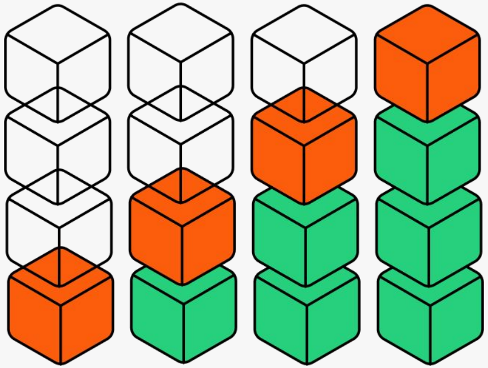
CAMUNDA

Benchmark Performance with Camunda Platform's Zeebe Engine

February 2023



Agenda

1 **Scalable cloud-native architecture**

What makes Zeebe fast?

2 **Deep dive into performance benchmarking**

Tools for load testing Zeebe

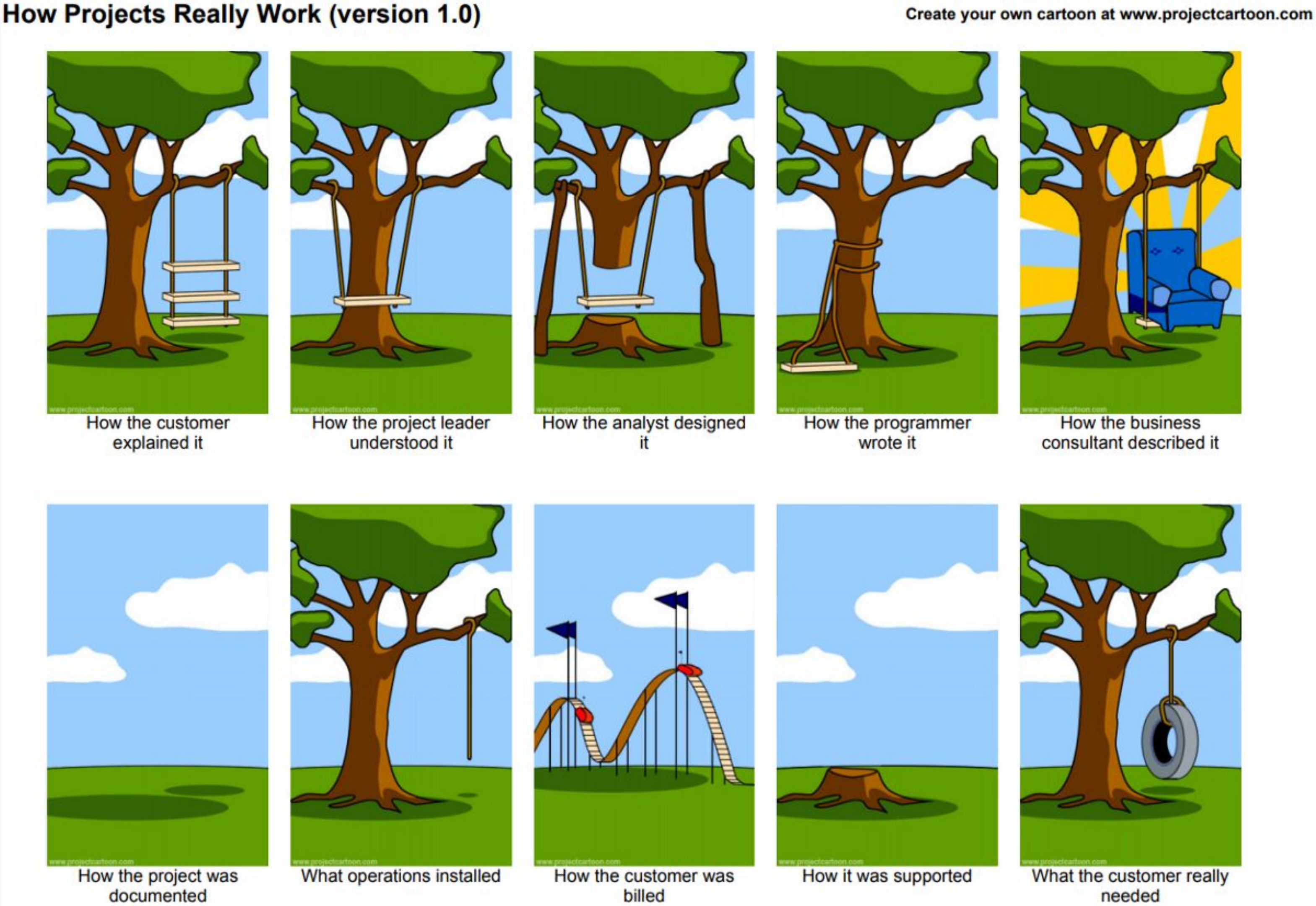
3 **Best practices**

How to perform benchmarking and optimize performance

Introductions

Olga Inozemtceva
Senior Product Marketing Manager
Camunda

Falko Menge
Senior Principal Solution Architect
Camunda


How Projects Really Work (version 1.0)
Create your own cartoon at www.projectcartoon.com
How the customer explained it
How the project leader understood it
How the analyst designed it
How the programmer wrote it
How the business consultant described it
How the project was documented
What operations installed
How the customer was billed
How it was supported
What the customer really needed

Camunda Platform: The Universal Process Orchestrator

Design

Developers & business users collaborate to design & deploy processes with Camunda.

Modeler

Model & deploy business process diagrams with BPMN & DMN.

Available via web and desktop app.

Connectors

Out-of-the-box integrations to easily communicate with popular enterprise applications & protocols.

Integration Framework

Build & provision your own connectors to any system including home-grown & legacy applications.

Forms

Drag & drop creation of forms that power workflows that require human interaction.

Automate

Enterprise-grade automation platform. Built for today's business complexity, loved by developers.

POWERED BY ZEEBE

Workflow Engine

Next-generation, cloud-native BPMN workflow engine that unlocks unparalleled speed, scale & resilience.

Decision Engine

Automate decisions in end-to-end business processes via DMN.

Tasklist

Assign and execute tasks that require human interaction via easy-to-use Forms or via your own apps with the Tasklist API.

Operate

Real time visibility to monitor, analyze and resolve problems with any process instance.

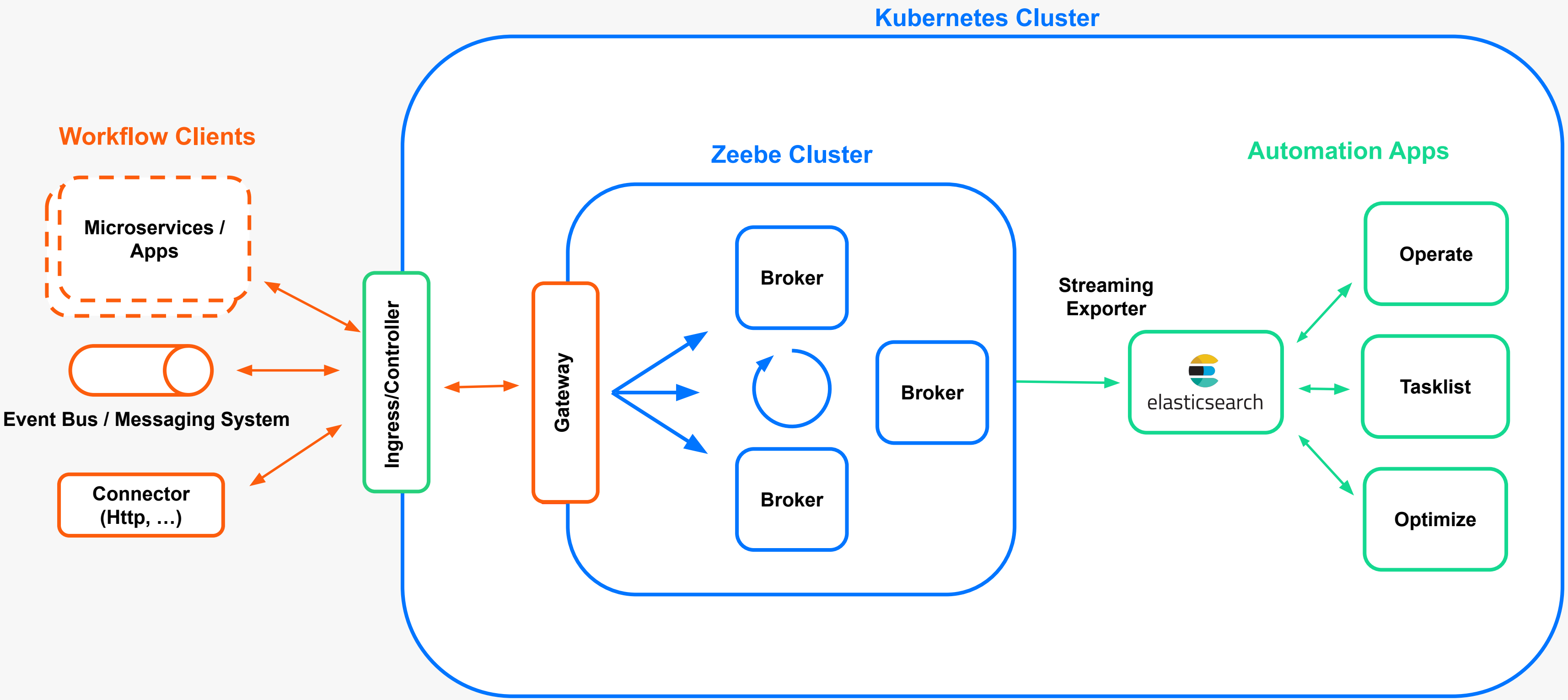
Improve

Teams have the insights to fix the highest ROI problems for their business processes.

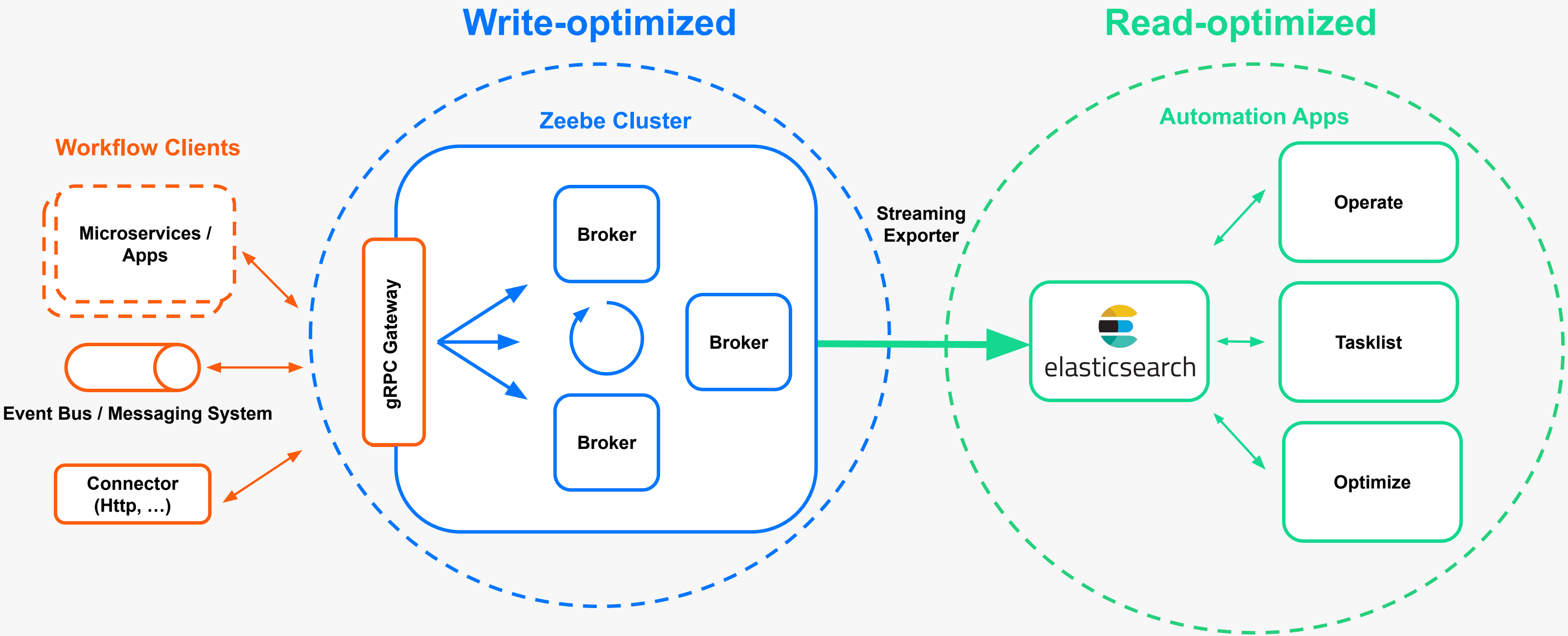
Optimize

Get the insights you need to understand and continuously improve your business processes.

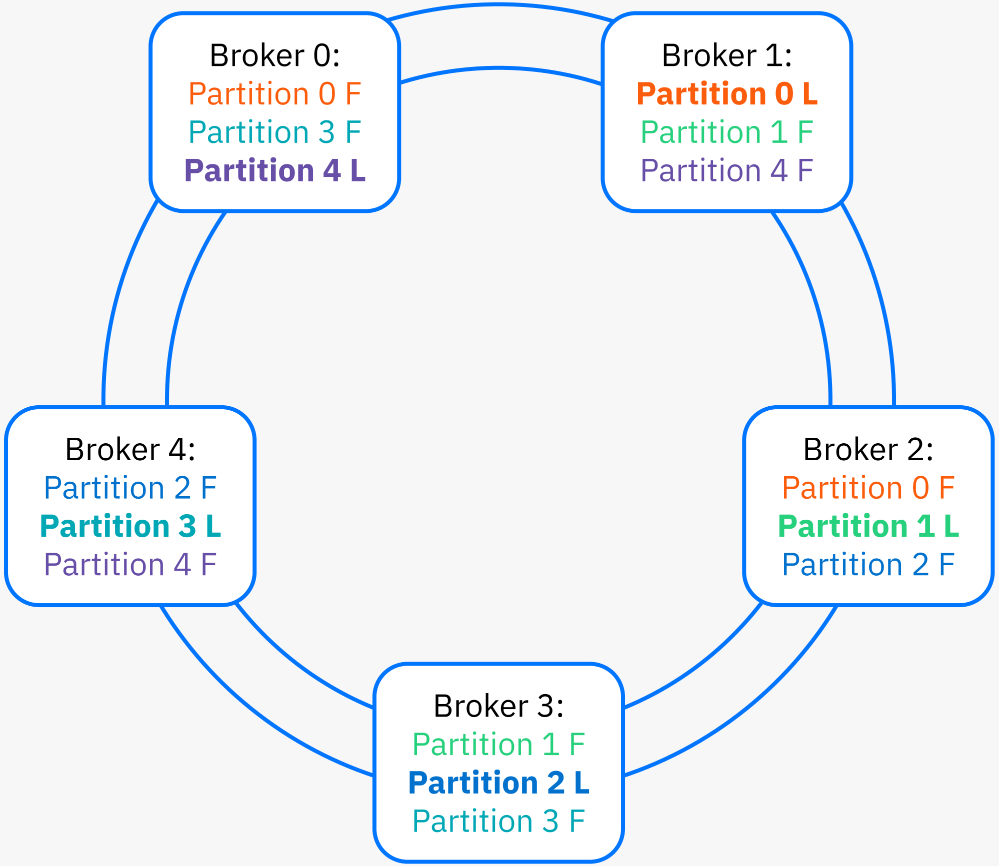
Scalable Cloud-Native Architecture



Command Query Responsibility Segregation (CQRS)



Partitions (Shards) and Replication using Raft

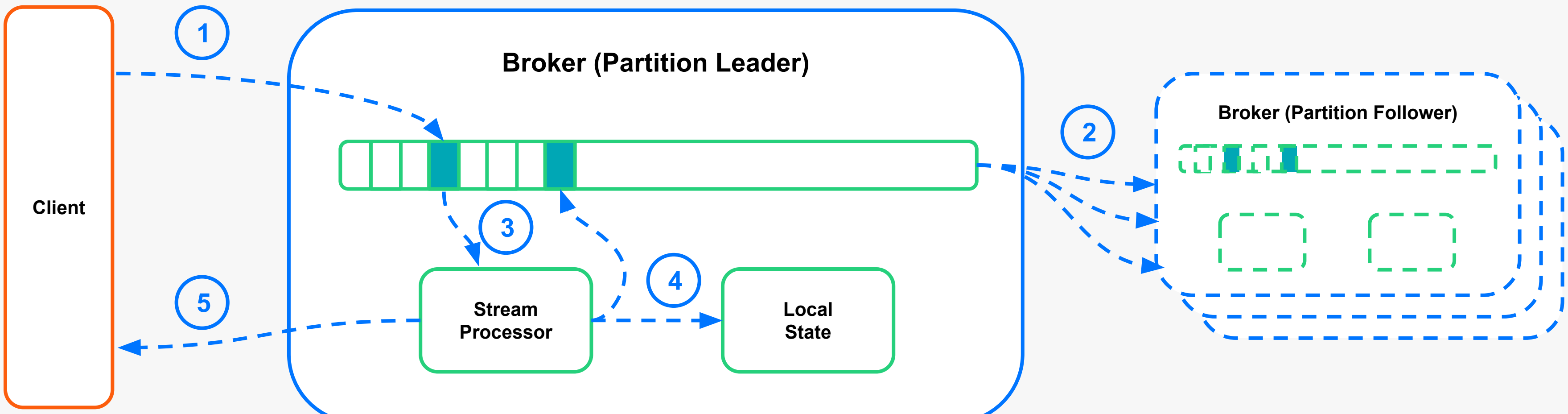


Example:

- 5 Brokers
- 5 Partitions
- Replication factor 3

- **L = Leader**
- F = Follower

Process Execution interpreted as Stream Processing



1 Send & Append Command

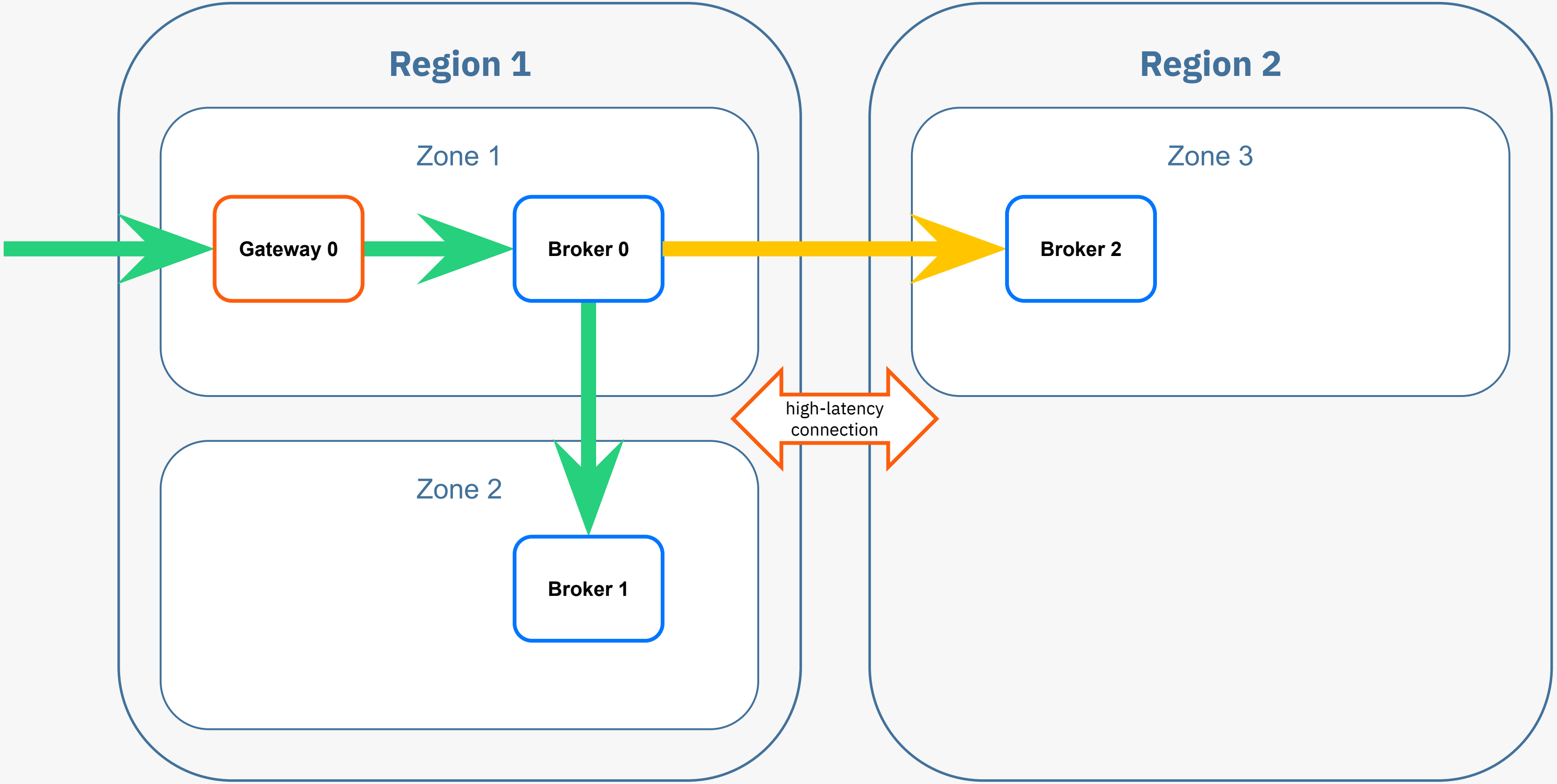
2 Replicate & Commit Command

3 Validate & Process Command

4 Apply to State & Write Event

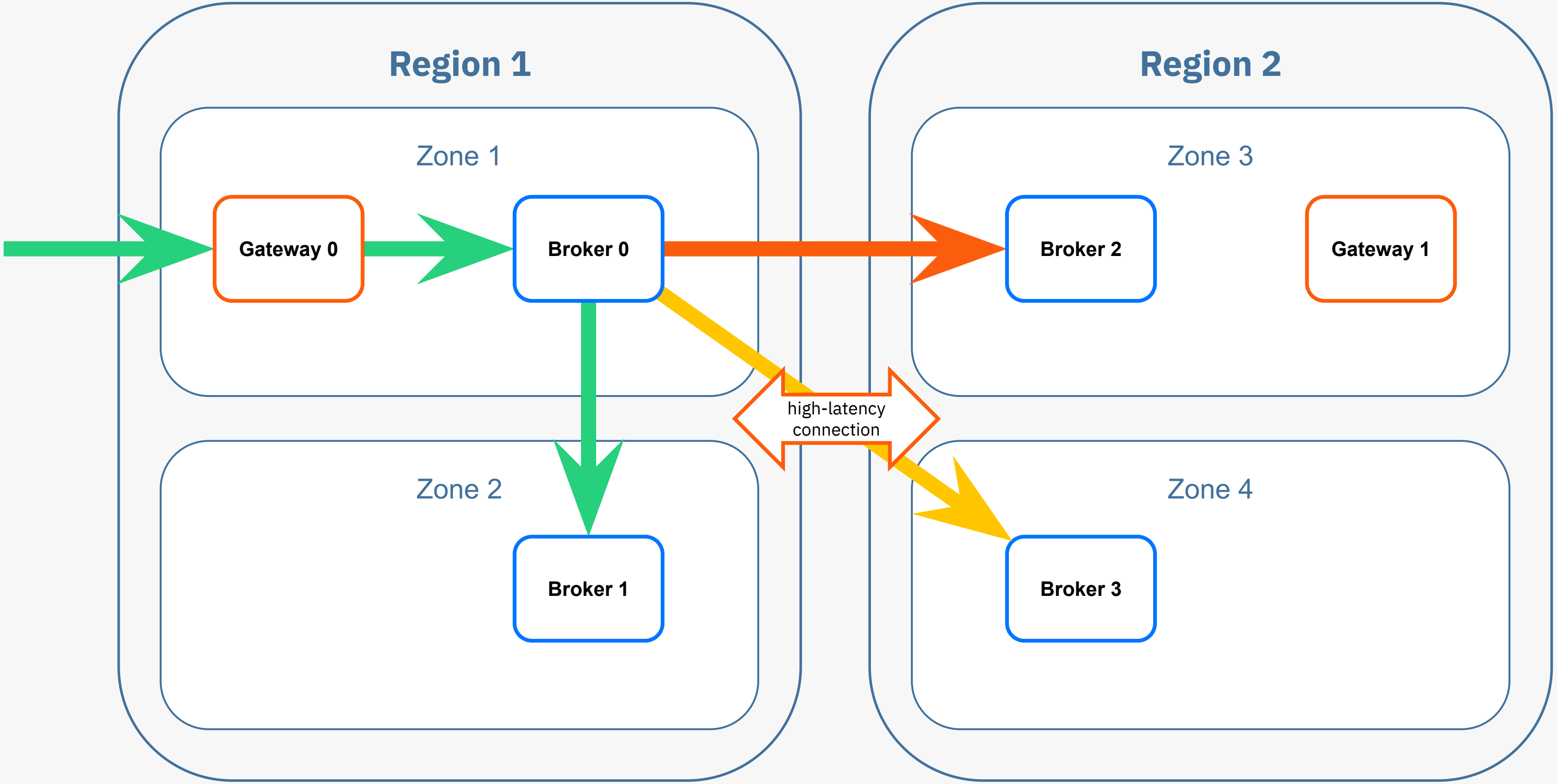
5 Send Response

Dual-region active-passive



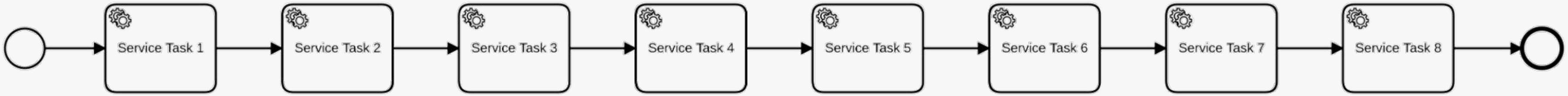
replication factor 3 => quorum 2 => commits stay local

Dual-region active-active



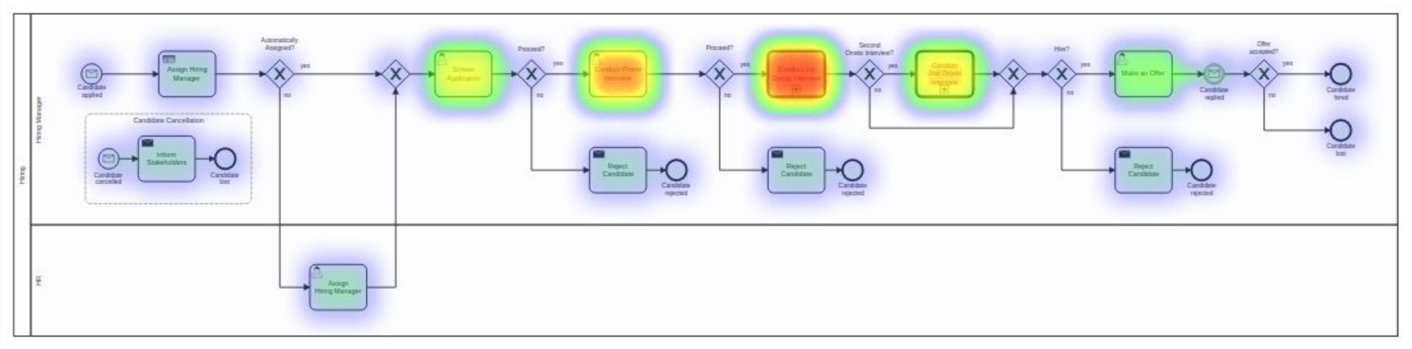
replication factor 4 => quorum 3 => commits must go cross-region

Key Process Performance Metrics



- **Throughput**
  - Number of process instances completed per second (PI/s)
- **Process size**
  - Number of tasks in the BPMN process model (tasks/PI)
  - Tasks dominate performance; gateways & events almost neglectable
- **Process latency (cycle time/process instance duration)**
  - Time to execute process instance from start to end (ms)
- **Inter-region network latency**
  - Traveling time of network packets between geographically distant regions (ms)

Key Process Performance Metrics



- **Throughput**
  - Number of process instances completed per second (PI/s)
- **Process size**
  - Number of tasks in the BPMN process model (tasks/PI)
- **Process latency (cycle time/process instance duration)**
  - Time to execute process instance from start to end (ms)
- **Inter-region network latency**
  - Traveling time of network packets between geographically distant regions (ms)

slido

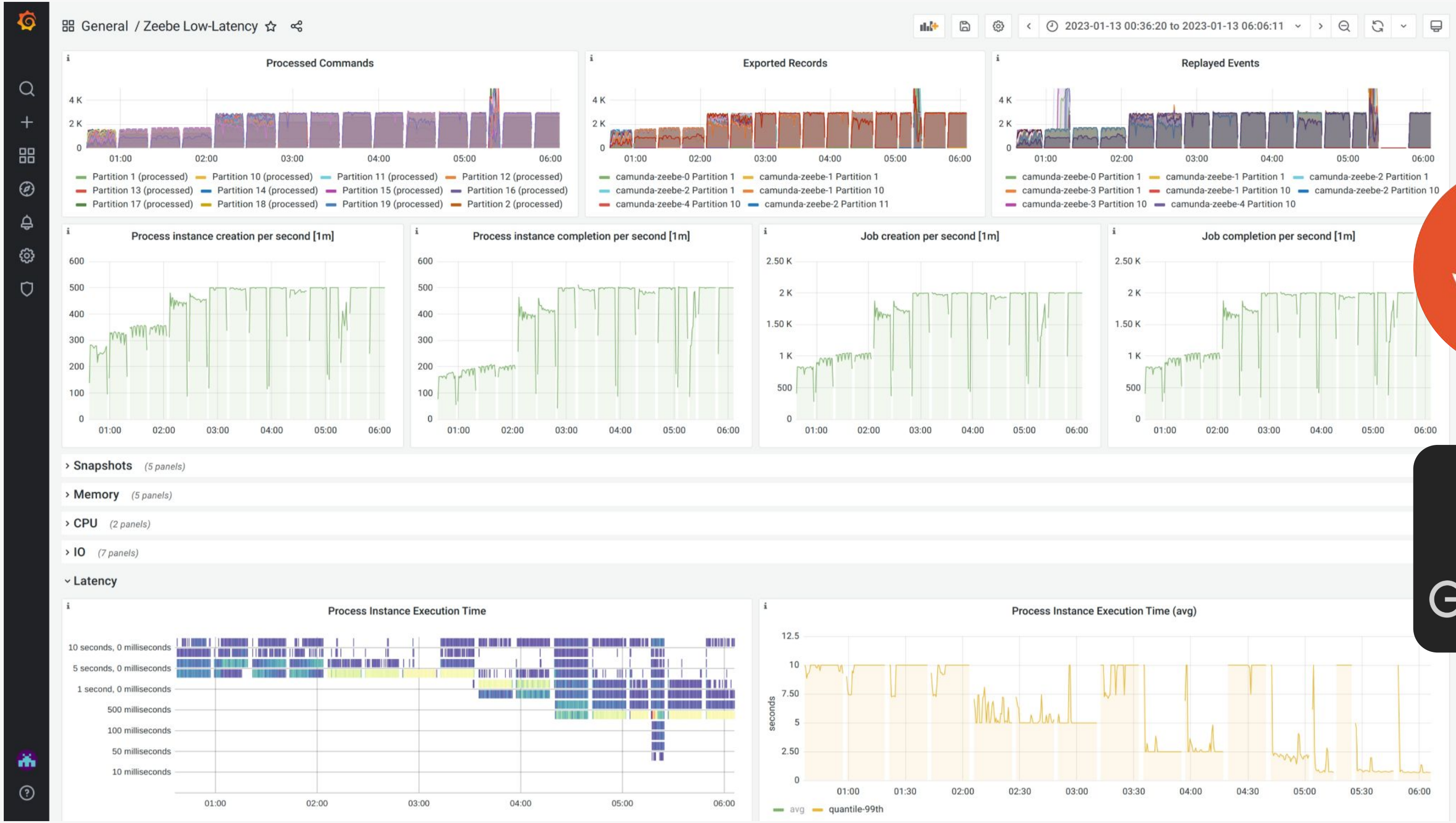
How many process instances per second (PI/s) are you running in production?

ⓘ Start presenting to display the poll results on this slide.

Workload Characteristics of Customers

Throughput (PI/s)	Process size (#tasks)	Latency (ms)	Multi-Region Setup
10,000	8 tasks	500 ms	active-passive east-west 60ms
500	3 tasks + 2 messages + 2 call activities	1,000 ms	active-active 10ms avg / 35ms max
2,400	10 tasks	1,200 ms	active-passive 52ms one way
1,700	10 tasks	120,000 ms	active-active-passive 2x east coast + 1x central
800	8 tasks	200 ms	active-passive 62ms
3,000	3 tasks	300 ms	single-region replication factor = 1

Zeebe Grafana Dashboard

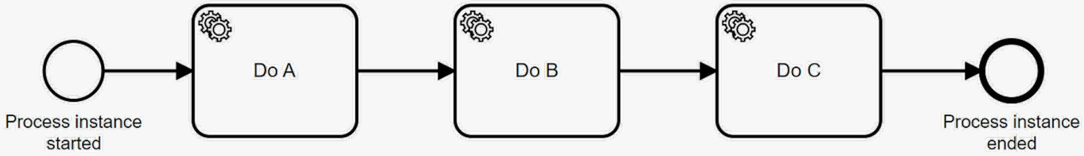


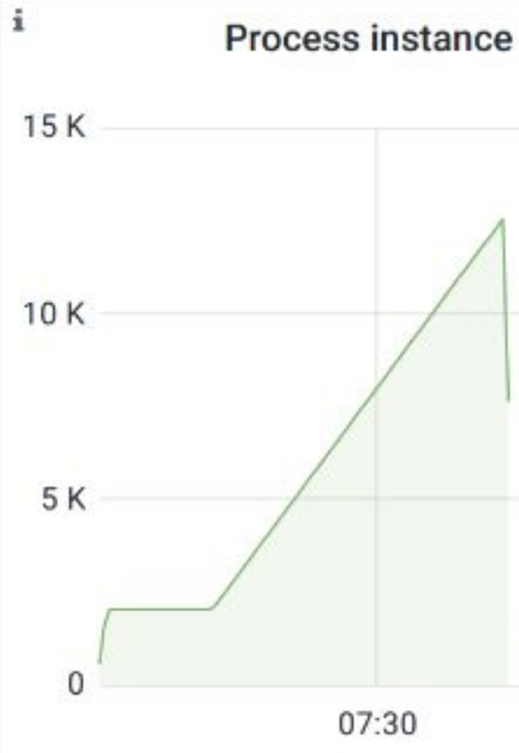
docs.camunda.io/docs/next/self-managed/zeebe-deployment/operations/metrics/

Load Generator: Camunda 8 Benchmark

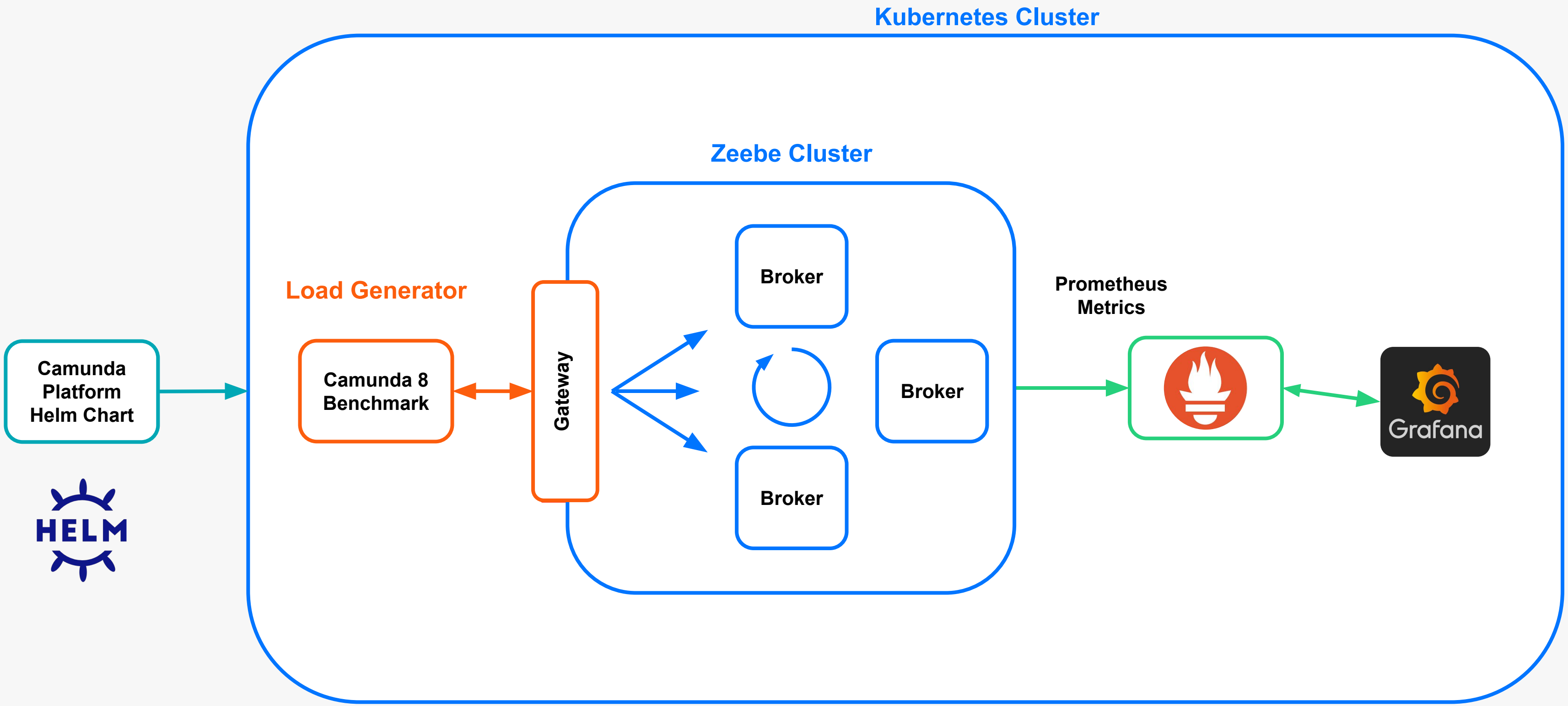
github.com/camunda-community-hub/camunda-8-benchmark

- Java-based load generator for Zeebe
- Simulates the gRPC workload of clients
- Starts thousands of process instances at fixed/increasing rate
  - Overcomes Java scheduler limitations
- Completes tens of thousands of jobs
  - Configurable delay & payload
  - Implemented as asynchronous/reactive as possible, i.e. no blocking of threads





Benchmark Setup – Don’t try this at home ;-)



helm.camunda.io

Zeebe Benchmark Template

		Load Generator/Starter									engine																						JVM	RocksDB		Gateway					Job Wor		Performance									
Test Case Name	Timestamp (CET)	Process Model	Starter Replicas	Load Generator Threads	Ramp up time (s)	Run Duration (s)	Start Throughput (PI/s)	Rate Adjustment Strategy	Start Pi Increase Factor	Message TTL (min)	Image	Engine Version	Machine Type	Cluster Size (nodes)	vCPUs (Hyperthreads/node)	RAM (GiB/node)	Exporters	CPU Thread Pool Size/Node	IO Thread Pool Size/Node	Partitions	Replication Factor	Log Segment Size (MB)	pre Allocate Segment Files	Disable Explicit Raft Flush	Disk Type	Disk Size	File System (Storage Class)	Backpressure	Inter-Region Latency (ms)	Network Compression	max Appends Per Follower	max Append Batch Size (KiB)	JVM MaxRAMPercentage	Write Buffer Size (MiB)	Buffer to Maintain (MiB)	Replicas (nodes)	vCPUs (Hyperthreads)	RAM (GiB)	Number of Threads	Inter-Region Latency (ms)	Client for Job Worker	Job duration (ms)	Throughput (finished kPI/s)	Throughput with 99% < 1s	Throughput (kTasks/s)	Throughput (kFNI/s)	Process Duration 99% (s)	Process Duration 50% (s)	Process Duration avg (s)	Standard Deviation (s)	Flow Node Duration (ms)	Test Duration (min)
		Default		1	-	-	200		0.4	60	camunda/zeebe	latest	n1-	1	5	4	elastic	2	2	1	1	128	☑	☐	ssd	128	ssd-ext4	☑		NO	2	32	25	64	128	3					java	50										
		Min tested		1	10	1800	18		0.0	60		0		3	2	4		2	2	3	3	16	☑	☐		14							25	64	128	2						0	0.08			0.32			0.1	0	0	21
		Max tested		100	180	3600	500		0.1	60	camunda/zeebe	0	n2-	24	14	32	metrics	14	25	36	4	512	☑	☐	disk	500							25	64	128	3					jobex	50	0.42			0.60			112000.	0	0	30
4		singleProce	1	20	10	3600	80	none	-	60	camunda/zeebe	8.1.2	n1-	8	4	16	metrics	4	4	24	4	128	☑	☐	disk	100	ssd-ext4	☑		NO	8	32	25	64	128	2	3	3	3	0	jobe:	0	0.08	0.08		0.32	0.5		0.3			30
5	"2022-10-27	singleProce	1	20	10	3600	80	none	-	60	camunda/zeebe	8.1.2	n1-	8	4	16	metrics	4	4	32	4	128	☑	☐	disk	200	ssd-ext4	☑		NO	8	32	25	64	128	2	3	3	3	0	jobex	0	0.08	0.08		0.32	0.5		0.4			30
6	"2022-10-27	singleProce	1	20	10	3600	300	none	-	60	camunda/zeebe	8.1.2	n1-	8	4	16	metrics	4	4	24	4	128	☑	☐	disk	200	ssd-ext4	☑		NO	8	32	25	64	128	2	3	3	3	0	jobex	0	0.25	0.00			>10		112000.			30
7		singleProce	1	20	10	3600	80	none	-	60	camunda/zeebe	8.1.2	n1-	8	4	16	metrics	4	4	24	4	128	☑	☐	disk	100	ssd-ext4	☑	35	NO	8	32	25	64	128	2	3	3	3	0	jobex	0										
8		singleProce	1	20	10	3600	80	none	-	60	camunda/zeebe	8.1.2	n1-	8	4	16	metrics	4	4	18	4	128	☑	☐	disk	100	ssd-ext4	☑	35	NO	8	32	25	64	128	2	3	3	3	0	jobex	0	0.08	0.08			0.5		0.3			30
9	11/2 9:28	singleProce	1	20	10	3600	160	none	-	60	camunda/zeebe	8.1.2	n1-	8	4	16	metrics	4	4	24	4	128	☑	☐	disk	100	ssd-ext4	☑	35	NO	8	32	25	64	128	2	3	3	3	0	jobex	0	0.16	0.16			0.8		0.5			30
10	11/2 10:37	singleProce	1	20	10	3600	300	none	-	60	camunda/zeebe	8.1.2	n1-	8	6	16	metrics	6	6	24	4	128	☑	☐	disk	100	ssd-ext4	☑	35	NO	8	32	25	64	128	2	3	3	3	0	jobex	0	0.30				2.5		0.7			30
11	11/2 11:32	singleProce	1	20	10	3600	300	none	-	60	camunda/zeebe	8.1.2	n1-	8	8	16	metrics	8	8	24	4	128	☑	☐	disk	100	ssd-ext4	☑	35	NO	8	32	25	64	128	2	3	3	3	0	jobex	0	0.30				2.0					30
12	11/2 12:47	singleProce	1	20	10	3600	300	none	-	60	camunda/zeebe	8.1.2	n1-	8	8	16	metrics	8	8	32	4	128	☑	☐	disk	100	ssd-ext4	☑	35	NO	8	32	25	64	128	2	3	3	3	0	jobex	0	0.30				2.5		0.7			30
13	11/2 13:28	singleProce	1	20	10	3600	300	none	-	60	camunda/zeebe	8.1.2	n1-	8	12	16	metrics	12	12	24	4	128	☑	☐	disk	100	ssd-ext4	☑	35	NO	2	32	25	64	128	2	3	3	3	0	jobex	0	0.30				1.1		0.6			30
14	11/2 14:38	singleProce	1	20	10	3600	300	none	-	60	camunda/zeebe	8.1.2	n1-	8	14	16	metrics	14	14	24	4	128	☑	☐	disk	100	ssd-ext4	☑	35	NO	2	32	25	64	128	2	3	3	3	0	jobex	0	0.30	0.30			0.8		0.5			30
15	11/3 16:02	singleProce	1	20	10	3600	80	none	-	60	camunda/zeebe	8.1.2	n1-	8	4	16	metrics	4	4	24	4	128	☑	☐	disk	100	ssd-ext4	☑	35	NO	2	32	25	64	128	2	3	3	3	35	jobex	0	0.08				5.0		2.0			30
16	11/3 15:25	singleProce	1	20	10	3600	80	none	-	60	camunda/zeebe	8.1.2	n1-	8	4	16	metrics	4	4	24	4	128	☑	☐	disk	100	ssd-ext4	☑	35	NO	2	32	25	64	128	2	3	3	3	35	jobex	0	0.08				5.0		3.4			30
17	11/4 11:04	singleProce	1	20	10	3600	80	none	-	60	camunda/zeebe	8.1.2	n1-	8	4	16	metrics	4	4	24	4	128	☑	☐	disk	100	ssd-ext4	☑	35	NO	2	32	25	64	128	2	3	3	3	35	jobex	0	0.08				5.0		3.4			30
18	11/4 9:20	singleProce	1	20	10	3600	80	none	-	60	camunda/zeebe	8.1.2	n1-	8	4	16	metrics	4	4	24	4	128	☑	☑	disk	100	ssd-ext4	☑	35	NO	2	32	25	64	128	2	3	3	3	35	jobex	0	0.08				5.0		3.3			30
19	11/4 13:09	singleProce	1	20	10	3600	80	none	-	60	camunda/zeebe	8.1.2	n1-	8	7	16	metrics	7	7	24	4	128	☑	☐	disk	100	ssd-ext4	☑	35	NO	2	32	25	64	128	2	3	3	3	35	jobex	0	0.08				5.0		3.5			30
20	11/4 13:37	singleProce	1	20	10	3600	80	none	-	60	camunda/zeebe	8.1.2	n1-	8	7	16	metrics	7	7	24	4	128	☑	☑	disk	100	ssd-ext4	☑	35	NO	2	32	25	64	128	2	3	3	3	35	jobex	0	0.08				5.0		3.1			30
21	11/4 14:41	singleProce	1	20	10	3600	300	none	-	60	camunda/zeebe	8.1.2	n1-	8	7	16	metrics	7	7	24	4	128	☑	☐	disk	100	ssd-ext4	☑	35	NO	2	32	25	64	128	2	3	3	3	35	jobex	0	0.13				>10					30
22	11/4 13:56	singleProce	1	20	10	3600	300	none	-	60	camunda/zeebe	8.1.2	n1-	8	7	16	metrics	7	7	24	4	128	☑	☑	disk	100	ssd-ext4	☑	35	NO	2	32	25	64	128	2	3	3	3	35	jobex	0	0.14				9.8		6.5			30
23	11/4 15:15	singleProce	1	20	10	3600	300	none	-	60	camunda/zeebe	8.1.2	n1-	12	7	16	metrics	7	7	24	4	128	☑	☐	disk	100	ssd-ext4	☑	35	NO	2	32	25	64	128	2	3	3	3	35	jobex	0	0.14				>10					30
24	11/4 15:43	singleProce	1	20	10	3600	300	none	-	60	camunda/zeebe	8.1.2	n1-	12	7	16	metrics	7	7	24	4	128	☑	☑	disk	100	ssd-ext4	☑	35	NO	2	32	25	64	128	2	3	3	3	35	jobex	0	0.15				9.7		5.9			30
25	11/4 16:13	singleProce	1	20	10	3600	300	none	-	60	camunda/zeebe	8.1.2	n1-	24	7	16	metrics	7	7	24	4	128	☑	☐	disk	100	ssd-ext4	☑	35	NO	2	32	25	64	128	2	3	3	3	35	jobex	0	0.30				4.9		2.0			30
25	11/7 14:16	singleProce	1	20	10	3600	300	none	-	60	camunda/zeebe	8.1.2	n1-	24	7	16	metrics	7	7	24	4	128	☑	☐	disk	100	ssd-ext4	☑	35	NO	2	32	25	64	128	2	3	3	3	35	jobex	0	0.25				9.5		??			30
26	11/7 10:18	singleProce	1	20	10	3600	300	none	-	60	camunda/zeebe	8.1.2	n1-	24	7	16	metrics	7	7	24	4	128	☑	☑	mem	100	ssd-ext4	☑	35	NO	2	32	25	64	128	2	3	3	3	35	jobex	0	0.26				7.5		??			30
27	11/7 10:38	singleProce	1	20	10	3600	300	none	-	60	camunda/zeebe	8.1.2	n1-	24	7	16	metrics	7	7	24	4	128	☑	☑	disk	100	ssd-ext4	☑	0	NO	2	32	25	64	128	2	3	3	3	0	jobex	0	0.30				0.1		0.1			30
28	11/7 12:27	singleProce	1	20	10	3600	300	none	-	60	camunda/zeebe	8.1.2	n1-	24	7	16	metrics	7	7	24	4	128	☑	☑	disk	100	ssd-ext4	☑	35	GZ	2	32	25	64	128	2	3	3	3	35	jobex	0	0.30				5.0		3.0			30
29	11/7 13:10	singleProce	1	20	10	3600	300	none	-	60	camunda/zeebe	8.1.2	n1-	24	7	16	metrics	7	7	24	4	128	☑	☑	disk	100	ssd-ext4	☑	35	SN.	2	32	25	64	128	2	3	3	3	35	jobex	0	0.30				5.0		3.0			30
30	11/7 17:23	singleProce	1	20	10	3600	300	none	-	60	camunda/zeebe	8.1.2	n1-	24	7	16	metrics	7	7	24	4	128	☑	☐	disk	100	ssd-ext4	☑	35	GZ	4	64	25	64	128	2	3	3	3	35	jobex	0	0.30				5.0		2.7			
31	11/7 17:00	singleProce	1	20	10	3600	300	none	-	60	camunda/zeebe	8.1.2	n1-	24	7	16	metrics	7	7	24	4	128	☑	☐	disk	100	ssd-ext4	☑	35	GZ	2	64	25	64	128	2	3	3	3	35	jobex	0	0.30				5.0		3.1			
32	11/7 17:52	singleProce	1	20	10	3600	300	none	-	60	camunda/zeebe	8.1.2	n1-	24	7	16	metrics	7	7	24	4	128	☑	☐	disk	100	ssd-ext4	☑	35	GZ	8	64	25	64	128	2	3	3	3	35	jobex	0	0.30				5.0		2.7			
33	11/8 8:49	singleProce	1	20	10	3600	300	none	-	60	camunda/zeebe	8.1.2	n1-	24	8	16	metrics	8	8	24	4	128	☑	☐	disk	100	ssd-ext4	☑	35	GZ	2	128	25	64	128	2	3	3	3	35	jobex	0	0.30				5.0		2.7			
34	11/8 9:00	singleProce	1	20	10	3600	300	none	-	60	camunda/zeebe	8.1.2	n1-	24	8	16	metrics	8	8	24	4	128	☑	☑	disk	100	ssd-ext4	☑	35	GZ	2	128	25	64	128	2	3	3	3	35	jobex	0	0.30				5.0		2.7			

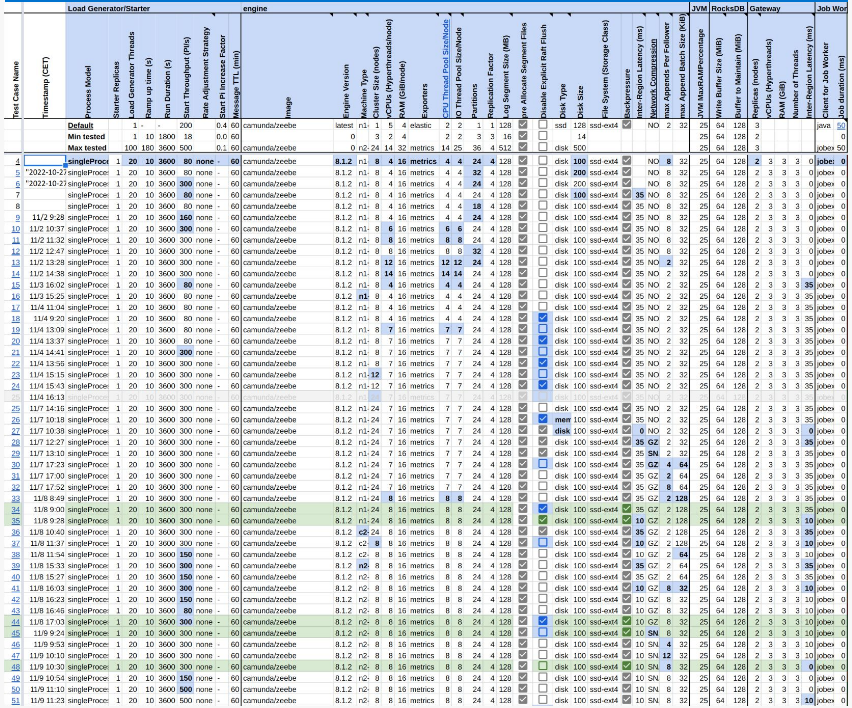
Zeebe Benchmark Result Template

- Precisely documents
  - Configuration parameters
  - Test results, e.g.
    - Throughput
    - Duration
- Graphical diff between configurations
- Heatmap of best results
- Planning new benchmark runs
- Driving k8s automation to iterate over planned configurations

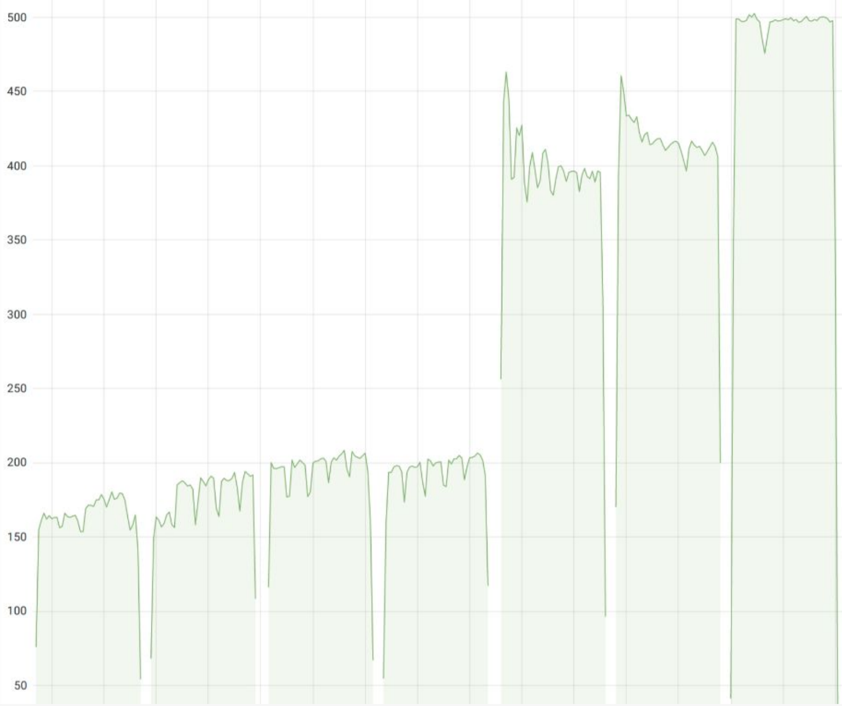
https://camunda.com/blog/2020/11/zeebe-performance-tool/

Zeebe Tuner (parameterized Kubernetes tests)

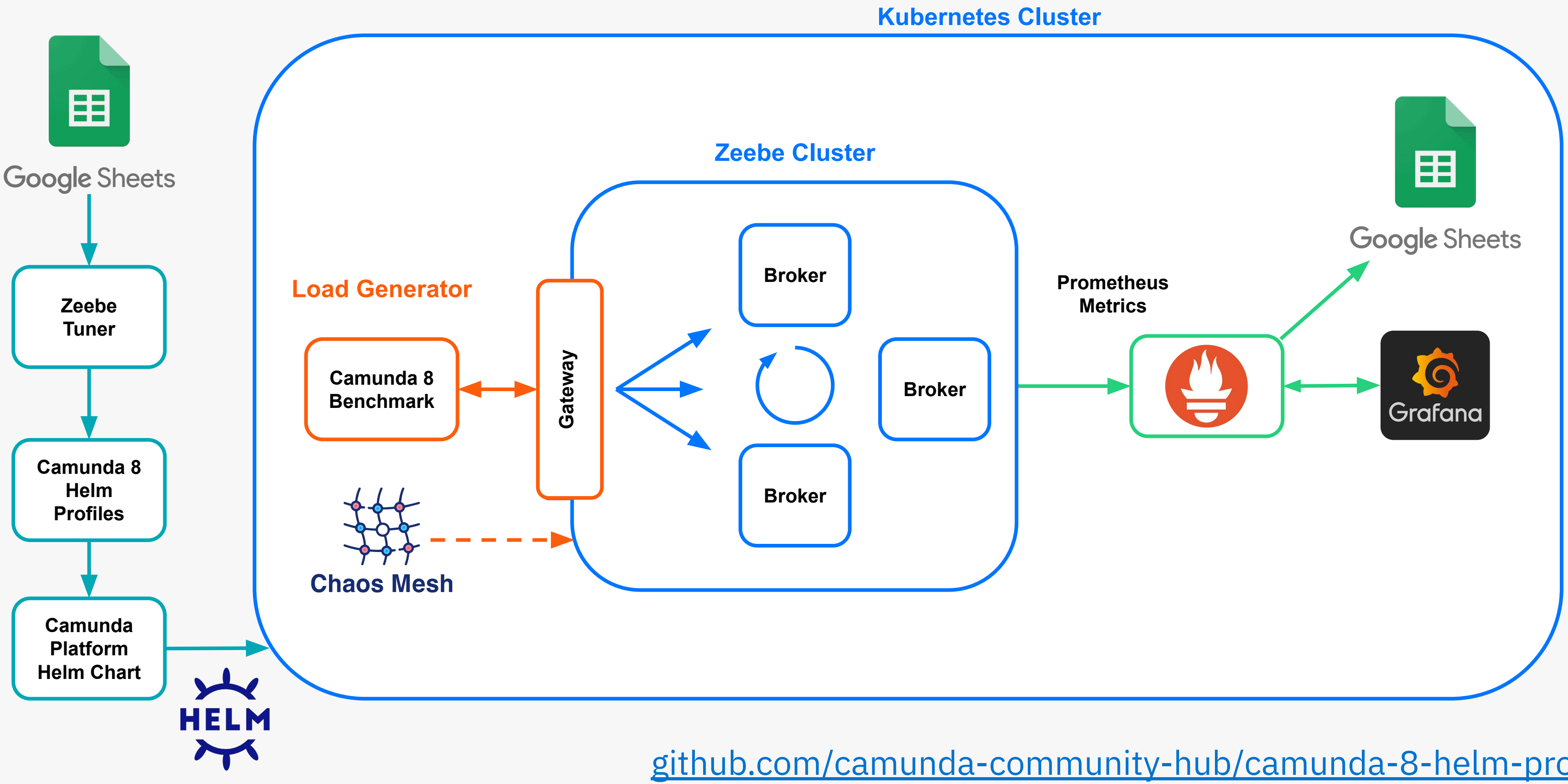
Input: Zeebe Configurations



Output: Prometheus Metrics



Iterative Benchmark Setup with Zeebe Tuner



github.com/camunda-community-hub/camunda-8-helm-profiles

Zeebe Tuner (parameterized Kubernetes tests)

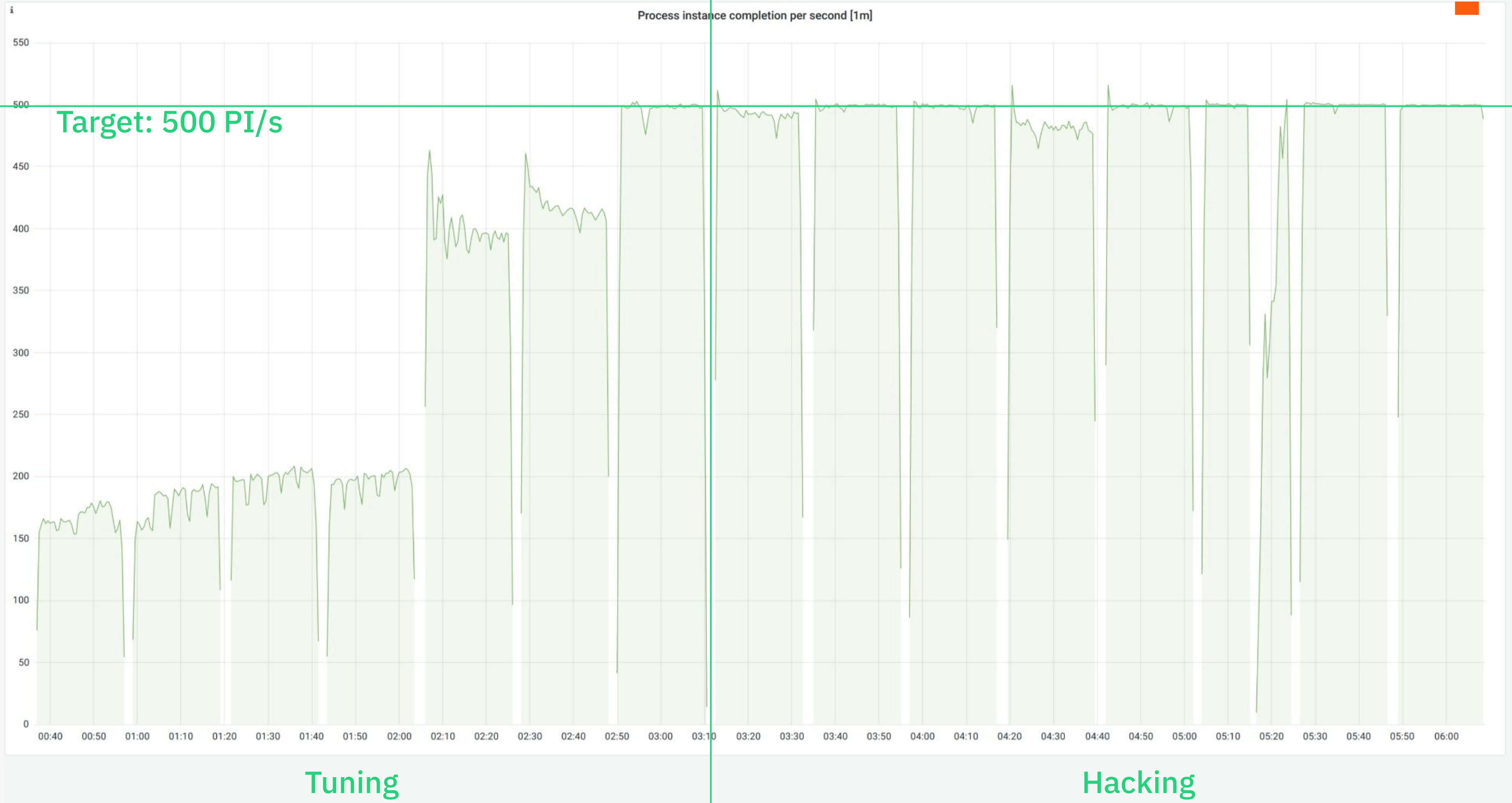
- Zeebe Tuner project (Spring Boot)
  - Programmatically reads Benchmark Template Spreadsheet
  - Creates directory + scripts to run each test
  - Tests can be shared and re-run
  - One Bash script to run multiple tests in sequence
  - Saves url to easily view results
  - Able to run tests unattended
  - Results can be viewed as Grafana Chart and analyzed

Google Sheets

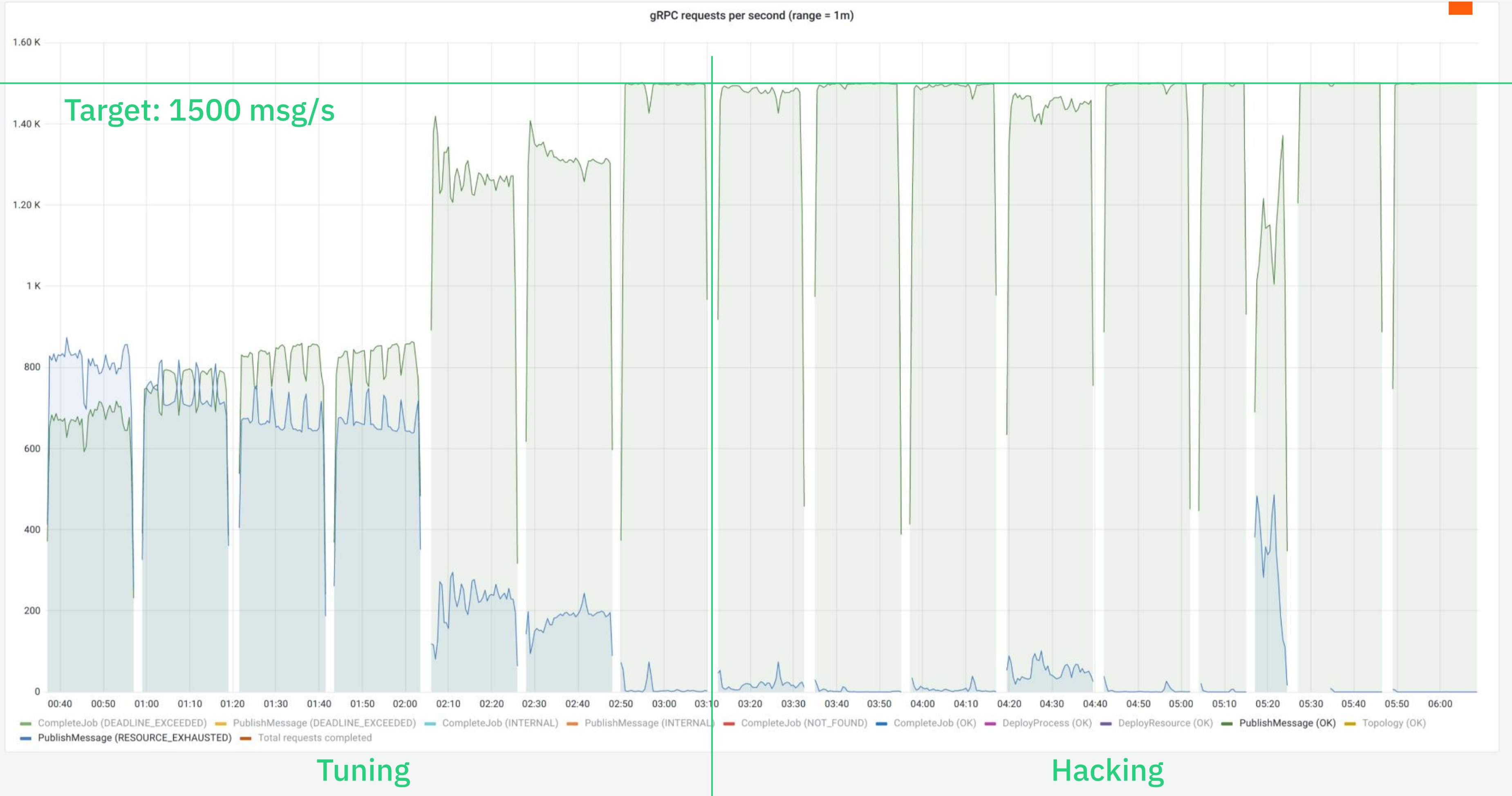
Grafana

github.com/camunda-consulting/zeebe-tuner

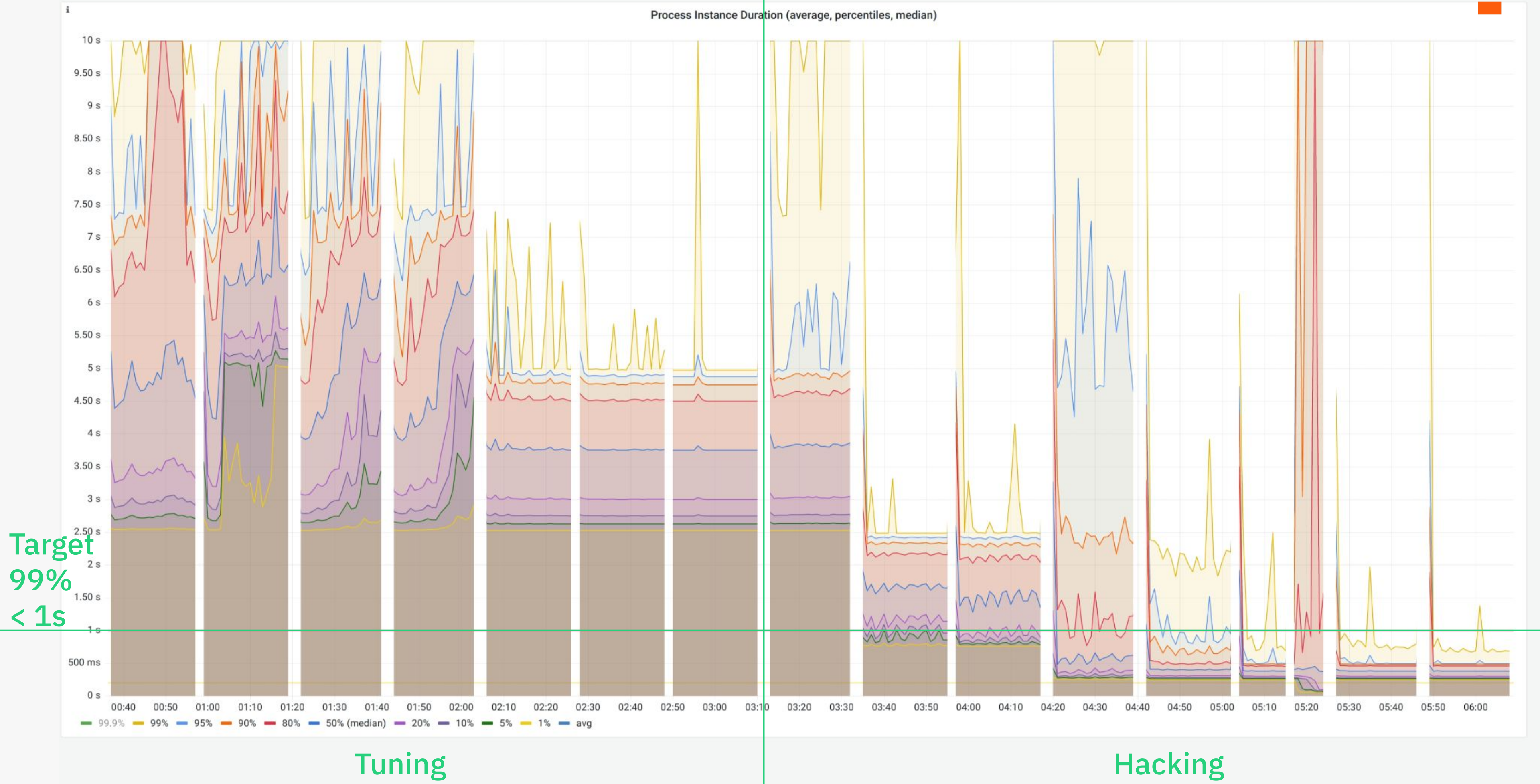
Throughput (PI/s)



Message Throughput & Backpressure

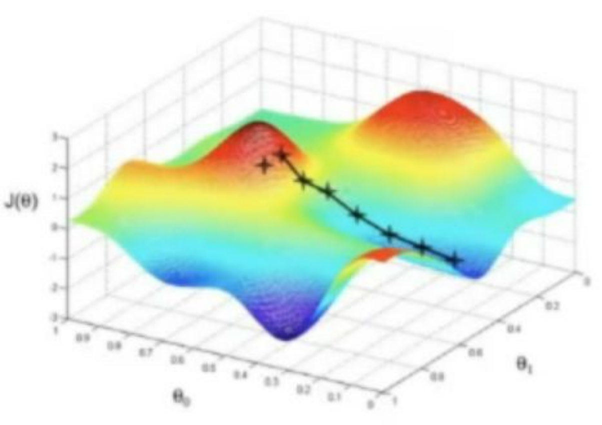


Process Instance Duration (Latency)



Test Strategies

- Exploratory tests: starting from a baseline change one parameter at a time to find new directions
- Navigating the terrain: iterate through various values within a parameter's value range to find local optimum, then iterate over other parameters to find global optimum



Optimize Performance First, Hardware Cost Second

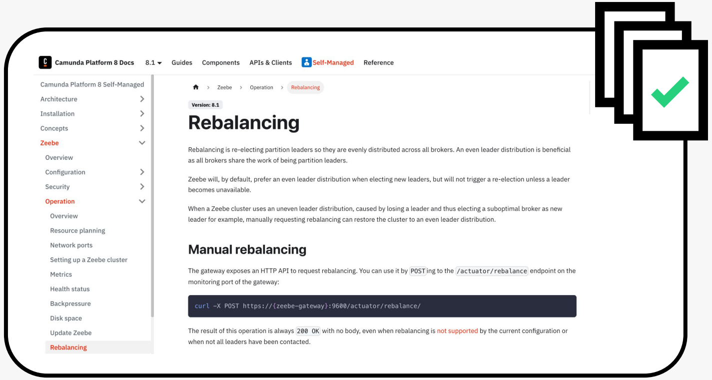
- First test with “unlimited” hardware, e.g. reserve more CPUs and memory than the brokers could possibly use
  - That reduces the number benchmark parameters to iterate over
  - Find optimal number of partitions per broker and other parameters
- Then measure CPU and memory consumption and reduce hardware limits to optimize costs
- Also long-running tests to check stability should be done later

Performance engineering is a process

- A change in the code may invalidate prior optimization results, e.g.
  - Number of workers
  - Number of job types
- Parameters are interrelated, i.e. changing one requires changing others, e.g.
  - Number of partitions & brokers
  - vCPUs & thread pool sizes
- That's why optimization is an ongoing process

Current Tuning Best Practices

- Always enforce leader balancing
- Scale partitions & brokers
- Latest generation CPUs (Arm)
- Fastest possible disks & file systems, e.g. XFS
- Disable RocksDB WAL
- Disable explicit Raft flush (takes disk out of critical path)
- Experimental
    - Job exporter
    - Prefer local brokers by selecting correlation key



Resources

Bernd Ruecker's Blog Articles

- How to Benchmark Your Camunda 8 Cluster
- How to Achieve Geo-redundancy with Zeebe

GitHub

- camunda-consulting/zeebe-tuner
- camunda-community-hub/camunda-8-benchmark

Camunda Platform 8 Docs

- Metrics
- Deployment options

Contact Us

- Contact Form
- mailto:info@camunda.com
- Try Camunda Platform 8 for free

CAMUNDA COMMUNITY SUMMIT 2023

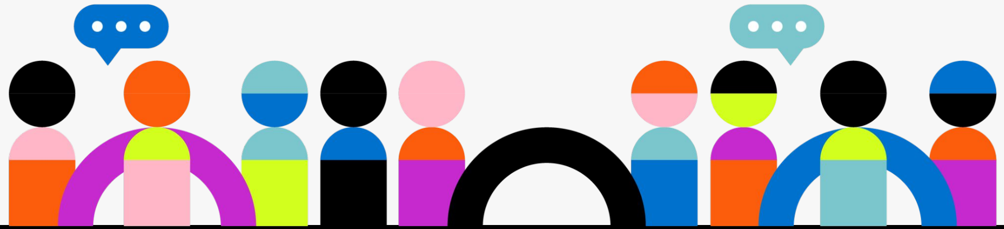
The [hybrid] conference for Camunda Developers

May 10-11

The Camunda Community Summit will be a unique, interactive two-day event exclusively for developers, enterprise architects, and process automation experts.

Featuring deep-dive technical topics, live coding demonstrations, and the latest advances in process automation, our annual summit will be held in an immersive, hybrid and collaborative format.

register here → https://bit.ly/3IHGZip



THANK YOU

twitter.com/camunda

linkedin.com/company/camunda

camunda.com