

Storing 16 Bytes at Scale

Fabian Reinartz

Software Engineer

@fabxc

We're hiring: coreos.com/careers

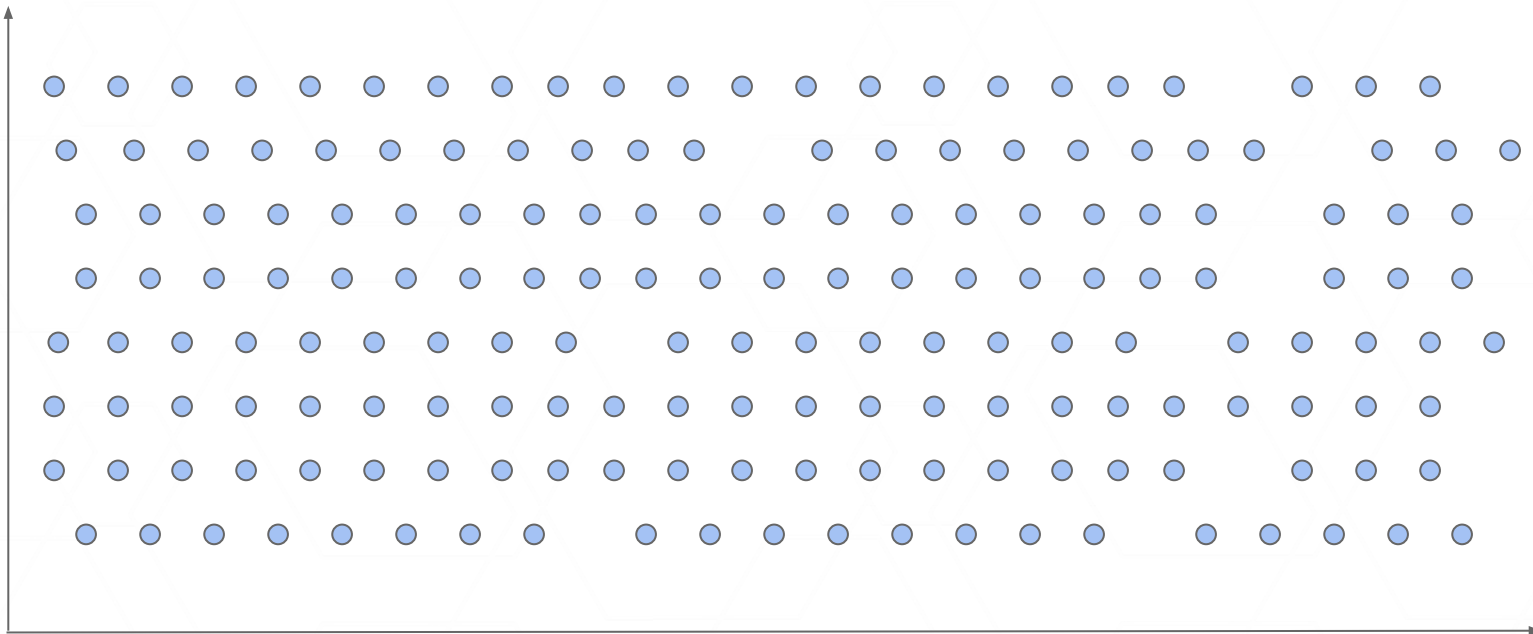
Time Series



time

Time Series

series



time

Time Series

```
requests_total{path="/status", method="GET", instance="10.0.0.1:80"}
```

```
requests_total{path="/status", method="POST", instance="10.0.0.3:80"}
```

```
requests_total{path="/", method="GET", instance="10.0.0.2:80"}
```

...

Time Series

```
requests_total{path="/status", method="GET", instance="10.0.0.1:80"}
```

```
requests_total{path="/status", method="POST", instance="10.0.0.3:80"}
```

```
requests_total{path="/", method="GET", instance="10.0.0.2:80"}
```

Select: *requests_total*

Time Series

```
requests_total{path="/status", method="GET", instance="10.0.0.1:80"}
```

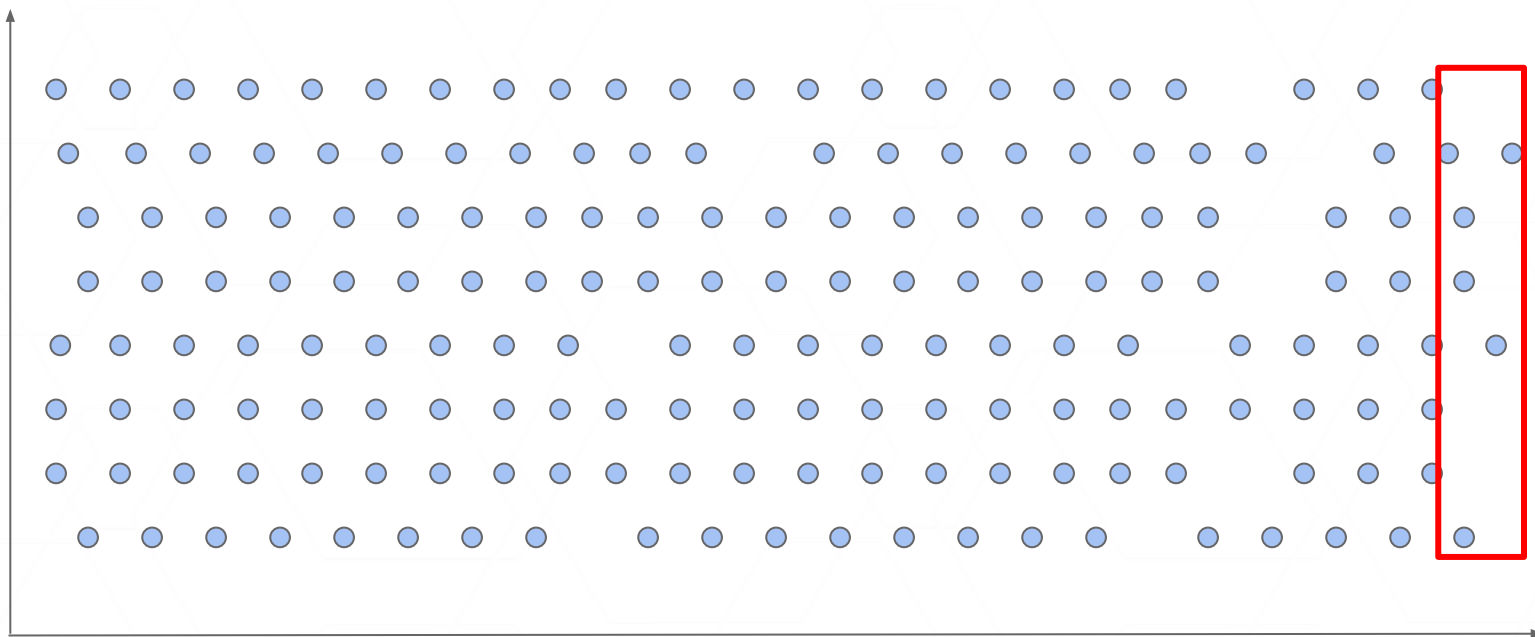
```
requests_total{path="/status", method="POST", instance="10.0.0.3:80"}
```

```
requests_total{path="/", method="GET", instance="10.0.0.2:80"}
```

Select: *requests_total{method="GET"}*

Time Series

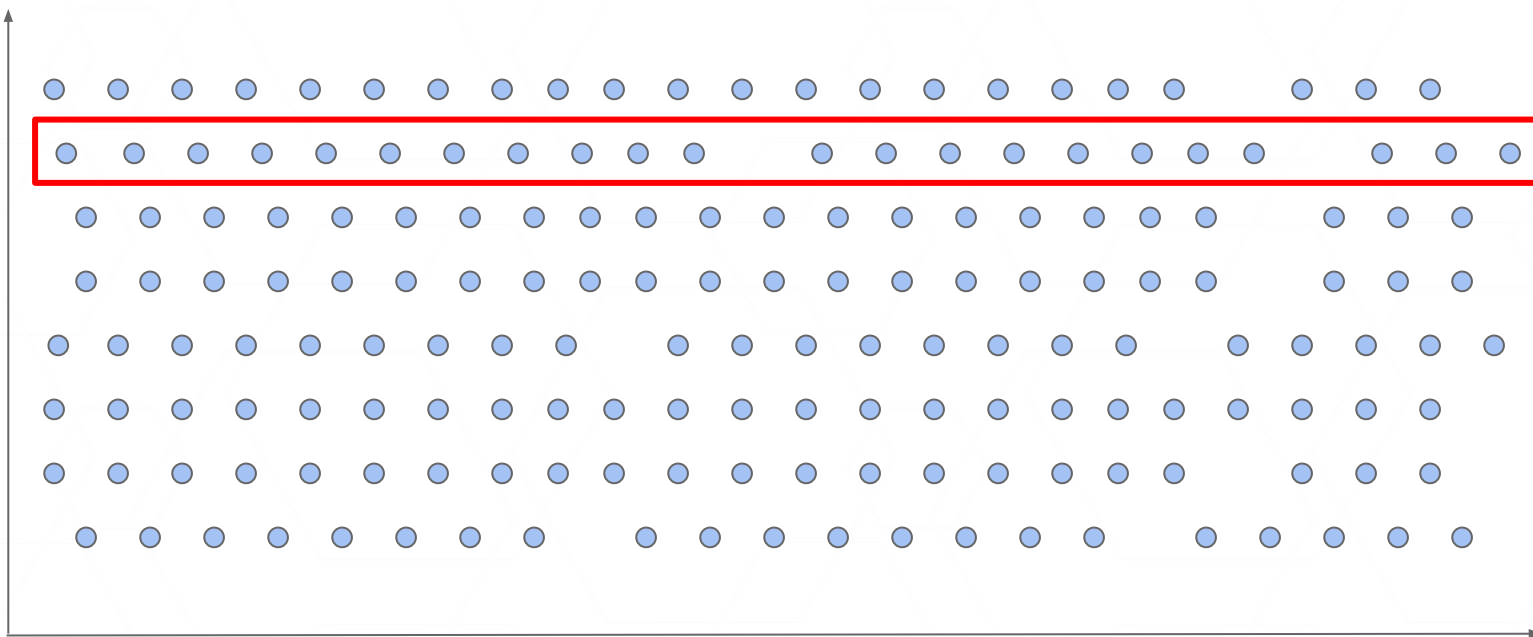
series



time

Time Series

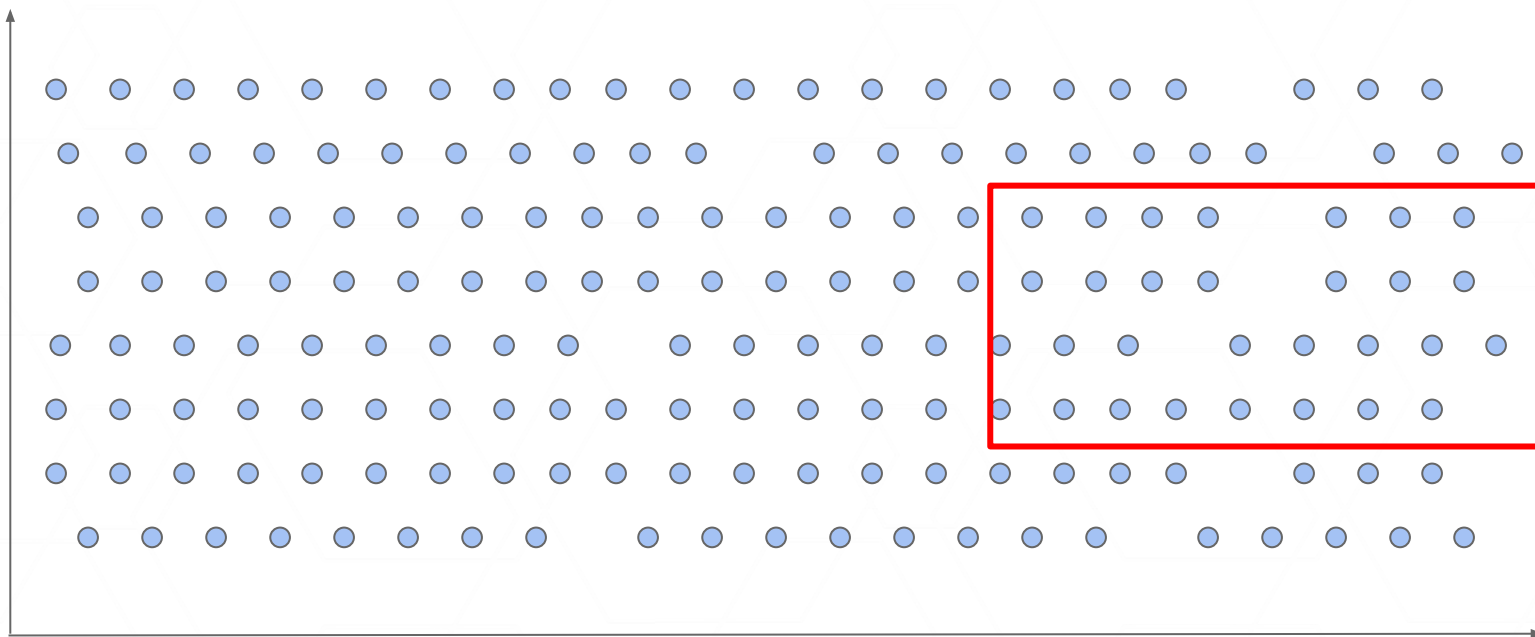
series



time

Time Series

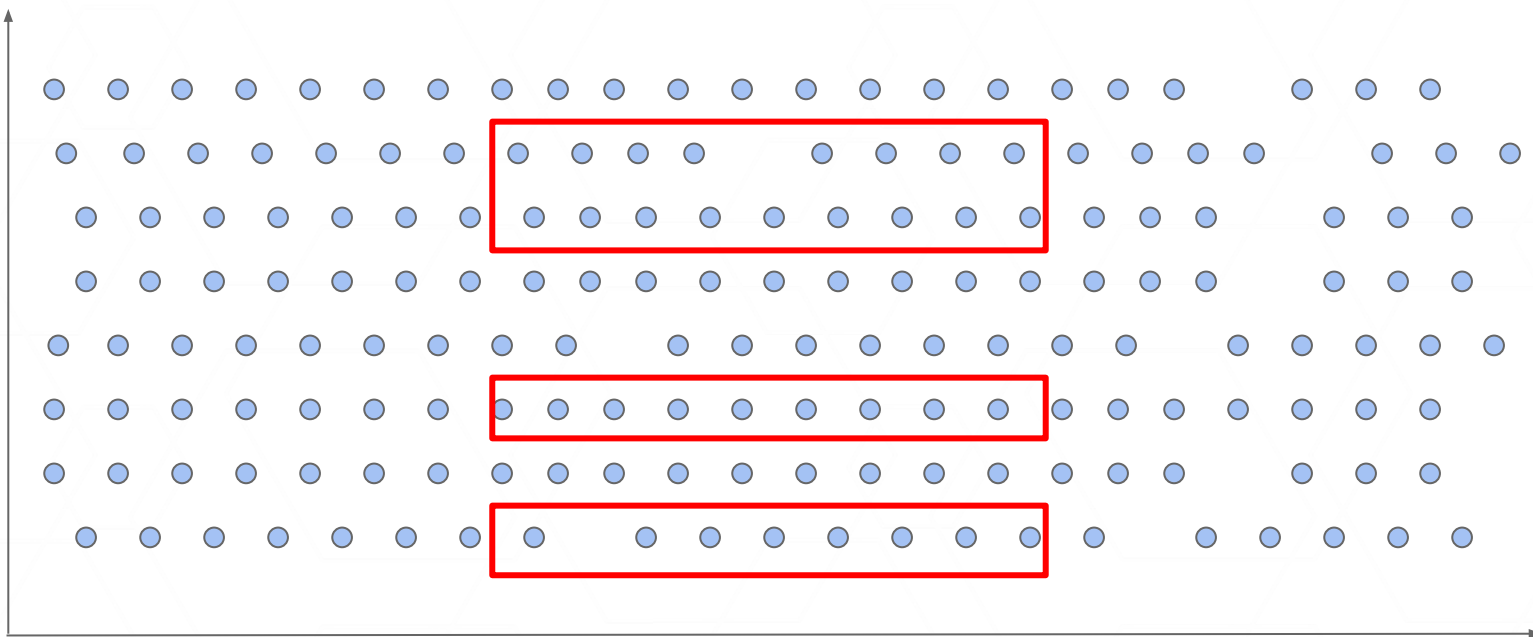
series



time

Time Series

series



time

Order of Scale

5 million active time series
30 second scrape interval
1 month of retention

← 2,000 - 15,000
targets

166,000 samples/second
432 billion samples

Order of Scale

5 million active time series
30 second scrape interval
1 month of retention

← 2,000 - 15,000
targets

166,000 samples/second
432 billion samples

8 byte timestamp + 8 byte value \Rightarrow 7 TB on disk

← seems OK

Order of Scale

5 million active time series

30 second scrape interval

6 month of retention

← 2,000 - 15,000
targets

166,000 samples/second

2592 billion samples

8 byte timestamp + 8 byte value \Rightarrow **42** TB on disk

Sample Compression – Timestamps

1496163646

1496163676

1496163706

1496163735

1496163765

Sample Compression – Timestamps

1496163646

1496163676

1496163706

1496163735

1496163765



1496163646

+30

+30

+29

+30

Sample Compression – Timestamps

1496163646

1496163676

1496163706

1496163735

1496163765

Δ

1496163646

+30

+30

+29

+30

ΔΔ

1496163646

+30

+0

-1

+1

Sample Compression – Values

| Decimal | Double Representation | XOR with previous |
|---------|-----------------------|--------------------|
| 12 | 0x4028000000000000 | |
| 24 | 0x4038000000000000 | 0x0010000000000000 |
| 15 | 0x402e000000000000 | 0x0016000000000000 |
| 12 | 0x4028000000000000 | 0x0006000000000000 |
| 35 | 0x4041800000000000 | 0x0069800000000000 |

| Decimal | Double Representation | XOR with previous |
|---------|-----------------------|--------------------|
| 15.5 | 0x402f000000000000 | |
| 14.0625 | 0x402c200000000000 | 0x0003200000000000 |
| 3.25 | 0x400a000000000000 | 0x0026200000000000 |
| 8.625 | 0x4021400000000000 | 0x002b400000000000 |
| 13.1 | 0x402a333333333333 | 0x000b733333333333 |

Sample Compression

Raw: 16 bytes/sample

Compressed: 1.37 bytes/sample



@beorn7



@dgryski

Order of Scale

5 million active time series
30 second scrape interval
1 month of retention

← 2,000 - 15,000
targets

166,000 samples/second
432 billion samples

8 byte timestamp + 8 byte value \Rightarrow ~~7 TB~~ on disk

0.8 TB

Storing ^{1.37}16 Bytes at Scale

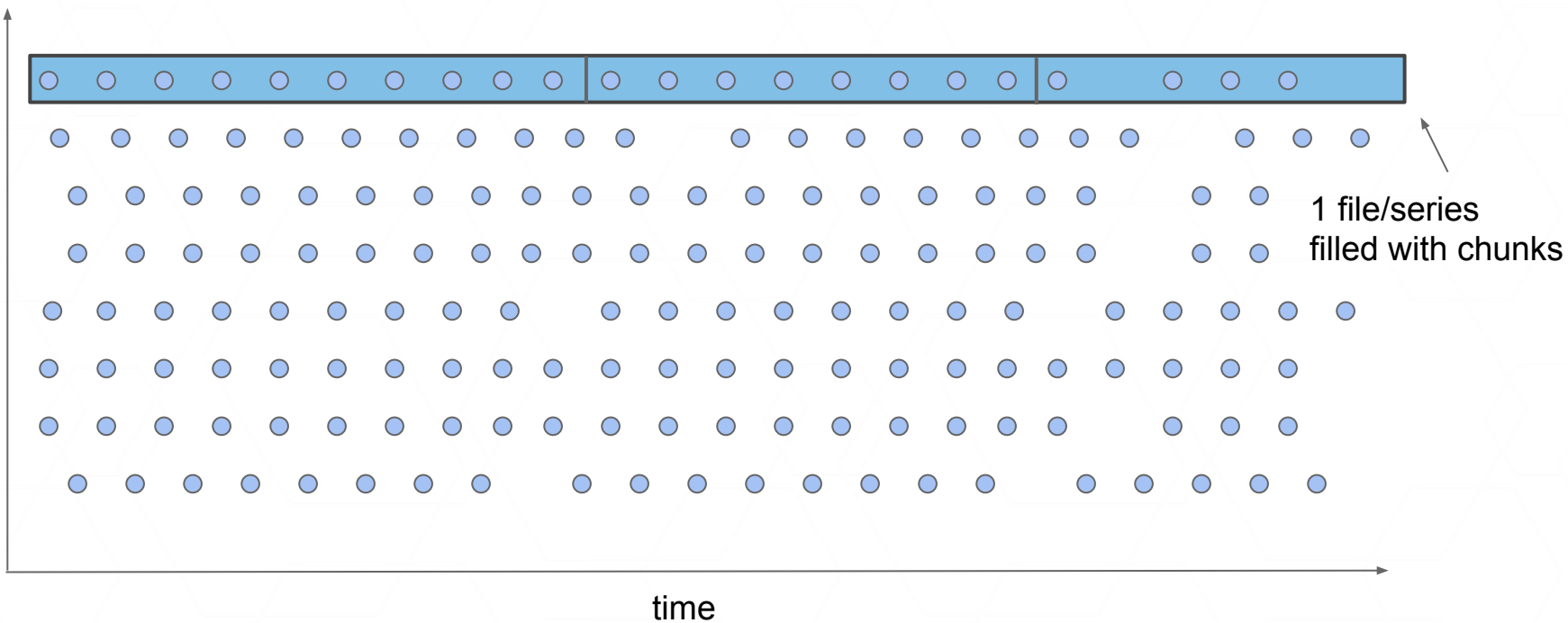
Fabian Reinartz

Software Engineer, CoreOS

@fabxc

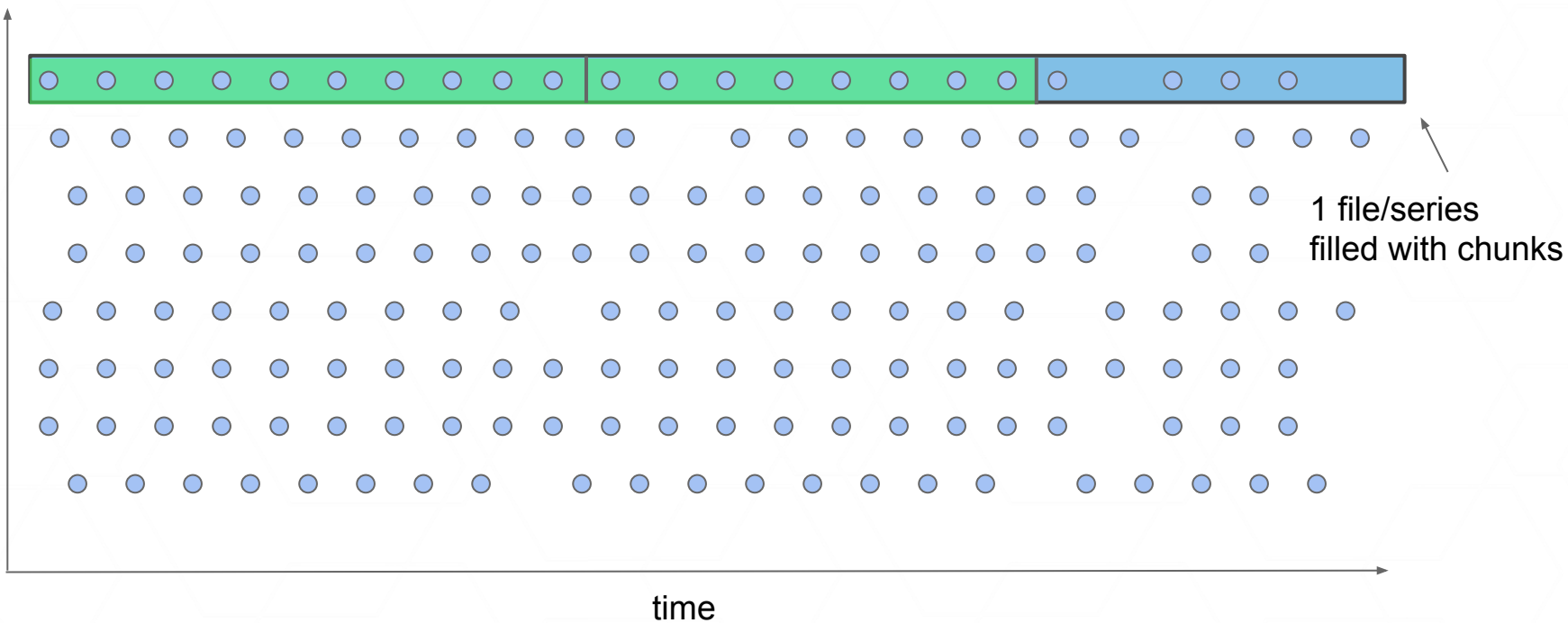
Time Series – Chunks

series



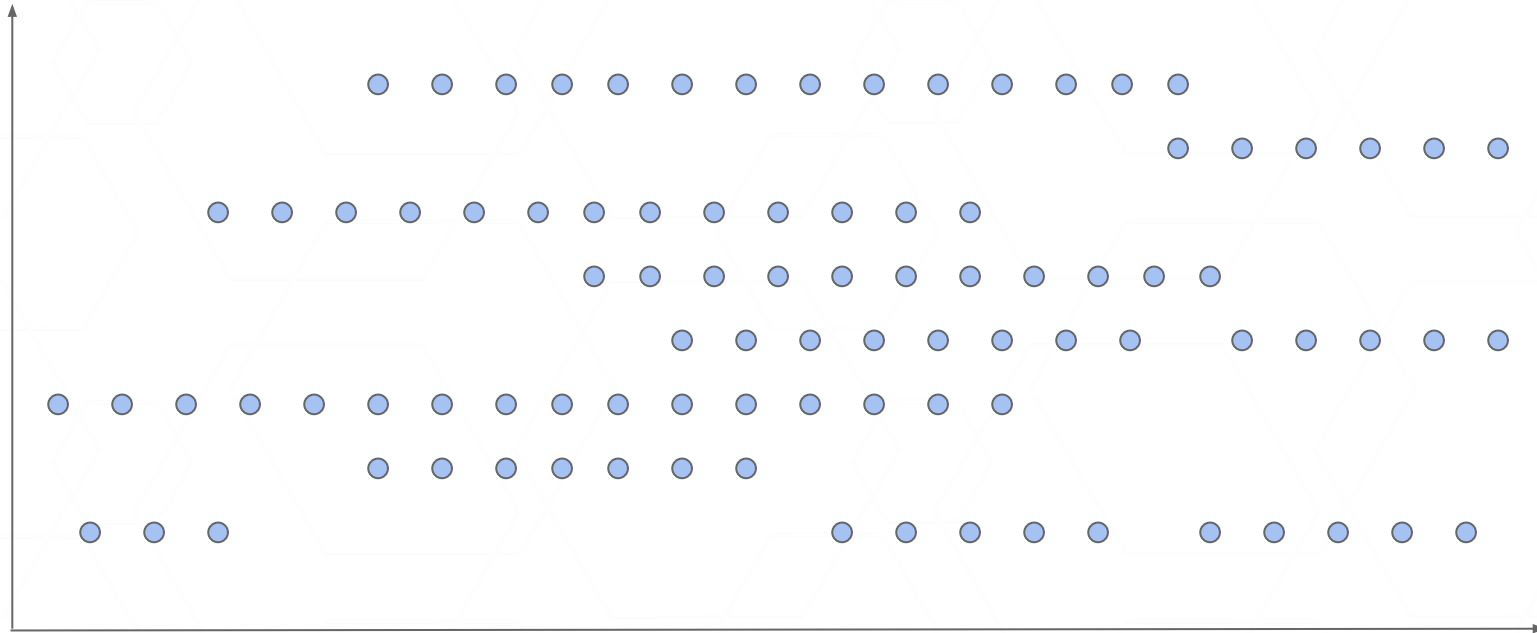
Time Series – Chunks

series



Churn

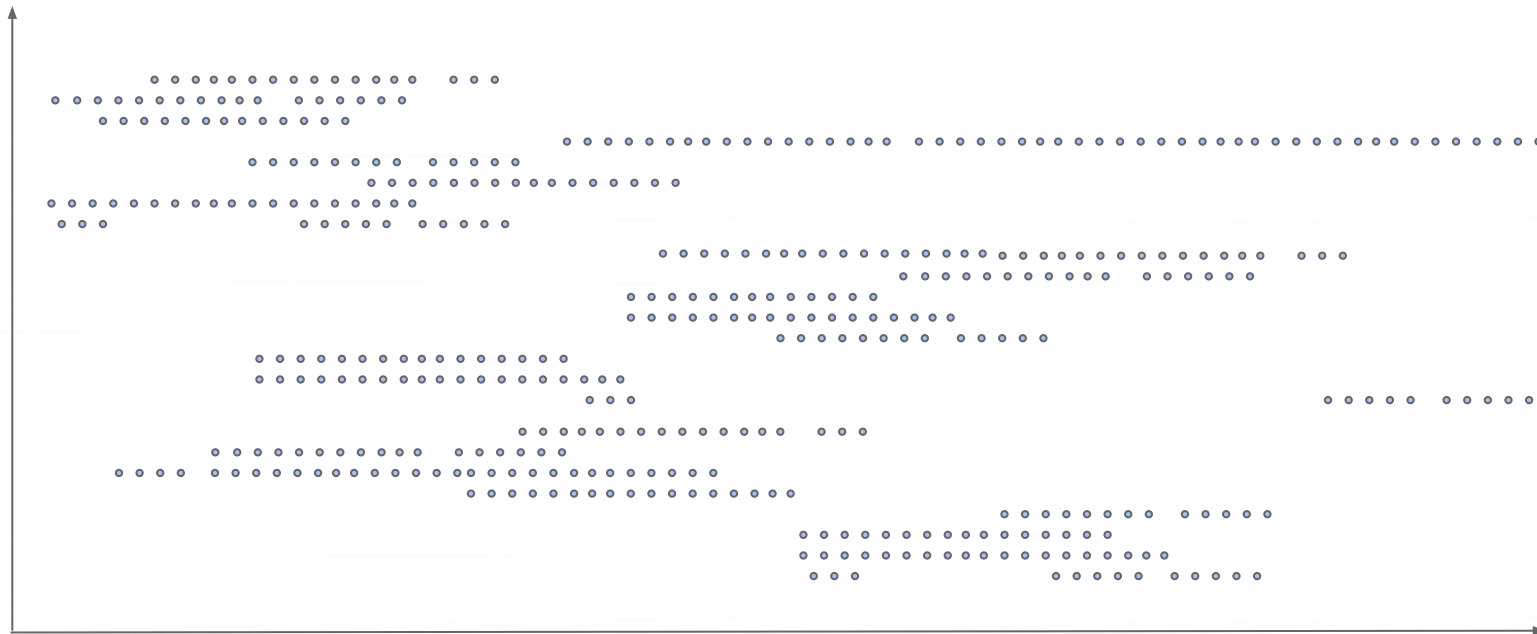
series



time

Churn

series



time

Order of Scale – with Churn

5 million **active** time series

150 million total time series

30 second scrape interval

1 month of retention

166,000 samples/second

432 billion samples

Churn

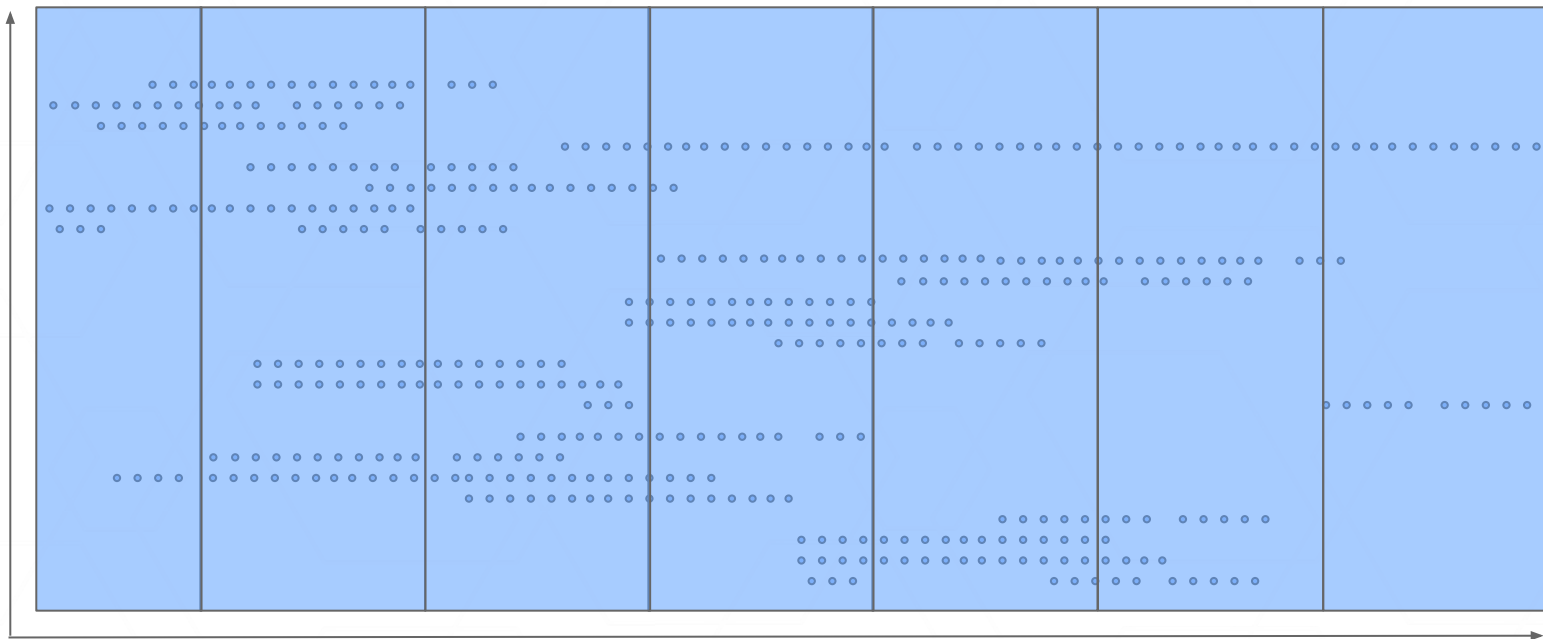
series



time

Blocks

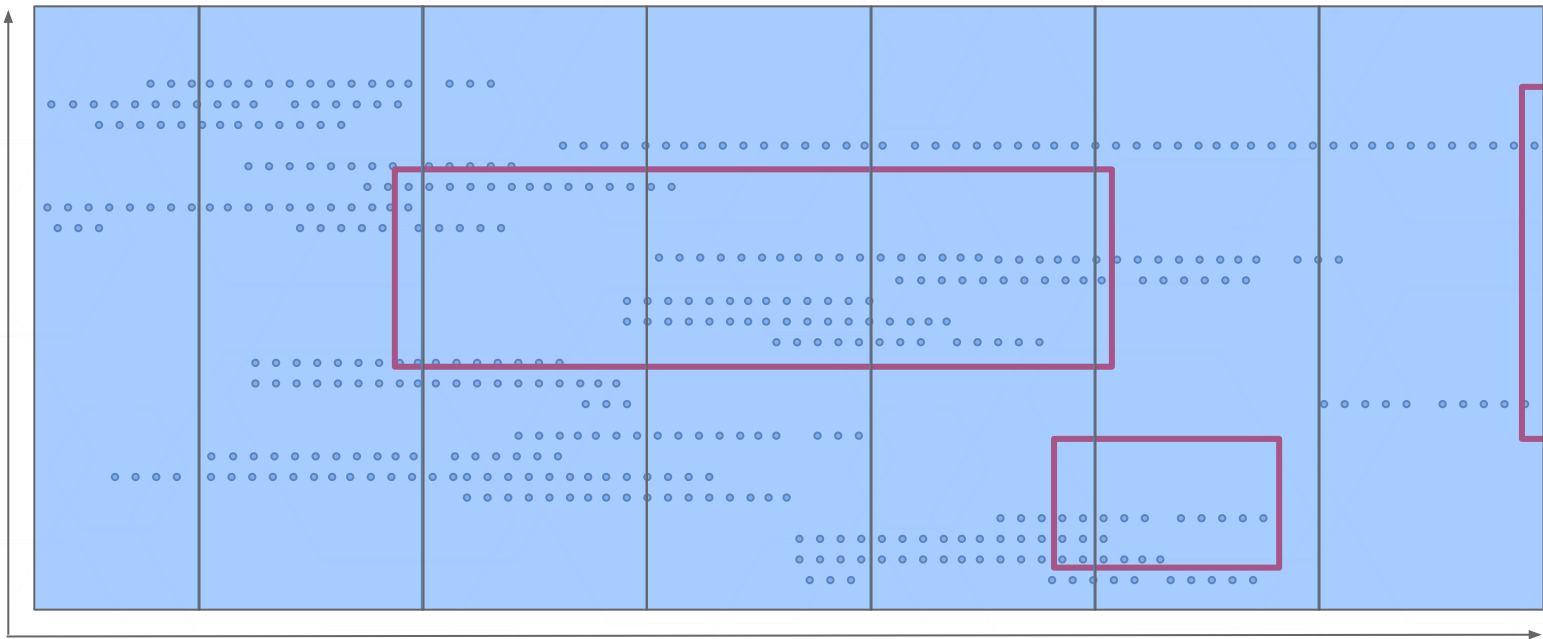
series



time

Blocks

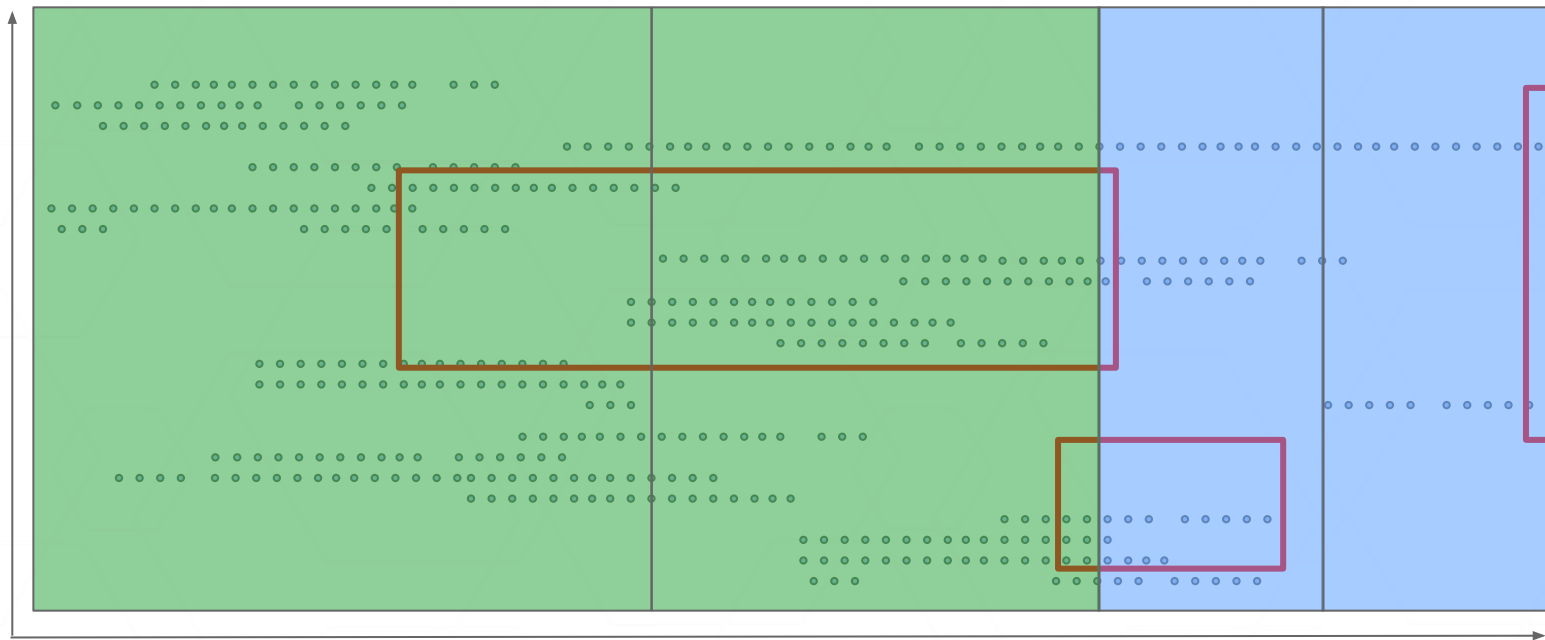
series



time

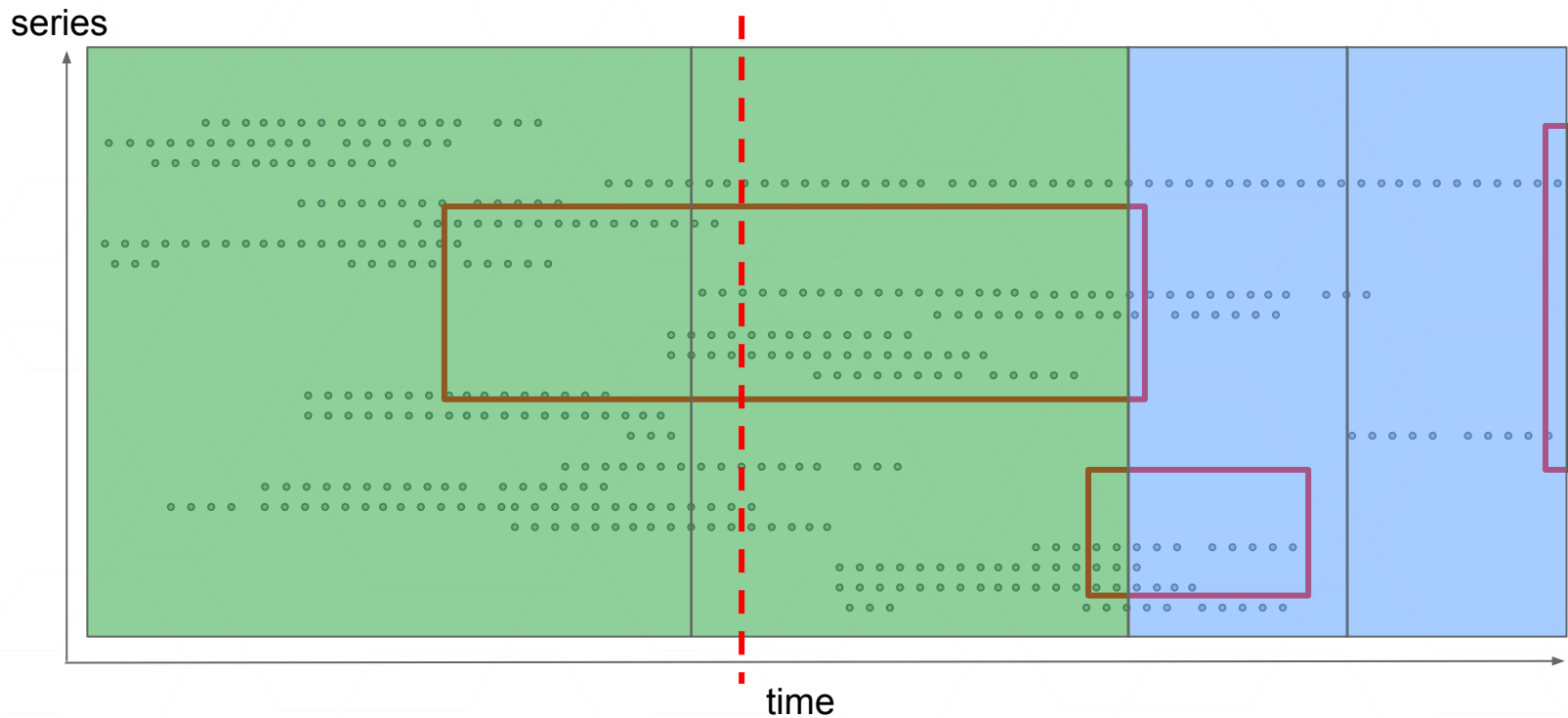
Compaction

series

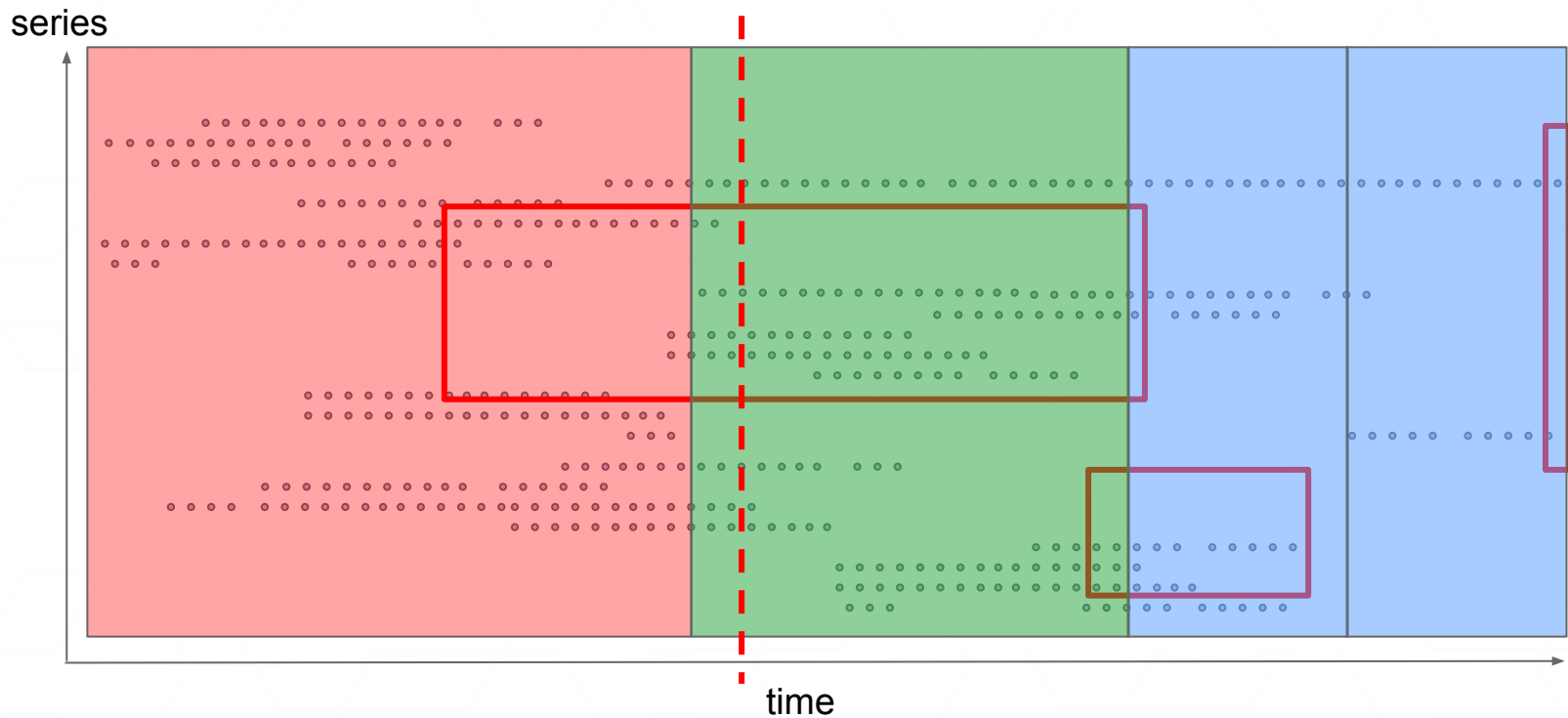


time

Retention



Retention



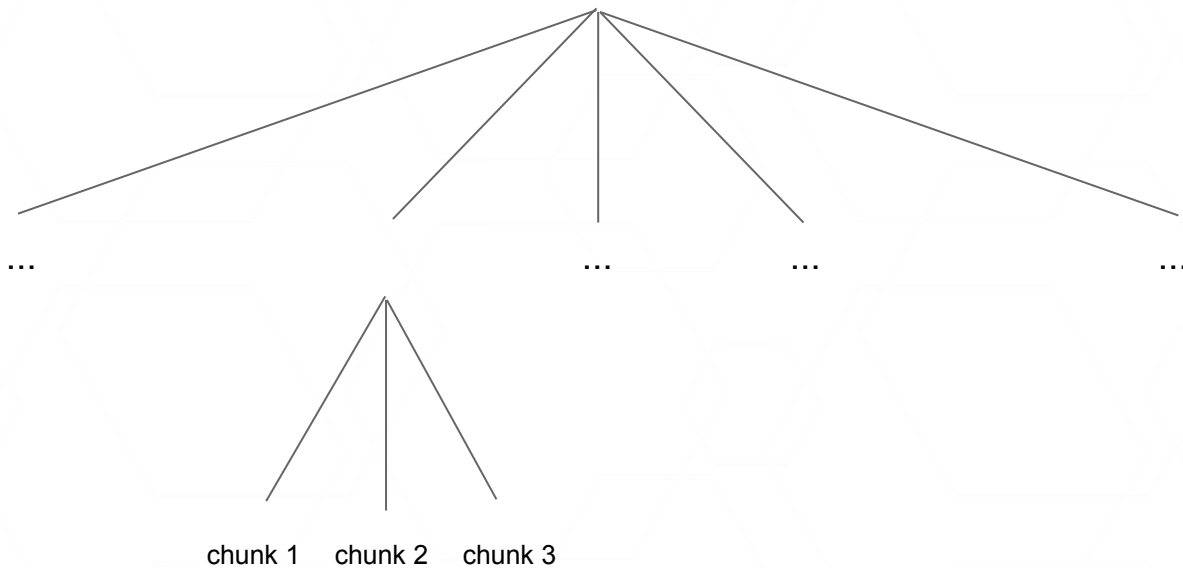
Blocks

“OMG, this is so dumb. That’s basically an LSM tree.

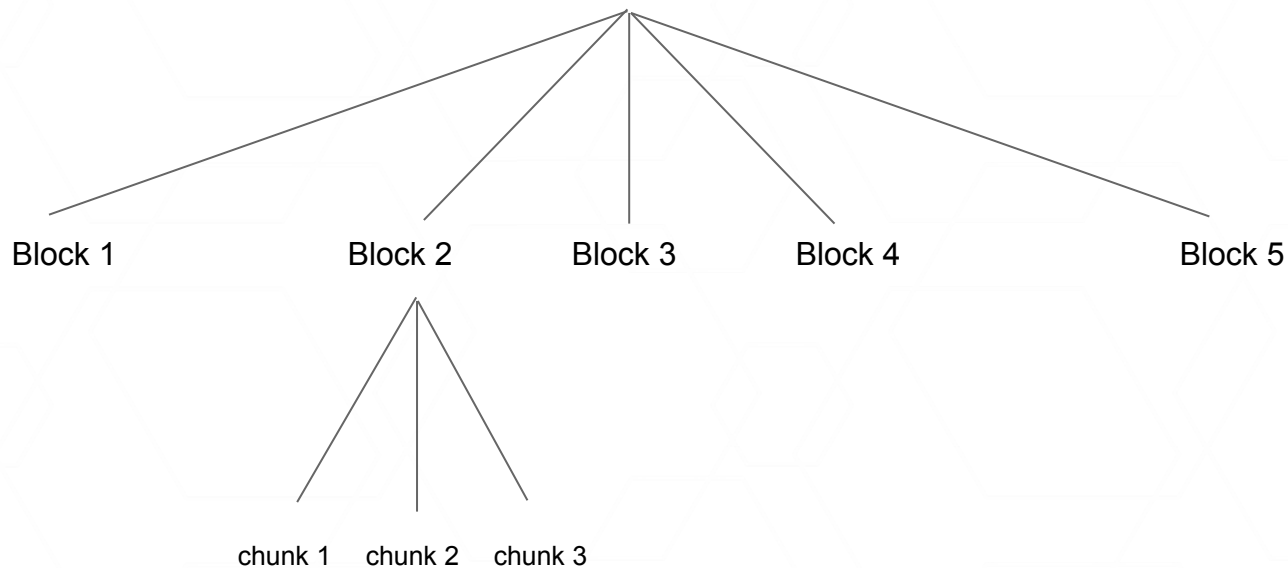
Time series must be in a B+ Tree.”

– *someone on HackerNews (paraphrased)*

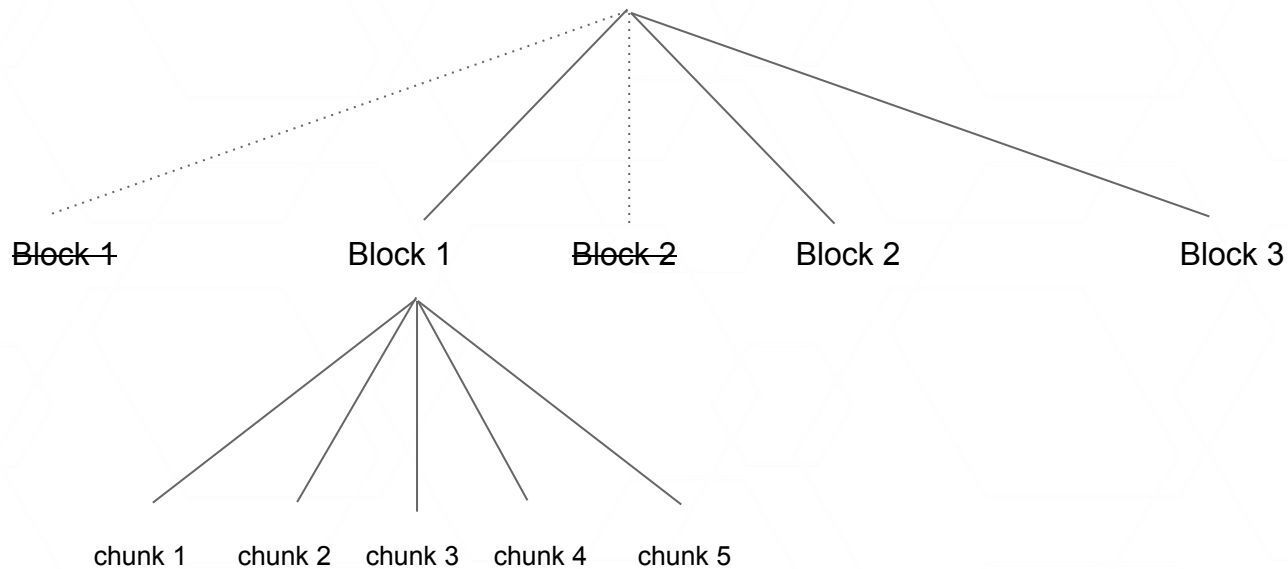
Tree



Tree

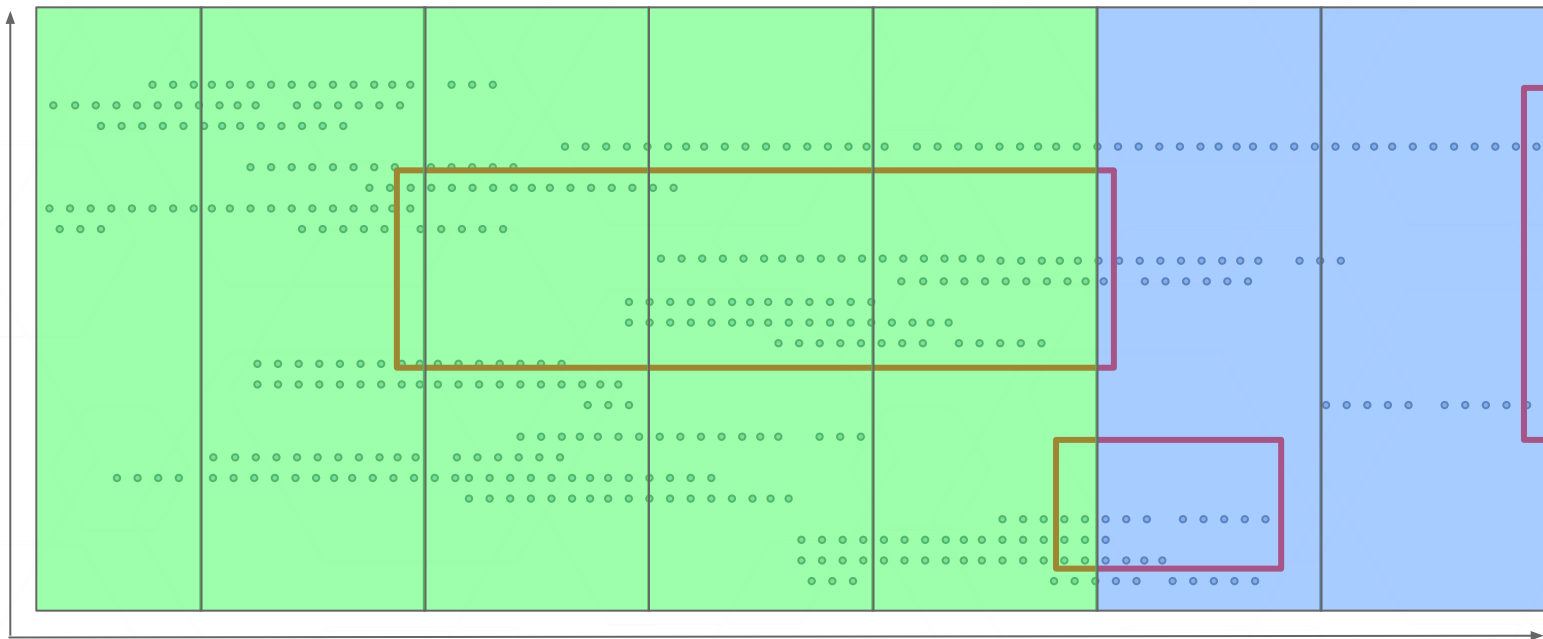


Tree



Blocks

series



time

Index

```
{  
  __name__="requests_total",  
  pod="nginx-34534242-abc723"  
  job="nginx",  
  path="/api/v1/status",  
  status="200",  
  method="GET",  
}
```

Index

```
{  
  __name__="requests_total",  
  pod="nginx-34534242-abc723"  
  job="nginx",  
  path="/api/v1/status",  
  status="200",  
  method="GET",  
}
```

- In each block, a series has a unique ID

↑
ID: 5

Index

```
{  
  __name__="requests_total",  
  pod="nginx-34534242-abc723"  
  job="nginx",  
  path="/api/v1/status",  
  status="200",  
  method="GET",  
  ...  
}
```

↑
ID: 5

- In each block, a series has a unique ID
- Maintain sorted lists from label pair to IDs

```
status="200":  1  2  5  99 1000 1001 1500 1502 500000  
method="GET":  2  3  4  5  6   9  10 1502 999999 ...  
...
```


Index

```
{  
  __name__="requests_total",  
  pod="nginx-34534242-abc723"  
  job="nginx",  
  path="/api/v1/status",  
  status="200",  
  method="GET",  
}
```

- In each block, a series has a unique ID
- Maintain sorted lists from label pair to IDs
- Efficient k-way set operations (merge, intersect)

```
status="200": 1 2 5 99 1000 1001 1500 1502 500000  
method="GET": 2 3 4 5 6 9 10 1502 999999 ...
```

Index

```
{  
  __name__="requests_total",  
  pod="nginx-34534242-abc723"  
  job="nginx",  
  path="/api/v1/status",  
  status="200",  
  method="GET",  
}
```

- In each block, a series has a unique ID
- Maintain sorted lists from label pair to IDs
- Efficient k-way set operations (merge, intersect)

```
status="200": 1 2 5 99 1000 1001 1500 1502 500000  
method="GET": 2 3 4 5 6 9 10 1502 999999 ...
```

Intersect:

Index

```
{  
  __name__="requests_total",  
  pod="nginx-34534242-abc723"  
  job="nginx",  
  path="/api/v1/status",  
  status="200",  
  method="GET",  
}
```

- In each block, a series has a unique ID
- Maintain sorted lists from label pair to IDs
- Efficient k-way set operations (merge, intersect)

```
status="200": 1 2 5 99 1000 1001 1500 1502 500000  
method="GET": 2 3 4 5 6 9 10 1502 999999 ...
```

Intersect:

Index

```
{  
  __name__="requests_total",  
  pod="nginx-34534242-abc723"  
  job="nginx",  
  path="/api/v1/status",  
  status="200",  
  method="GET",  
}
```

- In each block, a series has a unique ID
- Maintain sorted lists from label pair to IDs
- Efficient k-way set operations (merge, intersect)

```
status="200": 1 2 5 99 1000 1001 1500 1502 500000  
method="GET": 2 3 4 5 6 9 10 1502 999999 ...  
  
Intersect: 2
```

Index

```
{  
  __name__="requests_total",  
  pod="nginx-34534242-abc723"  
  job="nginx",  
  path="/api/v1/status",  
  status="200",  
  method="GET",  
}
```

- In each block, a series has a unique ID
- Maintain sorted lists from label pair to IDs
- Efficient k-way set operations (merge, intersect)

```
status="200": 1 2 5 99 1000 1001 1500 1502 500000  
method="GET": 2 3 4 5 6 9 10 1502 999999 ...  
  
Intersect: 2
```

Index

```
{  
  __name__="requests_total",  
  pod="nginx-34534242-abc723"  
  job="nginx",  
  path="/api/v1/status",  
  status="200",  
  method="GET",  
}
```

- In each block, a series has a unique ID
- Maintain sorted lists from label pair to IDs
- Efficient k-way set operations (merge, intersect)

```
status="200": 1 2 5 99 1000 1001 1500 1502 500000  
method="GET": 2 3 4 5 6 9 10 1502 999999 ...  
  
Intersect: 2
```

Index

```
{  
  __name__="requests_total",  
  pod="nginx-34534242-abc723"  
  job="nginx",  
  path="/api/v1/status",  
  status="200",  
  method="GET",  
}
```

- In each block, a series has a unique ID
- Maintain sorted lists from label pair to IDs
- Efficient k-way set operations (merge, intersect)

```
status="200": 1 2 5 99 1000 1001 1500 1502 500000  
method="GET": 2 3 4 5 6 9 10 1502 999999 ...  
  
Intersect: 2 5
```

Index

```
{  
  __name__="requests_total",  
  pod="nginx-34534242-abc723"  
  job="nginx",  
  path="/api/v1/status",  
  status="200",  
  method="GET",  
}
```

- In each block, a series has a unique ID
- Maintain sorted lists from label pair to IDs
- Efficient k-way set operations (merge, intersect)

```
status="200": 1 2 5 99 1000 1001 1500 1502 500000  
method="GET": 2 3 4 5 6 9 10 1502 999999 ...  
  
Intersect: 2 5
```


Index

```
{  
  __name__="requests_total",  
  pod="nginx-34534242-abc723"  
  job="nginx",  
  path="/api/v1/status",  
  status="200",  
  method="GET",  
}
```

- In each block, a series has a unique ID
- Maintain sorted lists from label pair to IDs
- Efficient k-way set operations (merge, intersect)

```
status="200":  1  2  5  99  1000  1001  1500  1502  500000  
method="GET":  2  3  4  5  6  9  10  1502  999999  ...  
  
Intersect:    2  5
```

Index

```
{  
  __name__="requests_total",  
  pod="nginx-34534242-abc723"  
  job="nginx",  
  path="/api/v1/status",  
  status="200",  
  method="GET",  
}
```

- In each block, a series has a unique ID
- Maintain sorted lists from label pair to IDs
- Efficient k-way set operations (merge, intersect)

```
status="200": 1 2 5 99 1000 1001 1500 1502 500000  
method="GET": 2 3 4 5 6 9 10 1502 999999 ...  
  
Intersect: 2 5
```

Index

```
{  
  __name__="requests_total",  
  pod="nginx-34534242-abc723"  
  job="nginx",  
  path="/api/v1/status",  
  status="200",  
  method="GET",  
}
```

- In each block, a series has a unique ID
- Maintain sorted lists from label pair to IDs
- Efficient k-way set operations (merge, intersect)

```
status="200": 1 2 5 99 1000 1001 1500 1502 500000  
method="GET": 2 3 4 5 6 9 10 1502 999999 ...  
  
Intersect: 2 5 1502
```

Index

```
{  
  __name__="requests_total",  
  pod="nginx-34534242-abc723"  
  job="nginx",  
  path="/api/v1/status",  
  status="200",  
  method="GET",  
}
```

- In each block, a series has a unique ID
- Maintain sorted lists from label pair to IDs
- Efficient k-way set operations (merge, intersect)

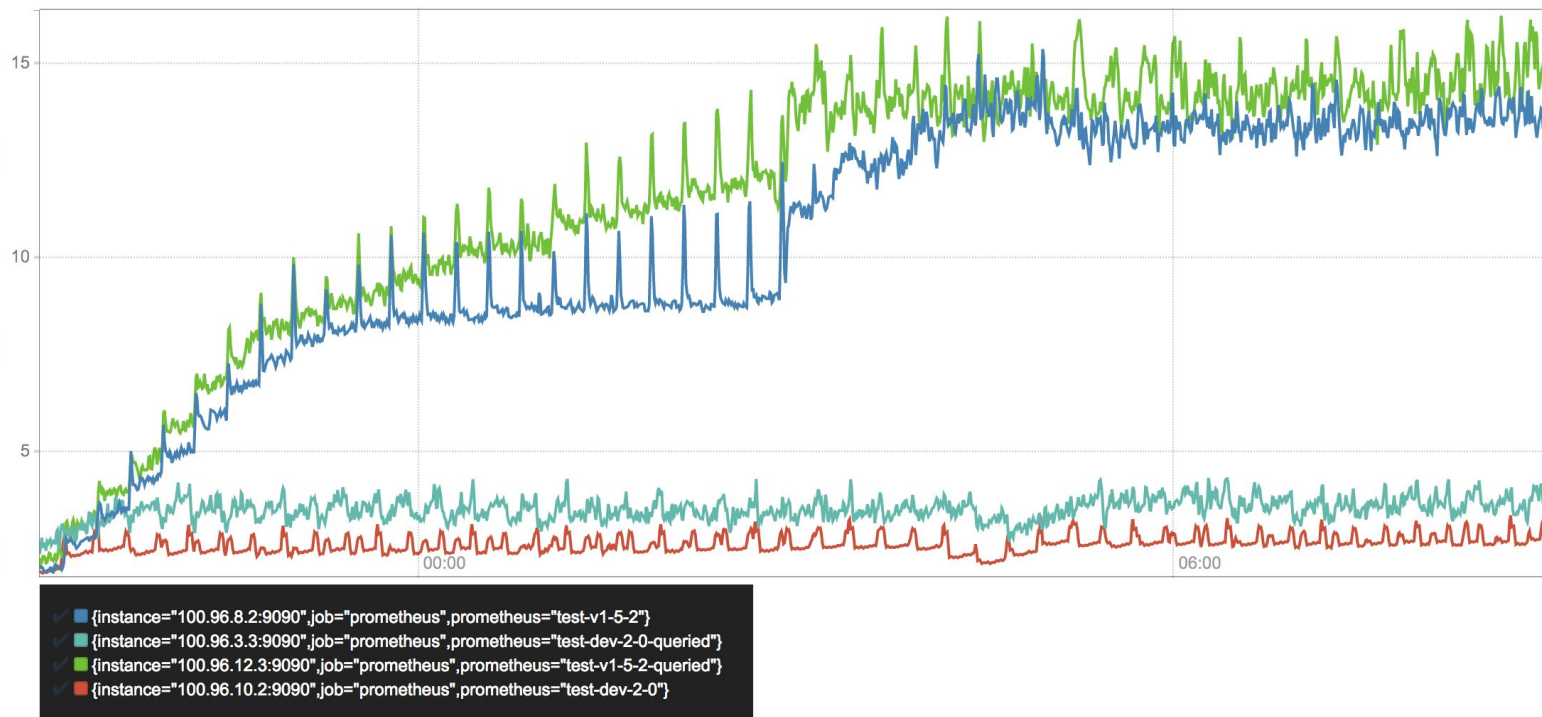
```
status="200": 1 2 5 99 1000 1001 1500 1502 500000  
method="GET": 2 3 4 5 6 9 10 1502 999999 ...  
  
Intersect: 2 5 1502
```

Benchmarks

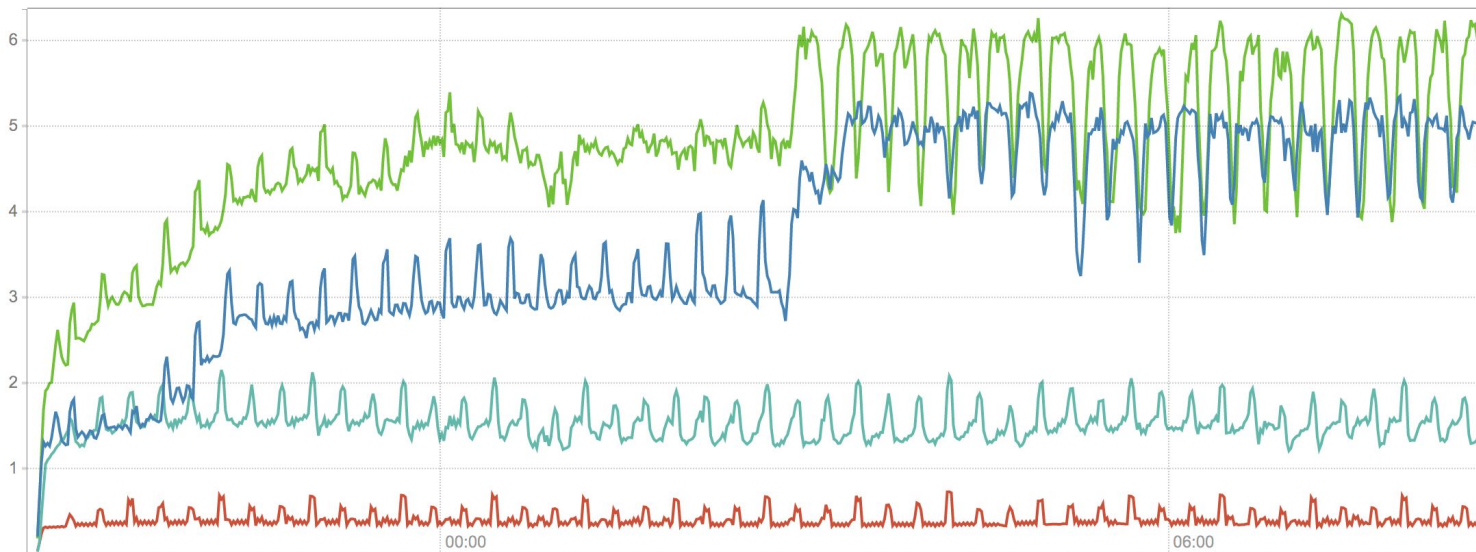
Benchmarks

- Kubernetes cluster + dedicated Prometheus nodes
- 800 microservice instances + Kubernetes components
- 120,000 samples/second
- 300,000 active time series
- Swap out 50% of pods every 10 minutes

Benchmarks (memory)

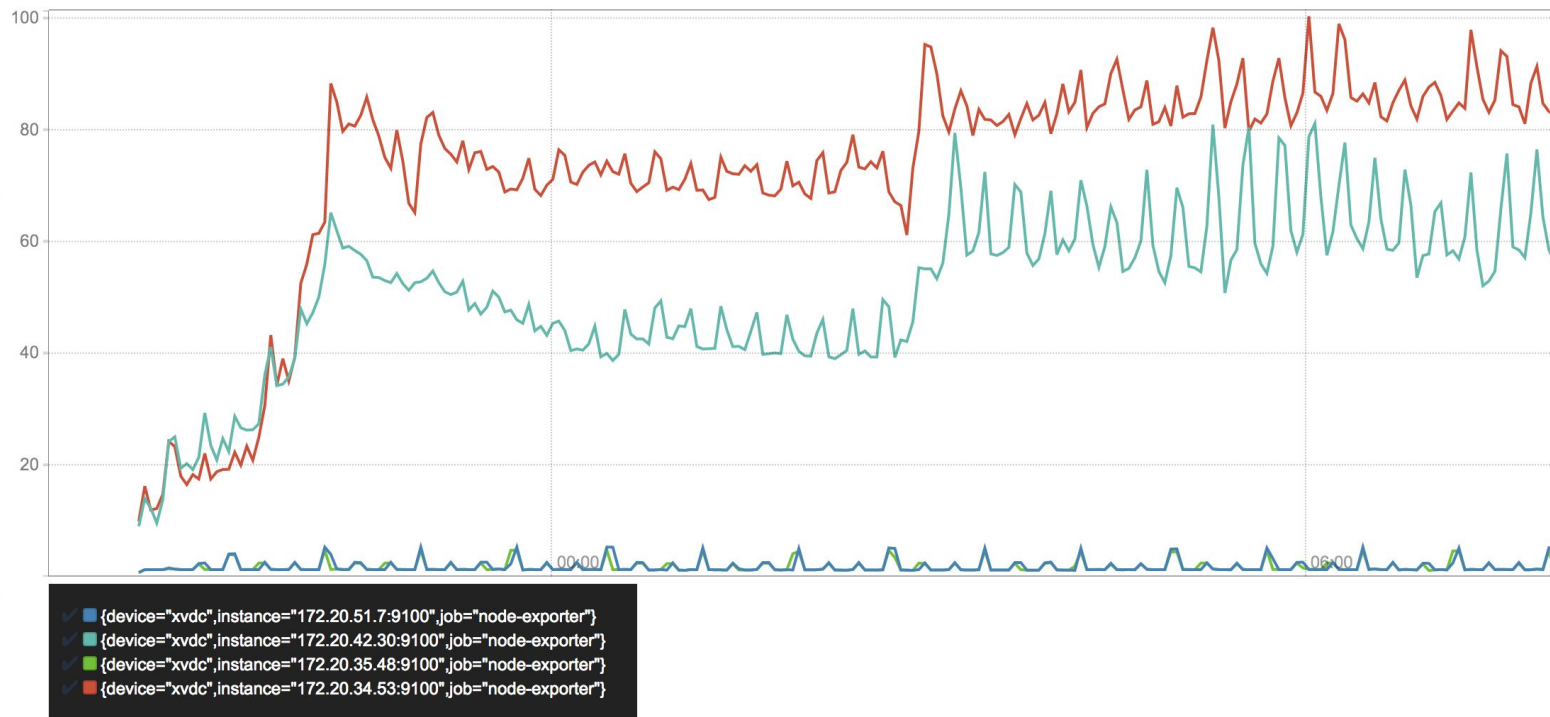


Benchmarks (CPU)

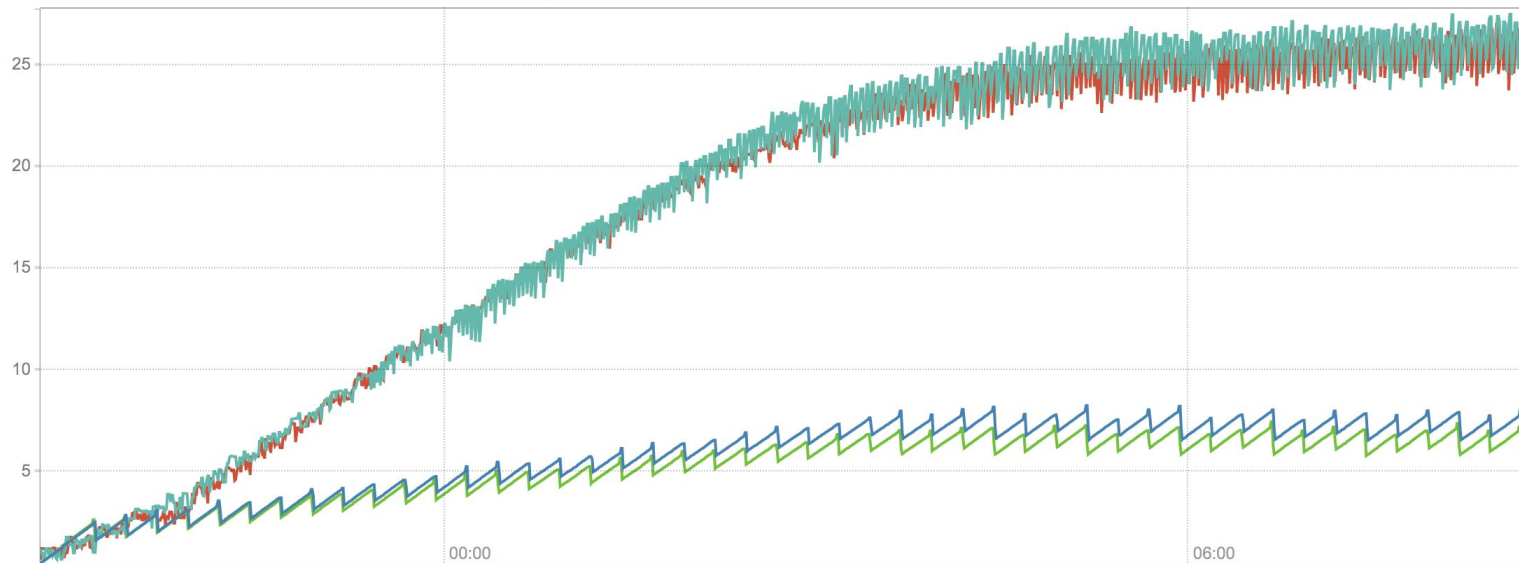


- {instance="100.96.8.2:9090",job="prometheus",prometheus="test-v1-5-2"}
- {instance="100.96.3.3:9090",job="prometheus",prometheus="test-dev-2-0-queried"}
- {instance="100.96.12.3:9090",job="prometheus",prometheus="test-v1-5-2-queried"}
- {instance="100.96.10.2:9090",job="prometheus",prometheus="test-dev-2-0"}

Benchmarks (disk writes)

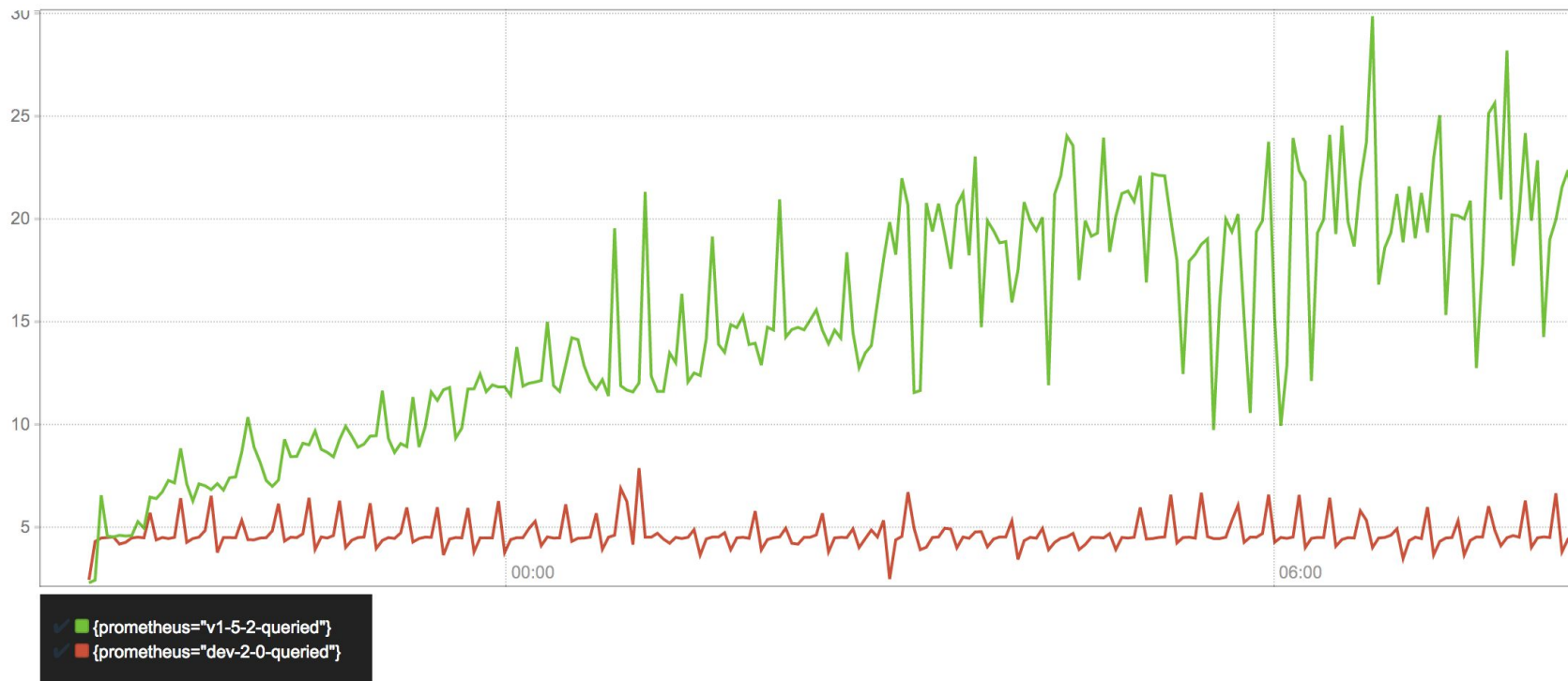


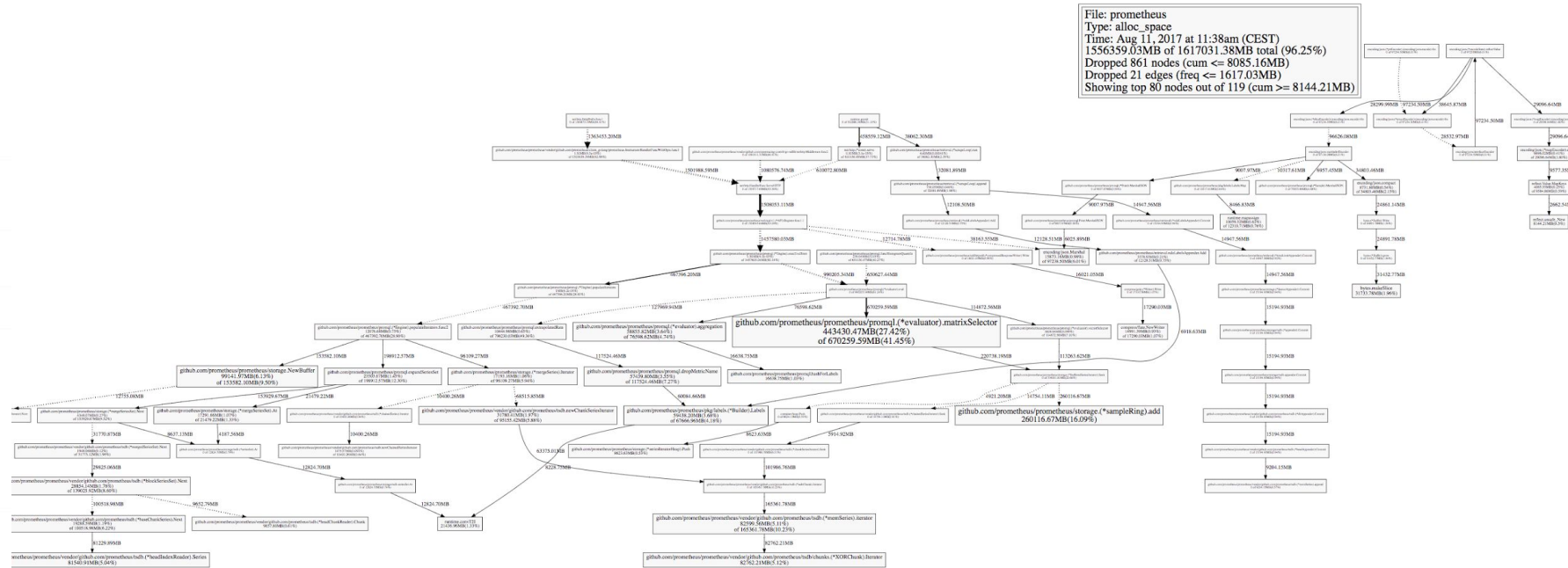
Benchmarks (on-disk size)



✓ {device="/dev/xvdc",id="",instance="ip-172-20-51-7.eu-west-1.compute.internal",job="kubelets"}
✓ {device="/dev/xvdc",id="",instance="ip-172-20-42-30.eu-west-1.compute.internal",job="kubelets"}
✓ {device="/dev/xvdc",id="",instance="ip-172-20-35-48.eu-west-1.compute.internal",job="kubelets"}
✓ {device="/dev/xvdc",id="",instance="ip-172-20-34-53.eu-west-1.compute.internal",job="kubelets"}

Benchmarks (query latency)





max_over_time(go_memstats_heap_inuse_bytes(prometheus=~"test.+")[3m])

Execute

- insert metric at cursor -

Load time: 555m
Resolution: 10ss
Total time series:

Graph Console

-

16h

+

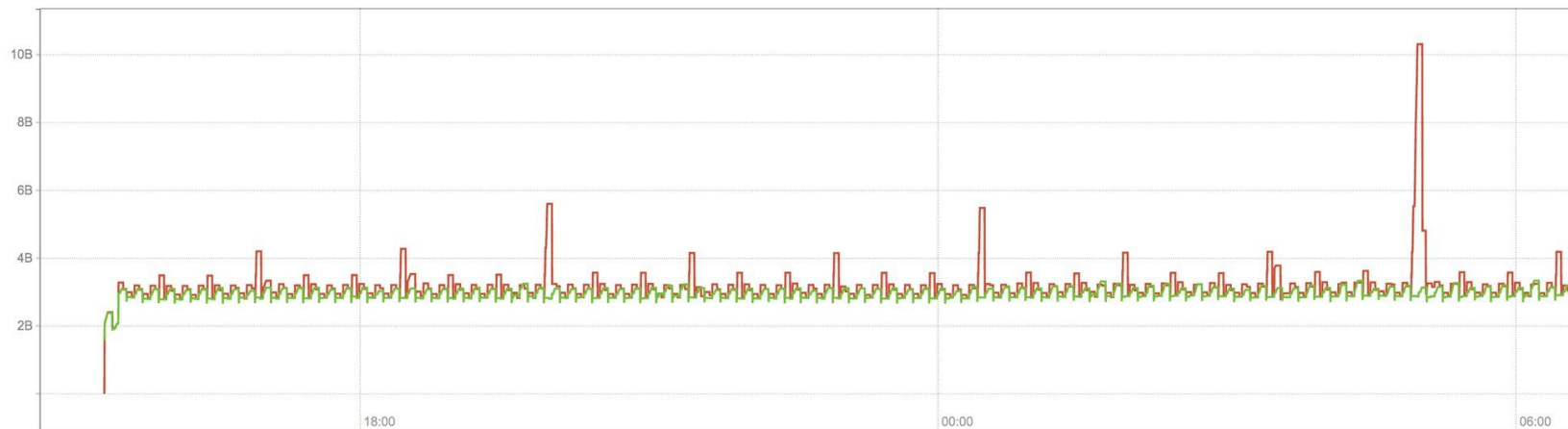
◀

Until

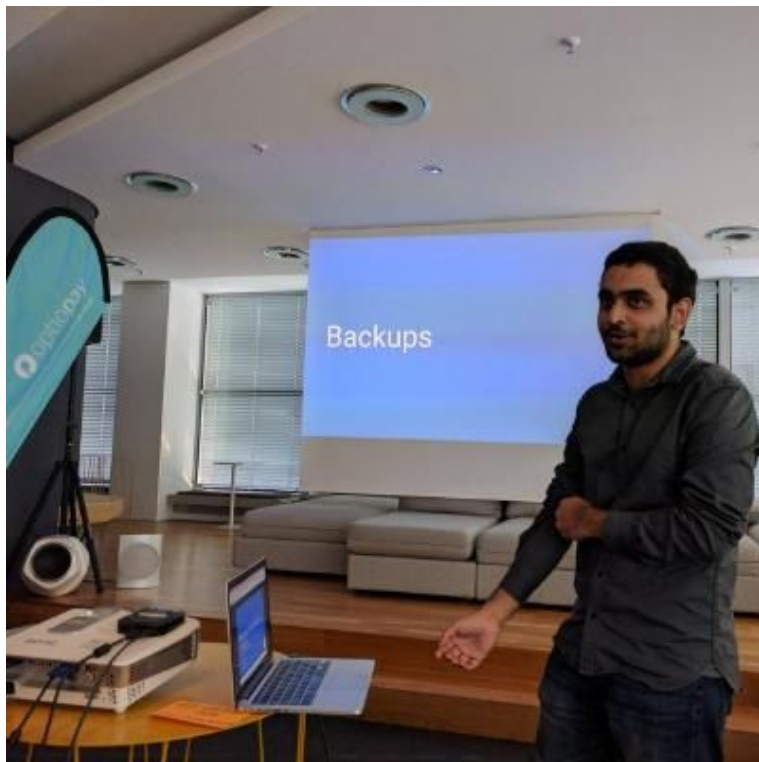
▶

10s

☐ stacked




■ {instance="100.96.2.4:9090",job="prometheus",prometheus="test-v2-0-0-dev"}
■ {instance="100.96.1.4:9090",job="prometheus",prometheus="test-v2-0-0-beta"}



Try it out!

Draft

2.0.0-beta.2 / 2017-08-17

 fabxc drafted this 26 minutes ago

This release includes numerous changes to the new storage layer. The main changes are:

- [CHANGES] Deterministic block boundaries
- [ENHANCEMENTS] Avoid memory usage spikes during compactions

It's generally advised to start with a clean storage directory. As a best effort, running `sed -i .bkp 's/generation/level/g' */meta.json` from within the directory should be sufficient to migrate data written by v2.0.0-beta.0.

Edit

We're hiring: coreos.com/careers

More

- <https://fabxc.org/blog/2017-04-10-writing-a-tsdb/>
- <https://github.com/prometheus/tsdb>