Implementing I/O best practices to improve system performance with Ellexus

Dr Rosemary Francis
CEO Ellexus: The I/O profiling company

Bad I/O harms shared storage and limits performance significantly, problems that no amount of extra hardware or cloud spend can fix. In this talk, Rosemary will explain common I/O problems and best practices to implement that provide easy performance wins. She will outline I/O profiling for improving application performance as well as I/O monitoring for improving HPC system performance through good application deployment. Understanding I/O patterns can give easy wins as often a simple fix can make the biggest difference, but knowing where to look is crucial. Rosemary will demo I/O profiling tools from Ellexus that provide system telemetry and application performance metrics.

Agenda

- Slides on I/O best practices with customer case studies
  - Questions

- Ellexus tool demo
  - More questions!

Please submit questions via the “questions” tab in the webinar app

Ellexus Ltd: The I/O Profiling Company

Products: We make tools to help you

- improve system and application performance,
- protect shared storage,
- manage application dependencies,
- manage hybrid cloud.

Customers include:

TACC TEXAS The University of Texas at Austin diamond

wellcome sanger institute DELL EMC

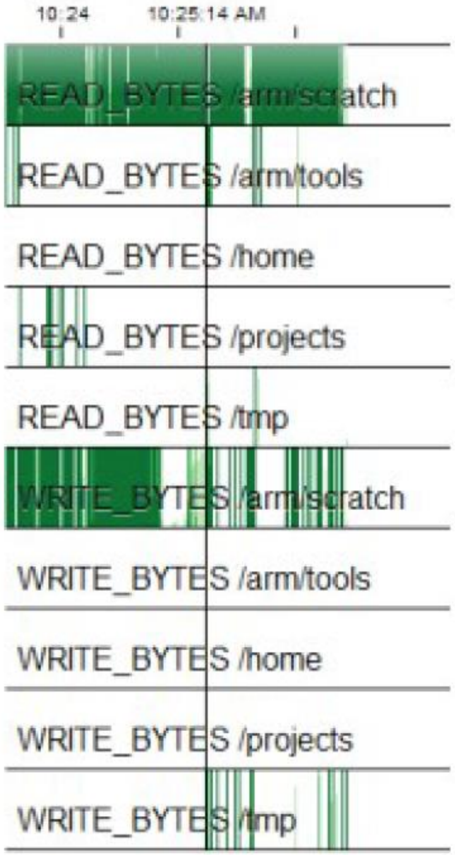
ANSYS® arm SYNOPSYS®

Solving the noisy neighbour problem

How we worked with Arm to develop our technology

It is possible for a small number of jobs to overload shared file systems and to cause system bottlenecks.

Example of a rogue job from Arm:

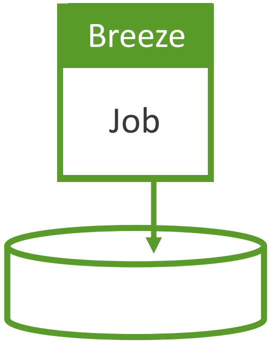


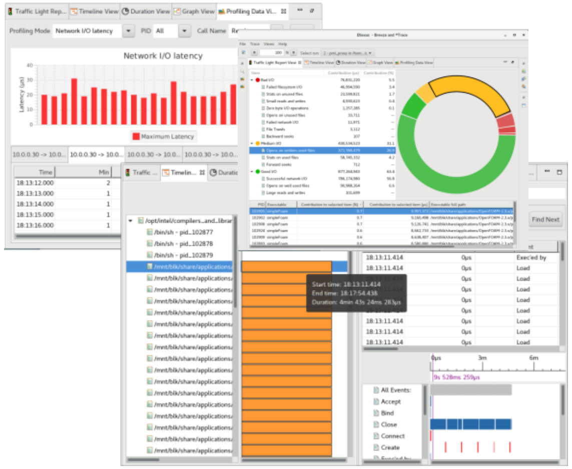
This application is overloading shared storage by putting data in the wrong place.

Temporary data is written to shared storage

Local storage is unused

Ellexus product overview





Detailed application tracing

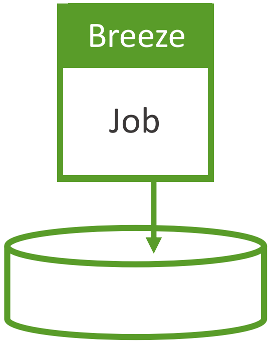
Application file and network dependencies
I/O patterns and performance bottlenecks
Data for migration, optimization and debug

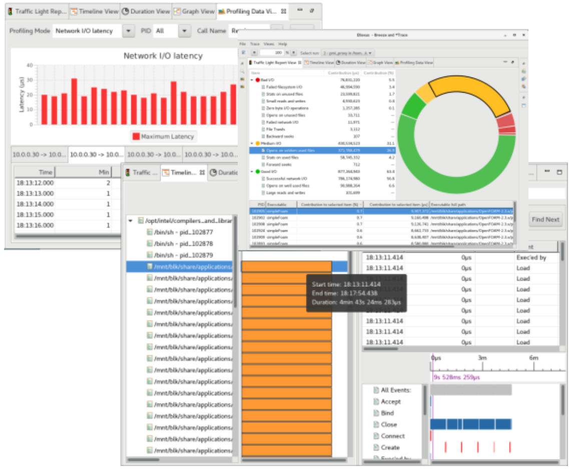
Breeze is not designed to be run in production

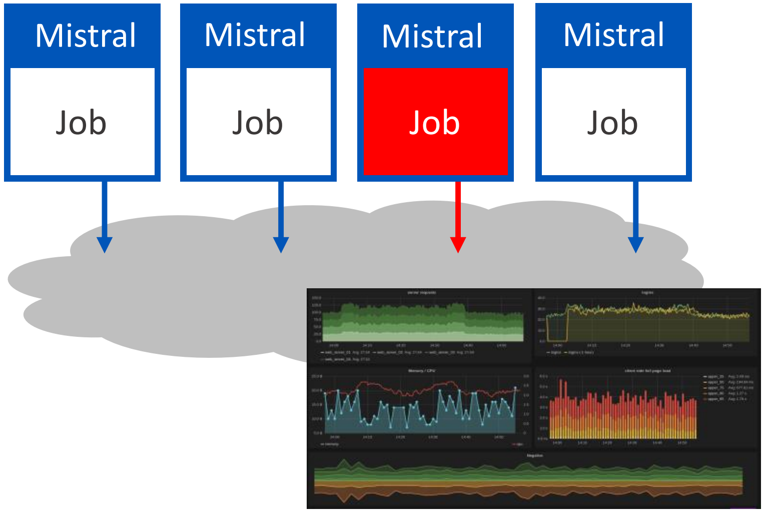
Ellexus product overview

breeze

mistral







Detailed application tracing

Application file and network dependencies
I/O patterns and performance bottlenecks
Data for migration, optimization and debug

Breeze is not designed to be run in production

Live system telemetry

Scalable lightweight profiling and monitoring
Per-job I/O and compute metrics
Insight for managing and tuning hybrid cloud
Data feeds for operational dashboards and ML

Why profile I/O

Detect dependencies for containerization and migration

Application correctness in delivery and deployment

Understand resources for sizing and procurement of IT resources or cloud

Profile I/O for tuning and optimization

Monitor I/O for chargeback and troubleshooting in the field

Cost management in hybrid cloud and system tuning

Where does bad I/O come from?

Third party tools and libraries

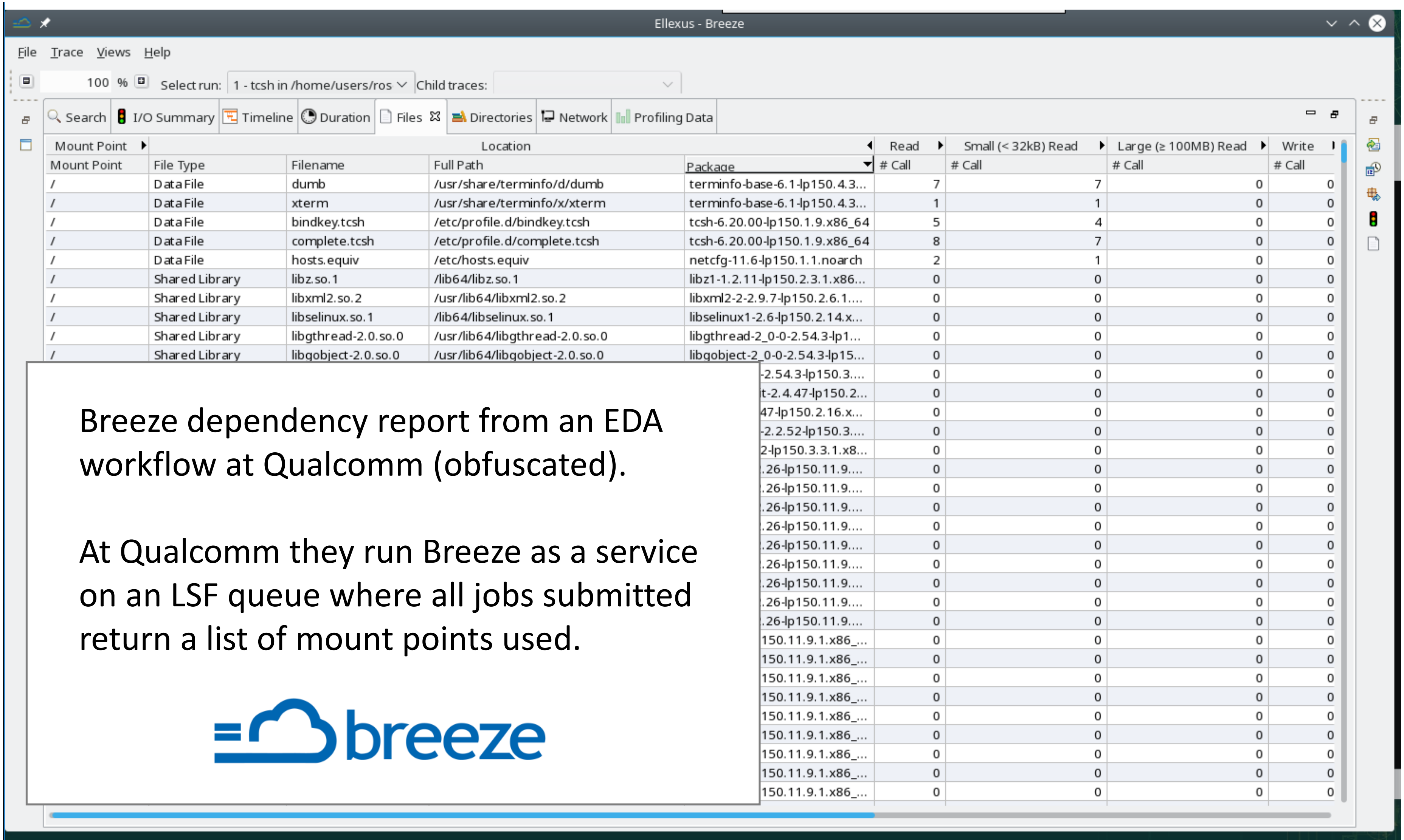
Legacy code

Misunderstandings about IT infrastructure

Changes in your working environment

Starting with file and network dependencies

Application dependencies are needed for migration, containerisation and compliance.



Mount Point	Location				Read	Small (< 32kB) Read	Large (≥ 100MB) Read	Write
Mount Point	File Type	Filename	Full Path	Package	# Call	# Call	# Call	# Call
/	Data File	dumb	/usr/share/terminfo/d/dumb	terminfo-base-6.1-lp150.4.3...	7	7	0	0
/	Data File	xterm	/usr/share/terminfo/x/xterm	terminfo-base-6.1-lp150.4.3...	1	1	0	0
/	Data File	bindkey.tcsh	/etc/profile.d/bindkey.tcsh	tcsh-6.20.00-lp150.1.9.x86_64	5	4	0	0
/	Data File	complete.tcsh	/etc/profile.d/complete.tcsh	tcsh-6.20.00-lp150.1.9.x86_64	8	7	0	0
/	Data File	hosts.equiv	/etc/hosts.equiv	netcfg-11.6-lp150.1.1.noarch	2	1	0	0
/	Shared Library	libz.so.1	/lib64/libz.so.1	libz1-1.2.11-lp150.2.3.1.x86...	0	0	0	0
/	Shared Library	libxml2.so.2	/usr/lib64/libxml2.so.2	libxml2-2-2.9.7-lp150.2.6.1....	0	0	0	0
/	Shared Library	libselinux.so.1	/lib64/libselinux.so.1	libselinux1-2.6-lp150.2.14.x...	0	0	0	0
/	Shared Library	libgthread-2.0.so.0	/usr/lib64/libgthread-2.0.so.0	libgthread-2_0-0-2.54.3-lp1...	0	0	0	0
/	Shared Library	libgobject-2.0.so.0	/usr/lib64/libgobject-2.0.so.0	libgobject-2_0-0-2.54.3-lp15...	0	0	0	0
				-2.54.3-lp150.3....	0	0	0	0
				t-2.4.47-lp150.2...	0	0	0	0
				47-lp150.2.16.x...	0	0	0	0
				-2.2.52-lp150.3....	0	0	0	0
				2-lp150.3.3.1.x8...	0	0	0	0
				.26-lp150.11.9....	0	0	0	0
				.26-lp150.11.9....	0	0	0	0
				.26-lp150.11.9....	0	0	0	0
				.26-lp150.11.9....	0	0	0	0
				.26-lp150.11.9....	0	0	0	0
				.26-lp150.11.9....	0	0	0	0
				.26-lp150.11.9....	0	0	0	0
				.26-lp150.11.9....	0	0	0	0
				.26-lp150.11.9....	0	0	0	0
				150.11.9.1.x86_...	0	0	0	0
				150.11.9.1.x86_...	0	0	0	0
				150.11.9.1.x86_...	0	0	0	0
				150.11.9.1.x86_...	0	0	0	0
				150.11.9.1.x86_...	0	0	0	0
				150.11.9.1.x86_...	0	0	0	0
				150.11.9.1.x86_...	0	0	0	0
				150.11.9.1.x86_...	0	0	0	0
				150.11.9.1.x86_...	0	0	0	0

Breeze dependency report from an EDA workflow at Qualcomm (obfuscated).

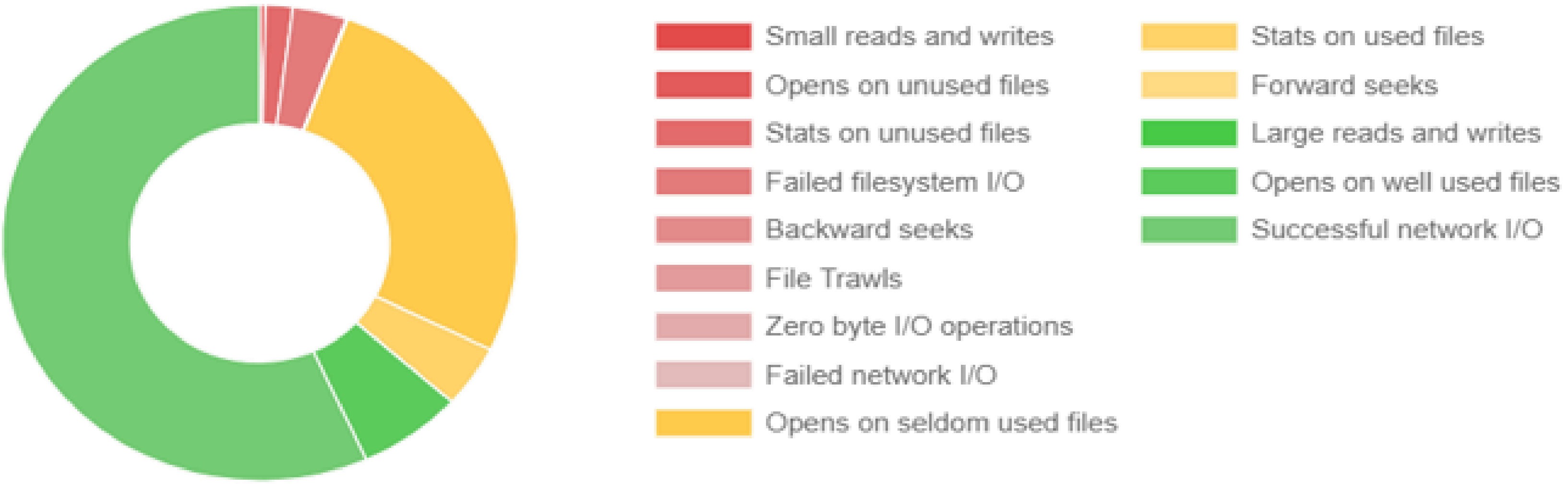
At Qualcomm they run Breeze as a service on an LSF queue where all jobs submitted return a list of mount points used.

breeze

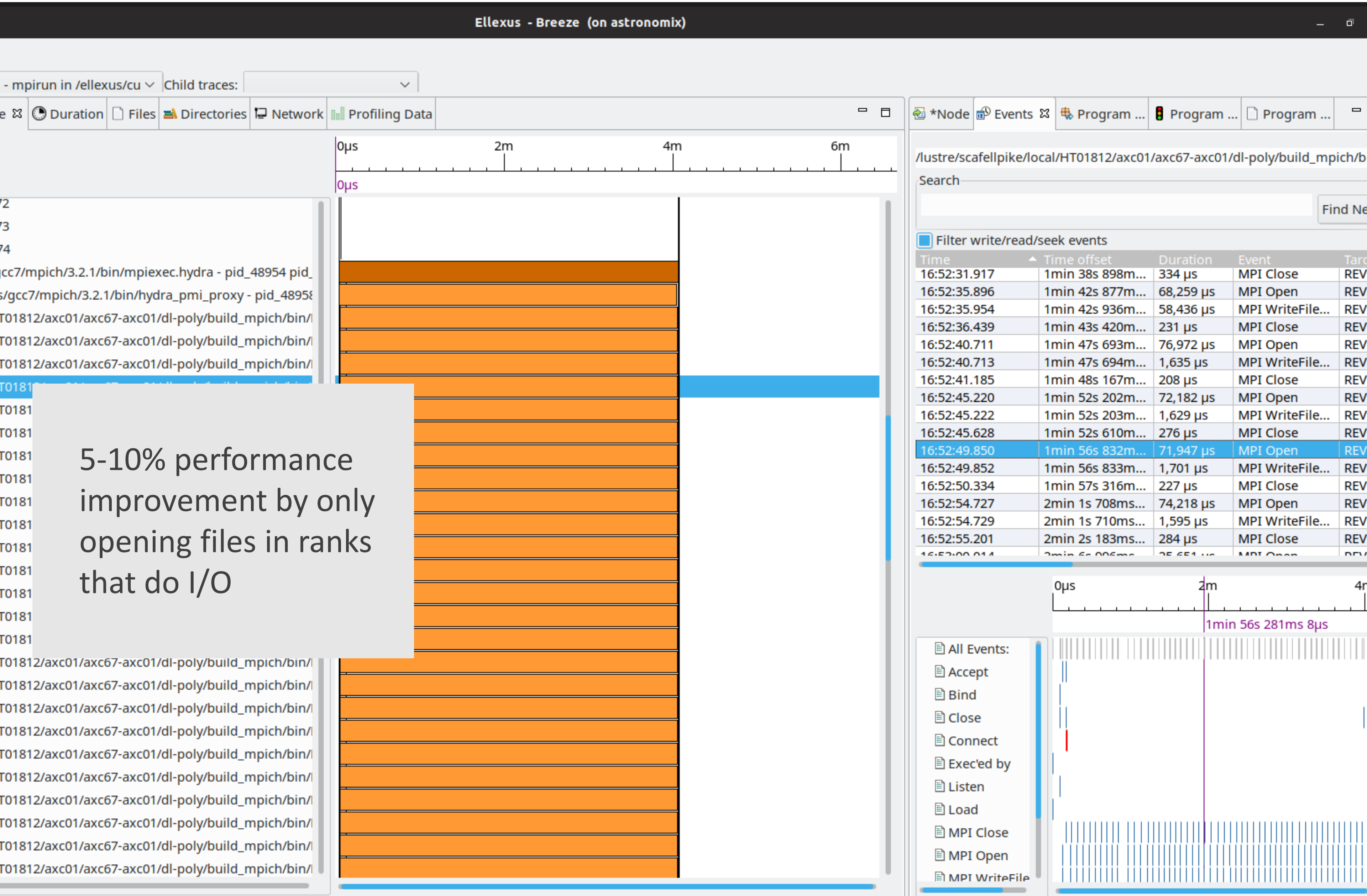
How much time are you wasting doing bad I/O?

The Breeze I/O summary breaks down good and bad I/O.

Users can easily identify bottlenecks, areas for optimisation and can understand the impact of bad I/O patterns.



Optimising open()s at the STFC Hartree Centre



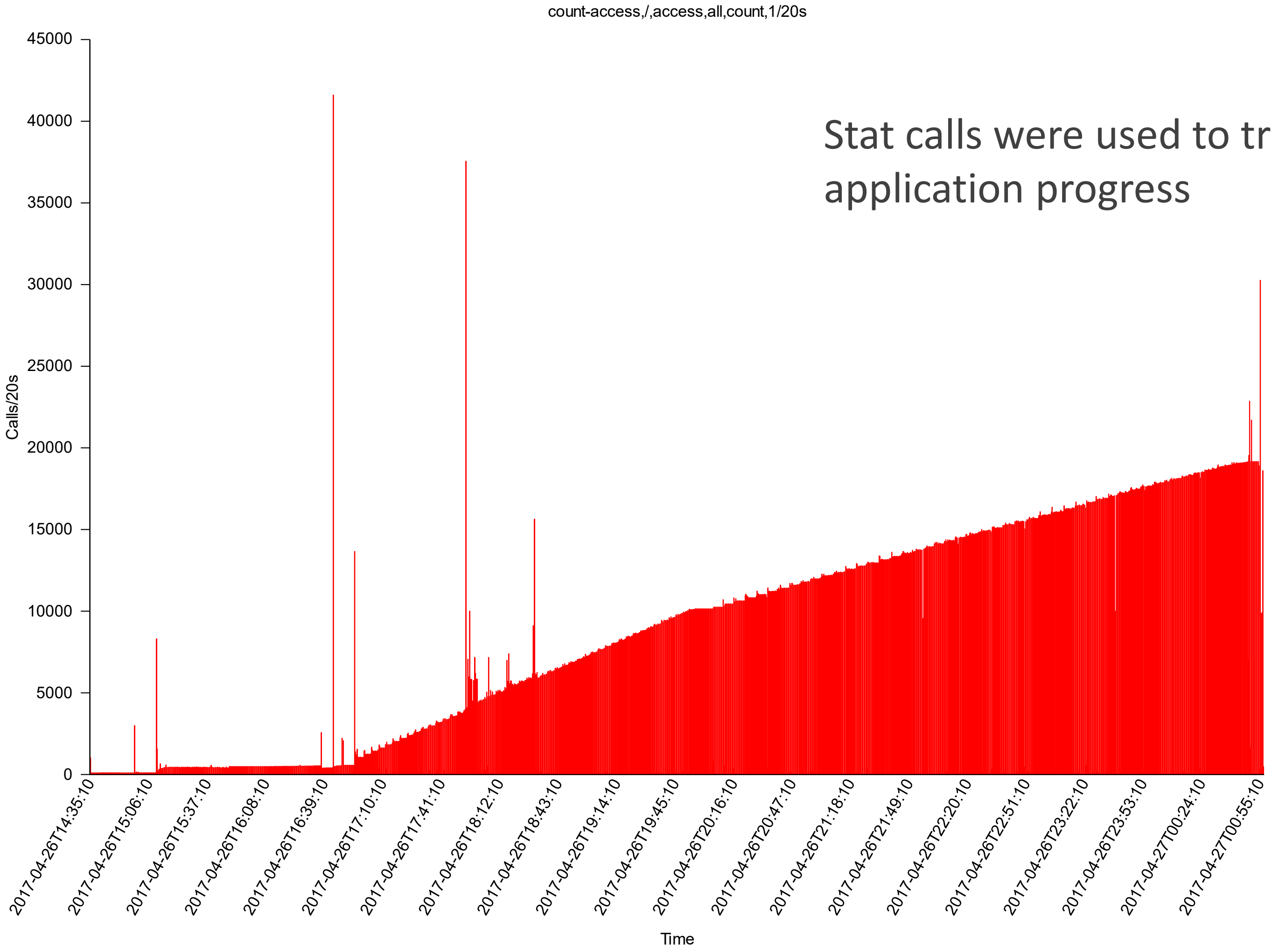
/lustre/scafellpike/local/HT01812/axc01/axc67-axc01/dl-poly/build_mpich/b

☑ Filter write/read/seek events

Time	Time offset	Duration	Event	Targ
16:52:31.917	1min 38s 898m...	334 μs	MPI Close	REV
16:52:35.896	1min 42s 877m...	68,259 μs	MPI Open	REV
16:52:35.954	1min 42s 936m...	58,436 μs	MPI WriteFile...	REV
16:52:36.439	1min 43s 420m...	231 μs	MPI Close	REV
16:52:40.711	1min 47s 693m...	76,972 μs	MPI Open	REV
16:52:40.713	1min 47s 694m...	1,635 μs	MPI WriteFile...	REV
16:52:41.185	1min 48s 167m...	208 μs	MPI Close	REV
16:52:45.220	1min 52s 202m...	72,182 μs	MPI Open	REV
16:52:45.222	1min 52s 203m...	1,629 μs	MPI WriteFile...	REV
16:52:45.628	1min 52s 610m...	276 μs	MPI Close	REV
16:52:49.850	1min 56s 832m...	71,947 μs	MPI Open	REV
16:52:49.852	1min 56s 833m...	1,701 μs	MPI WriteFile...	REV
16:52:50.334	1min 57s 316m...	227 μs	MPI Close	REV
16:52:54.727	2min 1s 708ms...	74,218 μs	MPI Open	REV
16:52:54.729	2min 1s 710ms...	1,595 μs	MPI WriteFile...	REV
16:52:55.201	2min 2s 183ms...	284 μs	MPI Close	REV

5-10% performance improvement by only opening files in ranks that do I/O

Meta-data accesses
Case study: genome pipeline



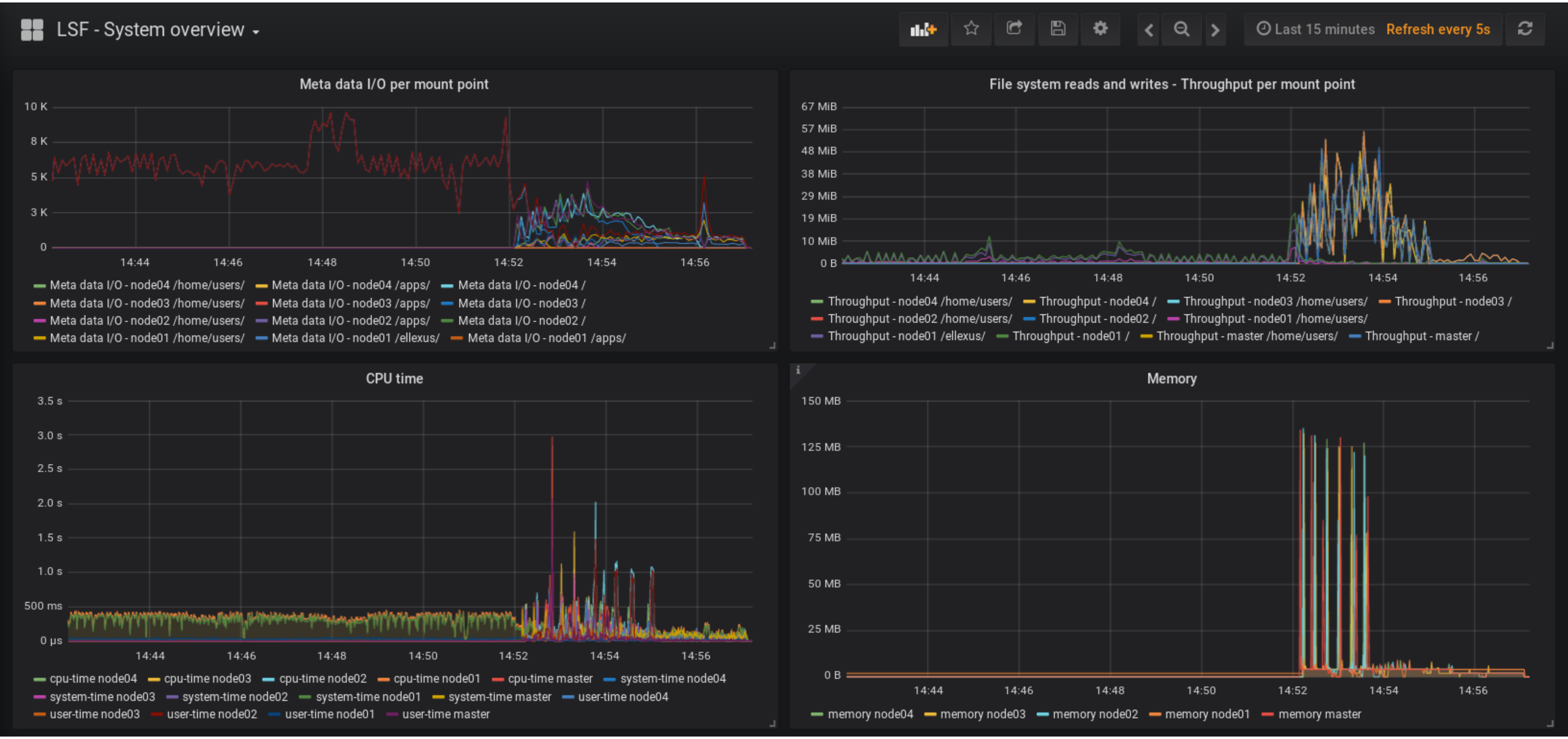
Stat calls were used to track application progress

Ellexus: The I/O Profiling Company
www.ellexus.com

What is normal?

Operations telemetry with per-job metrics and storage performance

Find rogue jobs and bottleneck quickly. Track normal usage for planning and tuning.



Tuning cancer pipelines at the Sanger Institute

The Pancancer project: 2000 whole genomes at multiple HPC sites

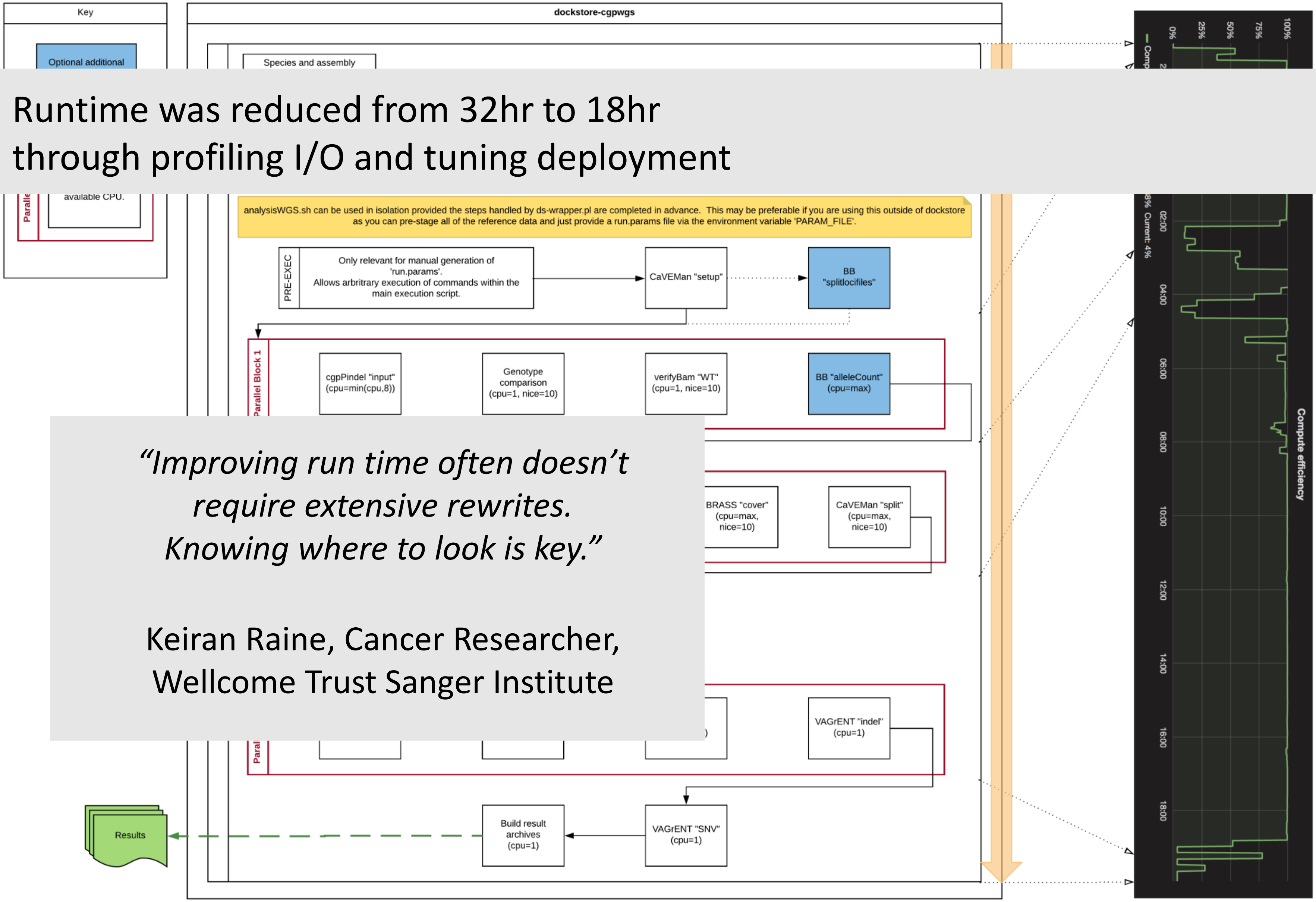
→ Containerised pipelines for portability

→ I/O tuned with Ellexus tools

→ Storage now needs to be sized correctly

Tuning cancer pipelines at the Sanger Institute

Runtime was reduced from 32hr to 18hr
through profiling I/O and tuning deployment


Key
Optional additional
available CPU.
Paralle
dockstore-cgpwgs
Species and assembly
analysisWGS.sh can be used in isolation provided the steps handled by ds-wrapper.pl are completed in advance. This may be preferable if you are using this outside of dockstore as you can pre-stage all of the reference data and just provide a run.params file via the environment variable 'PARAM_FILE'.
PRE-EXEC
Only relevant for manual generation of 'run.params'. Allows arbitrary execution of commands within the main execution script.
CaVEMan "setup"
BB "splitlocifiles"
Parallel Block 1
cgpPindel "input" (cpu=min(cpu,8))
Genotype comparison (cpu=1, nice=10)
verifyBam "WT" (cpu=1, nice=10)
BB "alleleCount" (cpu=max)
BRASS "cover" (cpu=max, nice=10)
CaVEMan "split" (cpu=max, nice=10)
Paral
VAGrENT "indel" (cpu=1)
VAGrENT "SNV" (cpu=1)
Build result archives (cpu=1)
Results
0%
25%
50%
75%
100%
Comp
8% Current: 4%
02:00
04:00
06:00
08:00
10:00
12:00
14:00
16:00
18:00
Compute efficiency

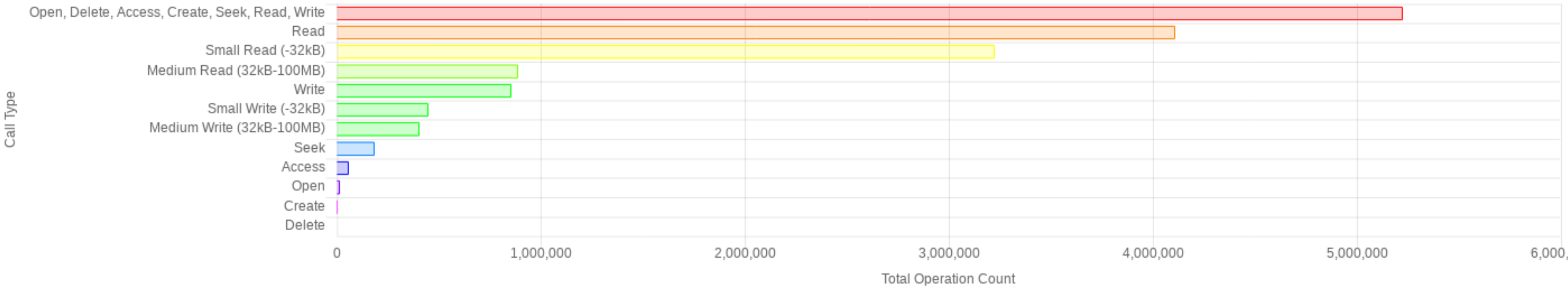
“Improving run time often doesn’t require extensive rewrites. Knowing where to look is key.”

Keiran Raine, Cancer Researcher,
Wellcome Trust Sanger Institute

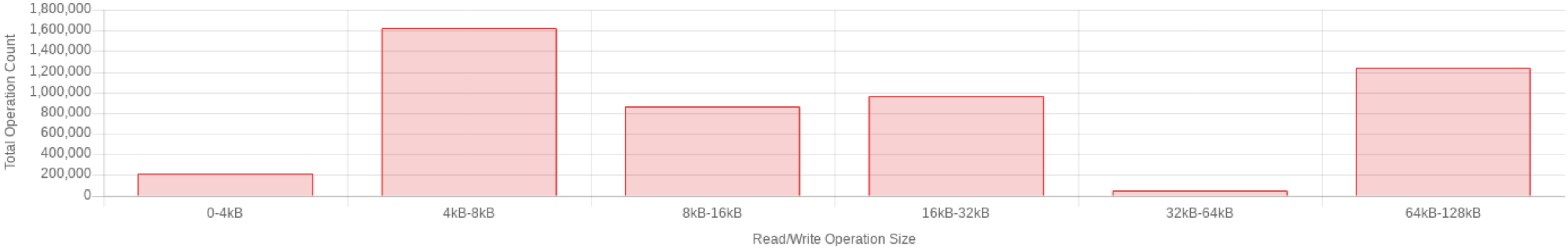
Profiling the cancer pipeline

AWS m5.xlarge 4vCPU 16GB

Number of I/O operations() by type



Size of read and write operations()



Storage comparison

	Time*		Cost per month	
GP2	52m 23s	100%	174.11	100%
Magnetic EBS	1h 01m 44s	118%	174.43	100%
Provisioned 100 IOPS	1h 42m 01s	195%	184.61	106%
Throughput optimised HDD	1h 19m 32s	152%	189.01	109%
150GB NVMe	51m 27s	98%	191.79	110%
Provisioned 500 IOPS	54m 22s	104%	215.01	123%

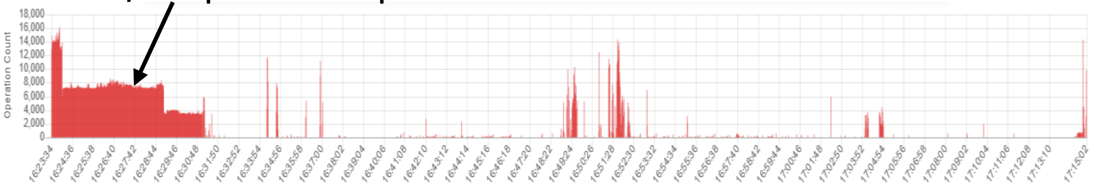
⇒ AWS default option, GP2 is the best

  ⇒ NVMe was only 2% faster for a 10% price increase

I/O Operations() over time

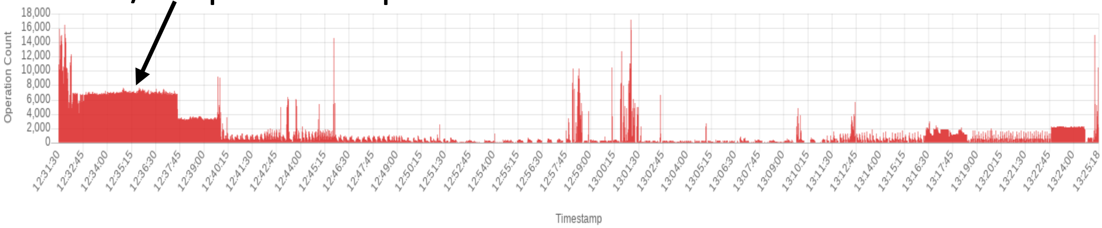
NVMe SSD

~7k I/O operations per second



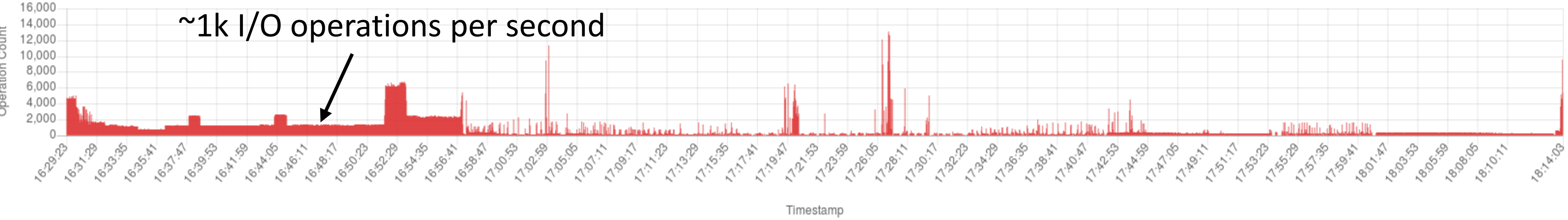
GP2 SSD

~7k I/O operations per second



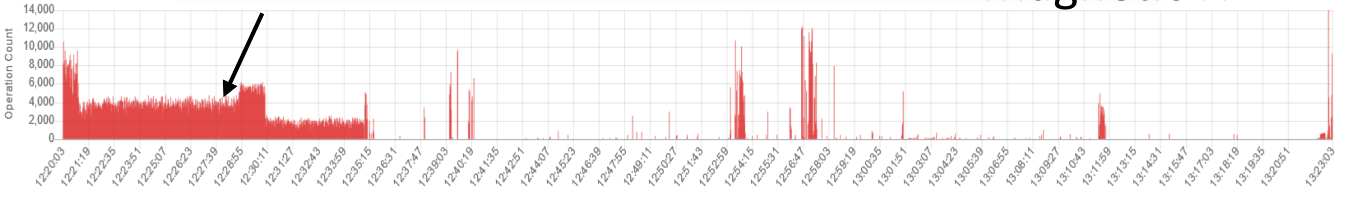
Provisioned 100IOPS SSD

~1k I/O operations per second



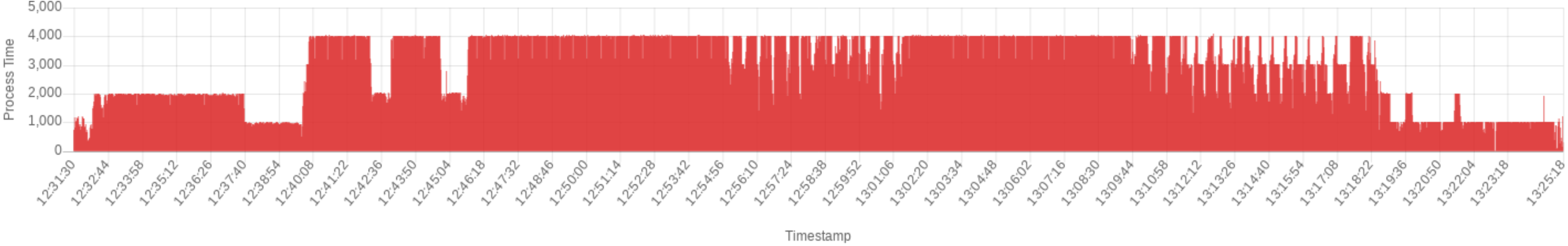
Magnetic HDD

~4k I/O operations per second

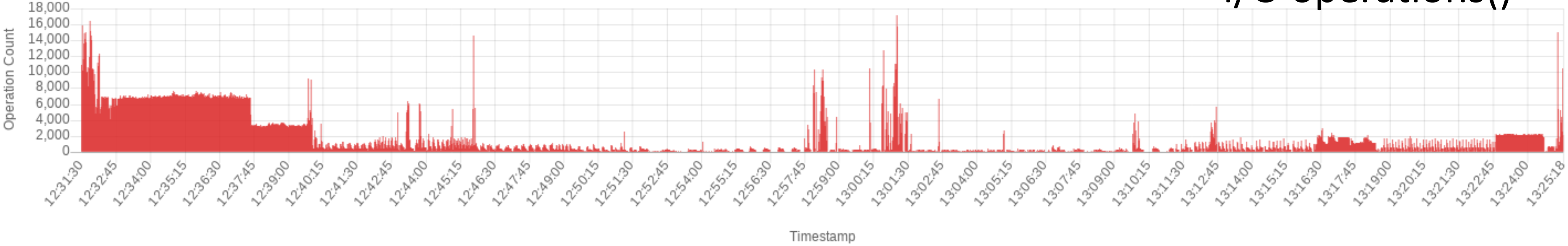


CPU and I/O Profile (on GP2 SSD)

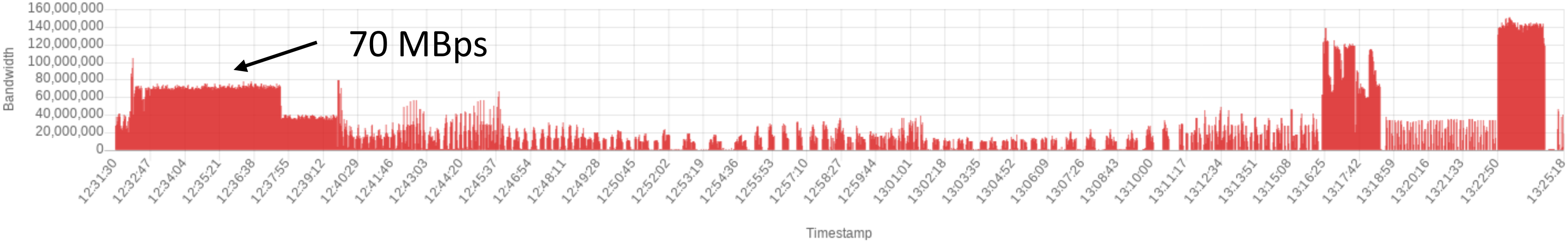
CPU



I/O operations()



Read bandwidth



70 MBps

More CPU and less memory: m5.xlarge vs c5.xlarge

(still on GP2 SSD default storage)

M5.xlarge	**c5.xlarge**
4 vCPU	4 vCPU
16GB	8GB
Runtime: 53min	Runtime: 44min
Cost: $0.21	Cost: $0.16

This ran the pipeline faster and increased reads bandwidth from 70MBps to 90MBps

How long did this work take?

Tuning the pipeline took a lot of effort

… but runtime went from 32hr to 18hr

Sizing the storage and compute correctly took three days

… and we saved >10-25% of cloud costs for the project

Lessons learnt

⇒ Containers make it easy to deploy applications

⇒ Optimising I/O can save you money

⇒ Need to understand all variables to find bottlenecks:

- ⇒ CPU
- ⇒ memory
- ⇒ I/O patterns
- ⇒ I/O performance

Ellexus best practices for good I/O

Dependencies as the number one check
 - Wrong libraries
 - Wrong config files
 - Changes need to be checked by a human

Regression testing for I/O behaviour

Zero tolerance on bad I/O

Test in production - Often problems are in set up scripts

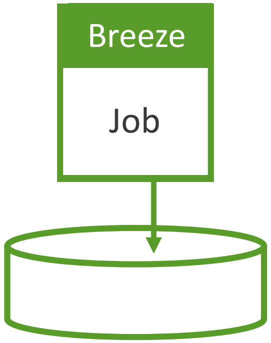
Tuning and optimisation - if you have time

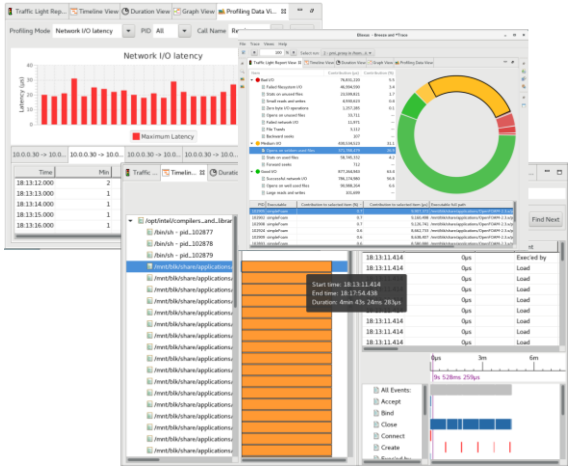
Demo!

Ellexus product overview

breeze

mistral



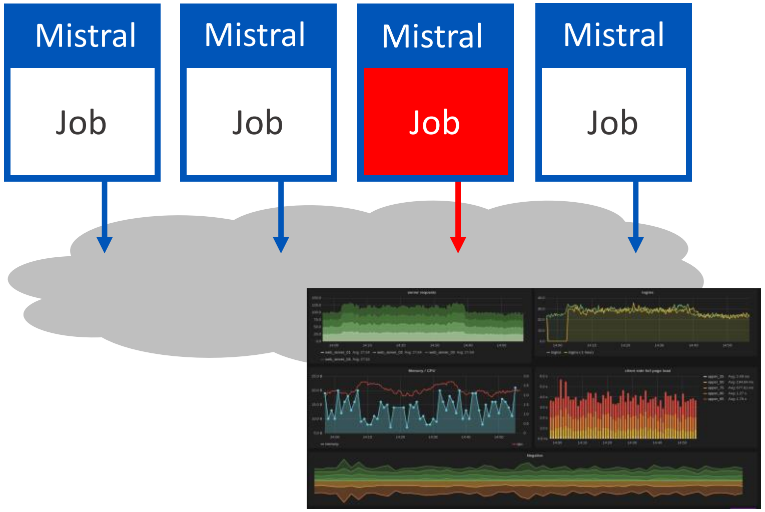

Traffic Light Rep... | Timeline View | Duration View | Graph View | Profiling Data Vi...
Profiling Mode | Network I/O latency | PID | All | Call Name
Network I/O latency
Latency (µs)
40
30
20
10
0
Maximum Latency
10.0.0.30 -> 10.0... | 10.0.0.30 -> 10.0... | 10.0.0.30 -> 10.0... | 10.0.0.30 -> 10.0...
Ellexus - Breeze and Trace
File | Trace | Views | Help
Traffic Light Report View | Timeline View | Duration View | Graph View | Profiling Data View
Bad I/O
Medium I/O
Good I/O
Traffic ... | Timelin... | Duratio
Find Next
0µs
3m
6m
9s 528ms 259µs
All Events:
Accept
Bind
Close
Connect
Create

Time	Min
18:13:12.000	2
18:13:13.000	1
18:13:14.000	1
18:13:15.000	1
18:13:16.000	1

/opt/intel/compilers_and_librar
/bin/sh - pid_102877
/bin/sh - pid_102878
/bin/sh - pid_102879
/mnt/blk/share/applications

Start time: 18:13:11.414
End time: 18:17:54.438
Duration: 4min 43s 24ms 283µs

		Exec'd by
18:13:11.414	0µs	Load
	0µs	Load
	0µs	Load
	0µs	Load
	0µs	Load
18:13:11.414	0µs	Load
18:13:11.414	0µs	Load
18:13:11.414	0µs	Load



Detailed application tracing

Application file and network dependencies
I/O patterns and performance bottlenecks
Data for migration, optimization and debug

Breeze is not designed to be run in production

Live system telemetry

Scalable lightweight profiling and monitoring
Per-job I/O and compute metrics
Insight for managing and tuning hybrid cloud
Data feeds for operational dashboards and ML

Contact us!

Our tools are trusted by research organisations, financial institutions, semiconductor companies and software vendors around the globe.

- Take control of your I/O on-premise and in the cloud
- Whole-system monitoring with APM solutions
- Detailed dependency analysis and bottleneck resolution

Proven to improve performance, increase up time and keep your customers happy.

Dr Rosemary Francis
CEO and director of technology
rosemary@ellexus.com