Meta

INFRASTRUCTURE

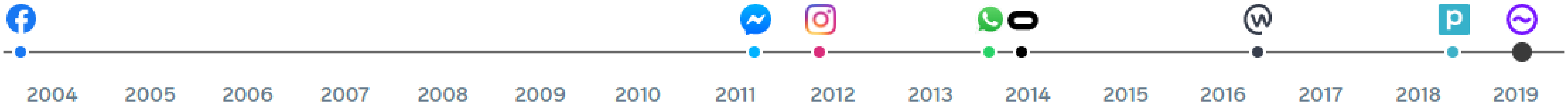
Debugging of Flash Issues Observed in Hyperscale Environment

Vineet Parekh
Hardware Systems Engineer

Venkat Ramesh
Hardware Systems Engineer

Family MAP : 3.59B

Globally, there are more than 3.59B people using Facebook, WhatsApp, Instagram or Messenger each month.

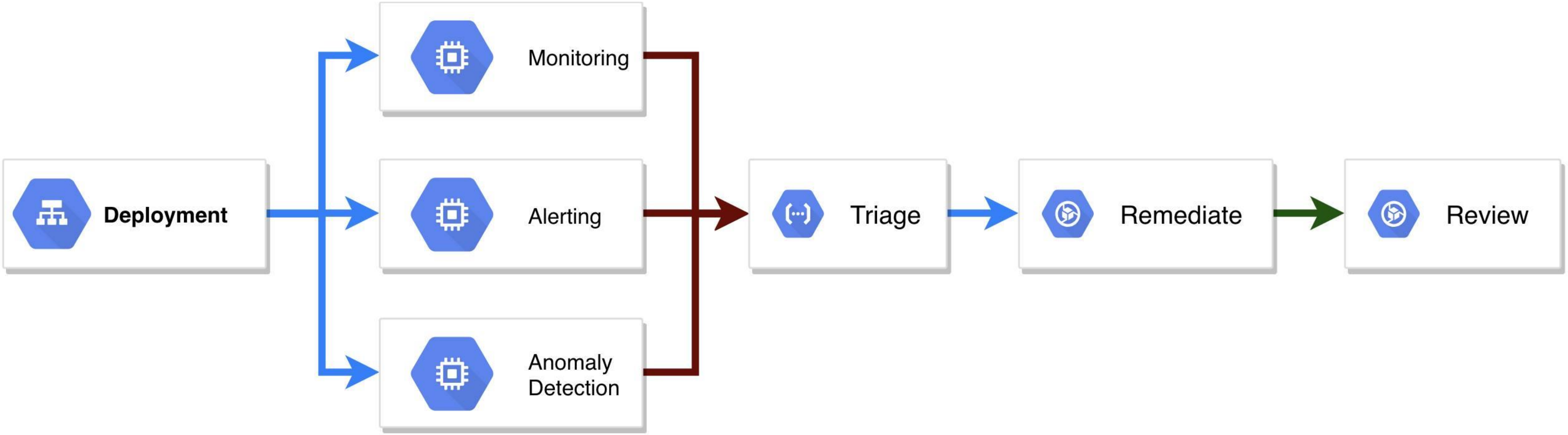


Meta

*MAP - Monthly Active People

Source: Meta Platforms Inc. Q4 2021

Flash Failure Debug Overview



Data Collection for Flash Reliability

Periodic Collection - At Scale (Automated)

SMART Attributes

DMESG Logs

Latency Monitoring Log (based on OCP NVMe Cloud SSD V2.0)

SMART Cloud Health Log (0xCO)
(Based on NVMe Cloud SSD V1.0)

- Nand Statistics
- PCIe Statistics
- Health Statistics

Vendor & model independent logs can be captured efficiently by automation

Non- Periodic – Individual Drive (Manual)

Telemetry Logs

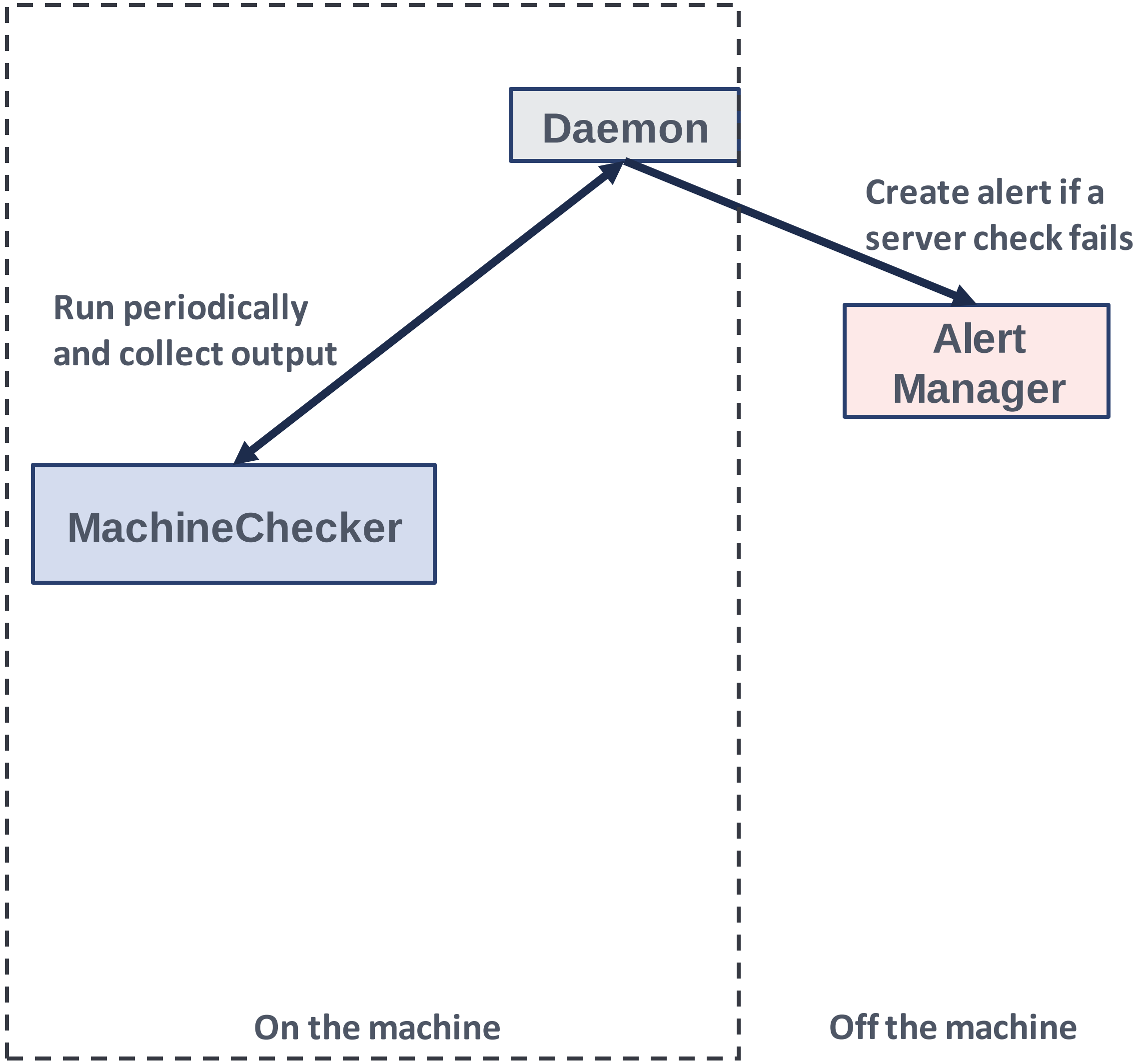
Drive Event Logs

More Drives = Requires More Resources

Hardware Remediation @ Scale

Failure Detection – MachineChecker

- Runs hardware checks periodically
- Host ping, memory, CPU, NIC, dmesg, S.M.A.R.T., power supply, SEL, etc.

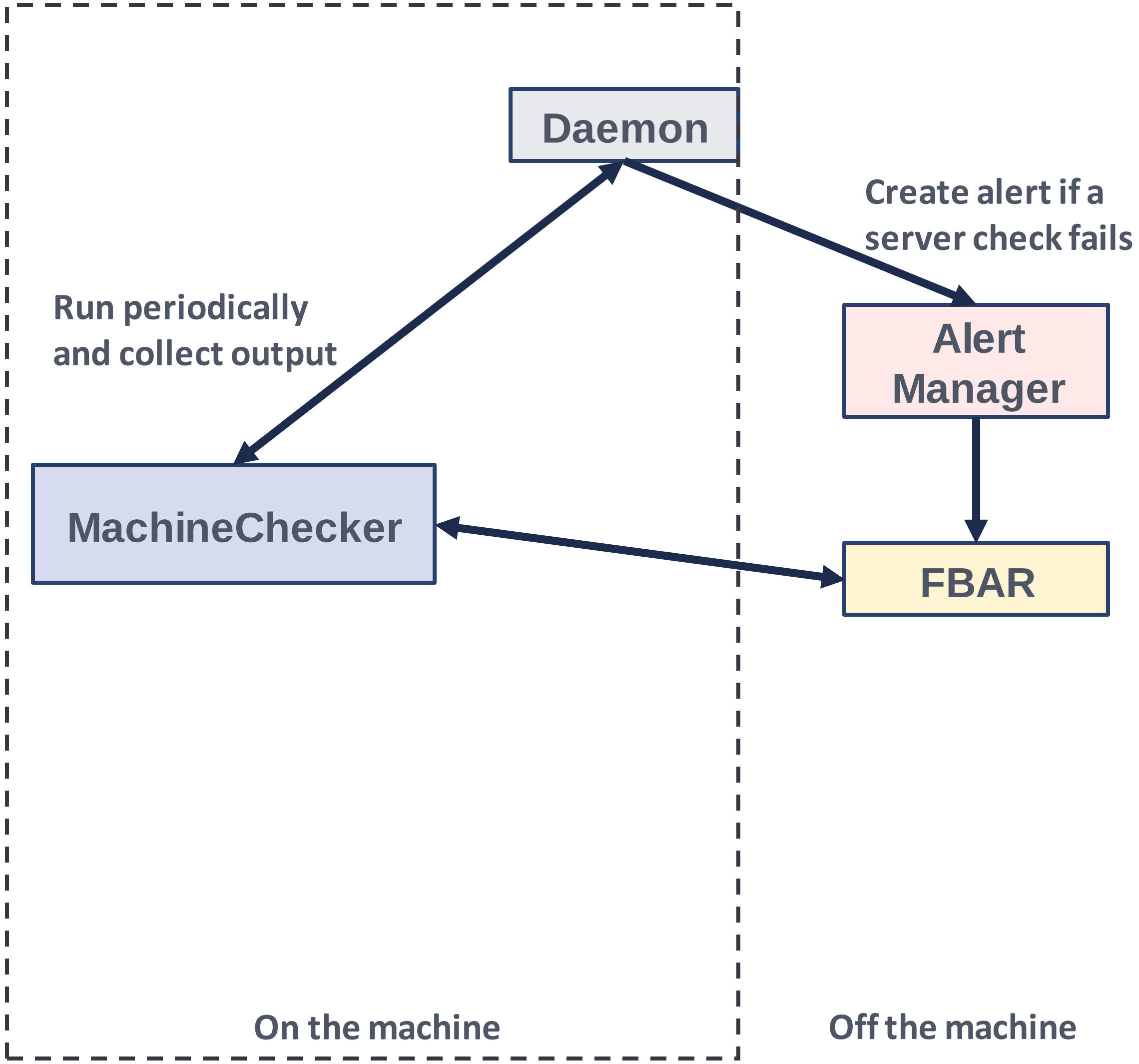

Daemon
Create alert if a server check fails
Run periodically and collect output
Alert Manager
MachineChecker
On the machine
Off the machine

Hardware Remediation @ Scale

Failure Detection – MachineChecker

Failure Digestion – FBAR

- Facebook Auto Remediation
- Picks up hardware failures, process logged information, and execute custom-made remediation accordingly


Daemon
Create alert if a server check fails
Run periodically and collect output
Alert Manager
MachineChecker
FBAR
On the machine
Off the machine

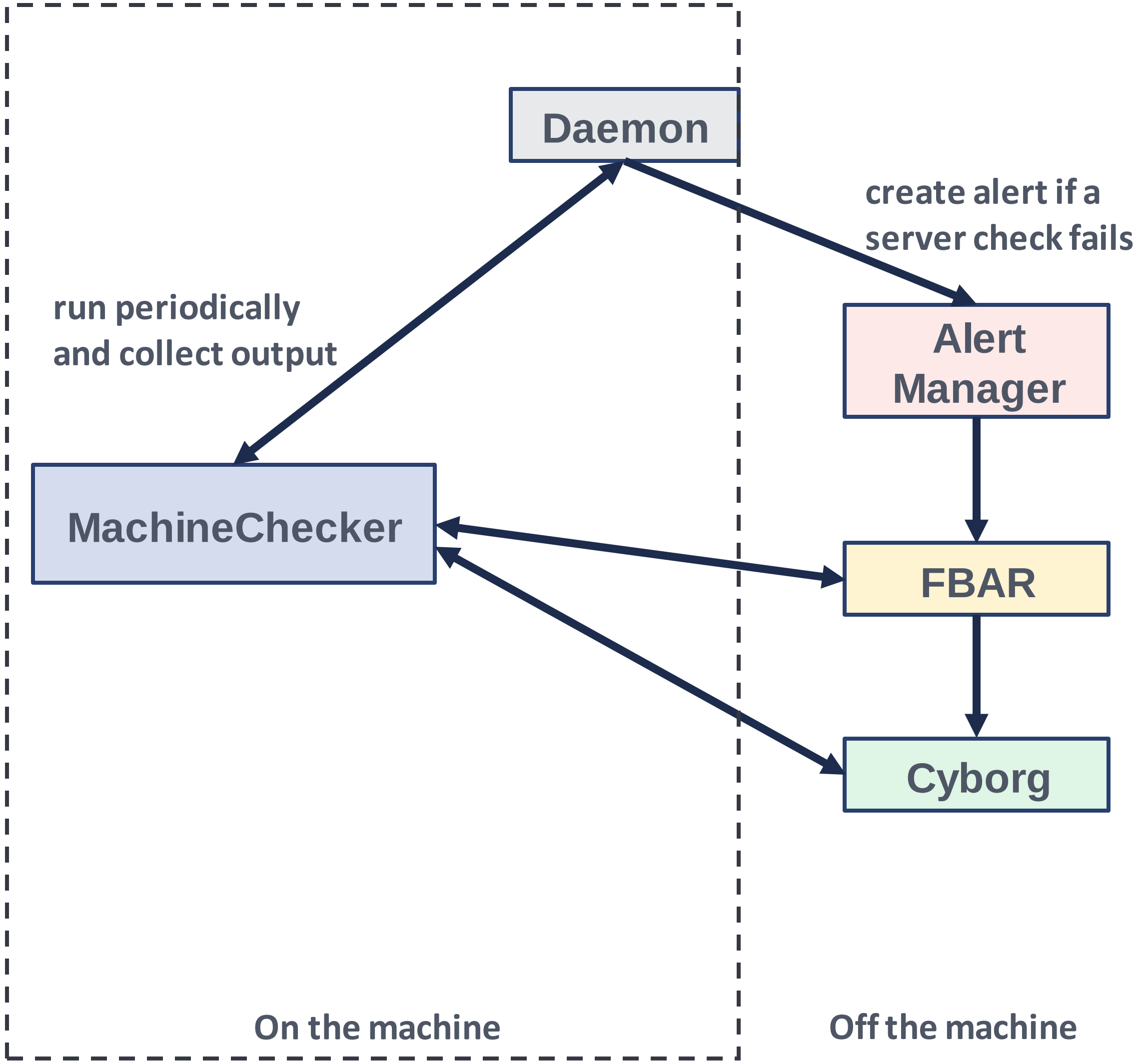
Hardware Remediation @ Scale

Failure Detection – MachineChecker

Failure Digestion – FBAR

Low-Level Software Fix – Cyborg

- Handles low-level software fixes such as firmware update and reimaging



Hardware Remediation @ Scale

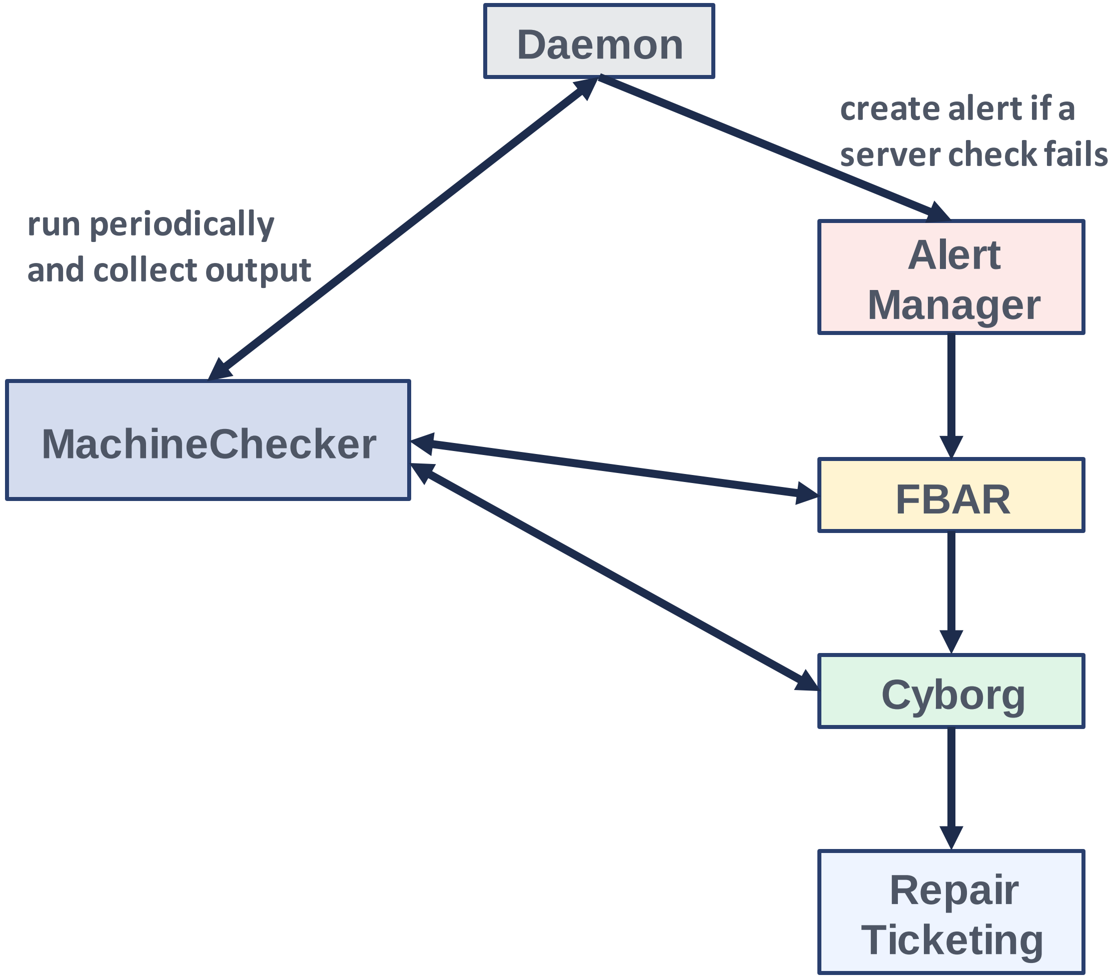
Failure Detection – MachineChecker

Failure Digestion – FBAR

Low-Level Software Fix – Cyborg

Manual Fix – Repair Ticketing

- Creates repair tickets for DC technicians to swap SSD
- Provides detailed logs throughout the auto-remediation
- Logs repair actions for further analysis



Daemon
create alert if a server check fails
run periodically and collect output
Alert Manager
MachineChecker
FBAR
Cyborg
Repair Ticketing

Failure Types - Examples

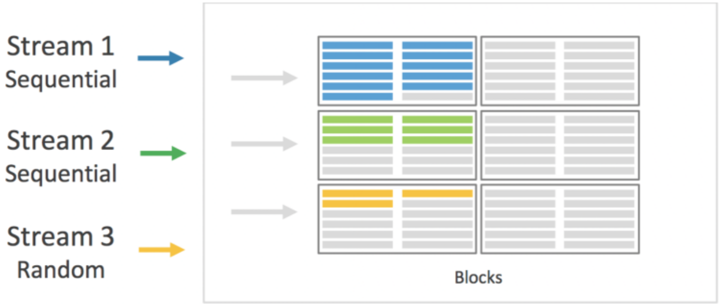
Application Level	I/O Stalls	Bandwidth Reduction	Data Corruption	Capacity Disabled
Fleet Monitoring	Endurance	SMART	Read and Write Errors	Protocol Errors

Debug Challenges

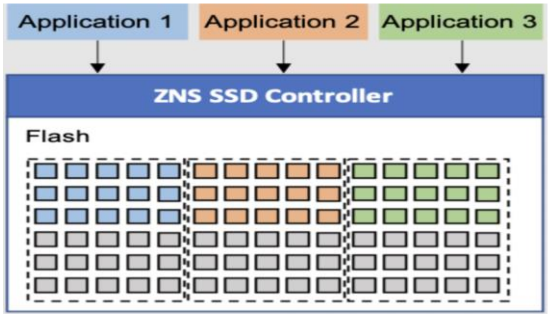
2.5inch
SATA

PCIe/NVMe Add in Card

E1.S SSD



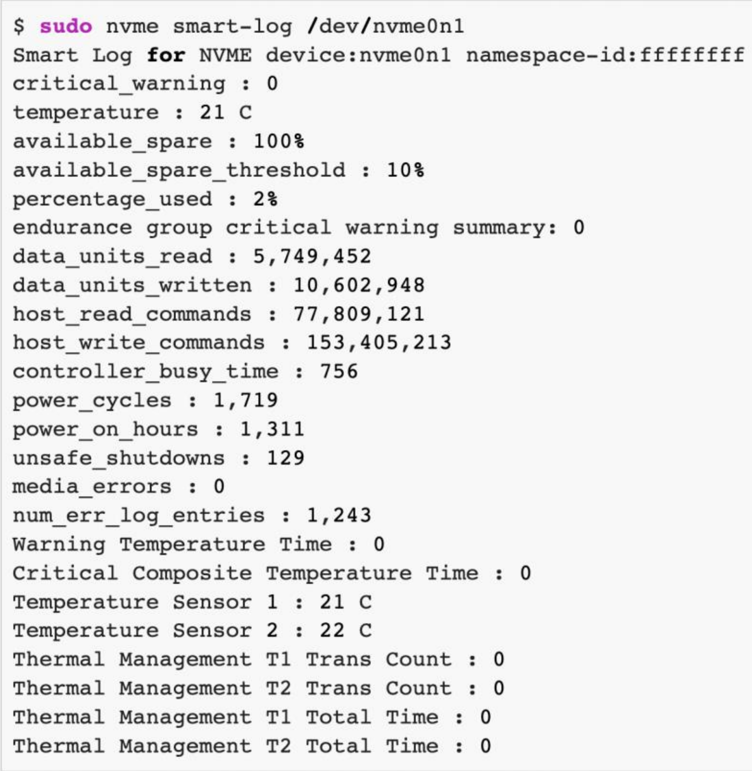
Streams



ZNS

Evolution of flash drives into complex storage system

Telemetry and SMART can help debug all problems....



```
$ sudo nvme smart-log /dev/nvme0n1
Smart Log for NVME device:nvme0n1 namespace-id:ffffffff
critical_warning : 0
temperature : 21 C
available_spare : 100%
available_spare_threshold : 10%
percentage_used : 2%
endurance group critical warning summary: 0
data_units_read : 5,749,452
data_units_written : 10,602,948
host_read_commands : 77,809,121
host_write_commands : 153,405,213
controller_busy_time : 756
power_cycles : 1,719
power_on_hours : 1,311
unsafe_shutdowns : 129
media_errors : 0
num_err_log_entries : 1,243
Warning Temperature Time : 0
Critical Composite Temperature Time : 0
Temperature Sensor 1 : 21 C
Temperature Sensor 2 : 22 C
Thermal Management T1 Trans Count : 0
Thermal Management T2 Trans Count : 0
Thermal Management T1 Total Time : 0
Thermal Management T2 Total Time : 0
```

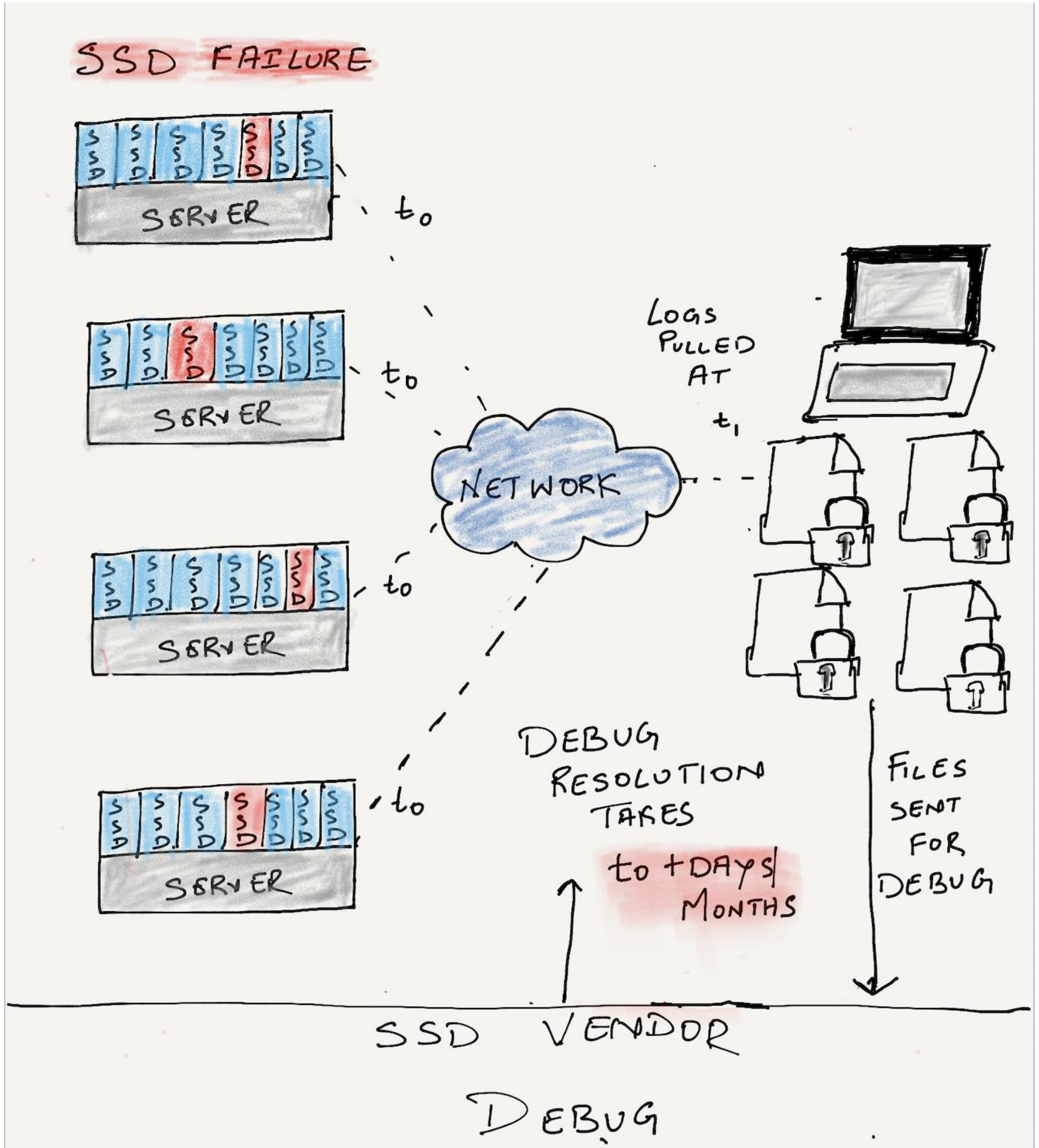
But can they????

Debug Challenges – Telemetry Is Overrated

- ‘SMART’ is not *that* Smart!
  - SMART attributes are not enough to help hyperscalers to debug SSD problems
  - Barely provides any insight into the internal condition of the drive

- Telemetry Challenges
  - Current model of telemetry log collection **does not** work at Scale
  - Hyperscalers left in dark while vendors debug/root cause
  - Long turnaround time for first level debug

Need more human readable logs for at scale debug



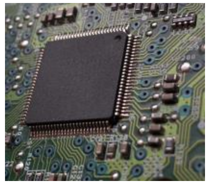
Debugging flash Issues in hyperscale environment is inefficient

Focusing on a real problem...

Latency stall – A single I/O event taking more than the expected time to complete

1 Read/Write/Trim > 1 second

Latency Stalls in SSD



Firmware or ASIC bugs

Hard to detect

Extremely difficult and long debug

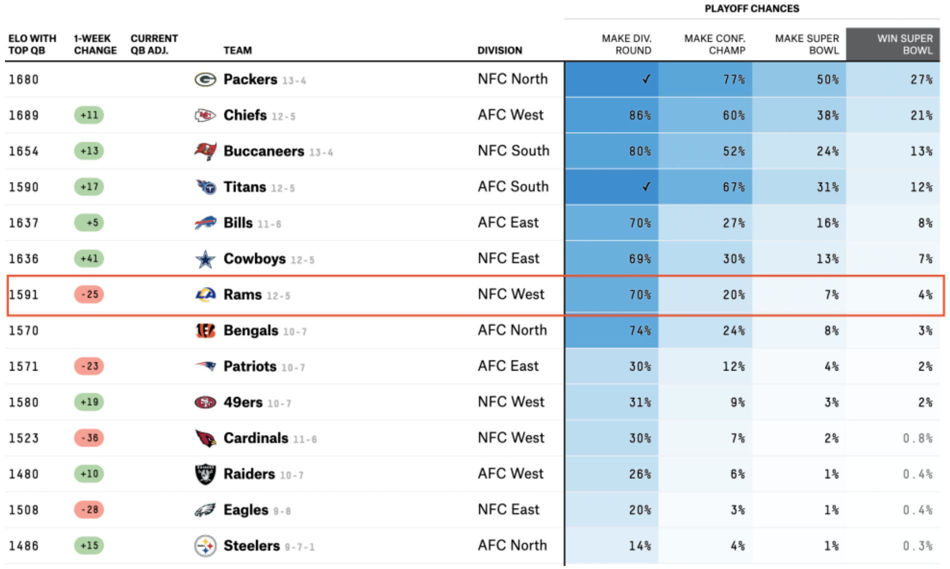
Significant impact to services

Odds per Day

Greater than 1 second - I/O stalls per day

40 distinct SSD stalls in a thousand devices

Equivalent Odds

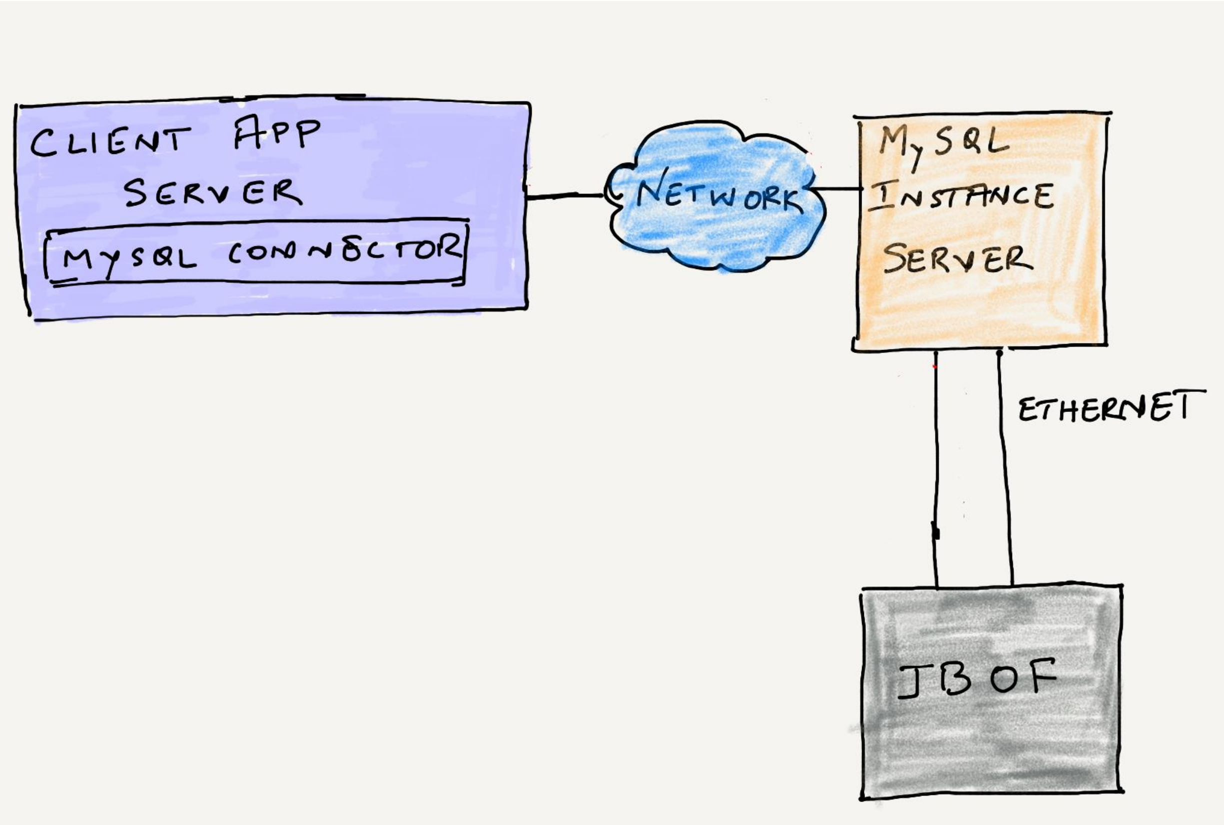


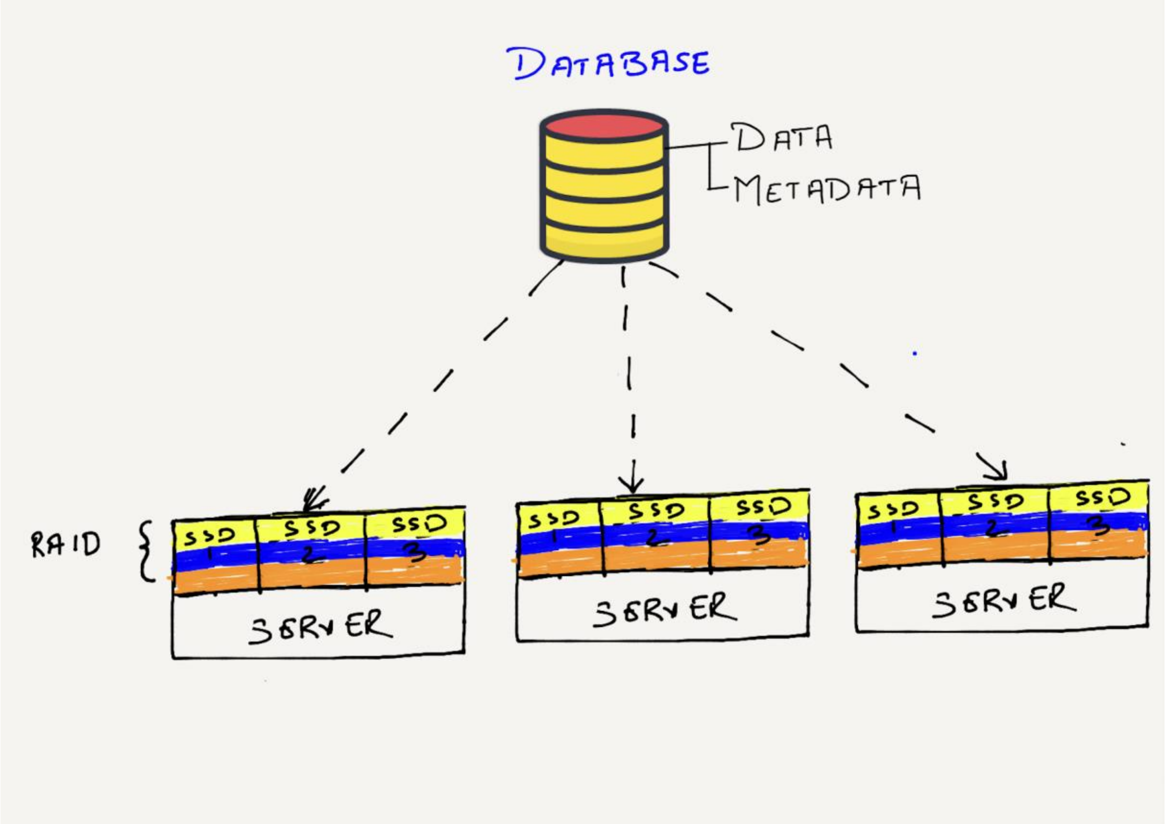
					PLAYOFF CHANCES			
ELO WITH TOP QB	1-WEEK CHANGE	CURRENT QB ADJ.	TEAM	DIVISION	MAKE DIV. ROUND	MAKE CONF. CHAMP	MAKE SUPER BOWL	WIN SUPER BOWL
1680			**Packers** 13-4	NFC North	✓	77%	50%	27%
1689	+11		**Chiefs** 12-5	AFC West	86%	60%	38%	21%
1654	+13		**Buccaneers** 13-4	NFC South	80%	52%	24%	13%
1590	+17		**Titans** 12-5	AFC South	✓	67%	31%	12%
1637	+5		**Bills** 11-6	AFC East	70%	27%	16%	8%
1636	+41		**Cowboys** 12-5	NFC East	69%	30%	13%	7%
1591	-25		**Rams** 12-5	NFC West	70%	20%	7%	4%
1570			**Bengals** 10-7	AFC North	74%	24%	8%	3%
1571	-23		**Patriots** 10-7	AFC East	30%	12%	4%	2%
1580	+19		**49ers** 10-7	NFC West	31%	9%	3%	2%
1523	-36		**Cardinals** 11-6	NFC West	30%	7%	2%	0.8%
1480	+10		**Raiders** 10-7	AFC West	26%	6%	1%	0.4%
1508	-28		**Eagles** 9-8	NFC East	20%	3%	1%	0.4%
1486	+15		**Steelers** 9-7-1	AFC North	14%	4%	1%	0.3%

Los Angeles Rams winning Superbowl

(2021 Playoff odds by FiveThirtyEight)

High Level Storage Architecture





A single I/O stall can lead to multiple application requests stalled

Latency Stalls – Fleet Data

- Probability of latency stalls calculated over fleet over a week. Looks familiar?
- Let's consider an SSD doing 1000 IOPS of 4K. Moderate?

Latency Stalls

Read Latency (Upper Bound)	IO Percentile	Percent of Reads in this band	Number of Reads in a SECOND
1ms	52.40	52.40	524
10ms	98.70	46.30	463
100ms	99.99	1.29	13
1s	99.999999	0.006	0
10s	99.9999999	8.07e-07	0
> 10s	100	1.008e-07	0

Latency Stalls

Read Latency (Upper Bound)	IO Percentile	Percent of Reads in this band	Number of Reads in a MINUTE
1ms	52.40	52.40	31,440
10ms	98.70	46.30	27,782
100ms	99.99	1.29	774
1s	99.999999	0.006	4
10s	99.9999999	8.07e-07	0
> 10s	100	1.008e-07	0

Access time of first
commercial HDD: 1956

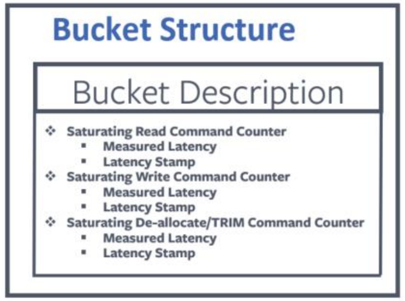
Latency Stalls

Read Latency (Upper Bound)	IO Percentile	Percent of Reads in this band	Number of Reads in 3 DAYS
1ms	52.40	52.40	135,818,259
10ms	98.70	46.30	120,020,237
100ms	99.99	1.29	3,345,464
1s	99.999999	0.006	16,035
10s	99.9999999	8.07e-07	2
> 10s	100	1.008e-07	0

Access time of first commercial HDD: 1956

Usain Bolt 100m sprint record

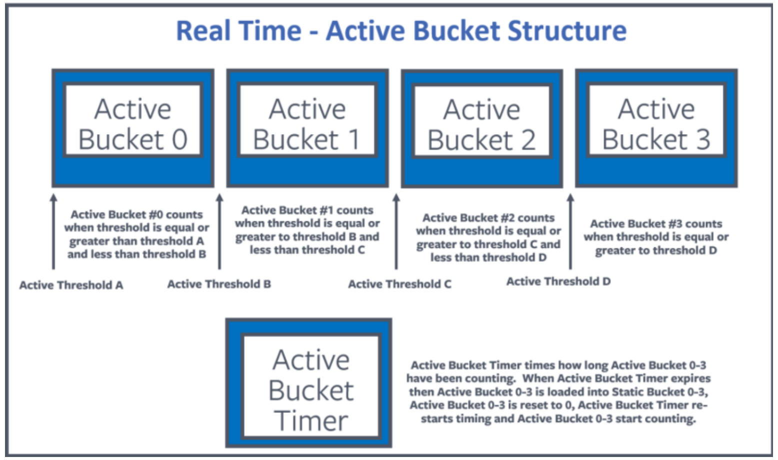
Efficient Debugging – Latency Monitoring Log



Bucket Structure

Bucket Description

- Saturating Read Command Counter
  - Measured Latency
  - Latency Stamp
- Saturating Write Command Counter
  - Measured Latency
  - Latency Stamp
- Saturating De-allocate/TRIM Command Counter
  - Measured Latency
  - Latency Stamp



Real Time - Active Bucket Structure

Active Bucket 0

Active Bucket 1

Active Bucket 2

Active Bucket 3

Active Bucket #0 counts when threshold is equal or greater than threshold A and less than threshold B

Active Bucket #1 counts when threshold is equal or greater to threshold B and less than threshold C

Active Bucket #2 counts when threshold is equal or greater to threshold C and less than threshold D

Active Bucket #3 counts when threshold is equal or greater to threshold D

Active Threshold A

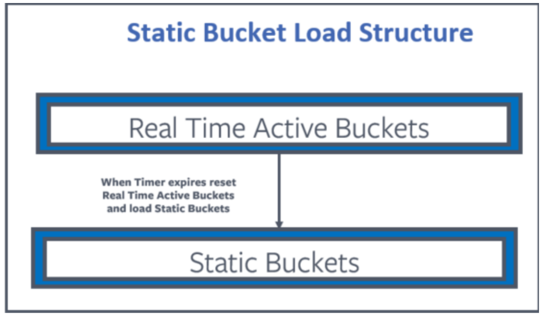
Active Threshold B

Active Threshold C

Active Threshold D

Active Bucket Timer

Active Bucket Timer times how long Active Bucket 0-3 have been counting. When Active Bucket Timer expires then Active Bucket 0-3 is loaded into Static Bucket 0-3, Active Bucket 0-3 is reset to 0, Active Bucket Timer re-starts timing and Active Bucket 0-3 start counting.

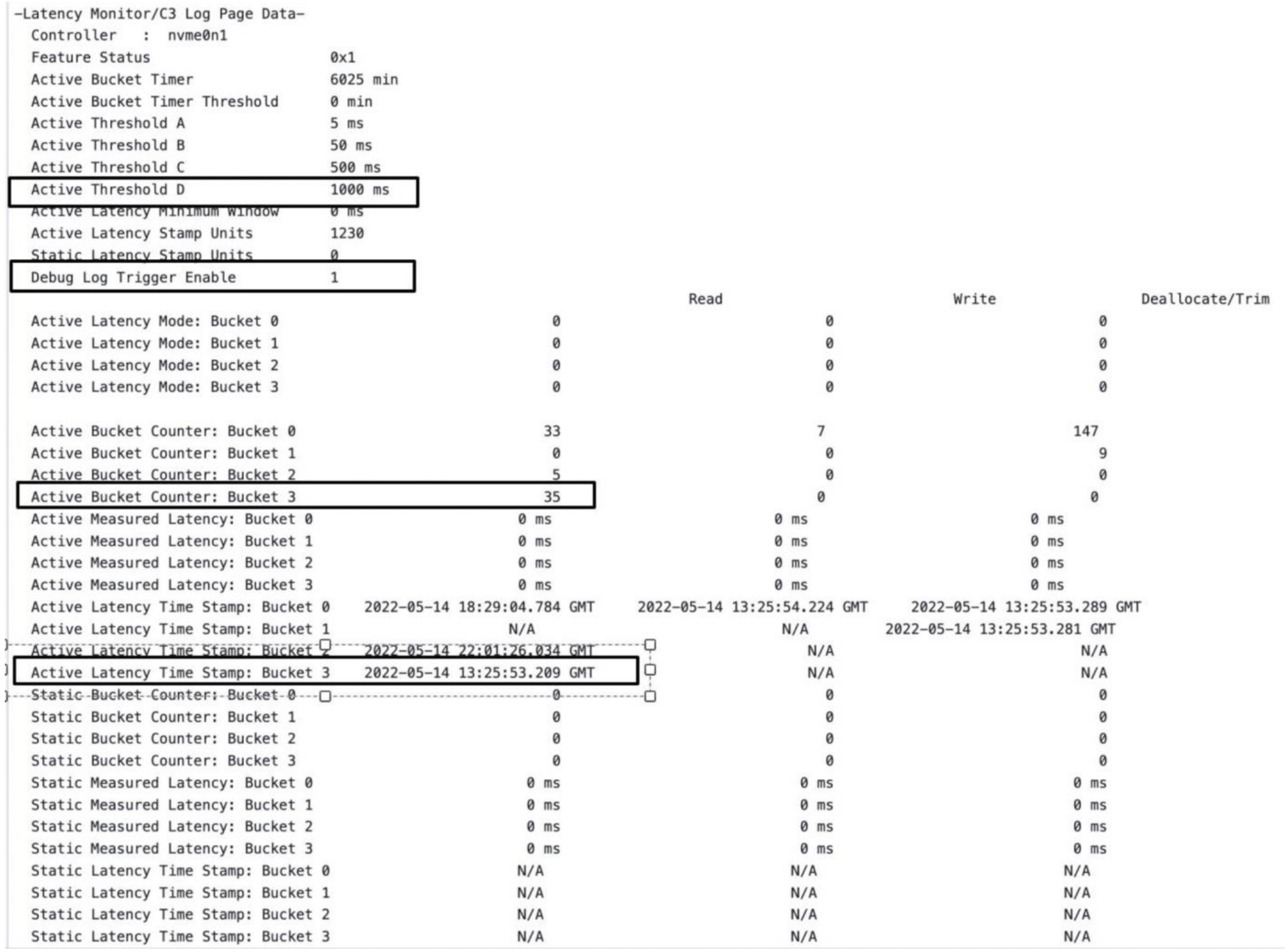


Static Bucket Load Structure

Real Time Active Buckets

When Timer expires reset Real Time Active Buckets and load Static Buckets

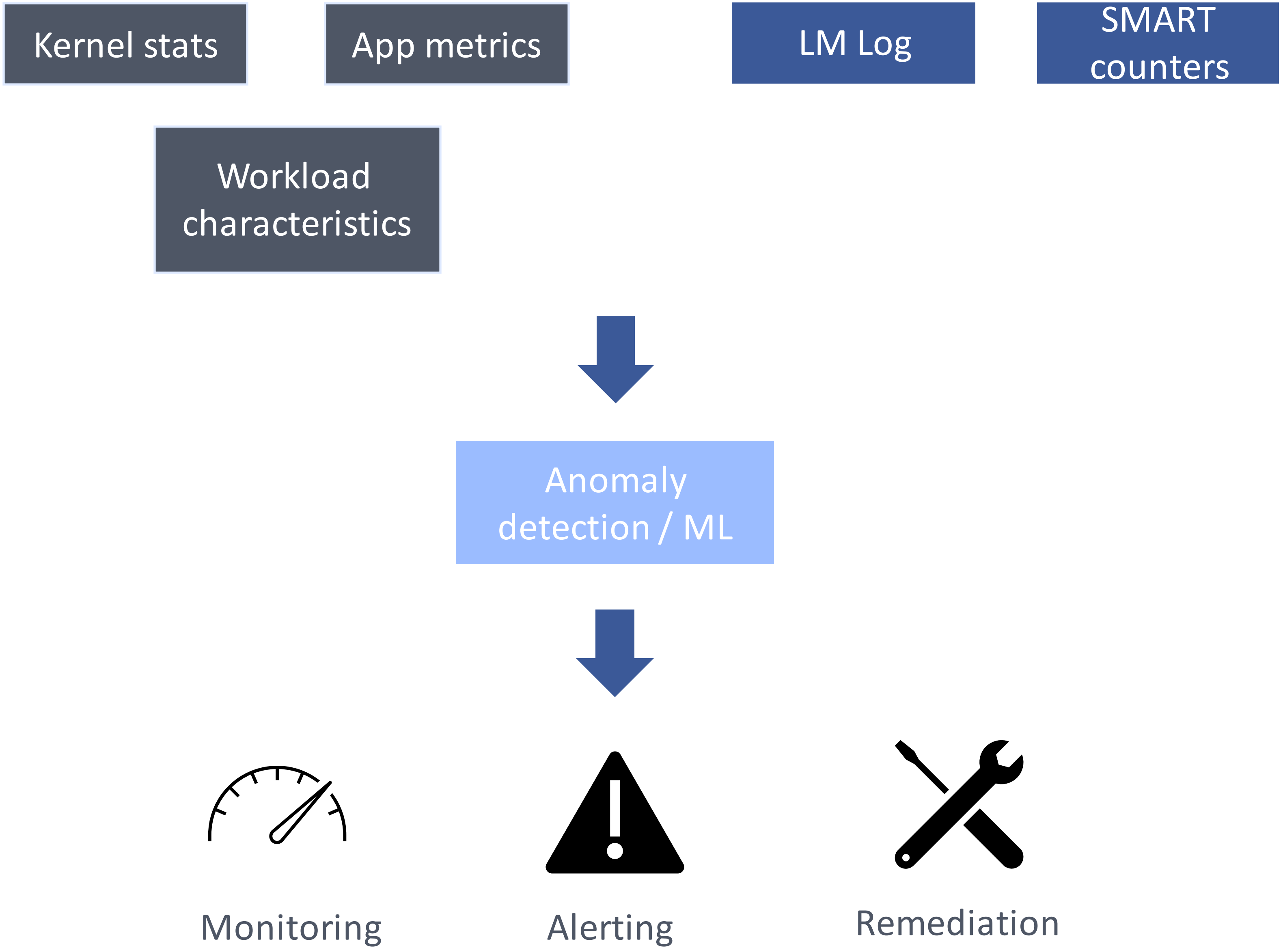
Static Buckets



```
-Latency Monitor/C3 Log Page Data-
  Controller    :  nvme0n1
  Feature Status                      0x1
  Active Bucket Timer                 6025 min
  Active Bucket Timer Threshold       0 min
  Active Threshold A                  5 ms
  Active Threshold B                  50 ms
  Active Threshold C                  500 ms
  Active Threshold D                  1000 ms
  Active Latency Minimum Window       0 ms
  Active Latency Stamp Units          1230
  Static Latency Stamp Units          0
  Debug Log Trigger Enable            1
                                                                     Read                           Write                   Deallocate/Trim
  Active Latency Mode: Bucket 0                             0                               0                               0
  Active Latency Mode: Bucket 1                             0                               0                               0
  Active Latency Mode: Bucket 2                             0                               0                               0
  Active Latency Mode: Bucket 3                             0                               0                               0

  Active Bucket Counter: Bucket 0                          33                               7                             147
  Active Bucket Counter: Bucket 1                           0                               0                               9
  Active Bucket Counter: Bucket 2                           5                               0                               0
  Active Bucket Counter: Bucket 3                          35                              0                               0
  Active Measured Latency: Bucket 0                     0 ms                            0 ms                            0 ms
  Active Measured Latency: Bucket 1                     0 ms                            0 ms                            0 ms
  Active Measured Latency: Bucket 2                     0 ms                            0 ms                            0 ms
  Active Measured Latency: Bucket 3                     0 ms                            0 ms                            0 ms
  Active Latency Time Stamp: Bucket 0   2022-05-14 18:29:04.784 GMT   2022-05-14 13:25:54.224 GMT   2022-05-14 13:25:53.289 GMT
  Active Latency Time Stamp: Bucket 1                 N/A                               N/A   2022-05-14 13:25:53.281 GMT
  Active Latency Time Stamp: Bucket 2   2022-05-14 22:01:26.034 GMT                           N/A                             N/A
  Active Latency Time Stamp: Bucket 3   2022-05-14 13:25:53.209 GMT                           N/A                             N/A
  Static Bucket Counter: Bucket 0                           0                               0                               0
  Static Bucket Counter: Bucket 1                           0                               0                               0
  Static Bucket Counter: Bucket 2                           0                               0                               0
  Static Bucket Counter: Bucket 3                           0                               0                               0
  Static Measured Latency: Bucket 0                      0 ms                            0 ms                            0 ms
  Static Measured Latency: Bucket 1                      0 ms                            0 ms                            0 ms
  Static Measured Latency: Bucket 2                      0 ms                            0 ms                            0 ms
  Static Measured Latency: Bucket 3                      0 ms                            0 ms                            0 ms
  Static Latency Time Stamp: Bucket 0                  N/A                             N/A                             N/A
  Static Latency Time Stamp: Bucket 1                  N/A                             N/A                             N/A
  Static Latency Time Stamp: Bucket 2                  N/A                             N/A                             N/A
  Static Latency Time Stamp: Bucket 3                  N/A                             N/A                             N/A
```

Debug Workflow @ Scale



Kernel stats

App metrics

LM Log

SMART counters

Workload characteristics

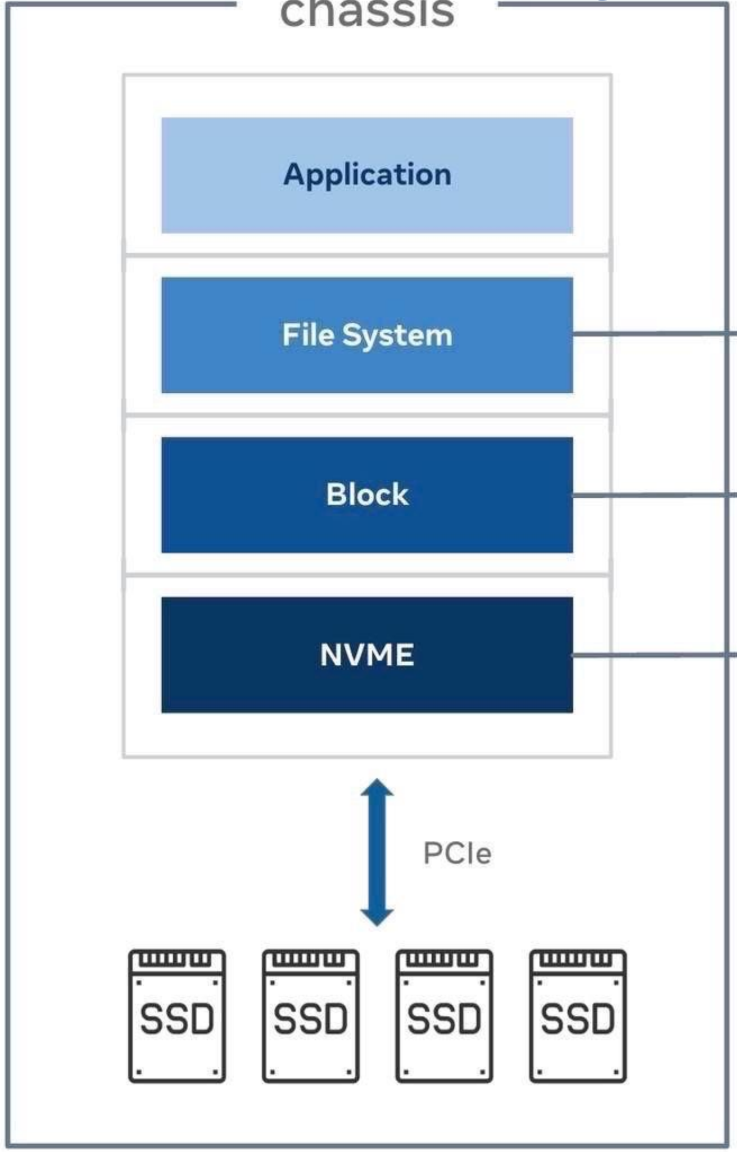
Anomaly detection / ML

Monitoring

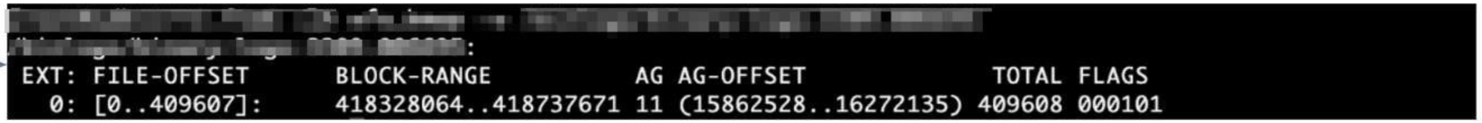
Alerting

Remediation

Observability throughout the I/O lifecycle

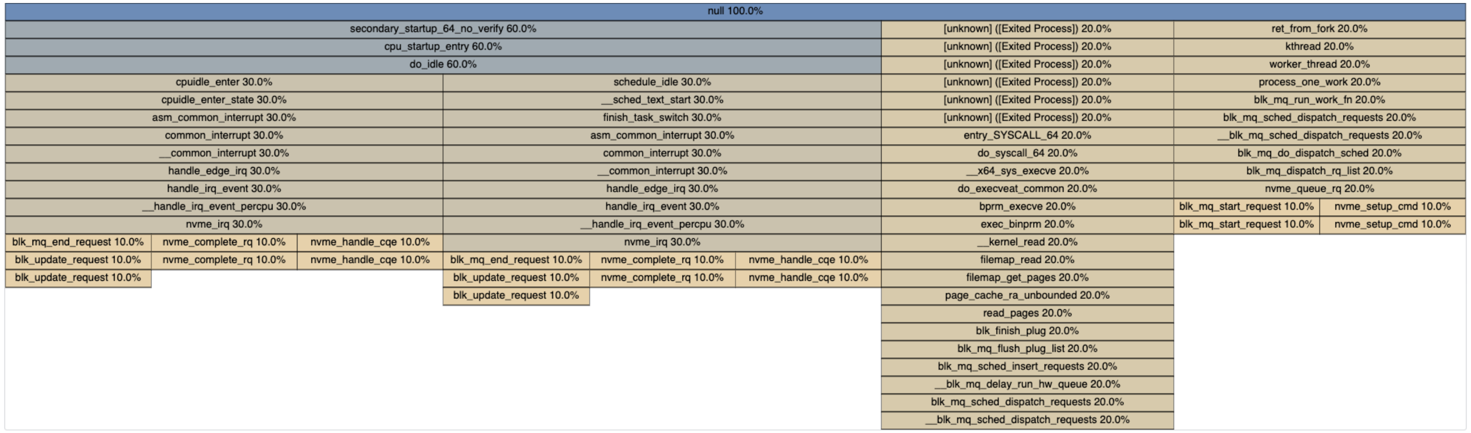


A Storage Stack Example



```
[redacted]:
 EXT: FILE-OFFSET      BLOCK-RANGE          AG AG-OFFSET             TOTAL FLAGS
   0: [0..409607]:     418328064..418737671 11 (15862528..16272135) 409608 000101
```

EVENT	TIME	COMM	TID	CPU	DETAIL
block_bio_queue	2.75%	mysqld	800131	[009]	9,2 R 125574800 + 16 [mysqld]
block_bio_remap	0.68%	mysqld	800131	[009]	259,0 R 63051664 + 16 <- (9,2) 125574800
block_bio_remap	2.75%	mysqld	800131	[009]	259,0 R 418520976 + 16 <- (259,3) 63051664
block_getrq	1.37%	mysqld	800131	[009]	259,0 R 418520976 + 16 [mysqld]
nvme_setup_cmd	3.44%	mysqld	800131	[009]	disk=nvme1n1 ctrl_id=1 qid=10 opcode=2 flags=0 fctype=1 cid=294 nsid=1 metadata=0 cdw10=ARRAY[f2, 43, 1e, 03, 00, 00, 00, 00, 01, 00, 00, 00, 00, 00, 00, 00, 00, 00, 00, 00, 00, 00, 00, 00]
block_rq_issue	88.96%	mysqld	800131	[009]	259,0 R 8192 () 418520976 + 16 [mysqld]
nvme_complete_rq	0%	swapper	0	[009]	disk=nvme1n1 ctrl_id=1 qid=10 cid=294 result=0 retries=0 flags=0 status=0



Summary

- At scale debug is extremely challenging due to inefficient design of debug logs for use at hyperscale environment
- Let's converge on debug-ability initiatives
  - BPF scripts for triage
  - Latency Monitoring Spec - Link
  - NVMe-CLI/ plugins / OCP - Link
- Meta welcomes Industry Partner's ideas on how to improve debug @ Scale

Together we can make debugging SSDs better!

Meta