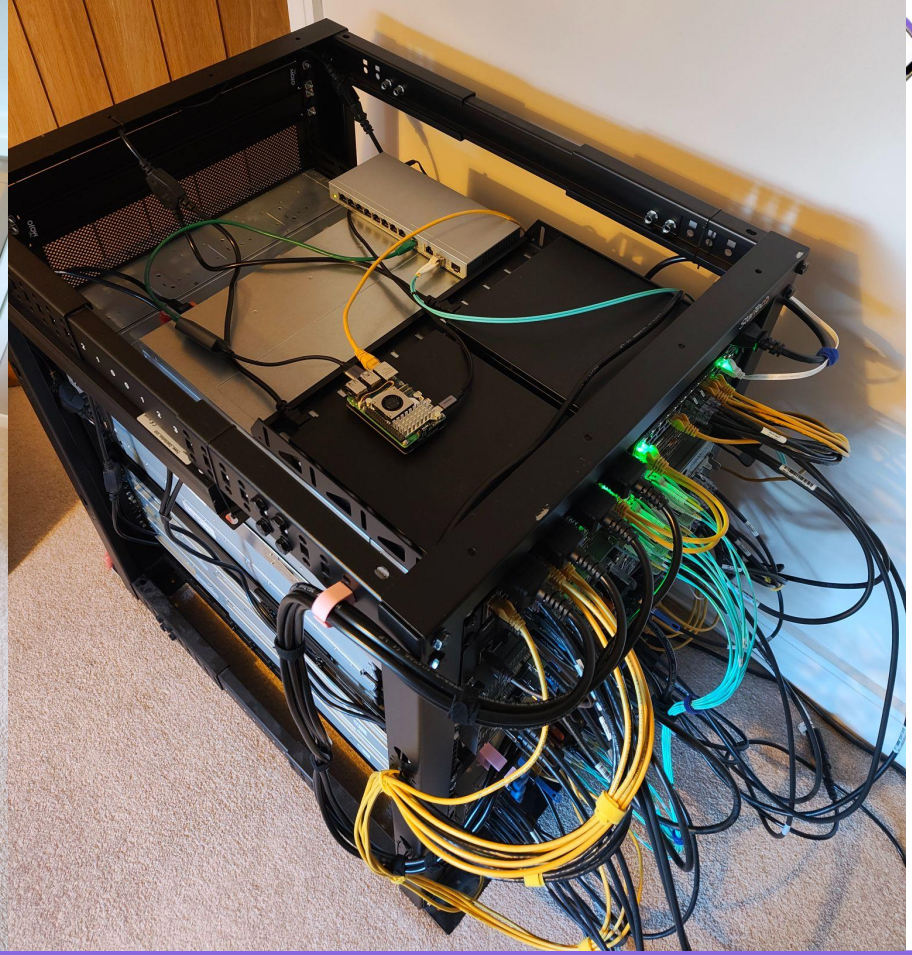

THE SKY TRADES
TAB
THE SUN FOR THE MOON




UGREEN


2 5GBe
(USB3)
~10W
4 CPUs
8GiB
NVME
(gen 2)


PCIE PIPE//O
ADDON
RPI 5
PIMORONI
RPi 5 PCIe
M.2 M-KEY
1x PCIe 2|3
2230
2242
2260
2280
+5V
NVMe BASE
PIMORONI
Made in Sheffield, UK
pimoroni.com/



chris@nexteam.co.uk

https://nexteam.co.uk

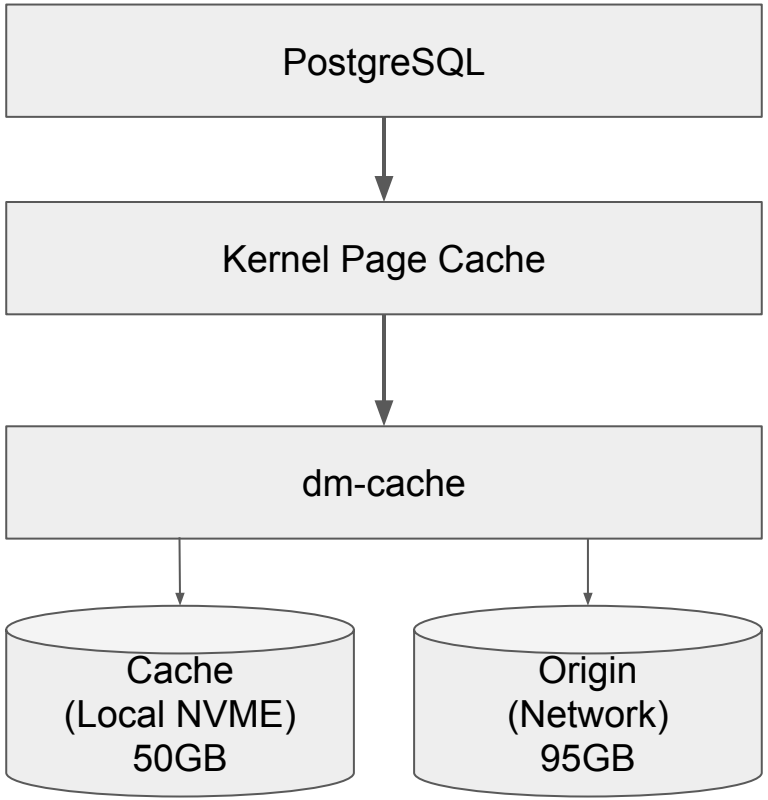
RPI Setup

```
pvcreate /dev/nvme0n1
vgcreate lcache /dev/nvme0n1
lvcreate -n local -L 95G lcache /dev/nvme0n1
mkfs.xfs -f -d agcount=32 /dev/mapper/lcache-local

ALTER SYSTEM SET shared_buffers = "2048MB";
ALTER SYSTEM SET checkpoint_timeout = "2min";
ALTER SYSTEM SET max_wal_size = "5GB";
```

I promised,
that,
one Interesting Trick...

lvm-cache
(dm-cache)


PostgreSQL
Kernel Page Cache
dm-cache
Cache
(Local NVME)
50GB
Origin
(Network)
95GB

RPI Setup

```
pvcreate /dev/nvme0n1
pvcreate /dev/rbd0
vgcreate lcache /dev/rbd0
vgcreate lcache /dev/nvme0n1
lvcreate -n main -L 95G lcache /dev/rbd0
lvcreate -n fast -L 50G lcache /dev/nvme0n1
```

RPI Setup

```
lvconvert --type cache \
  --cachemode writethrough \
  --cachevol fast \
  lcache/main

mkfs.xfs -f -d agcount=32 /dev/mapper/lcache-main
```

The Test!

Why Bother Benchmarking

- Data
  - Assumption makes an ass of you and me
- Validate
  - Configuration changes, hardware choices
- Compare
  - Across providers or hardware

Benchmarking Rigor

- Fair
  - The same test for each situation
- Repeatable
  - Same experiment each time
- Stressing
  - For a reasonable amount of time

Keeping It Simple - pgbench

- Built In
  - Comes out of the box with PostgreSQL
- Easy
  - Simple to run, and a simple output
- Useful
  - Update and Select only workloads

pgbench - Initialise

```
pgbench \
    -i \
    -s ${scale} \
    -F 80 \
    --foreign-keys \
    -h $HOST \
    -U bench \
    bench
```

pgbench - Read / Write

```
pgbench -j 4 \
   -M extended \
   -c $c \
   -T 60 \
   -h $HOST \
   -U bench \
   bench
```

pgbench - Read Only

```
pgbench -j 4 \
   -M extended \
   -c $c \
   -S \
   -T 60 \
   -h $HOST \
   -U bench \
   bench
```

pgbench - Output

```
pgbench (18.4, server 18.6 (Debian 18.6-1.pgdg13+2))
transaction type: <builtin: TPC-B (sort of)>
scaling factor: 250
query mode: extended
number of clients: 8
number of threads: 4
maximum number of tries: 1
duration: 60 s
number of transactions actually processed: 122471
number of failed transactions: 0 (0.000%)
latency average = 3.919 ms
initial connection time = 12.837 ms
tps = 2041.102676 (without initial connection time)
```

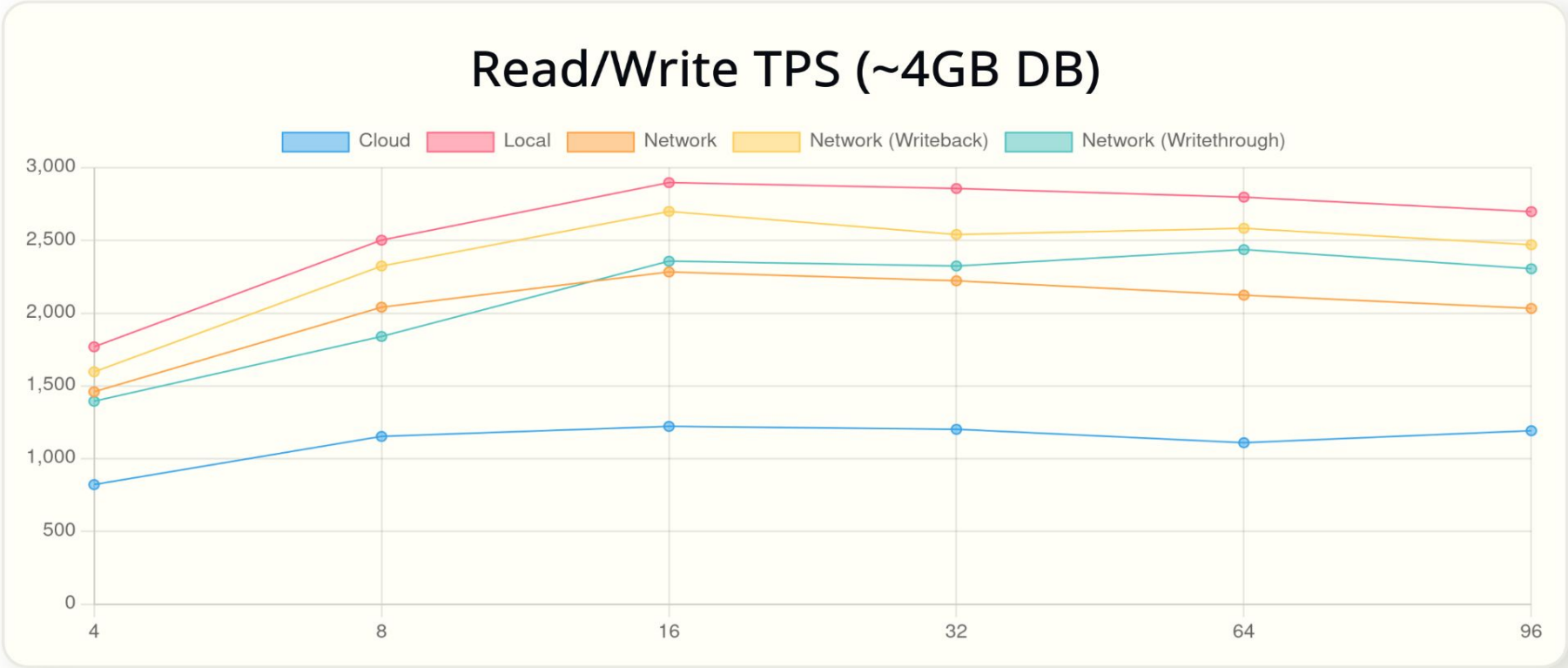
pgbench - Automated

```
#!/bin/sh
for scale in 250 1000; do
 pgbench -i -s ${scale} -F 80 --foreign-keys -h $HOST -U bench bench
 for r in 1; do
  for c in 4 8 16 32 64 96; do
   pgbench -j 4 -M extended -c $c -T 60 -h $HOST -U bench bench
  done
  for c in 4 8 16 32 64 96; do
   pgbench -j 4 -M extended -c $c -S -T 60 -h $HOST -U bench bench
  done
 done
done
```

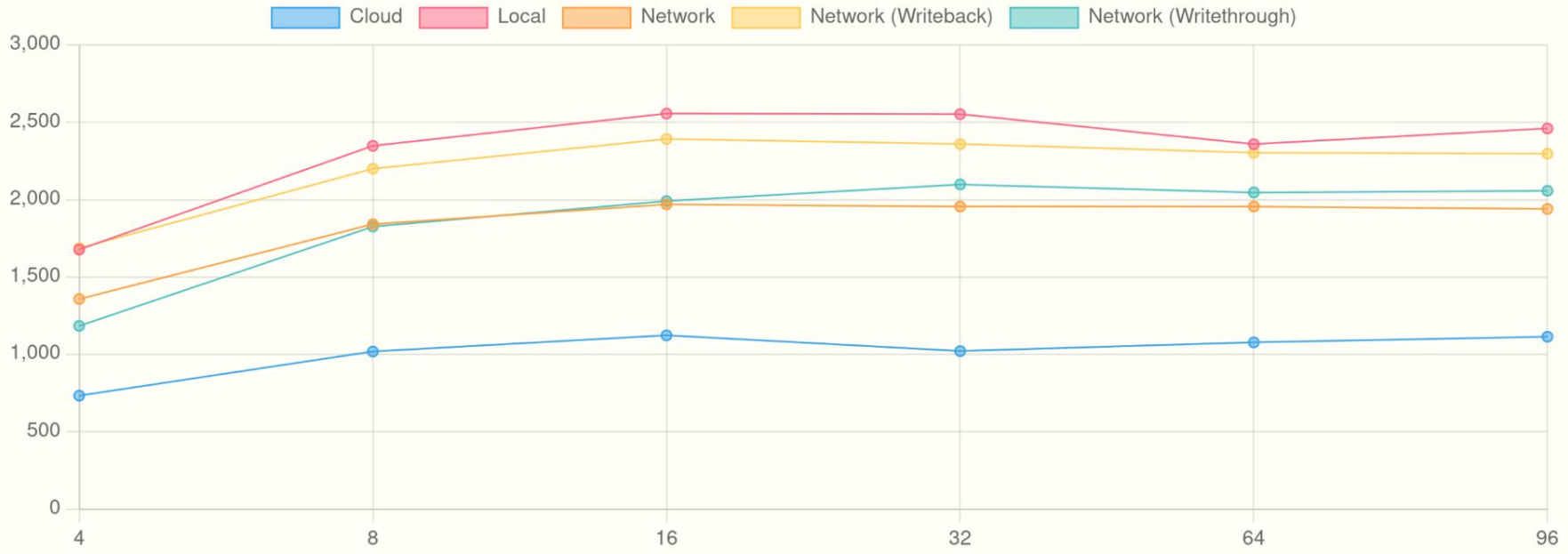
The Test

- ~4GB DB & ~18GB DB
  - Run pgbench using scale: 250 and 1000
- Multiple Runs
  - Scale up the number of concurrent clients
- Read / Write & Read Only
  - Test both update and select only performance

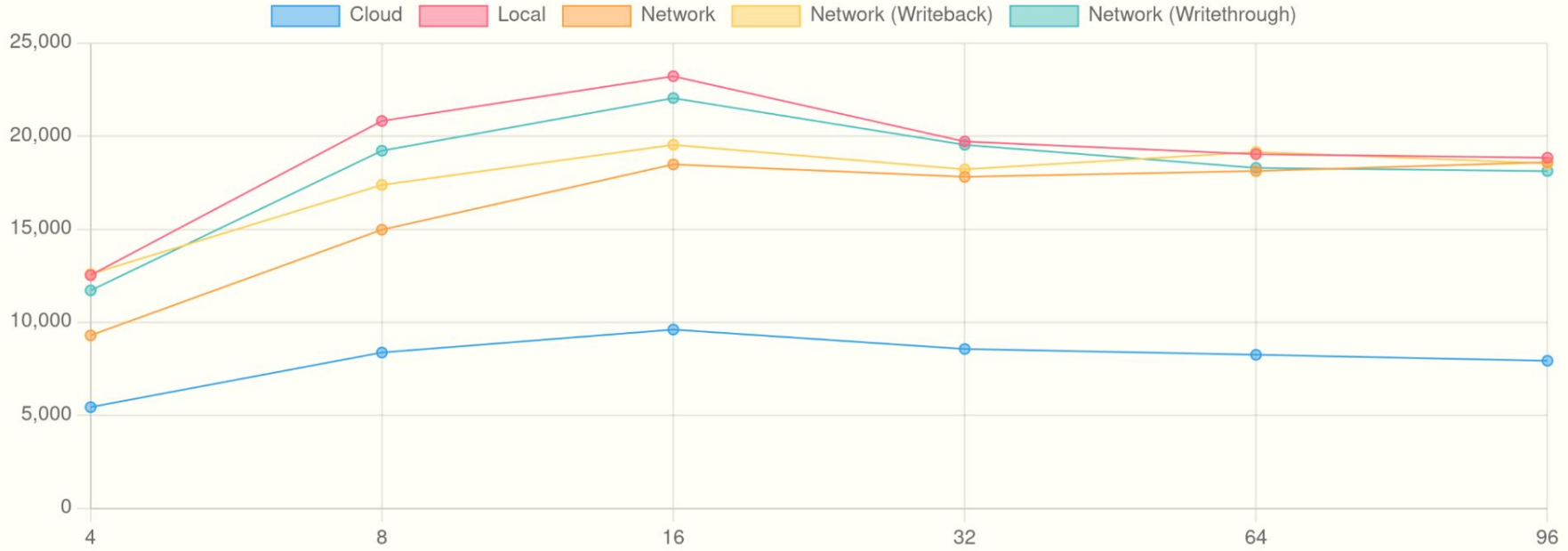
The Results!


Read/Write TPS (~4GB DB)
Cloud
Local
Network
Network (Writeback)
Network (Writethrough)
3,000
2,500
2,000
1,500
1,000
500
0
4
8
16
32
64
96

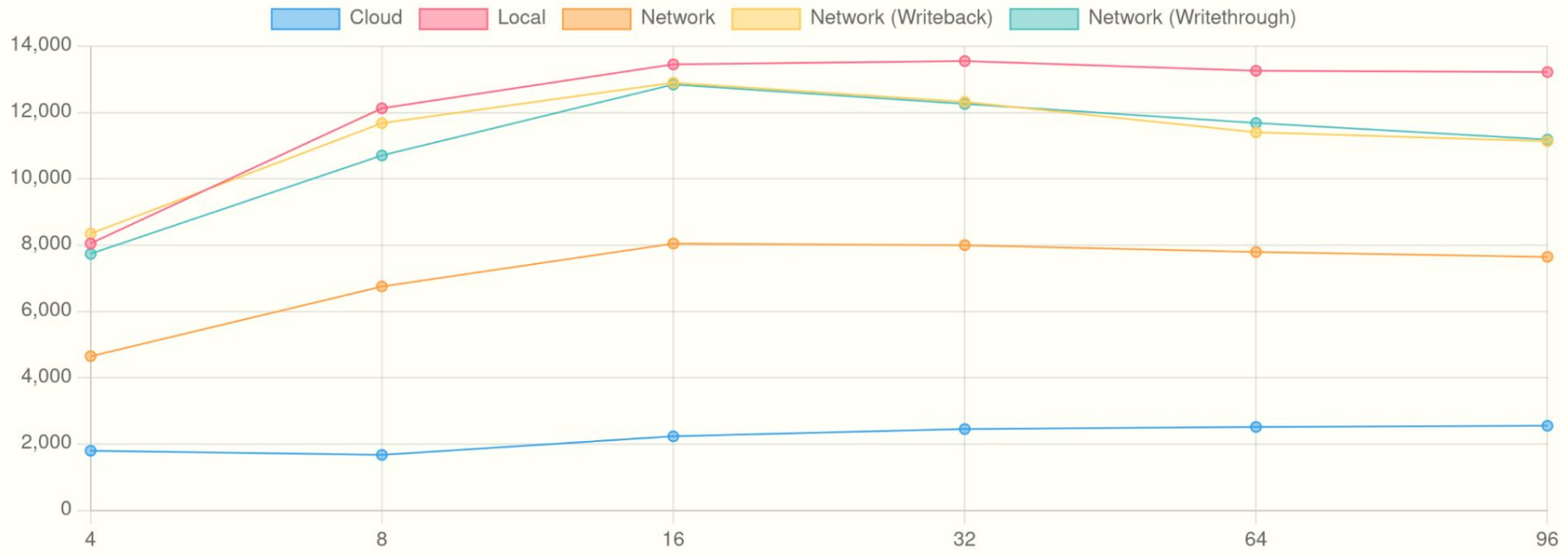
Read/Write TPS (~18GB DB)



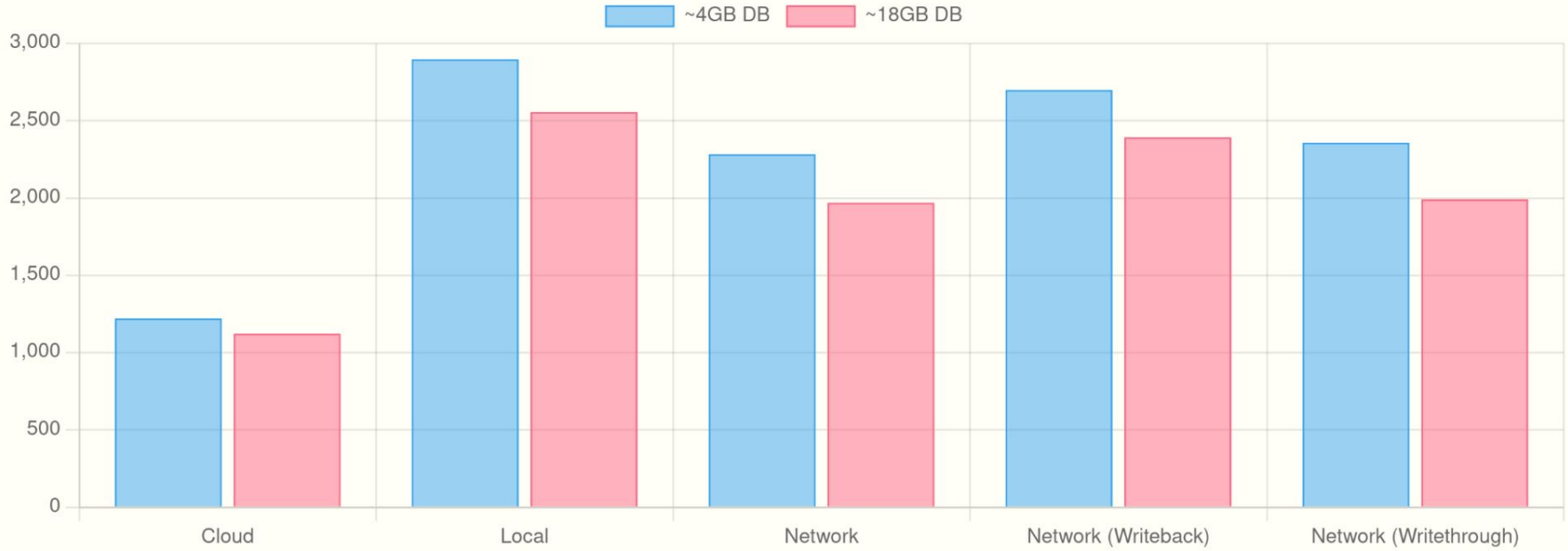
Read Only TPS (~4GB DB)

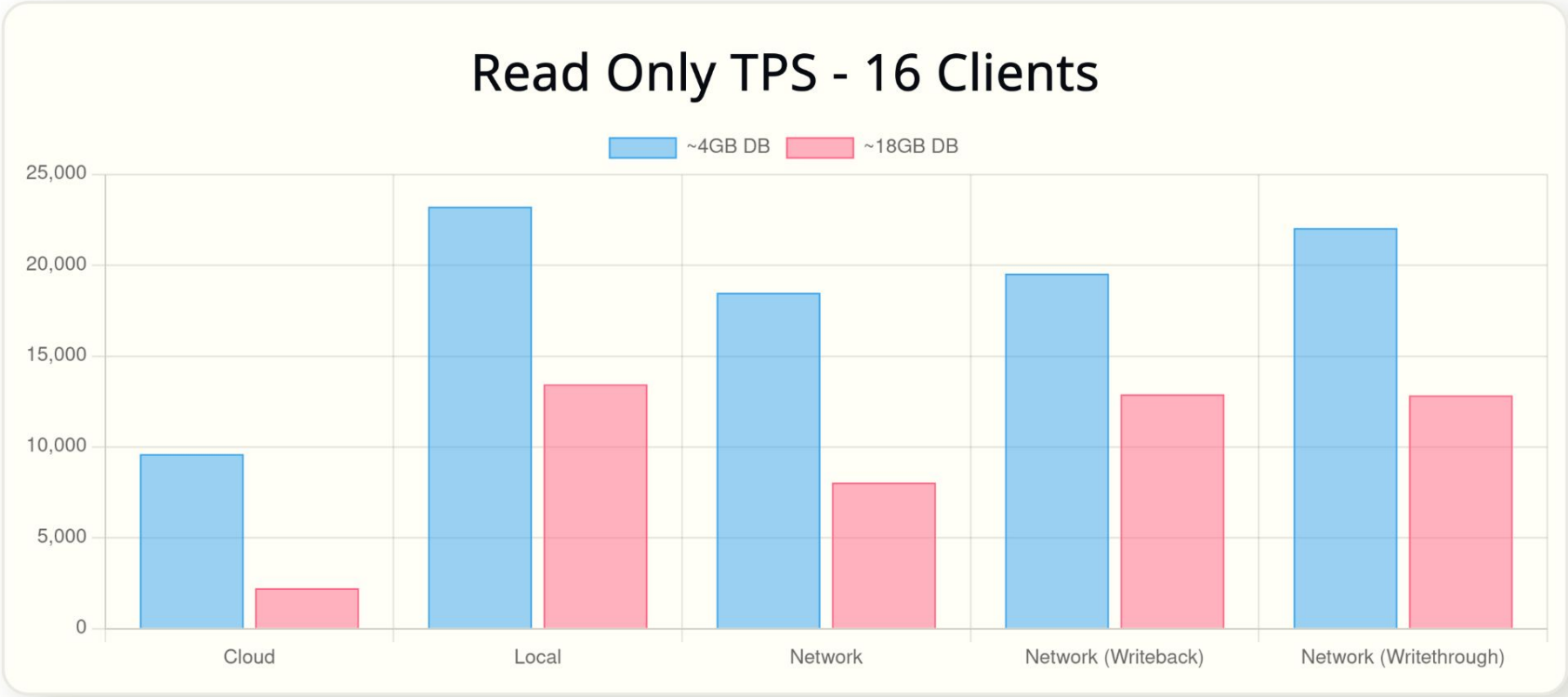


Read Only TPS (~18GB DB)



Read/Write TPS - 16 Clients




Read Only TPS - 16 Clients
~4GB DB
~18GB DB
25,000
20,000
15,000
10,000
5,000
0
Cloud
Local
Network
Network (Writeback)
Network (Writethrough)

So, Yes!

So, Yes!
PostgreSQL On My Raspberry Pi IS Faster Than SOME Cloud PostgreSQL Instances

So, Yes!
82% to 43% faster!?!?!

At What Cost?

Cloud:
~ $124 per month

= £92 per month

RPI: £63

NVME Adapter: £11

USB3 2.5GbE: £16

NVME: £55 (1TB)

= £145 - At the time

RPI CM5: £123

NVME + 2.5GbE: £25

NVME: £159

= £307

1U Colo: £45 per month

1U Colo: £45 per month

= 10 RPIs (easily)

IE:
~6 Month
ROI
(worst case)

Cloud: 8p per TPS
Vs
RPI: 4p per TPS

Granted,
borderline,
reductio ad absurdum

But.....

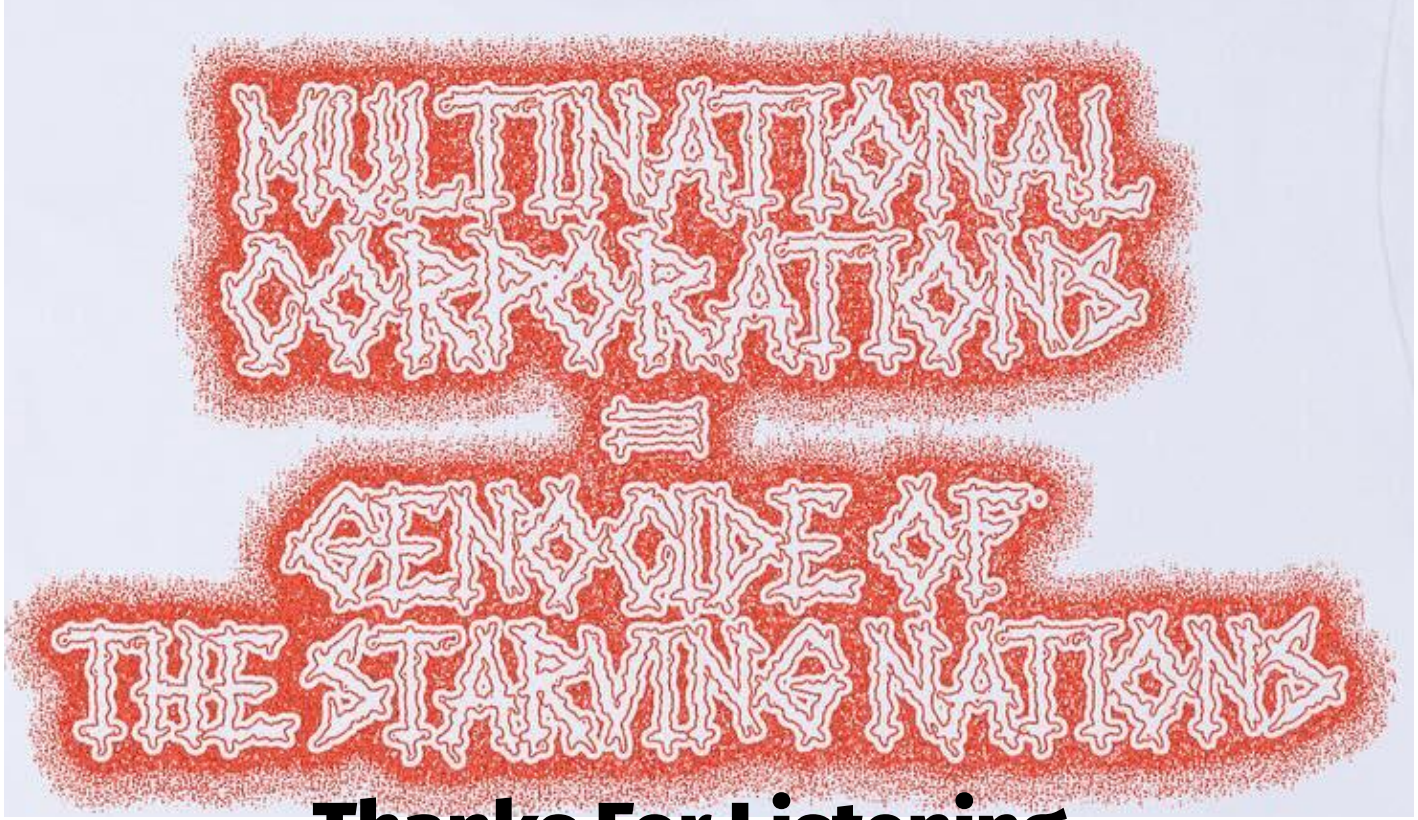
Some rules are fixed,
some,
we can bend

Apply some thought, before assuming that the cloud is the best / only solution.

Free / Open Source Software
can level the playing field,
giving all of us access
to good technology.

Outsourcing the core of your business isn't always best!

Do you want the worlds compute controlled by 4 companies?



Thanks For Listening, questions?