

Easy Tooling for Easy Cassandra Experimentation

Jon Haddad, RustyRazorBlade Consulting

Jordan West, Netflix

How do you qualify a Cassandra upgrade (or any change)?

Jon Haddad

- DataStax
- The Last Pickle
- Apple
- Netflix
- RRB Consulting!

Jordan West

- Basho (Riak)
- Apple
- Netflix
- Cassandra Committer & PMC

**Consulting has unpredictable
challenges**

Cassandra developers want to
qualify new features / changes

**Cassandra operators want to
qualify upgrades and configurations**

CCM?

- Local Cluster
- Nice for testing process
- Not great for performance tests
- Does mimic a real system in some ways

K8ssandra?

- Requires Kubernetes Knowledge
- Fighting the Operator
- No built in - support for arbitrary builds
- No support for older C* versions
- Running Prod Clusters != Running Labs
- Stateful Sets expect Disaggregated Storage

Properties in the ideal testing tool

- Easy to use different versions, even development branches
- Easy to change configs
- Under 10 minutes from start to load test
- Batteries included - deep observability

easy-cass-lab
easy-cass-stress



How about a live demo?
These never go wrong

Initialize The Environment

```
$ easy-cass-lab init -c 6 -s 1 --up coc
```

```
> easy-cass-lab init -c 6 -s 1 --up
WARNING: sun.reflect.Reflection.getCallerClass is not supported. This will impact performance.
Initializing directory
Directory Initialized Configuring Terraform
Writing Config
Calling init
Setting working directory inside container to /local
Starting ghcr.io/opentofu/opentofu:1.7 container (41515ff15024)
Attaching to running container
Starting container 41515ff15024541e222efaae304450330b301c850946b32c85acc5fb3eddbec0
```

Initializing the backend...

Initializing provider plugins...

- Finding latest version of hashicorp/aws...
- Installing hashicorp/aws v5.69.0...
- Installed hashicorp/aws v5.69.0 (signed, key ID **0C0AF313E5FD9F80**)

Providers are signed by their developers.

If you'd like to know more about provider signing, you can read about it here:
<https://opentofu.org/docs/cli/plugins/signing/>

OpenTofu has created a lock file **.terraform.lock.hcl** to record the provider selections it made above. Include this file in your version control repository so that OpenTofu can guarantee to make the same selections by default when you run "tofu init" in the future.

Pick a Version

```
$ easy-cass-lab use 5.0
```

```
> ecl use 5.0
```

```
WARNING: sun.reflect.Reflection.getCallerClass is not supported. This will impact performance.
```

```
Using version 5.0 on 6 hosts, filter: com.rustyrazorblade.easycasslab.commands.delegates.Hosts@65753040
```

```
Connecting to cassandra0 52.39.215.115
```

```
Executing remote command: sudo use-cassandra 5.0
```

```
'/usr/local/cassandra/current' -> '/usr/local/cassandra/5.0'
```

```
Using default java version from cassandra_versions.yaml: 11
```

```
Using Java version 11
```

```
Connecting to cassandra1 34.217.14.85
```

```
Executing remote command: sudo use-cassandra 5.0
```

```
'/usr/local/cassandra/current' -> '/usr/local/cassandra/5.0'
```

```
Using default java version from cassandra_versions.yaml: 11
```

```
Using Java version 11
```

```
Uploading cassandra.patch.yaml to Host(public=52.12.60.183, private=10.0.2.18, alias=cassandr
```

```
Patching Host(public=52.12.60.183, private=10.0.2.18, alias=cassandra5, availabilityZone=us-w
```

```
Connecting to cassandra5 52.12.60.183
```

```
Executing remote command: /usr/local/bin/patch-config cassandra.patch.yaml
```

```
Connecting to cassandra5 52.12.60.183
```



```
Executing remote command: sudo cp jvm.options /usr/local/cassandra/current/conf/jvm.options
```

```
Connecting to cassandra5 52.12.60.183
```

```
Executing remote command: sudo chown -R cassandra:cassandra /usr/local/cassandra/current/conf
```


```
You can update the cassandra.patch.yaml and jvm.options files then run easy-cass-lab update-
```



```
~/clusters/coc via  default on  (us-west-2) took 1m45s
> cat cassandra.patch.yaml
---
cluster_name: "test"
num_tokens: 4
seed_provider:
  class_name: "org.apache.cassandra.locator.SimpleSeedProvider"
  parameters:
    seeds: "10.0.0.143"
hints_directory: "/mnt/cassandra/hints"
data_file_directories:
- "/mnt/cassandra/data"
commitlog_directory: "/mnt/cassandra/commitlog"
concurrent_reads: 64
concurrent_writes: 64
trickle_fsync: true
endpoint_snitch: "Ec2Snitch"
```

Start it up!

```
$ easy-cass-lab start
```

```
~/clusters/coc via  default on  (us-west-2) took 7m47s
```

```
> source env.sh
```

```
[WARNING] We are creating aliases which override these commands:
```

```
ssh  
sftp  
scp  
rsync
```

```
The aliases point the commands they override to your new cluster.  
To undo these changes exit this terminal.
```

```
~/clusters/coc via  default on  (us-west-2)
```

```
> c0
```

```
Warning: Permanently added '52.39.215.115' (ED25519) to the list of known hosts.
```

```
Welcome to Ubuntu 22.04.5 LTS (GNU/Linux 6.8.0-1015-aws x86_64)
```

```
ubuntu@cassandra0:~$ nt status
```

```
Datacenter: us-west-2
```

```
=====
```

```
Status=Up/Down
```

```
|/ State=Normal/Leaving/Joining/Moving
```

--	Address	Load	Tokens	Owns (effective)	Host ID	Rack
UN	10.0.2.18	84.83 KiB	4	35.6%	091e3f5d-fec7-4604-881d-f530c4a90a13	us-west-2c
UN	10.0.1.13	79.78 KiB	4	31.1%	e0cf3c52-79b0-4bed-a7e8-a7c6a2aa1569	us-west-2b
UN	10.0.2.26	79.79 KiB	4	33.6%	520584e5-983b-481b-8db0-99ae23ac5b73	us-west-2c
UN	10.0.0.143	119.56 KiB	4	32.0%	a1b02809-2bd2-4120-8b85-aac24bd3d9e9	us-west-2a
UN	10.0.1.147	79.78 KiB	4	31.0%	457acd87-cb50-470b-bef5-f9f49b14a15d	us-west-2b
UN	10.0.0.145	79.78 KiB	4	36.7%	2f7ffa70-36ec-44d6-937b-5ef11c28d890	us-west-2a

Run a load test!

```
eubuntu@stress0:~$ easy-cass-stress run KeyValue -d 4h -p 10m --rate 50k
```

```
Creating easy_cass_stress:
```

```
CREATE KEYSPACE
```

```
IF NOT EXISTS easy_cass_stress
```

```
WITH replication = {'class': 'SimpleStrategy', 'replication_factor':3 }
```

```
Creating schema
```

```
Executing 0 operations with consistency level LOCAL_ONE and serial consistency level LOCAL_SERIAL
```

```
Connected to Cassandra cluster.
```

```
Creating Tables
```

```
CREATE TABLE IF NOT EXISTS keyvalue (
```

```
    key text PRIMARY KEY,
```

```
    value text
```

```
) WITH caching = {'keys': 'ALL', 'rows_per_partition': 'NONE'} AND default_time_to_live
```

```
Preparing queries
```

```
Initializing metrics
```

```
Not setting up prometheus endpoint.
```

```
Stepping rate limiter by 5000.0 to 50000.0
```

```
Connecting to Cassandra cluster ...
```

```
Creating generator random
```

```
1 threads prepared.
```

```
Prepopulating data with 0 records per thread (1)
```

```
Prometheus metrics are available at http://localhost:0/
```

```
Starting main runner
```

```
[Thread 0]: Running the profile for 240min...
```

Writes

Reads

Deletes

Count	Latency (p99)	1min (req/s)	Count	Latency (p99)	1min (req/s)	Count	Latency (p99)	1min (req/s)
10098	231.41	0	9856	242.79	0	0	0	0
17608	229.79	2999	17320	233.61	2965.4	0	0	0
25032	227.71	2999	24899	229.29	2965.4	0	0	0
37520	217.43	2959.21	37436	223.97	2928.16	0	0	0
50000	100.00	2100.00	50000	211.00	2000.00	0	0	0

MORE LOAD TESTING!

Writes			Reads			Deletes			Errors	
Count	Latency (p99)	1min (req/s)	Count	Latency (p99)	1min (req/s)	Count	Latency (p99)	1min (req/s)	Count	1min (errors/s)
15905433	75.71	24998.16	15904720	85.98	25007.37	0	0	0	0	0
15980550	77.74	25002.13	15979602	81.91	25003.34	0	0	0	0	0
16055545	75.71	25002.13	16054629	94.97	25003.34	0	0	0	0	0
16130771	77.74	24998.42	16129417	96.74	25007.15	0	0	0	0	0
16205864	69.98	25005.01	16204343	96.86	25000.49	0	0	0	0	0
16280656	69.98	25005.01	16279554	92.17	25000.49	0	0	0	0	0
16355472	64.71	24999.41	16354744	85.98	25006.06	0	0	0	0	0
16430223	55.98	24999.41	16429657	96.74	25006.06	0	0	0	0	0
16504978	48.53	24996.45	16505289	96.86	25009.08	0	0	0	0	0
16579848	73.19	24989.81	16580144	96.86	25013.9	0	0	0	0	0

Writes			Reads			Deletes			Errors	
Count	Latency (p99)	1min (req/s)	Count	Latency (p99)	1min (req/s)	Count	Latency (p99)	1min (req/s)	Count	1min (errors/s)
25539278	69.26	97695.62	258774	54.11	986.84	0	0	0	0	0
25836180	69.26	97695.62	261744	55.12	986.84	0	0	0	0	0
26133550	69.26	97808.91	264788	56.72	987.96	0	0	0	0	0
26430711	69.26	97808.91	267687	55.12	987.96	0	0	0	0	0
26727220	79.66	97907.23	270655	54.17	986.52	0	0	0	0	0
27024799	76.17	97995.69	273626	54.11	987.69	0	0	0	0	0
27321971	91.18	97995.69	276711	61.89	987.69	0	0	0	0	0
27618257	91.18	98075.03	279761	56.72	990.72	0	0	0	0	0
27915808	96.25	98075.03	282758	68.84	990.72	0	0	0	0	0
28212802	76.17	98132.92	285744	92.32	991.85	0	0	0	0	0

```
source env.sh
```

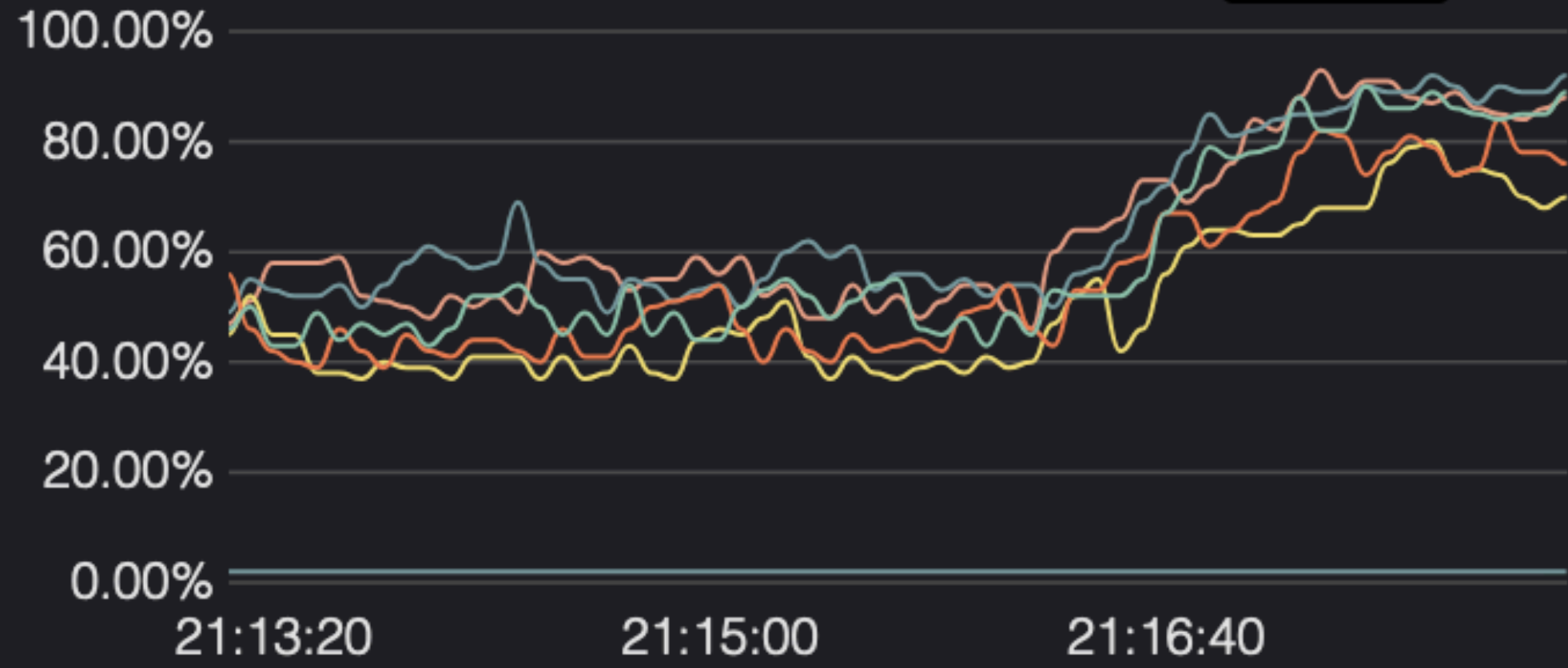
Count	Latency (p99)	1min (req/s)	Count	Latency (p99)	1min (req/s)	Count	Latency (p99)	1min (req/s)	Count	1min (errors/s)
31945	513.31	0	344	509.14	0	0	0	0	0	0
61520	509.15	10256.2	640	507.71	107.8	0	0	0	0	0
91215	506.47	10256.2	951	505.94	107.8	0	0	0	0	0
140772	489.71	10227.94	1474	486.91	107.21	0	0	0	0	0
200236	417.25	10988.58	2092	431.09	115.32	0	0	0	0	0
289369	367.62	10988.58	2958	403.55	115.32	0	0	0	0	0
388517	278.2	12476.52	3955	235.34	129.52	0	0	0	0	0
507376	145.79	12476.52	5099	167.37	129.52	0	0	0	0	0
646036	113.21	14630.39	6531	110.63	149.66	0	0	0	0	0
794734	100.51	17429.72	7956	104.12	177.35	0	0	0	0	0

Check out the metrics and logs!

CPU And Load ^

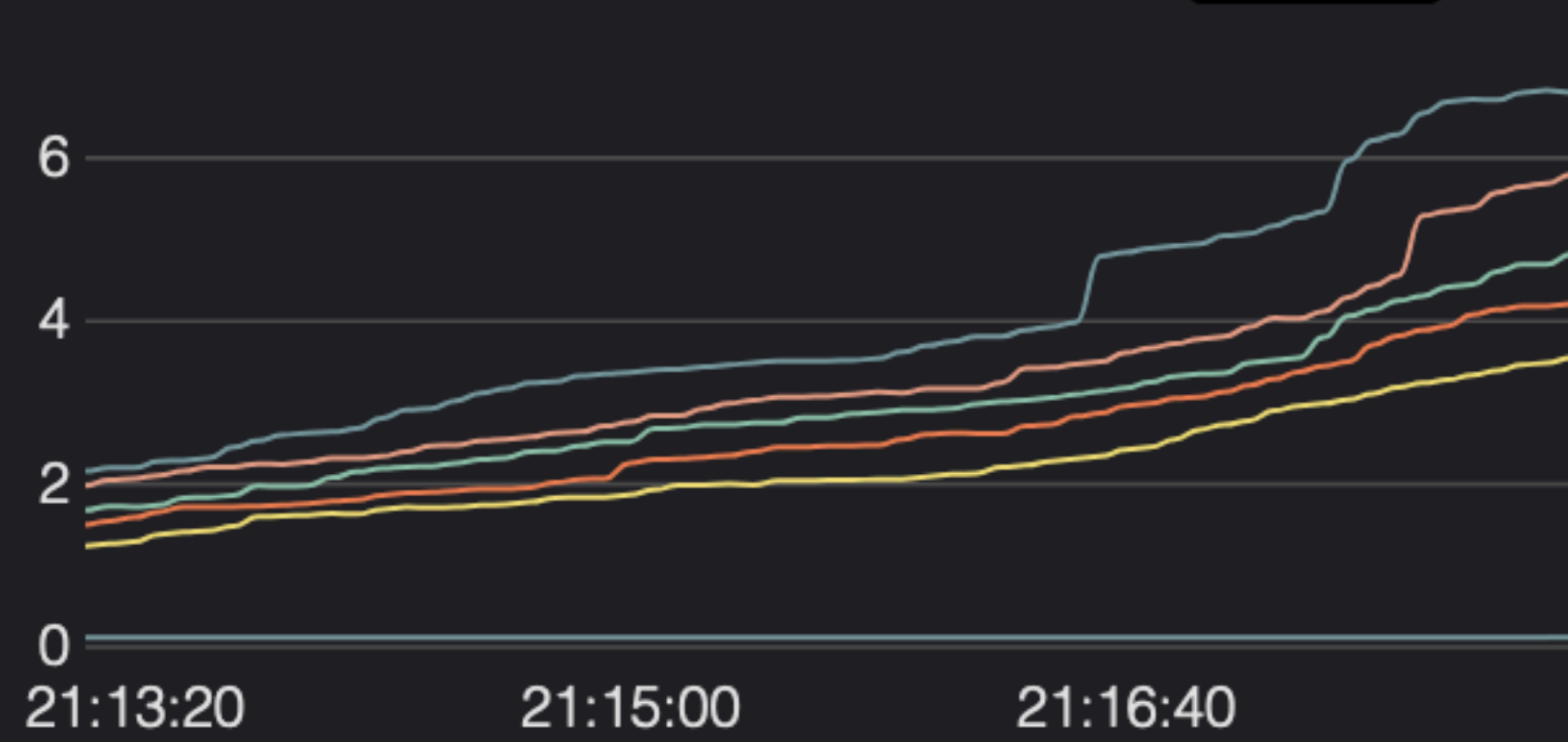
CPU usage per host

all



Load Average (15m)

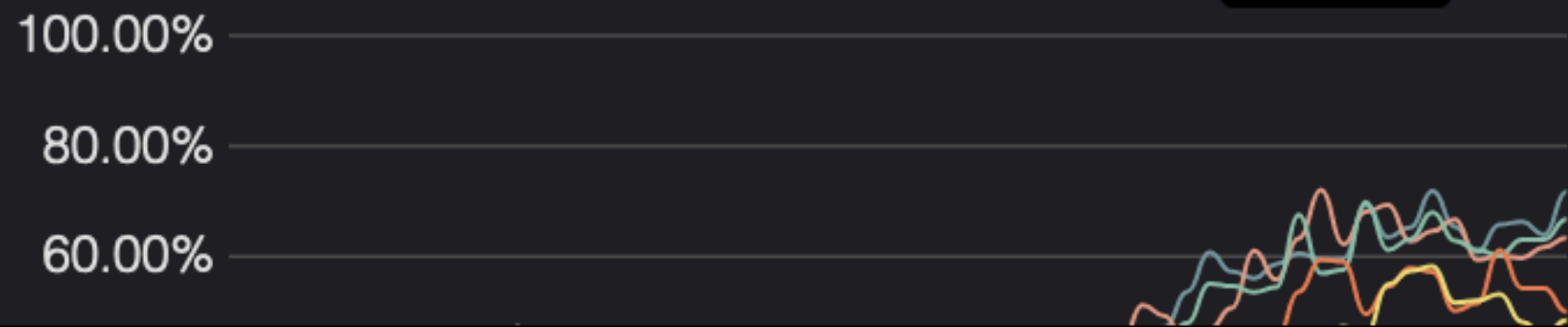
all



CPU Usage Detail ^

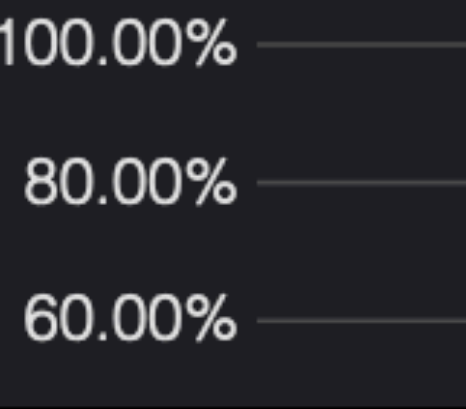
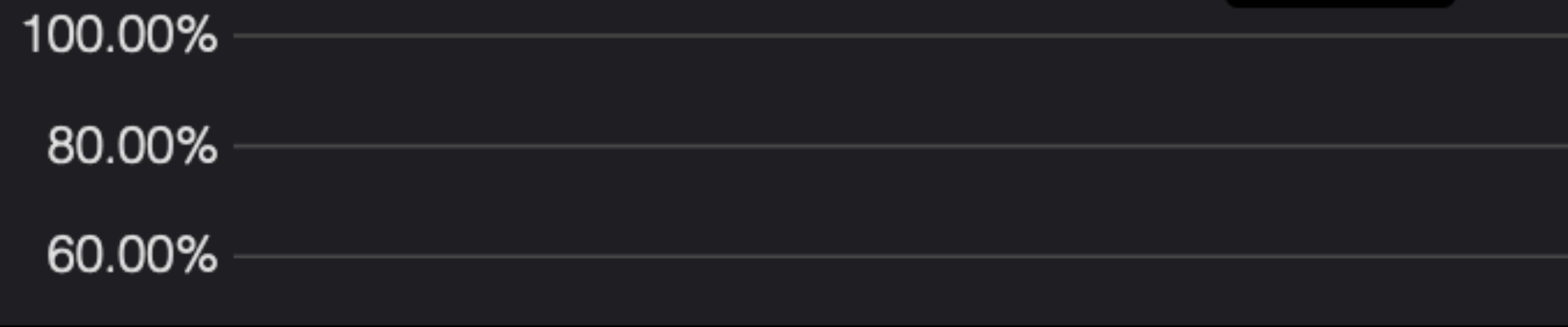
User

all



System

all



Level

Event Type

Source

Content

search...

LOGS & EVENTS

EVENTS ALERT DEFINITIONS

Data Center

Rack

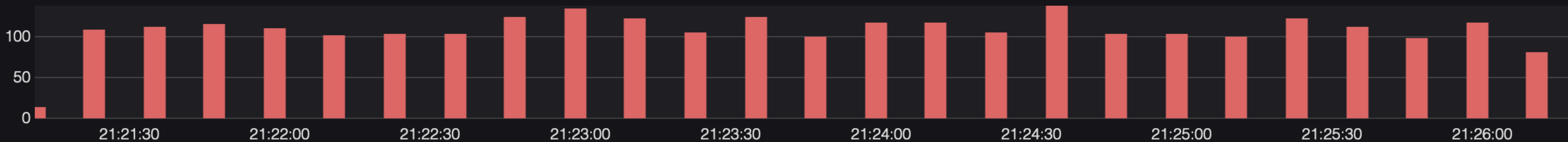
Node

Level

Event Type

Source

Content
search...

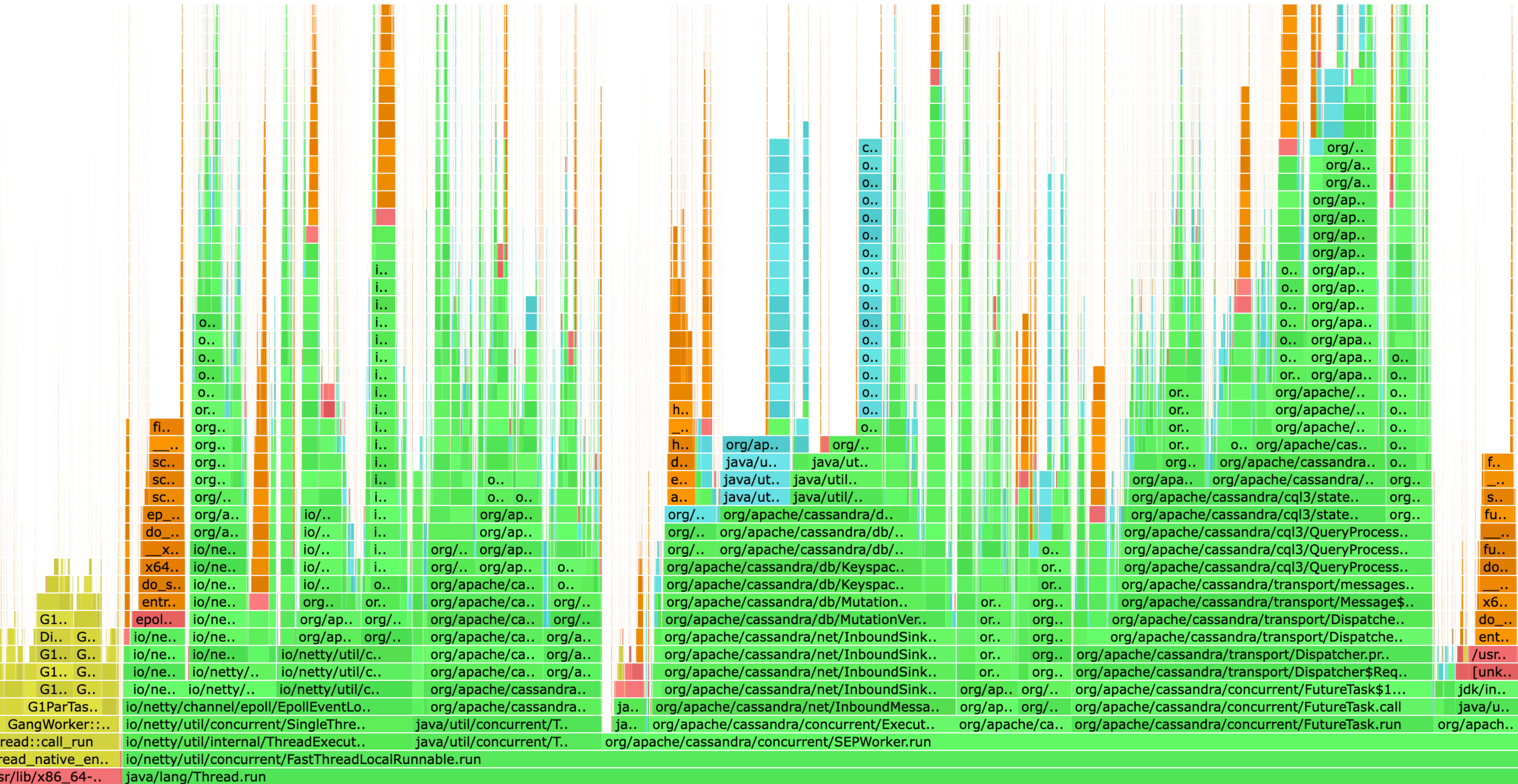


- Sep 28 21:22:28 ✓ 10.0.0.145:INFO [PerDiskMemtableFlushWriter_0:10] 2024-09-28 21:22:28,745 Flushing.java:179 - Completed flushing /mnt/cassandra/data/easy_cass_stress/sensor_data-f20569207ddd1
- Sep 28 21:22:23 ✓ 10.0.1.147:INFO [SlabPoolCleaner] 2024-09-28 21:22:23,617 ColumnFamilyStore.java:1052 - Enqueuing flush of easy_cass_stress.keyvalue, Reason: MEMTABLE_LIMIT, Usage: 241.189MiB (12%) on-he.
- Sep 28 21:22:23 ✓ 10.0.1.147:INFO [SlabPoolCleaner] 2024-09-28 21:22:23,617 AbstractAllocatorMemtable.java:293 - Flushing largest CFS(Keyspace='easy_cass_stress', ColumnFamily='keyvalue') to free up room. .
- Sep 28 21:22:23 ✓ 10.0.1.147:INFO [PerDiskMemtableFlushWriter_0:9] 2024-09-28 21:22:23,104 Flushing.java:179 - Completed flushing /mnt/cassandra/data/easy_cass_stress/sensor_data-f20569207ddd11ef8c9eb9c211.
- Sep 28 21:22:22 ✓ 10.0.0.145:INFO [SlabPoolCleaner] 2024-09-28 21:22:22,712 ColumnFamilyStore.java:1052 - Enqueuing flush of easy_cass_stress.random_access, Reason: MEMTABLE_LIMIT, Usage: 312.864MiB (16%) .
- Sep 28 21:22:22 ✓ 10.0.0.145:INFO [SlabPoolCleaner] 2024-09-28 21:22:22,711 AbstractAllocatorMemtable.java:293 - Flushing largest CFS(Keyspace='easy_cass_stress', ColumnFamily='random_access') to free up r.
- Sep 28 21:22:22 ✓ 10.0.0.143:INFO [PerDiskMemtableFlushWriter_0:11] 2024-09-28 21:22:22,437 Flushing.java:153 - Writing Memtable-sensor_data@1910303443(94.568MiB serialized bytes, 468326 ops, 293.709MiB (1.
- Sep 28 21:22:22 ✓ 10.0.0.143:INFO [SlabPoolCleaner] 2024-09-28 21:22:22,353 ColumnFamilyStore.java:1052 - Enqueuing flush of easy_cass_stress.sensor_data, Reason: MEMTABLE_LIMIT, Usage: 293.690MiB (15%) on.
- Sep 28 21:22:22 ✓ 10.0.0.143:INFO [SlabPoolCleaner] 2024-09-28 21:22:22,353 AbstractAllocatorMemtable.java:293 - Flushing largest CFS(Keyspace='easy_cass_stress', ColumnFamily='sensor_data') to free up roo.
- Sep 28 21:22:22 ✓ 10.0.0.145:INFO [PerDiskMemtableFlushWriter_0:10] 2024-09-28 21:22:22,008 Flushing.java:179 - Completed flushing /mnt/cassandra/data/easy_cass_stress/sensor_data-f20569207ddd11ef8c9eb9c21.
- Sep 28 21:22:21 ✓ 10.0.1.147:time= 2024-09-28T21:22:21Z level=info msg= agent waiting for component bootstrapping...
- Sep 28 21:22:21 ✓ 10.0.0.145:time= 2024-09-28T21:22:21Z level=info msg= agent waiting for component bootstrapping...
- Sep 28 21:22:20 ✓ 10.0.1.147:INFO [PerDiskMemtableFlushWriter_0:9] 2024-09-28 21:22:20,771 Flushing.java:153 - Writing Memtable-sensor_data@1936044305(70.060MiB serialized bytes, 346176 ops, 224.054MiB (12.
- Sep 28 21:22:20 ✓ 10.0.1.147:INFO [SlabPoolCleaner] 2024-09-28 21:22:20,695 ColumnFamilyStore.java:1052 - Enqueuing flush of easy_cass_stress.sensor_data, Reason: MEMTABLE_LIMIT, Usage: 224.049MiB (12%) on.
- Sep 28 21:22:20 ✓ 10.0.1.147:INFO [SlabPoolCleaner] 2024-09-28 21:22:20,695 AbstractAllocatorMemtable.java:293 - Flushing largest CFS(Keyspace='easy_cass_stress', ColumnFamily='sensor_data') to free up roo.

Built in Performance Tooling

Flame Graphs!!

```
$ c-flame cassandra0
```



```
root@cassandra0:~# bio latency 1 10
```

```
Tracing block device I/O... Hit Ctrl-C to end.
```

usecs		:	count	distribution
0	-> 1	:	0	
2	-> 3	:	0	
4	-> 7	:	0	
8	-> 15	:	0	
16	-> 31	:	0	
32	-> 63	:	3	
64	-> 127	:	315	*****
128	-> 255	:	105	*****
256	-> 511	:	14	*
512	-> 1023	:	1	

What else can we test?

- Any release (3.0, 3.11, 4.0, 4.1, 5.0)
- Any java version
- Any branch or commit
- Upgrades



How we use it
Real World Examples



[Cassandra](#) / [CASSANDRA-19477](#)

Do not go to disk to get HintsStore.getTotalFileSize

[Edit](#)

[Add comment](#)

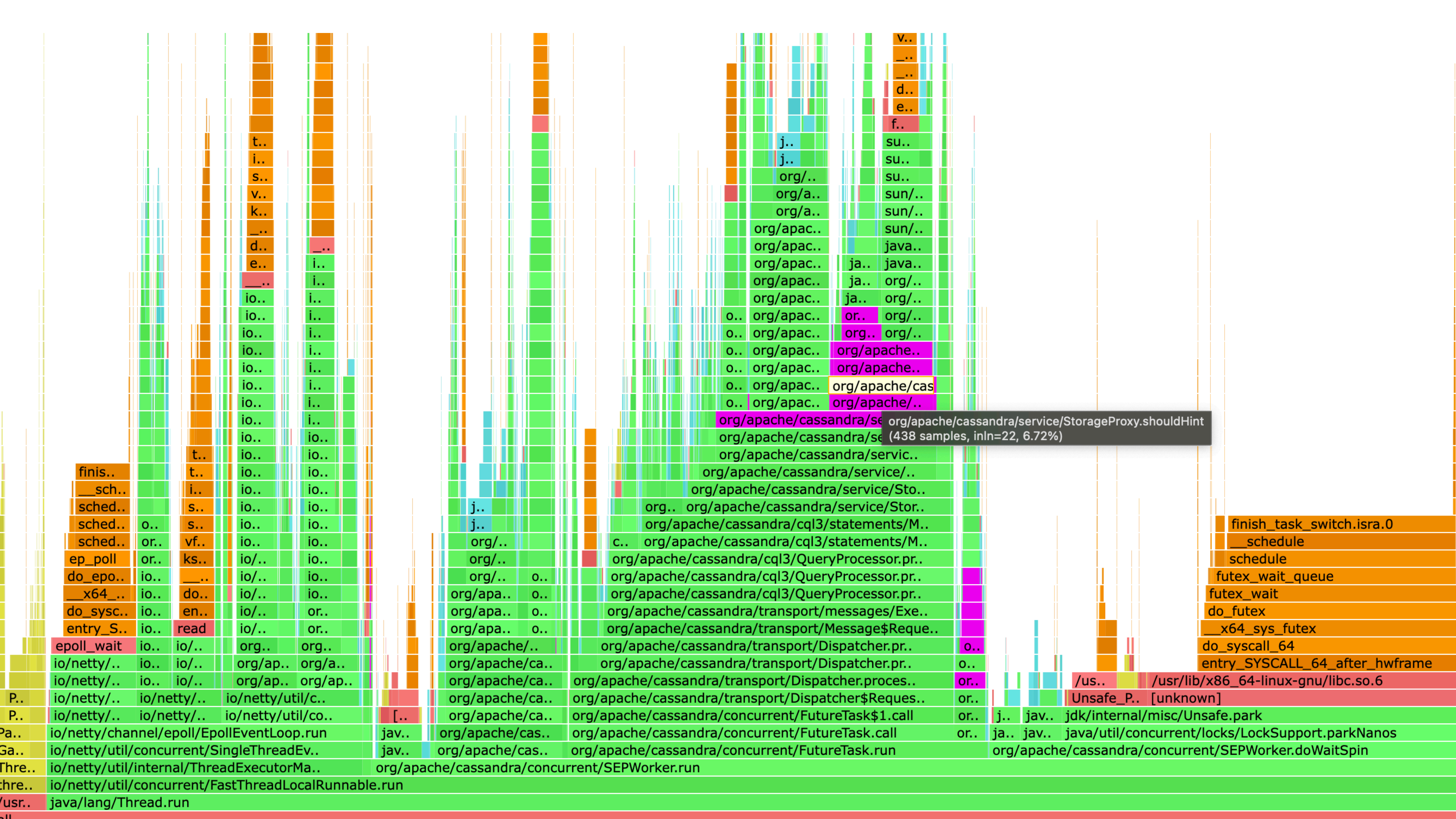
[Assign](#)

[More](#)

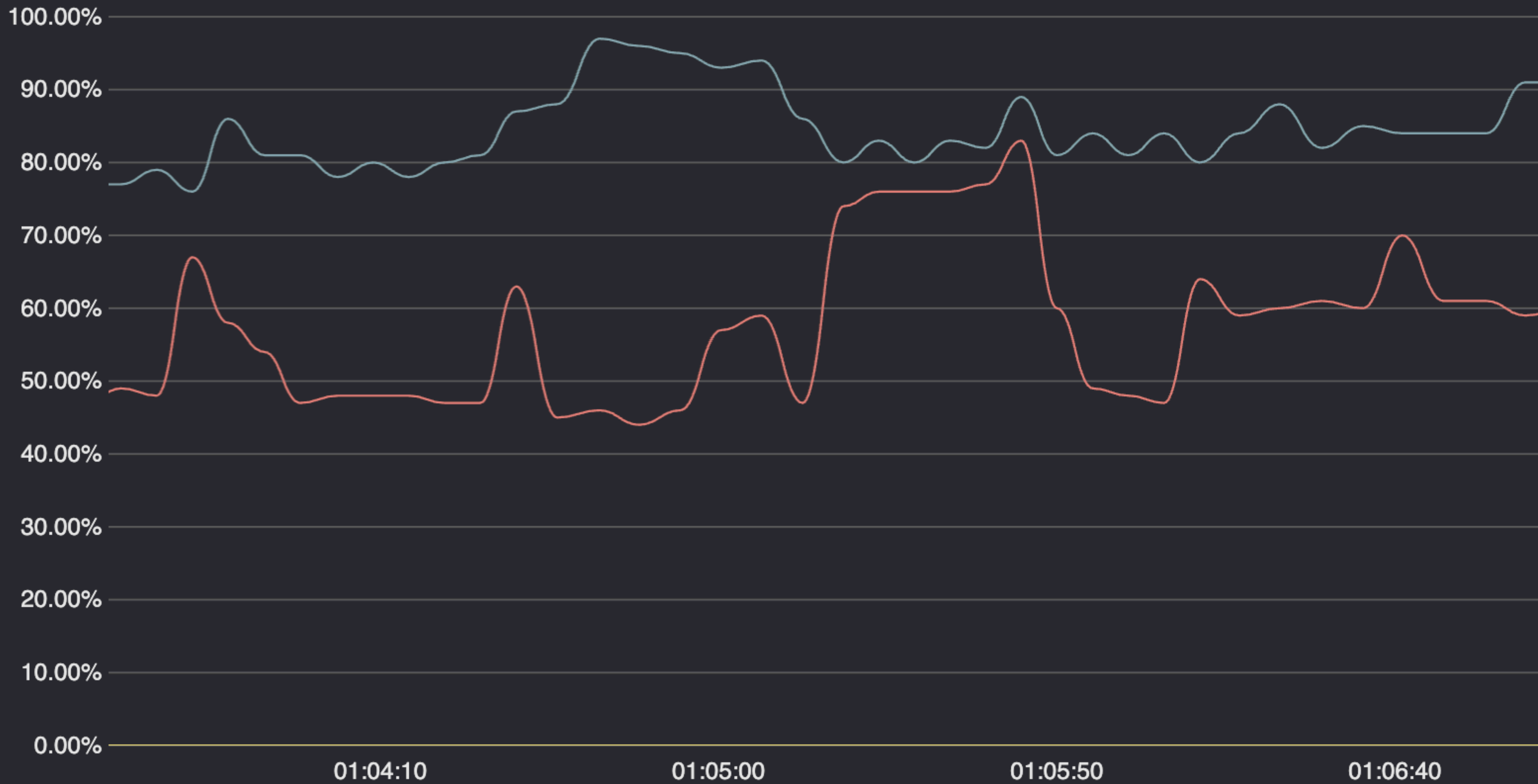
[Reopen](#)

▼ Details

Type:	Bug	Status:	RESOLVED
Priority:	Normal	Resolution:	Fixed
Component/s:	Consistency/Hints	Fix Version/s:	4.1.5 , 5.0-rc1 , 5.0 , 5.1-alpha1
Labels:	None		
Bug Category:	Code		
Severity:	Normal		
Complexity:	Normal		
Discovered By:	Adhoc Test		
Platform:	All		



CPU usage per host





[Cassandra](#) / [CASSANDRA-19534](#)

Unbounded queues in native transport requests lead to node instability

- [Edit](#)
- [Add comment](#)
- [Assign](#)
- [More](#) ▾
- [Reopen](#)

▼ Details

Type:	Bug	Status:	RESOLVED
Priority:	Urgent	Resolution:	Fixed
Component/s:	Legacy/Local Write-Read Paths	Fix Version/s:	4.1.6 , 5.0-rc1 , 5.0 , 5.1
Labels:	None		
Bug Category:	Availability		
Severity:	Critical		
Complexity:	Challenging		
Discovered By:	User Report		
Platform:	All		
Impacts:	None		
Since Version:	3.0.0		
Source Control Link:	https://github.com/apache/cassandra/commit/dc17c29724d86547538cc8116ff1a90d36a0bf3a		
Test and	▼ Includes tests, also was tested separately; screenshots and description attached		



Improve disk access patterns during compaction and streaming (big format)

▼ Details

Type:	Improvement	Status:	OPEN
Priority:	Normal	Resolution:	Unresolved
Component/s:	Legacy/Local Write-Read Paths, ... (1)	Fix Version/s:	None
Labels:	None		
Change Category:	Performance		
Complexity:	Normal		
Platform:	All		
Impacts:	None		

▼ Description

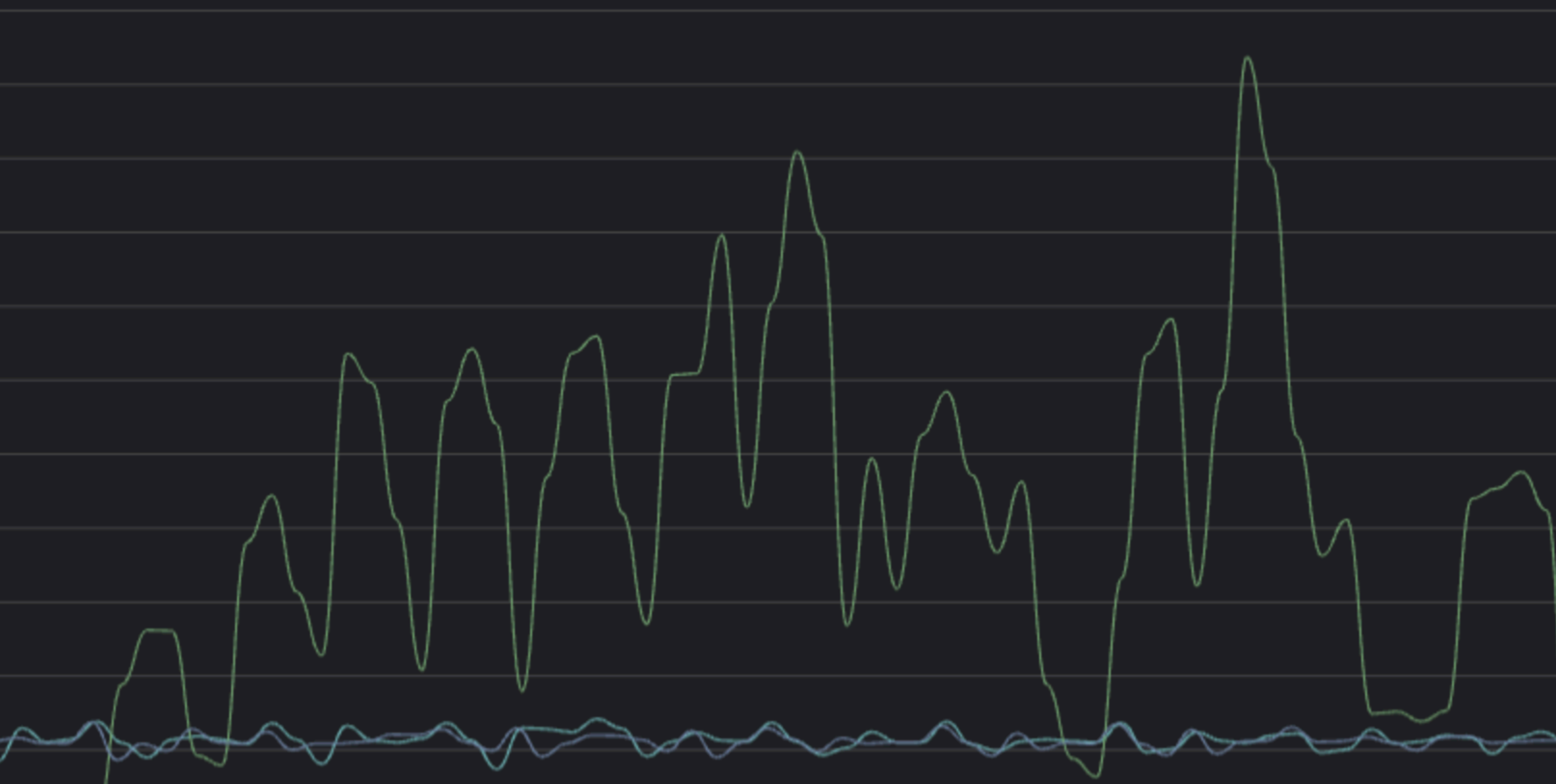
On read heavy workloads Cassandra performs much better when using a low read ahead setting. In my tests I've seen an 5x improvement in throughput and more than a 50% reduction in latency. However, I've also observed that it can have a negative impact on compaction and streaming throughput. It especially negatively impacts cloud environments where small reads incur hi costs in IOPS due to tiny requests

Here's a sample of what the I/O looks like at the block device level when running compact:

```
ubuntu@ip-172-31-38-58:~$ sudo /usr/share/bcc/tools/biosnoop -d xvdb | awk '$5 == "R" { print $0 }'
```

TIME(s)	COMM	PID	DISK	T	SECTOR	BYTES	LAT(ms)
0.000000	CompactionExec	26988	xvdb	R	48340000	16384	0.26
0.002350	CompactionExec	26988	xvdb	R	48339842	512	0.23
0.003560	CompactionExec	26988	xvdb	R	48339872	4096	0.22
0.003788	CompactionExec	26988	xvdb	R	48339864	4096	0.18
0.004550	CompactionExec	26988	xvdb	R	48339841	512	0.16
0.004719	CompactionExec	26988	xvdb	R	48339843	512	0.14
0.004906	CompactionExec	26988	xvdb	R	48339856	4096	0.15
0.005077	CompactionExec	26988	xvdb	R	48339848	4096	0.14
0.005304	CompactionExec	26988	xvdb	R	48340288	16384	0.17
0.009510	CompactionExec	26988	xvdb	R	48340320	16384	0.19
0.016991	NonPeriodicTas	26988	xvdb	R	32258112	16384	0.20
0.028200	CompactionExec	26988	xvdb	R	32262176	16384	0.27
0.029466	CompactionExec	26988	xvdb	R	32258944	16384	0.26
0.031511	CompactionExec	26988	xvdb	R	32226562	512	0.21
0.038502	CompactionExec	26988	xvdb	R	32226592	4096	0.22
0.038848	CompactionExec	26988	xvdb	R	32226584	4096	0.24
0.039842	CompactionExec	26988	xvdb	R	32226561	512	0.14
0.040079	CompactionExec	26988	xvdb	R	32226563	512	0.14

Bytes Read Per Second



How to get it?



easy-cass-lab Public

Unpin Unwatch 5 Fork 7 Starred 10

main 12 Branches 10 Tags

Go to file Add file

rustyrazorblade upgraded bcc tools 9250151 · 3 weeks ago

.circleci	commented out docker layer caching, not supported in fr...
bin	Fix for APP_HOME
config	first commit, yay
core	Convert project to use Kotlin for Gradle.
dashboards	Lots of little improvements
docker-grafonnet	fixed build
docs	More renaming
gradle/wrapper	Updated to Gradle 8.9
manual	Project cleanup and support for unrestricted SSH
packer	upgraded bcc tools



Releases 7

v7 Latest 3 weeks ago

```
brew tap rustyrazorblade/rustyrazorblade  
brew install easy-cass-lab
```



We also have a training program!

I like to look at the ROI of things I put money into, and in this case it's just too high to calculate.

Was already able to solve major issues for my customers, following the things I picked up here (and I don't think I was newbie, to say the least..), so again, thank you.

Thank you!
Questions?

