Grafana Labs

Where’s your money going?

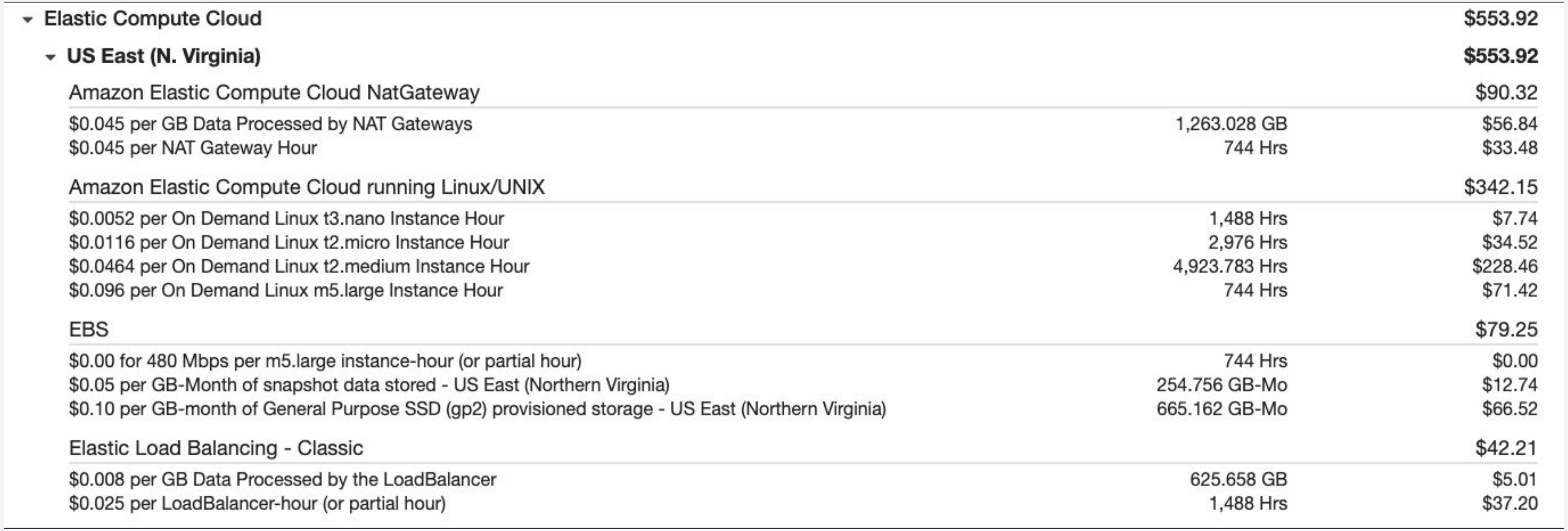
The Beginners Guide To Measuring Kubernetes Costs

Erik Sommer

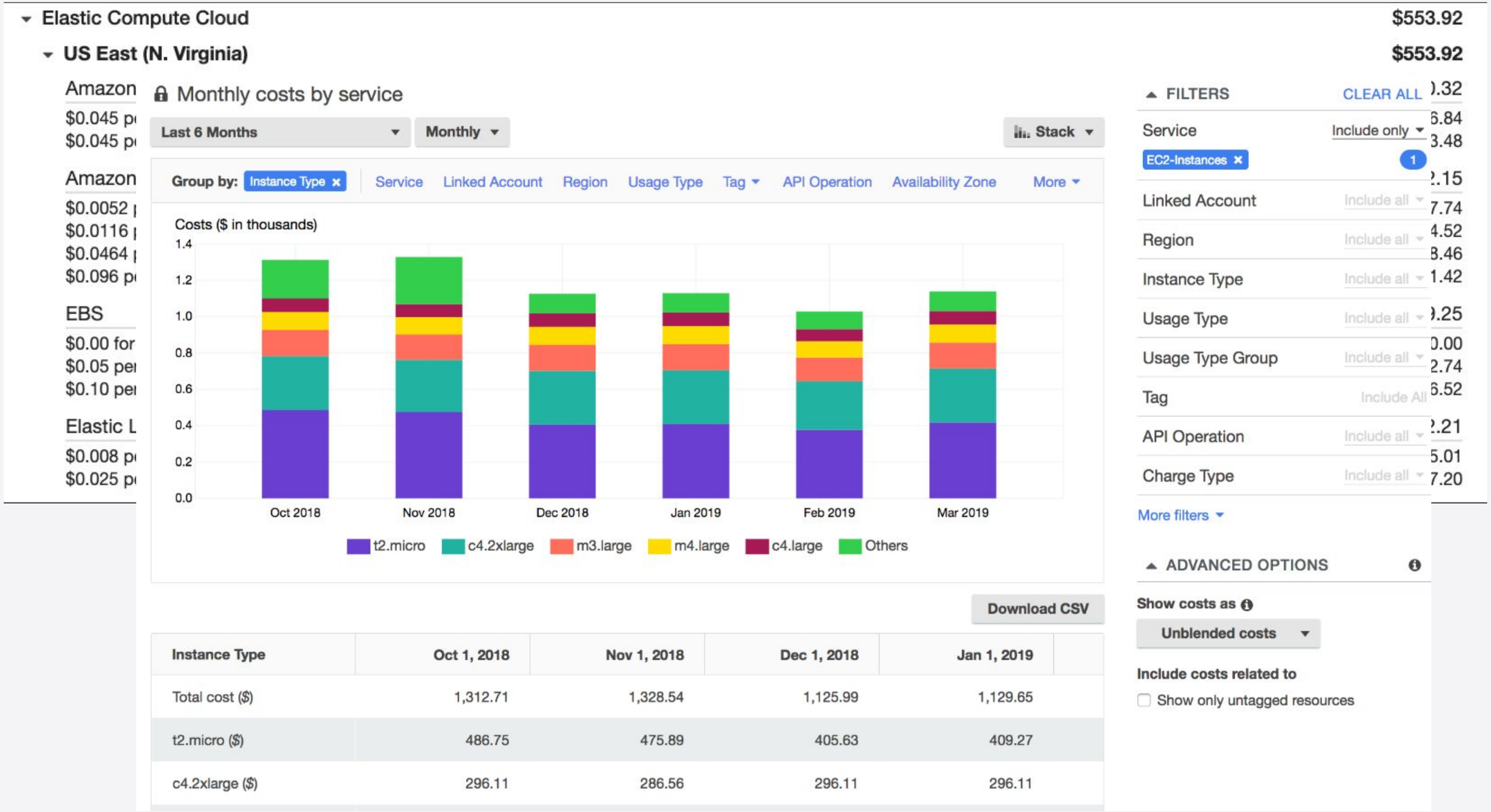
Sr. Software Engineer

What you can expect

- The Problem: When the bill doesn’t match your metrics
- Mental model to attributes costs in k8s
- How to compose PromQL queries
- Shortcomings / Where to go next



Elastic Compute Cloud		**$553.92**
US East (N. Virginia)		**$553.92**
Amazon Elastic Compute Cloud NatGateway		$90.32
$0.045 per GB Data Processed by NAT Gateways	1,263.028 GB	$56.84
$0.045 per NAT Gateway Hour	744 Hrs	$33.48
Amazon Elastic Compute Cloud running Linux/UNIX		$342.15
$0.0052 per On Demand Linux t3.nano Instance Hour	1,488 Hrs	$7.74
$0.0116 per On Demand Linux t2.micro Instance Hour	2,976 Hrs	$34.52
$0.0464 per On Demand Linux t2.medium Instance Hour	4,923.783 Hrs	$228.46
$0.096 per On Demand Linux m5.large Instance Hour	744 Hrs	$71.42
EBS		$79.25
$0.00 for 480 Mbps per m5.large instance-hour (or partial hour)	744 Hrs	$0.00
$0.05 per GB-Month of snapshot data stored - US East (Northern Virginia)	254.756 GB-Mo	$12.74
$0.10 per GB-month of General Purpose SSD (gp2) provisioned storage - US East (Northern Virginia)	665.162 GB-Mo	$66.52
Elastic Load Balancing - Classic		$42.21
$0.008 per GB Data Processed by the LoadBalancer	625.658 GB	$5.01
$0.025 per LoadBalancer-hour (or partial hour)	1,488 Hrs	$37.20



Elastic Compute Cloud $553.92

US East (N. Virginia) $553.92

Amazon).32

$0.045 p 6.84

$0.045 p 3.48

Amazon 2.15

$0.0052 7.74

$0.0116 4.52

$0.0464 3.46

$0.096 p 1.42

EBS).25

$0.00 for 0.00

$0.05 per 2.74

$0.10 per 6.52

Elastic L 2.21

$0.008 p 5.01

$0.025 p 7.20

Monthly costs by service

Last 6 Months ▾ | Monthly ▾ | Stack ▾

Group by: Instance Type ✕ | Service | Linked Account | Region | Usage Type | Tag ▾ | API Operation | Availability Zone | More ▾

Download CSV

Instance Type	Oct 1, 2018	Nov 1, 2018	Dec 1, 2018	Jan 1, 2019
Total cost ($)	1,312.71	1,328.54	1,125.99	1,129.65
t2.micro ($)	486.75	475.89	405.63	409.27
c4.2xlarge ($)	296.11	286.56	296.11	296.11

FILTERS CLEAR ALL

Service	Include only ▾
EC2-Instances ✕	1
Linked Account	Include all ▾
Region	Include all ▾
Instance Type	Include all ▾
Usage Type	Include all ▾
Usage Type Group	Include all ▾
Tag	Include All
API Operation	Include all ▾
Charge Type	Include all ▾

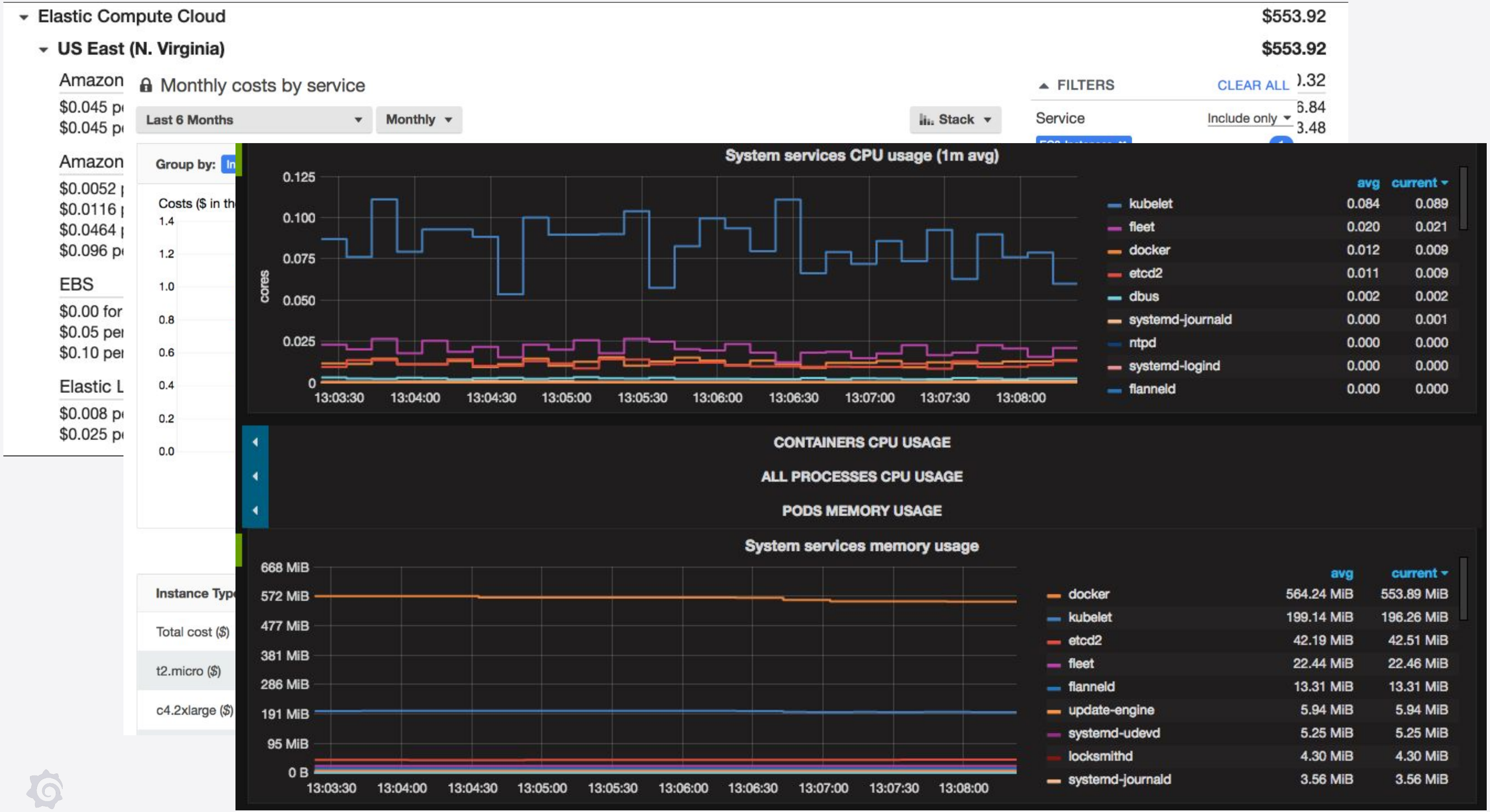
More filters ▾

ADVANCED OPTIONS

Show costs as: Unblended costs ▾

Include costs related to

□ Show only untagged resources



Elastic Compute Cloud $553.92

US East (N. Virginia) $553.92

Amazon).32

$0.045 p 6.84

$0.045 p 3.48

Amazon

$0.0052 |

$0.0116 |

$0.0464 |

$0.096 p

EBS

$0.00 for

$0.05 pe

$0.10 pe

Elastic L

$0.008 p

$0.025 p

Monthly costs by service

Last 6 Months ▾ Monthly ▾ Stack ▾

FILTERS CLEAR ALL

Service Include only ▾

Group by: In

Costs ($ in th

Instance Typ

Total cost ($)

t2.micro ($)

c4.2xlarge ($)

System services CPU usage (1m avg)

	avg	current ▾
kubelet	0.084	0.089
fleet	0.020	0.021
docker	0.012	0.009
etcd2	0.011	0.009
dbus	0.002	0.002
systemd-journald	0.000	0.001
ntpd	0.000	0.000
systemd-logind	0.000	0.000
flanneld	0.000	0.000

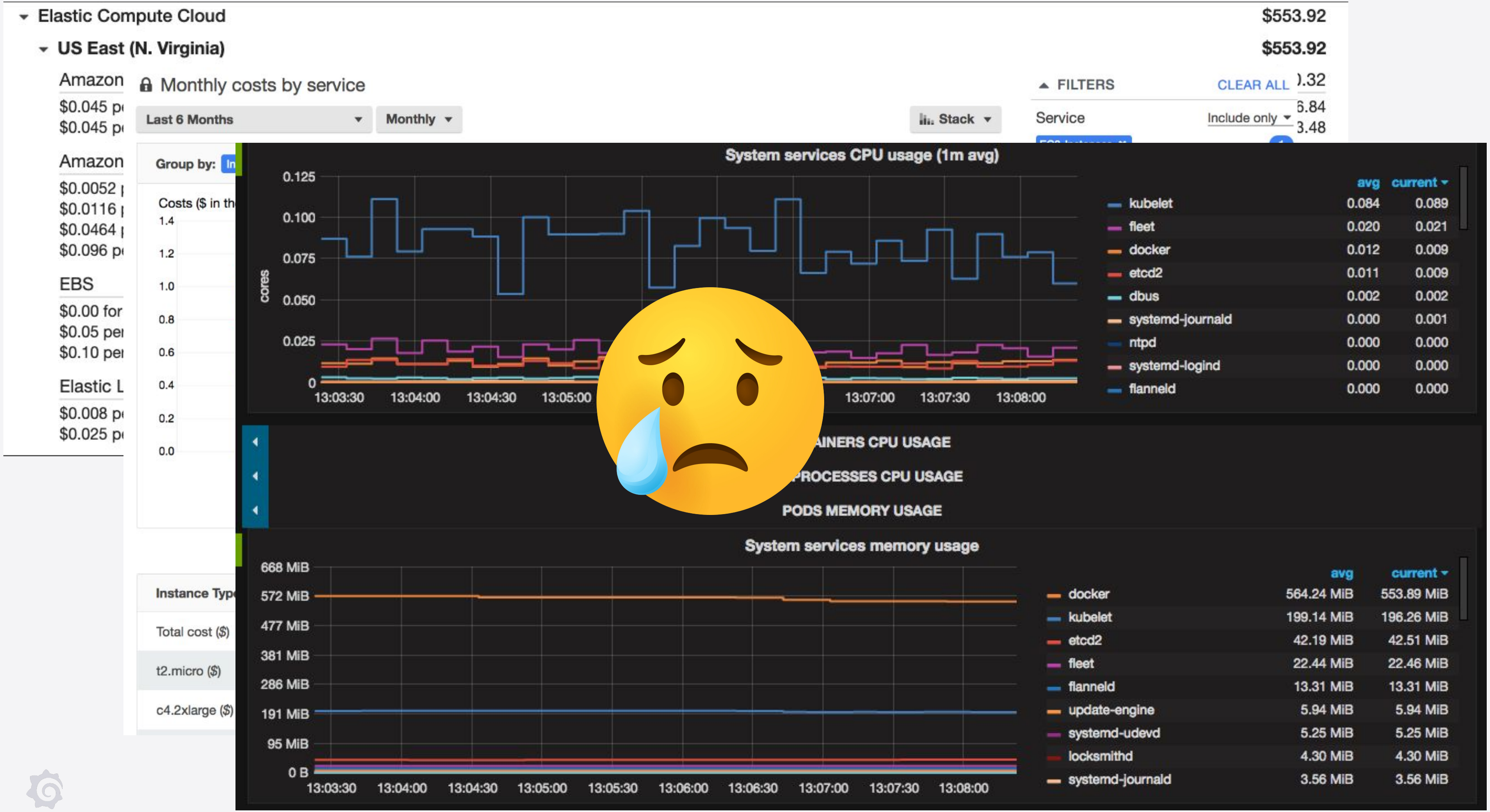
CONTAINERS CPU USAGE

ALL PROCESSES CPU USAGE

PODS MEMORY USAGE

System services memory usage

	avg	current ▾
docker	564.24 MiB	553.89 MiB
kubelet	199.14 MiB	196.26 MiB
etcd2	42.19 MiB	42.51 MiB
fleet	22.44 MiB	22.46 MiB
flanneld	13.31 MiB	13.31 MiB
update-engine	5.94 MiB	5.94 MiB
systemd-udevd	5.25 MiB	5.25 MiB
locksmithd	4.30 MiB	4.30 MiB
systemd-journald	3.56 MiB	3.56 MiB



Elastic Compute Cloud $553.92

US East (N. Virginia) $553.92

Amazon	).32
$0.045 p	6.84
$0.045 p	3.48
Amazon	
$0.0052	
$0.0116	
$0.0464	
$0.096 p	
EBS	
$0.00 for	
$0.05 per	
$0.10 per	
Elastic L	
$0.008 p	
$0.025 p	

Monthly costs by service

Instance Typ
Total cost ($)
t2.micro ($)
c4.2xlarge ($)

System services CPU usage (1m avg)

	avg	current
kubelet	0.084	0.089
fleet	0.020	0.021
docker	0.012	0.009
etcd2	0.011	0.009
dbus	0.002	0.002
systemd-journald	0.000	0.001
ntpd	0.000	0.000
systemd-logind	0.000	0.000
flanneld	0.000	0.000

AINERS CPU USAGE

PROCESSES CPU USAGE

PODS MEMORY USAGE

System services memory usage

	avg	current
docker	564.24 MiB	553.89 MiB
kubelet	199.14 MiB	196.26 MiB
etcd2	42.19 MiB	42.51 MiB
fleet	22.44 MiB	22.46 MiB
flanneld	13.31 MiB	13.31 MiB
update-engine	5.94 MiB	5.94 MiB
systemd-udevd	5.25 MiB	5.25 MiB
locksmithd	4.30 MiB	4.30 MiB
systemd-journald	3.56 MiB	3.56 MiB

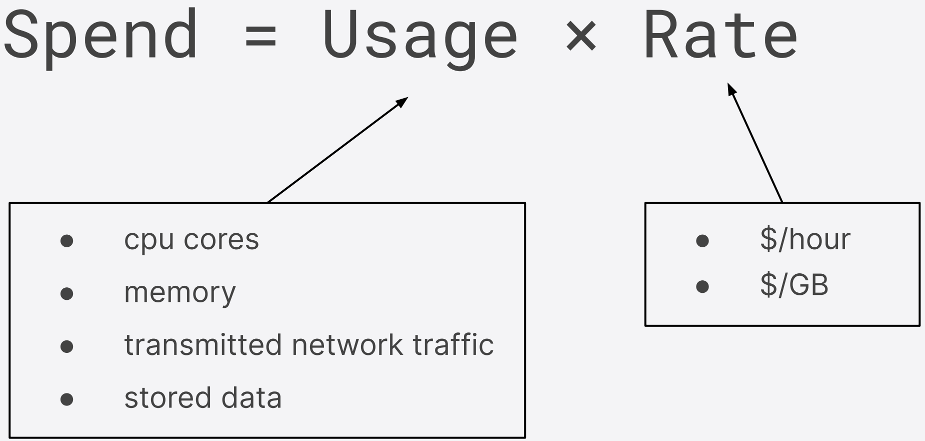
Problems

- Disconnect between billing statement and metrics
- Need to attribute costs of workloads running in Kubernetes
- Difficult to accurately measure Total Cost of Ownership (TCO) of our services

A simple formula with profound implications …

```
Spend = Usage × Rate
```

A simple formula with profound implications …



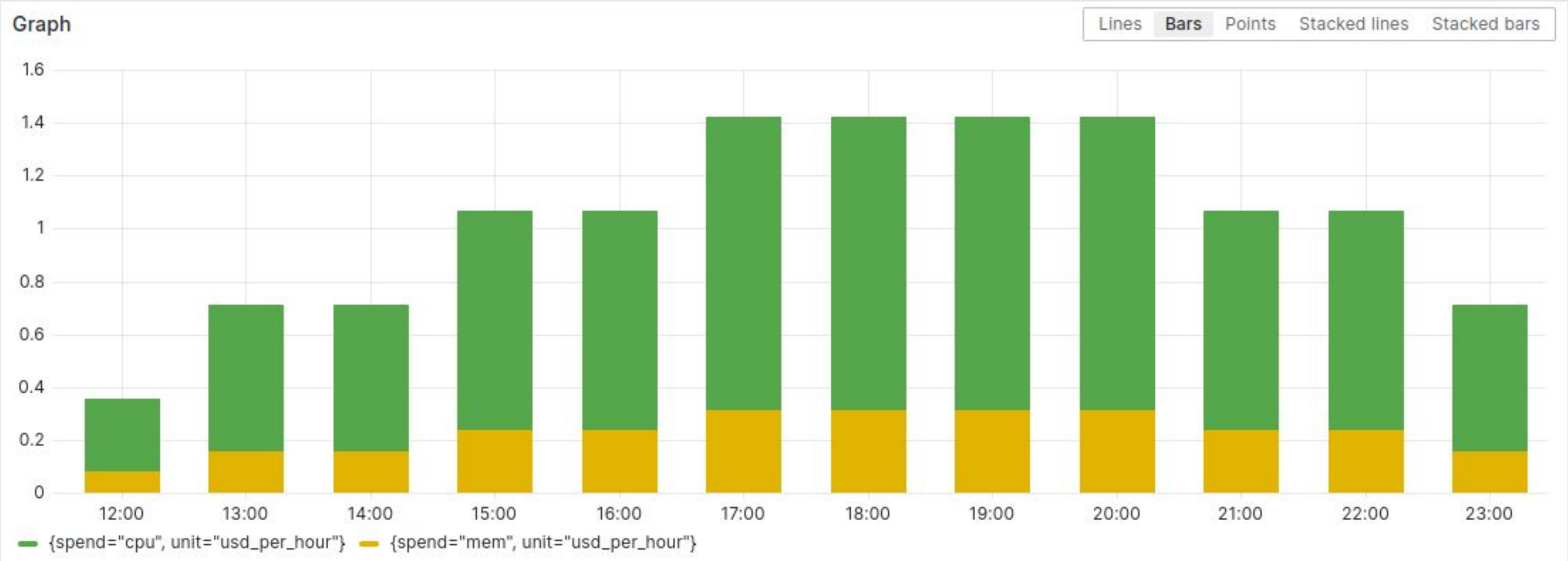
`Spend = Usage × Rate`

- cpu cores
- memory
- transmitted network traffic
- stored data

- $/hour
- $/GB

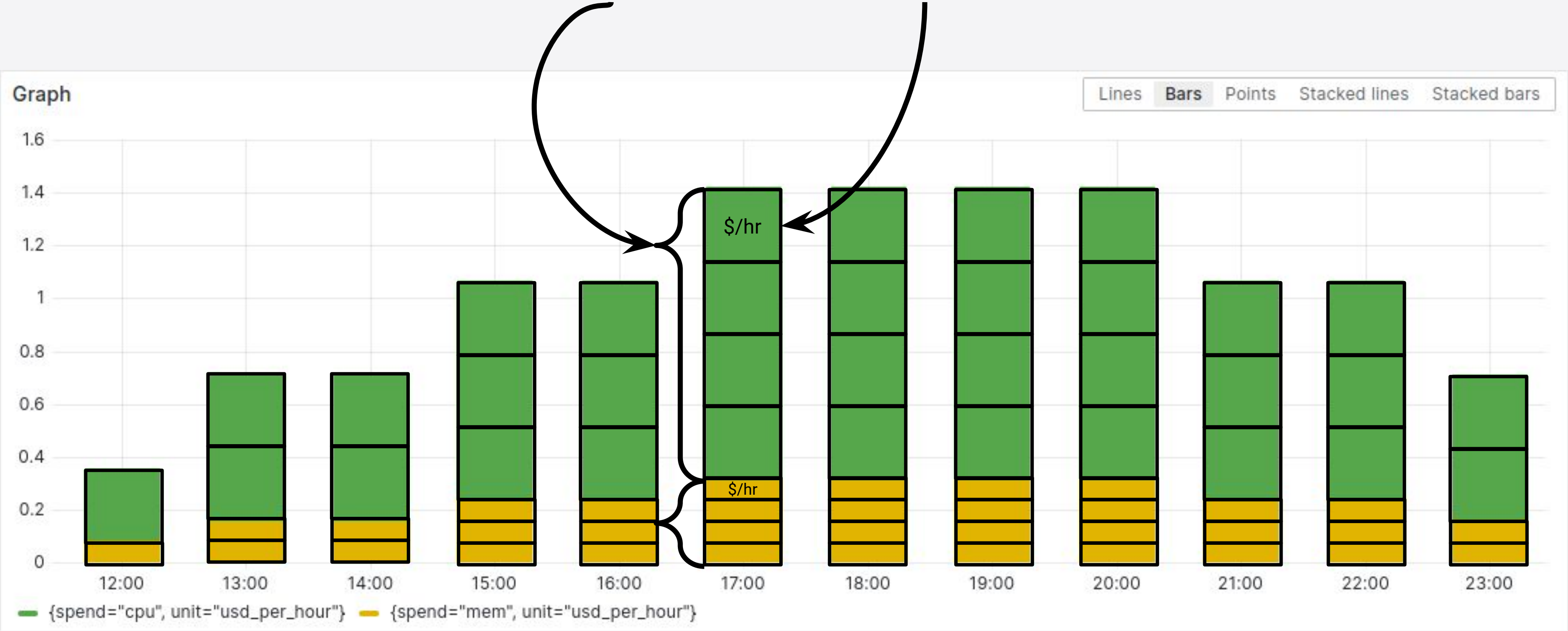
A simple formula with profound implications ...

```
Spend = Usage × Rate
```



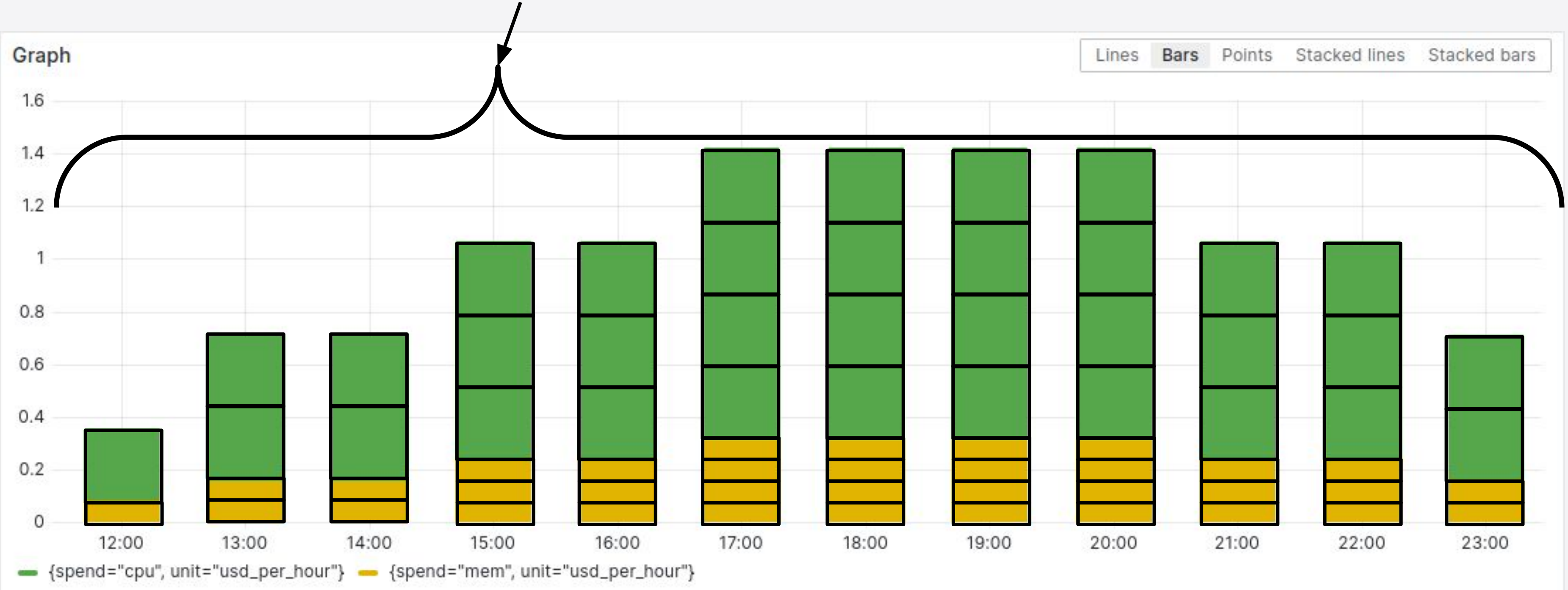
A simple formula with profound implications …

Spend = Usage × Rate



A simple formula with profound implications ...

$$\text{Spend} = \sum \text{Usage} \times \text{Rate} \quad \Delta t$$

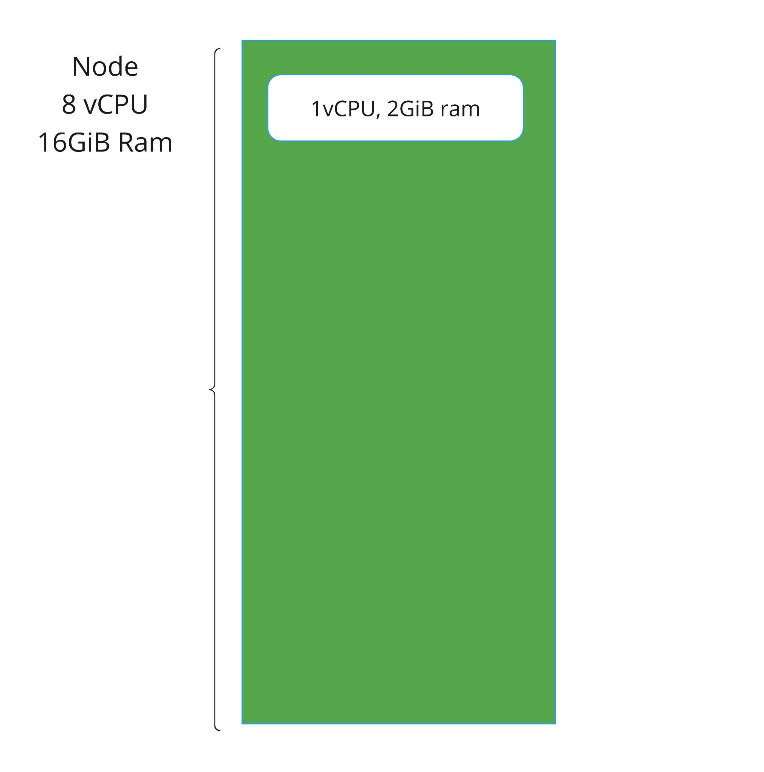


What drives k8s costs

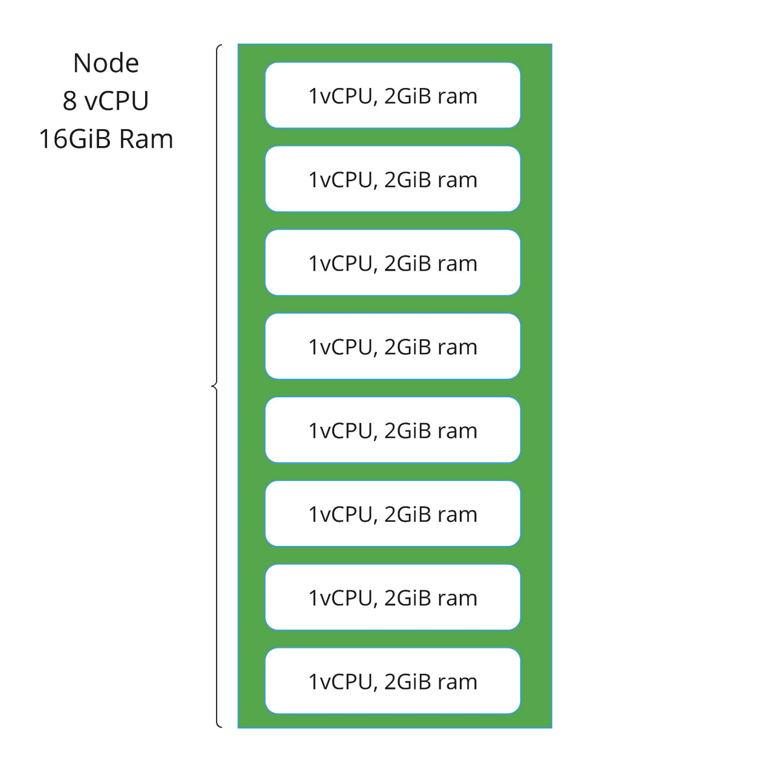
Web service

1vCPU, 2GiB ram

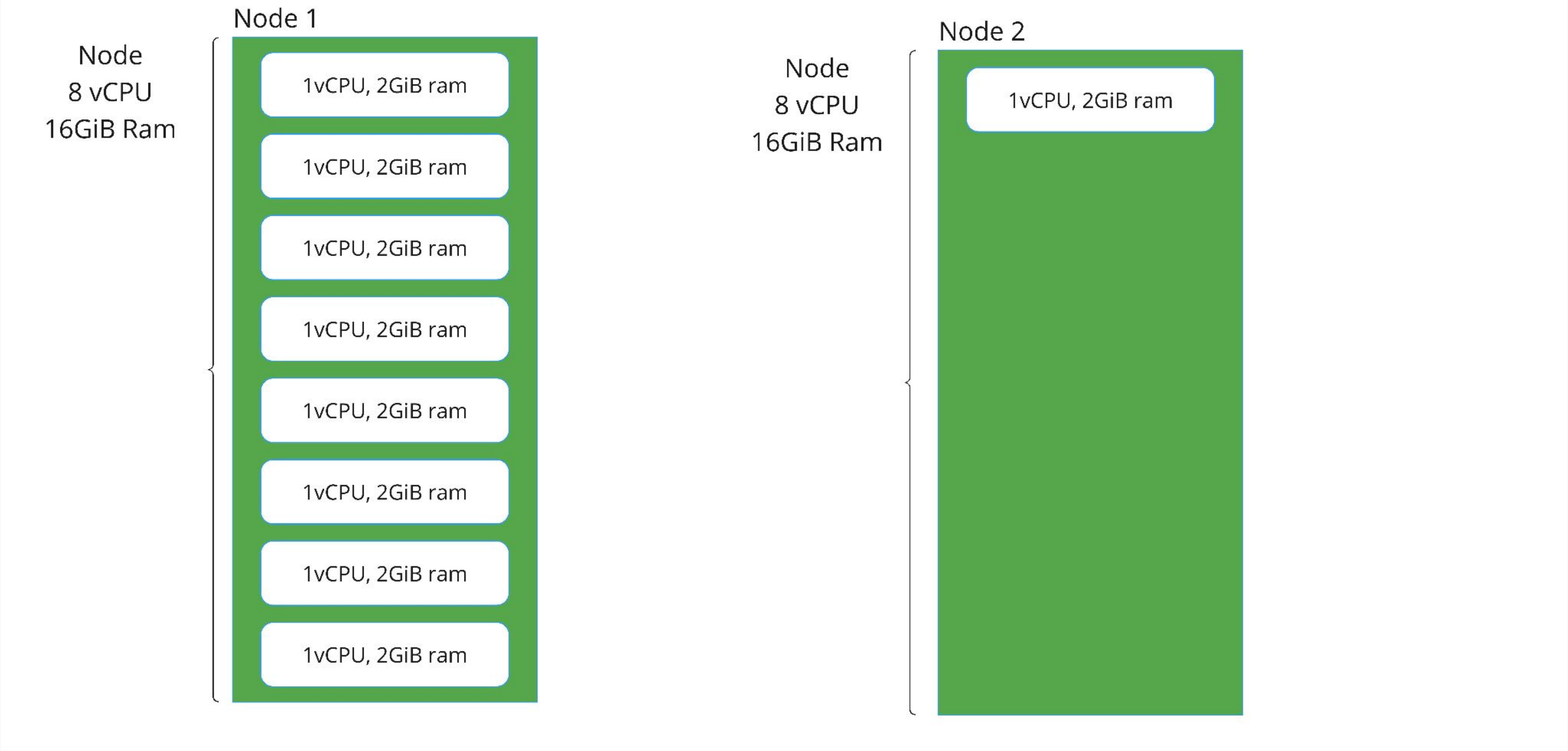
What drives k8s costs



What drives k8s costs



What drives k8s costs



How to measure usage

- cpu|memory of nodes

- cpu|memory requests of workloads

How to measure usage

cpu|memory of nodes

```
kube_node_status_capacity{cluster, resource, node}
```

cpu|memory requests of workloads

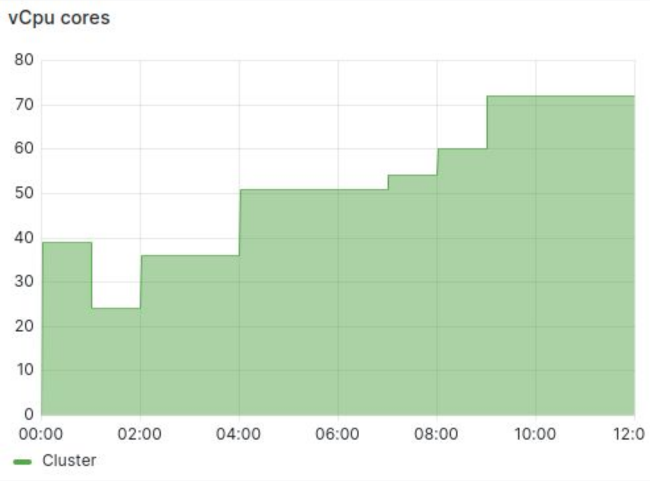
```
kube_pod_container_resource_requests{cluster, resource, node, namespace}
```

How to measure your nodes

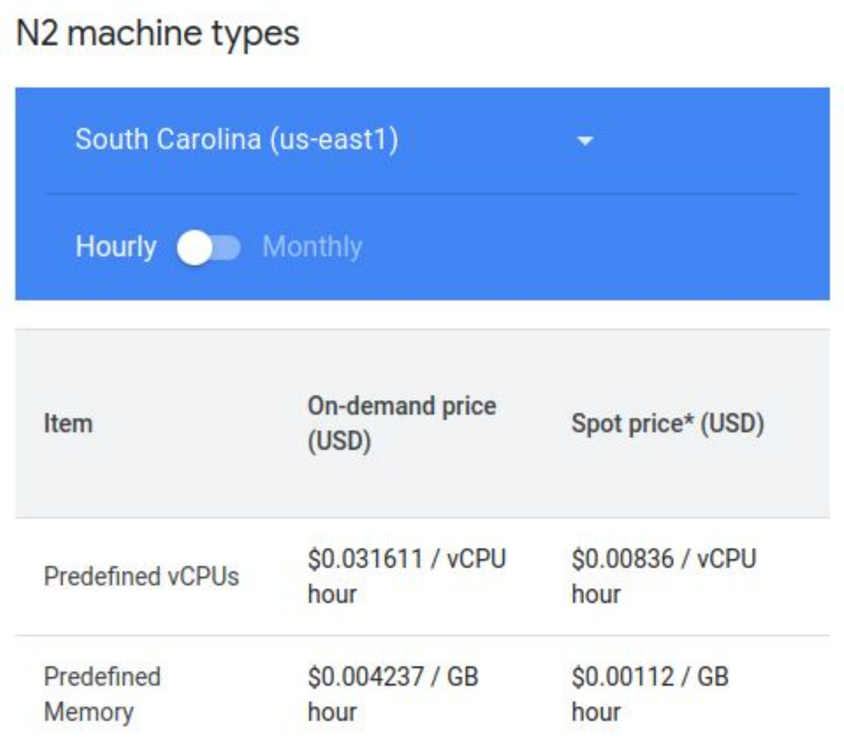
```
sum (
  usage
  *
  rate
)
```

How to measure your nodes

```
sum (
  kube_node_status_capacity(resource="cpu"}
  *
  rate
)
```



How to measure your nodes



N2 machine types

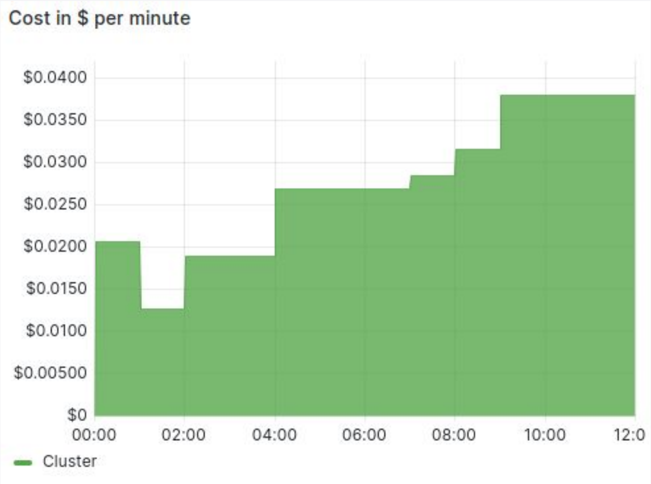
South Carolina (us-east1) ▾

[✓ Hourly] Monthly

Item	On-demand price (USD)	Spot price* (USD)
Predefined vCPUs	$0.031611 / vCPU hour	$0.00836 / vCPU hour
Predefined Memory	$0.004237 / GB hour	$0.00112 / GB hour

How to measure your nodes

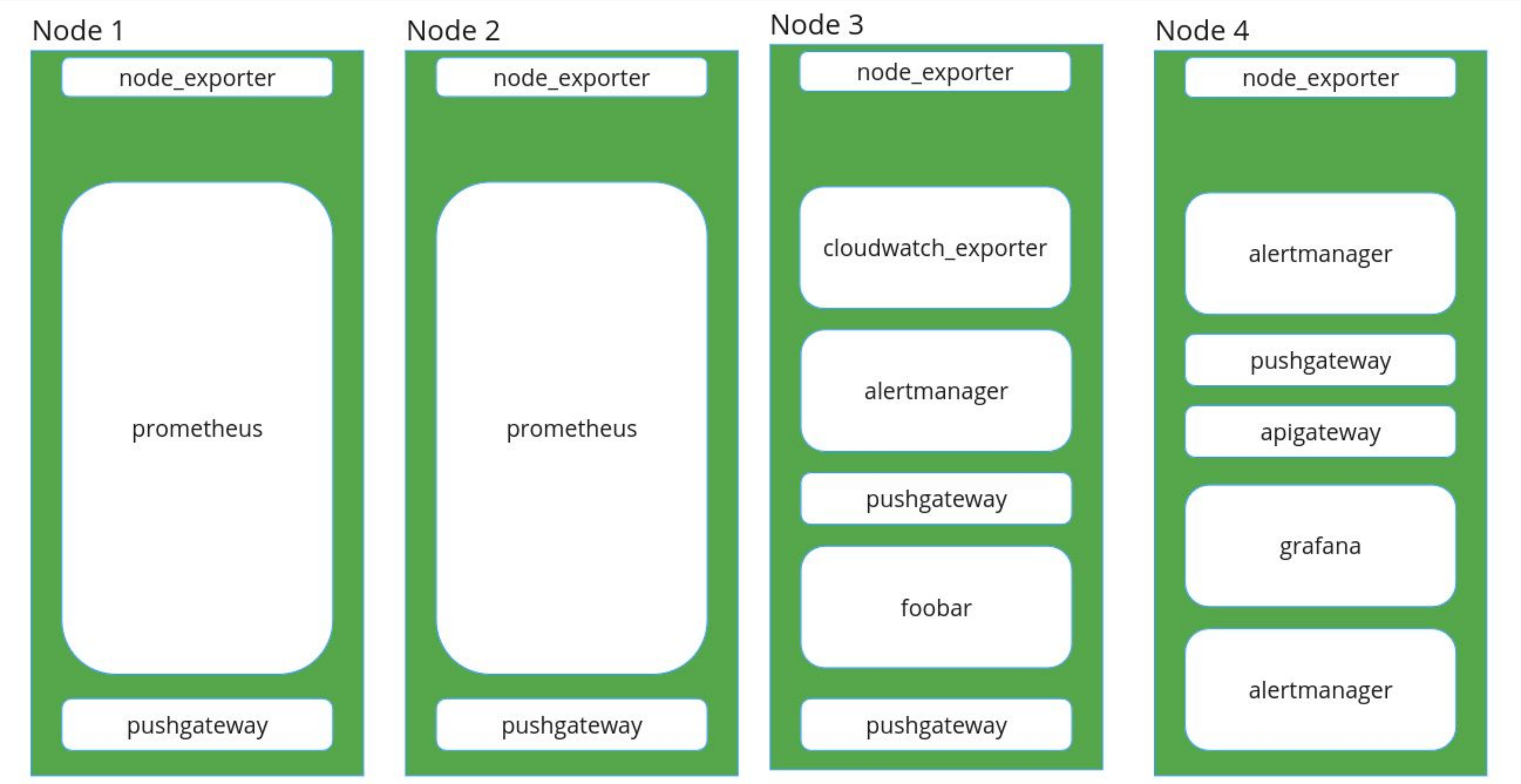
```
sum (
  kube_node_status_capacity(resource="cpu"}
  *
  (0.031611 / 60)
)
```



How to measure your nodes

```
- record: cluster:cost_per_minute:sum
  expr: |
    sum by (cluster) (
      kube_node_status_capacity(resource="cpu"}
      *
      (0.031611 / 60)
    )
  labels:
    resource: "cpu"
```

What drives k8s costs (or who)

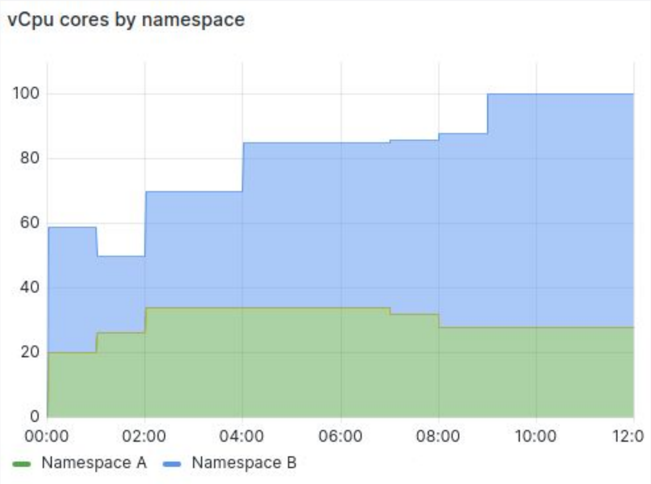


How to measure your workloads impact

```
sum by (namespace) (
  requests
  *
  rate
)
```

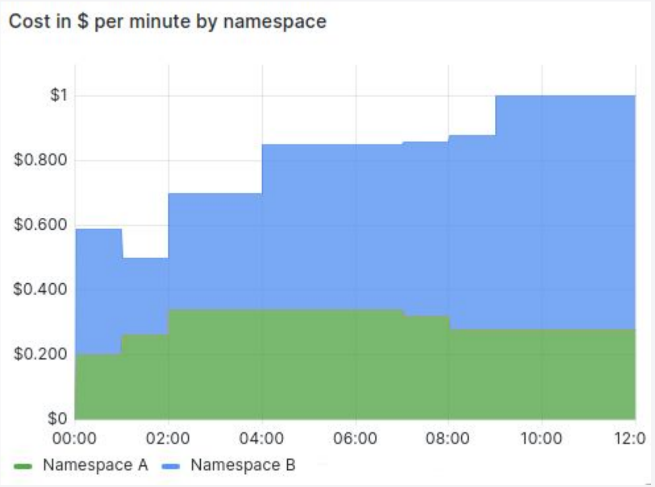
How to measure your workloads impact

```
sum by (namespace) (
  kube_pod_container_resource_requests{resource="cpu"}
  *
  rate
)
```



How to measure your workloads impact

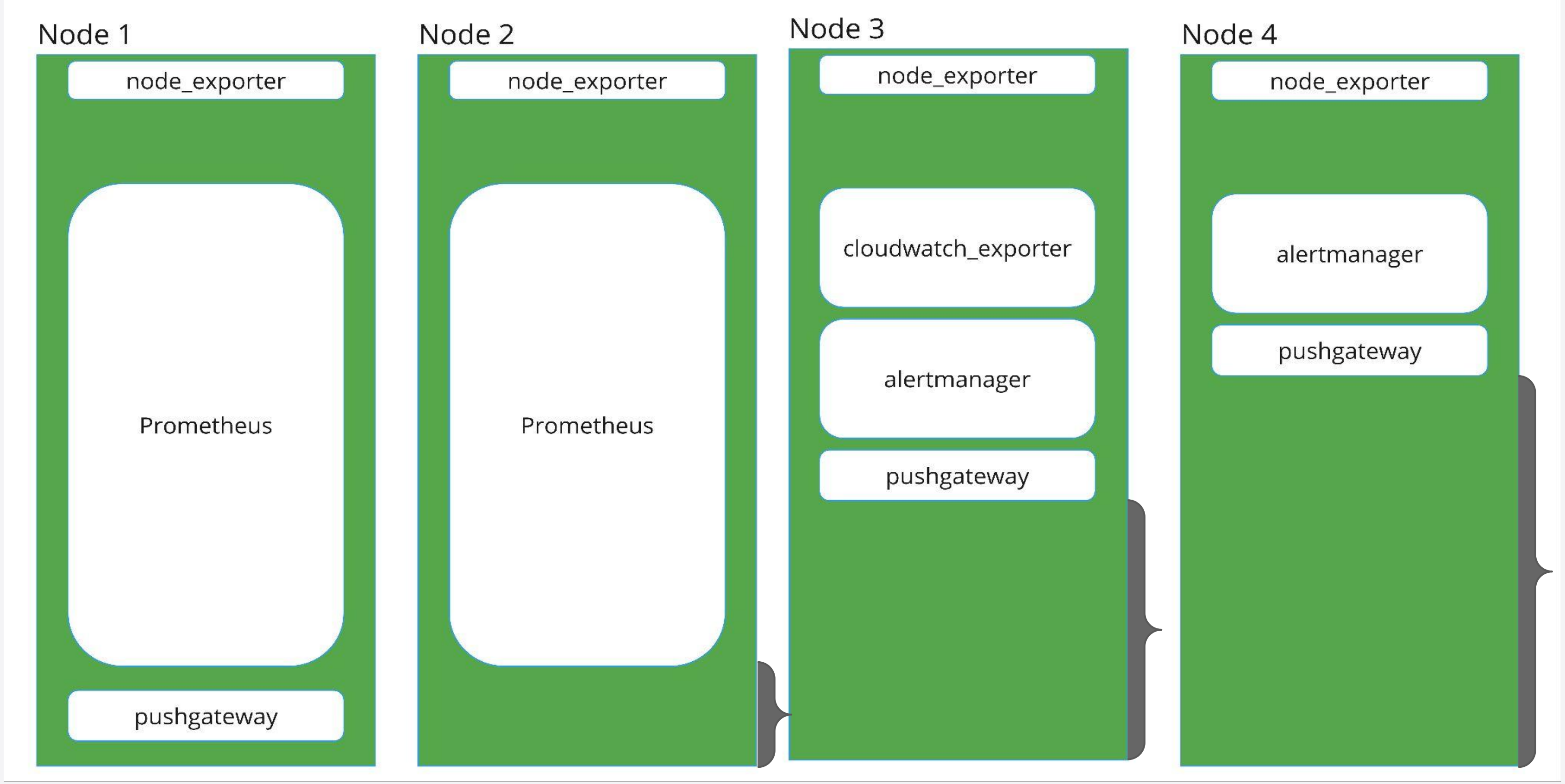
```
sum by (namespace) (
  kube_pod_container_resource_requests{resource="cpu"}
  *
  (0.031611 / 60)
)
```



How to measure your workloads impact

```
- record: cluster_namespace:cost_per_minute:sum
  expr: |
    sum by (cluster, namespace) (
      kube_pod_container_resource_requests{resource="cpu"}
      *
      (0.031611 / 60)
    )
  labels:
    resource: "cpu"
```

What drives k8s costs (realistic)

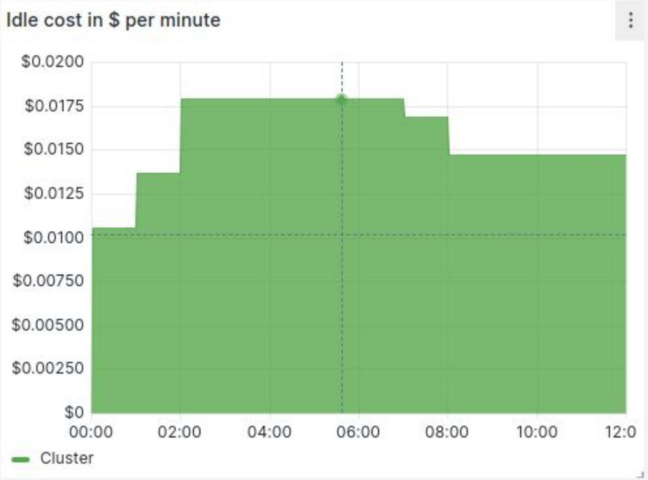


How to measure idle resources

```
sum (
  (
    capacity
    -
    requests
  )
  *
  rate
)
```

How to measure idle resources

```
sum (
  (
    sum by (node) (
      kube_node_status_capacity{resource="cpu"}
    )
    -
    sum by (node) (
      kube_pod_container_resource_requests{resource="cpu"}
    )
  )
  *
  (0.031611 / 60)
)
```



How to measure idle resources

```
- record: cluster_namespace:cost_per_minute:sum
  expr: |
    sum by (cluster) (
      (
        sum by (cluster, node) (
          kube_node_status_capacity{resource="cpu"}
        )
        -
        sum by (cluster, node) (
          kube_pod_container_resource_requests{resource="cpu"}
        )
      )
      *
      (0.031611 / 60)
    )
  labels:
    resource: "cpu"
    namespace: "__idle__"
```

Shortcomings

- This approach only works for homogeneous clusters
- Takes only compute resources into account
- Not all CSPs will give you the breakdown on compute resources costs
- Doesn’t reflect savings plans

What comes next

- Different nodes have different costs
- Projects like OpenCost can help you getting these cost metrics per node

What comes next

```
- record: cluster_namespace:cost_per_minute:sum
  expr: |
    sum by (cluster, namespace) (
      kube_pod_container_resource_requests{resource="cpu"}
      *
      rate
    )
  labels:
    resource: "cpu"
```

What comes next

```
- record: cluster_namespace:cost_per_minute:sum
  expr: |
    sum by (cluster, namespace) (
      kube_pod_container_resource_requests{resource="cpu"}
      *
      on (node) group_left node_cpu_hourly_cost / 60
    )
  labels:
    resource: "cpu"
```

Grafana Labs

Thank you

Grafana Labs

Core Concepts

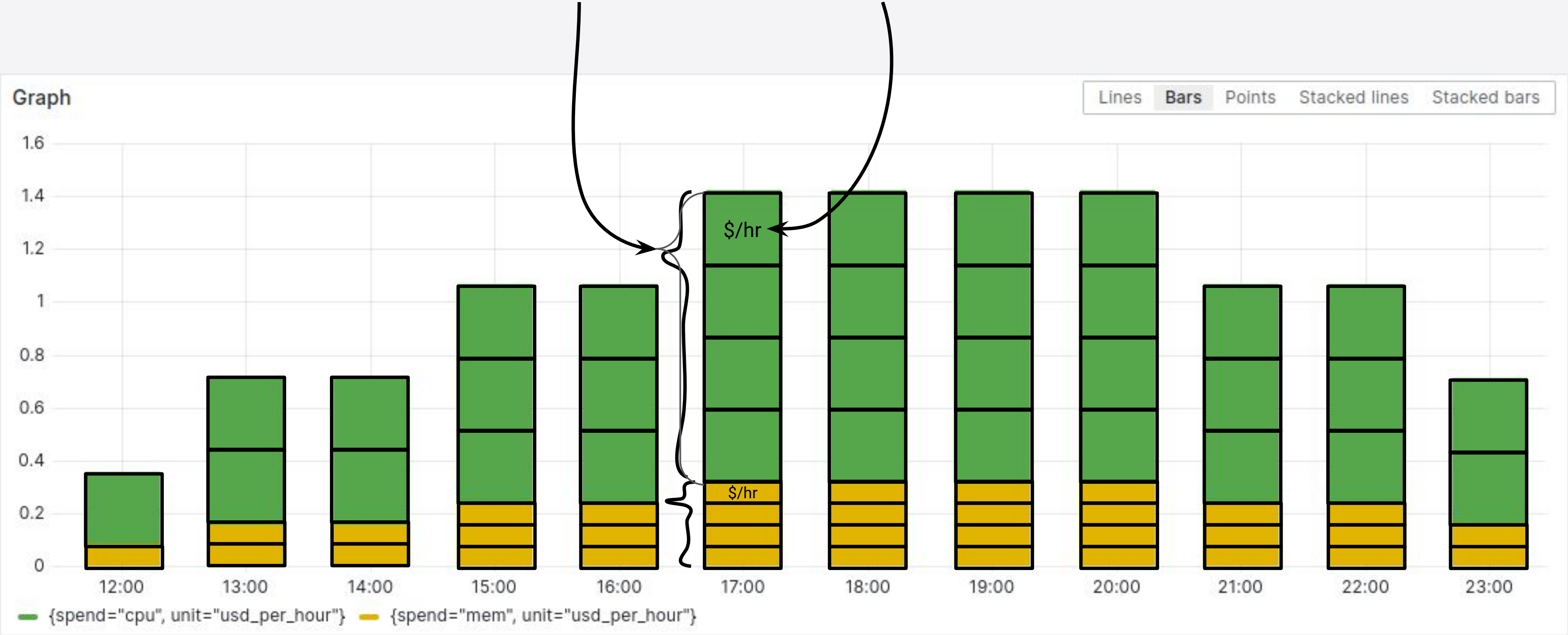
- What drives the costs
- How to measure this
- There is overhead

~~Rough~~ Solid Plan

- 25 minutes:
- 2 Intro slide
- 1 ~~intro matrix for usage and rate~~ just explore the formular rate x usage
- 2 explanation of utilization, usage and allocation
- 2 Where does the cost of a cluster comes from
  - Magic -> boils down to “It’s the nodes
  - Sum of the requests is driving the number of nodes‘
- 1 Establish k8s context
- 2 How to measure nodes -> KSM
- 2 How to measure your workloads impact -> KSM
- 2 Idle resources ….
  - Benefit: reduce idle resources
- 2 Simple step estimate memory/cpu for a node
- 2 combine everything together
  - Benefit: show costs for all workloads
- 2 OpenCost or alternatives

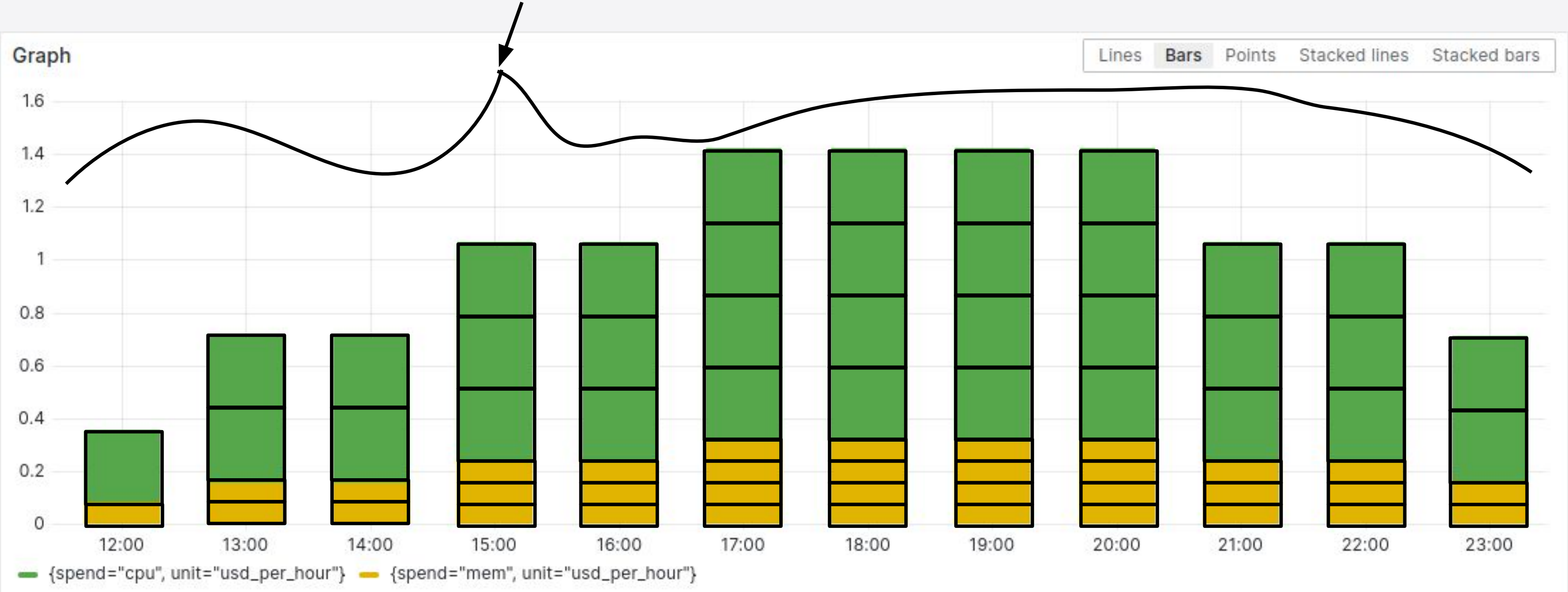
A simple formula with profound implications ...

$$\text{Spend} = \text{Usage} \times \text{Rate}$$



A simple formula with profound implications ...

$$\text{Spend} = \sum \text{Usage} \times \text{Rate}\ \Delta t$$


Graph
Lines
Bars
Points
Stacked lines
Stacked bars
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
0
12:00
13:00
14:00
15:00
16:00
17:00
18:00
19:00
20:00
21:00
22:00
23:00
🟩 {spend="cpu", unit="usd_per_hour"}
🟨 {spend="mem", unit="usd_per_hour"}