Joyent®

Node.js,
in a Galaxy Far,
Far Away
(and 300 million pockets)

Wyatt Preul

jsgeek.com/ni

Joyent®

- Original corporate steward of Node.js
- Running container-native public cloud for a decade
- Acquired by Samsung, which means worlds most popular mobile/IoT apps rely on our cloud

The not so good old days

The software that ran our public cloud services were…

- Ruby on Rails services
- Tightly coupled code
- Difficult to change

Monolith

- Slow release cycles
- Cascading failure due to tight coupling
- Difficult to work with

JavaScript on the Server

- Joyent started to switch to a SpiderMonkey implementation before Node.js existed
- Perfect for async programming where you interact with i/o devices

no.de

- Joyent hired Ryan and Isaac
- Began migrating to Node.js and microservices
- Supported core Node.js improvements

"We believe Node.js is a foundational technology that will enable developers to build the next generation of apps that will finally usher in the Internet of Things era."

"We knew the async model Node enforced was right, and we knew Javascript would work too, as we'd already fallen in love with Javascript-outside-the-browser building the Joyent Smart Platform."

– Mark Mayo, November 2010

Debuggability

- Node.js lacked enterprise tooling to help with observability
- Bryan added DTrace probes to core Node.js
- Post-mortem debugging support added to Node.js for use in mdb
- Modules created to help with observability; vasync, bunyan, restify

Microservices

- Aligns with Unix Philosophy
- Bounded concerns
- Iterate quickly

Node.js Everywhere!

- More than just API services
- Agents running at the heart of Triton
- Node.js can be a solution to most problems if you let it

joyent/binder

- DNS server running on Node.js
- Uses registrar agent on SmartOS, also written in Node.js

joyent/sdc-ufds

- UFDS server for user auth, account management

- Supports LDAP, even created ldapjs for LDAP client/server implementations

joyent/sdc-net-agent

- Compute node NIC agent
- Manage NICs on compute node

joyent/moray

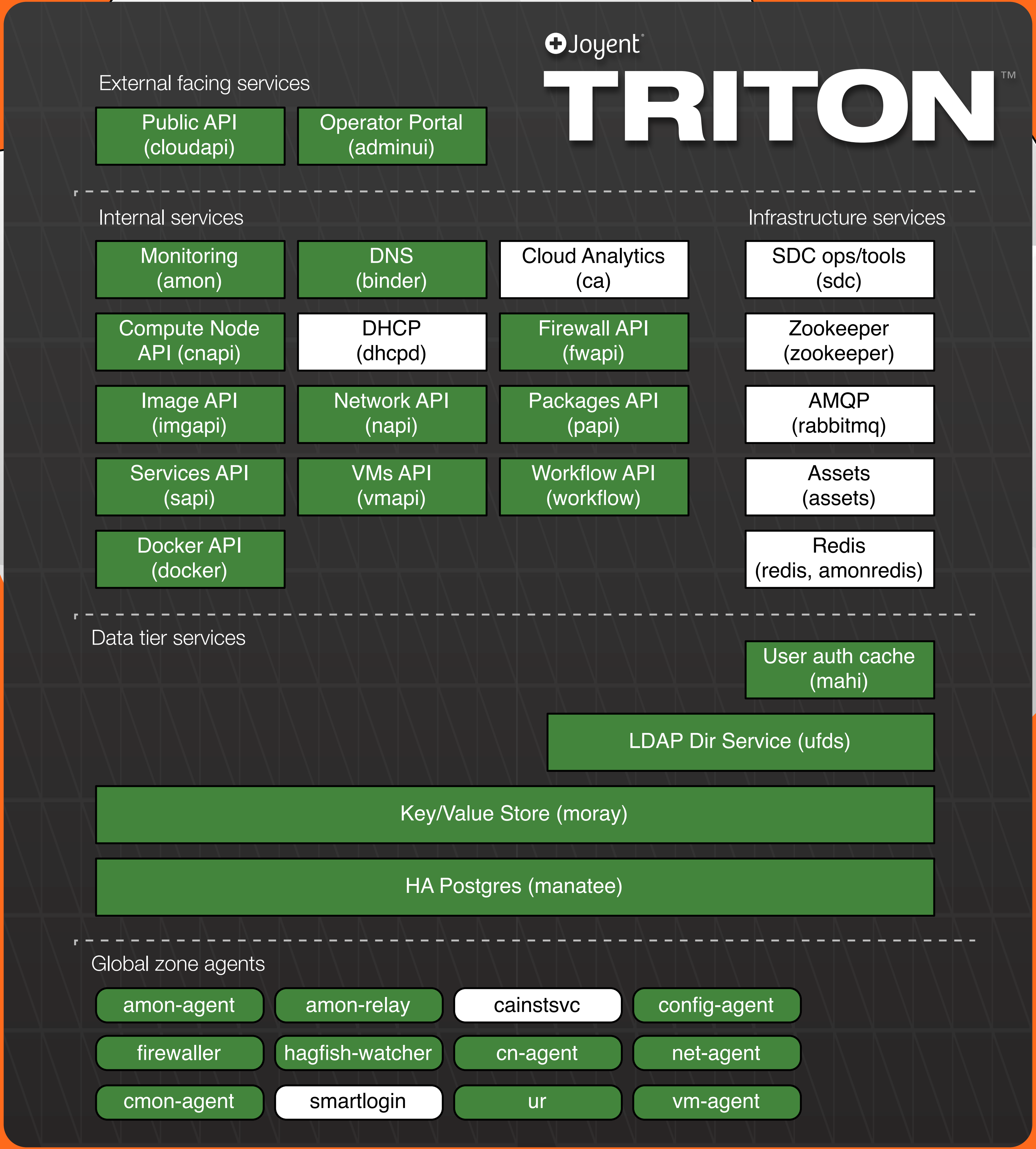
- Key/value store
- Backed by PostgreSQL
- Has a client/server and CLI

CLI with Node.js

- All CLI tools are built on Node.js
- triton
- manta
- Node.js is a prerequisite!

More Node.js Solutions

- Telemetry agent (hagfish-watcher)
- Log and data archival system (hermes)
- Container Name Service (triton-cns)
- DHCP server (sdc-booter)
- NFS server (sdc-nfs)



Joyent

TRITON™

External facing services

- Public API (cloudapi)
- Operator Portal (adminui)

Internal services

- Monitoring (amon)
- DNS (binder)
- Cloud Analytics (ca)
- Compute Node API (cnapi)
- DHCP (dhcpd)
- Firewall API (fwapi)
- Image API (imgapi)
- Network API (napi)
- Packages API (papi)
- Services API (sapi)
- VMs API (vmapi)
- Workflow API (workflow)
- Docker API (docker)

Infrastructure services

- SDC ops/tools (sdc)
- Zookeeper (zookeeper)
- AMQP (rabbitmq)
- Assets (assets)
- Redis (redis, amonredis)

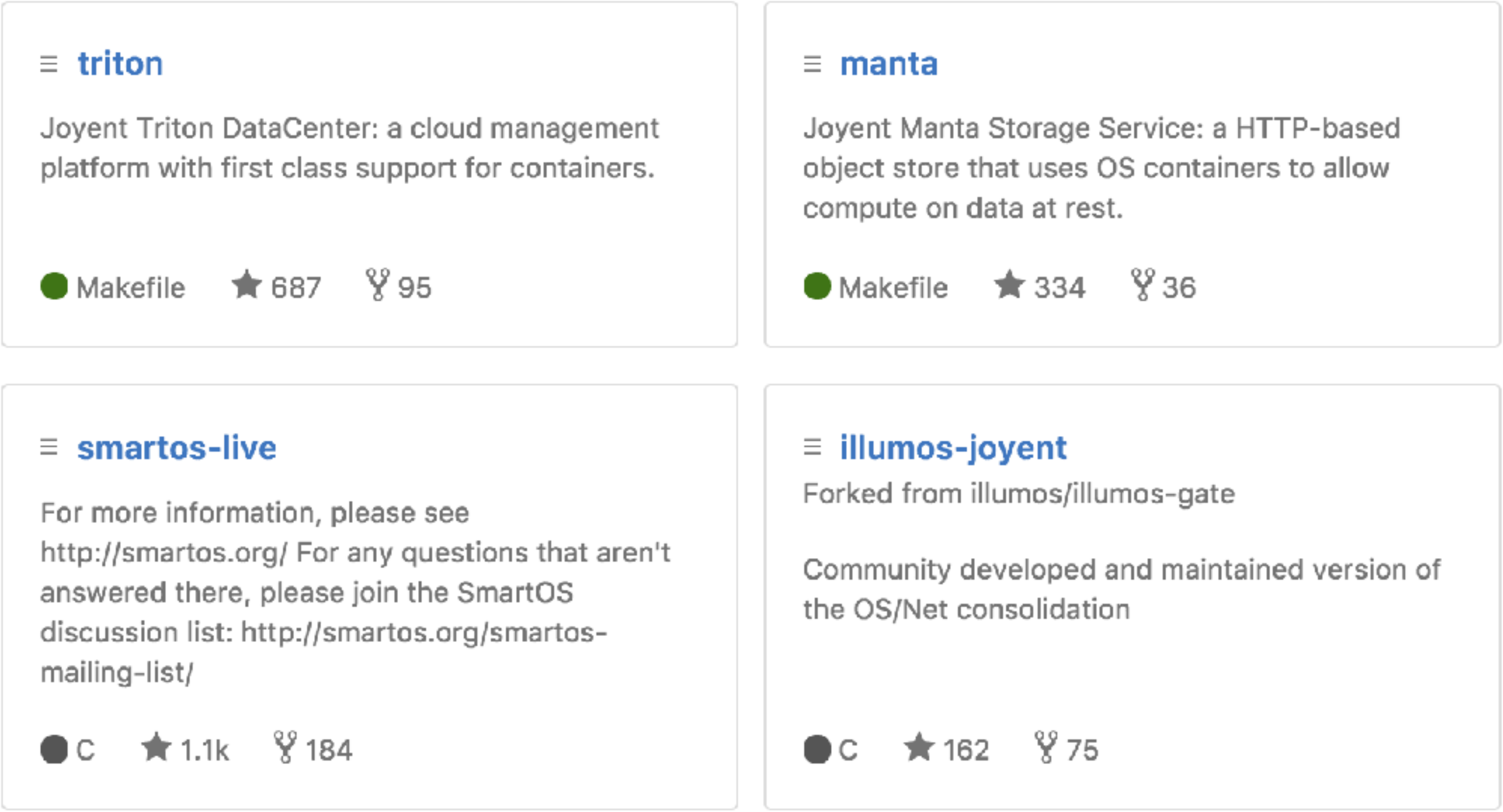
Data tier services

- User auth cache (mahi)
- LDAP Dir Service (ufds)
- Key/Value Store (moray)
- HA Postgres (manatee)

Global zone agents

- amon-agent
- amon-relay
- cainstsvc
- config-agent
- firewaller
- hagfish-watcher
- cn-agent
- net-agent
- cmon-agent
- smartlogin
- ur
- vm-agent

All Open Source



≡ **triton**

Joyent Triton DataCenter: a cloud management platform with first class support for containers.

● Makefile ★ 687 ⑂ 95

≡ **manta**

Joyent Manta Storage Service: a HTTP-based object store that uses OS containers to allow compute on data at rest.

● Makefile ★ 334 ⑂ 36

≡ **smartos-live**

For more information, please see http://smartos.org/ For any questions that aren't answered there, please join the SmartOS discussion list: http://smartos.org/smartos-mailing-list/

● C ★ 1.1k ⑂ 184

≡ **illumos-joyent**

Forked from illumos/illumos-gate

Community developed and maintained version of the OS/Net consolidation

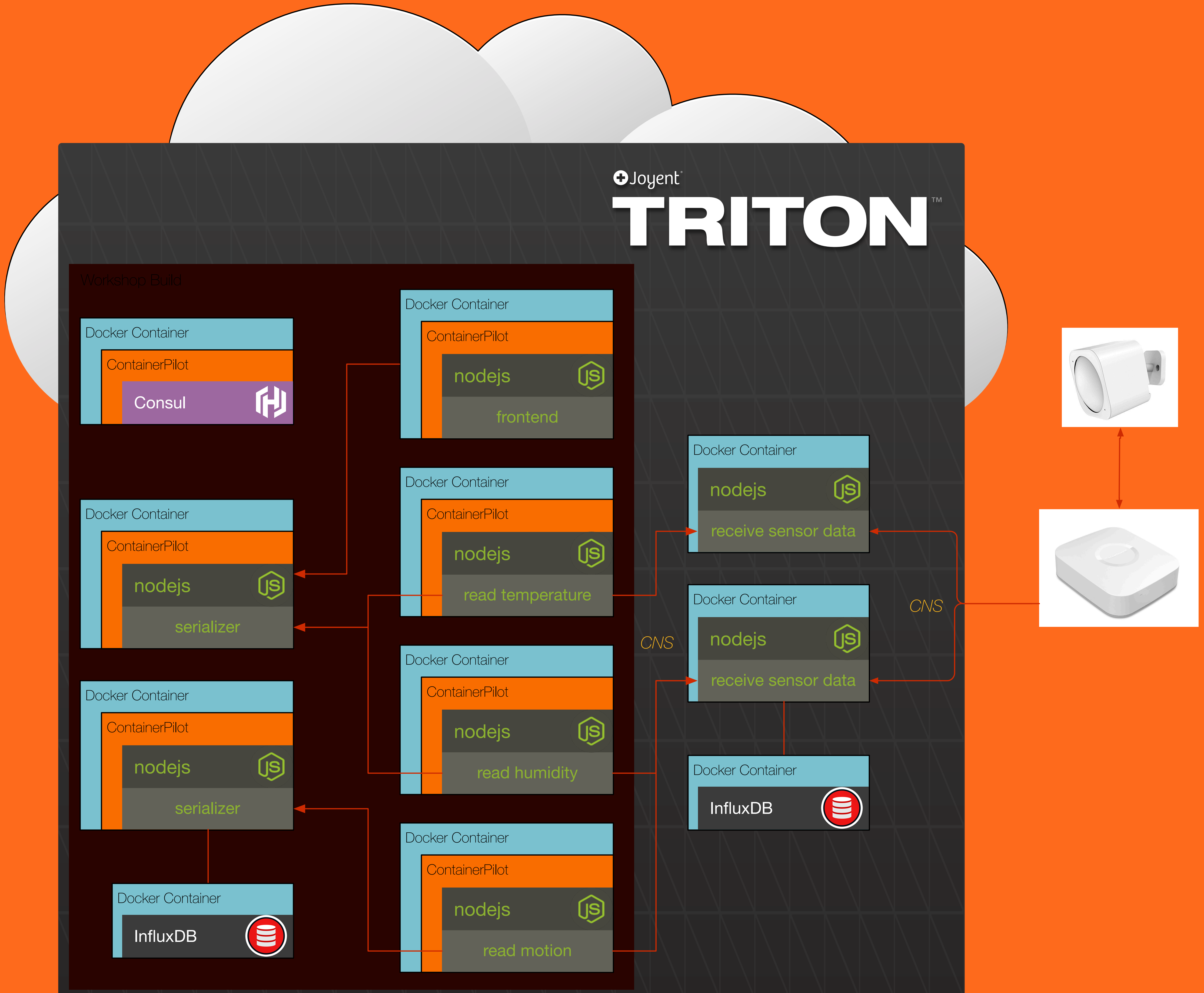
● C ★ 162 ⑂ 75

Acquired Joyent in June

Migration to Joyent

- Samsung has over a billion internet connected devices generating petabytes of data
- Strategic focus is on developing differentiated software and services that run on those devices
- Target architecture is microservices (lots of Node.js) running in containers on Joyent's public cloud

Node.js Microservices on Joyent Public Cloud



Joyent

TRITON™

Workshop Build

Docker Container

ContainerPilot

Consul

Docker Container

ContainerPilot

nodejs

serializer

Docker Container

ContainerPilot

nodejs

serializer

Docker Container

InfluxDB

Docker Container

ContainerPilot

nodejs

frontend

Docker Container

ContainerPilot

nodejs

read temperature

Docker Container

ContainerPilot

nodejs

read humidity

Docker Container

ContainerPilot

nodejs

read motion

CNS

Docker Container

nodejs

receive sensor data

Docker Container

nodejs

receive sensor data

Docker Container

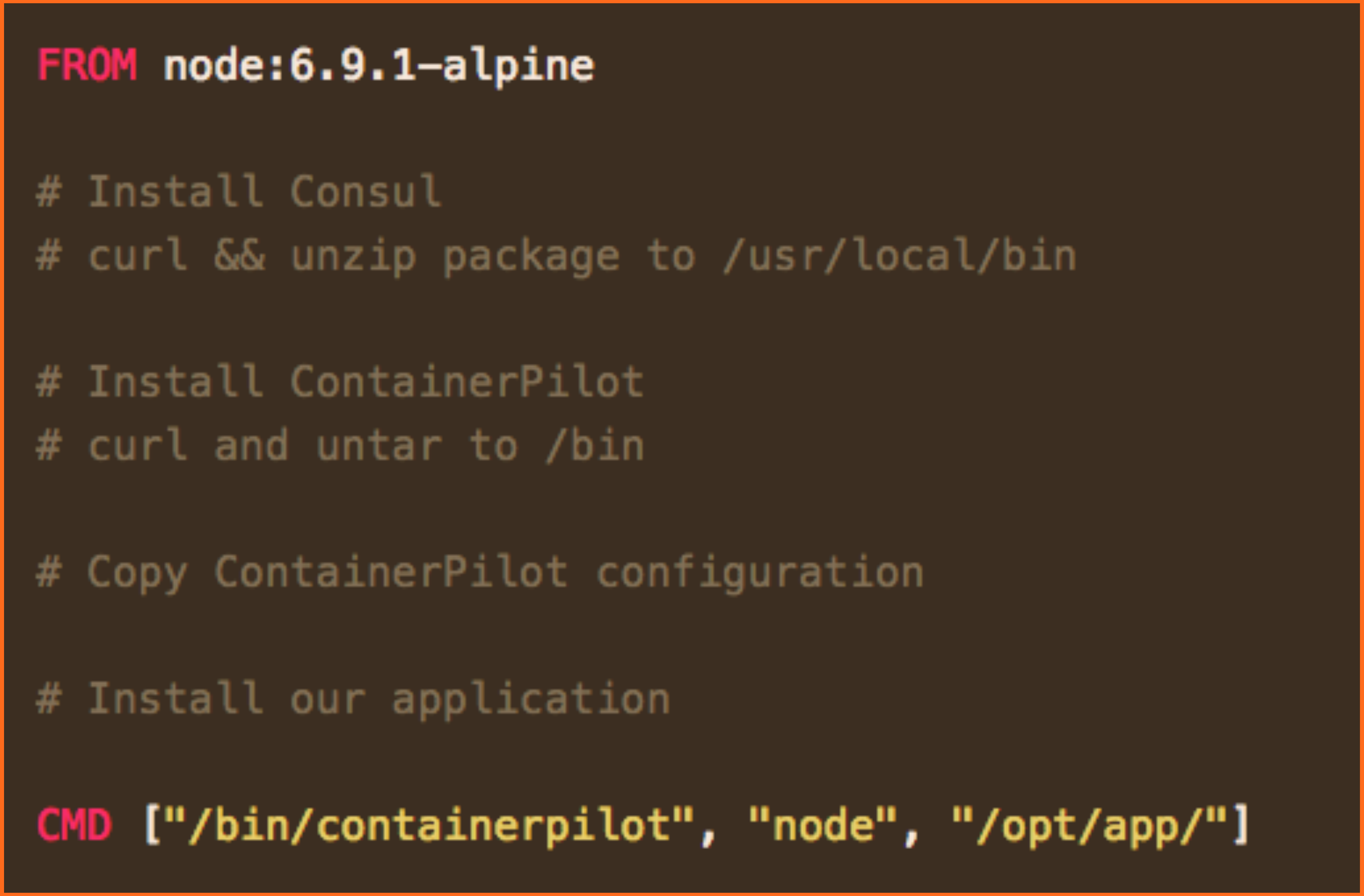
InfluxDB

CNS

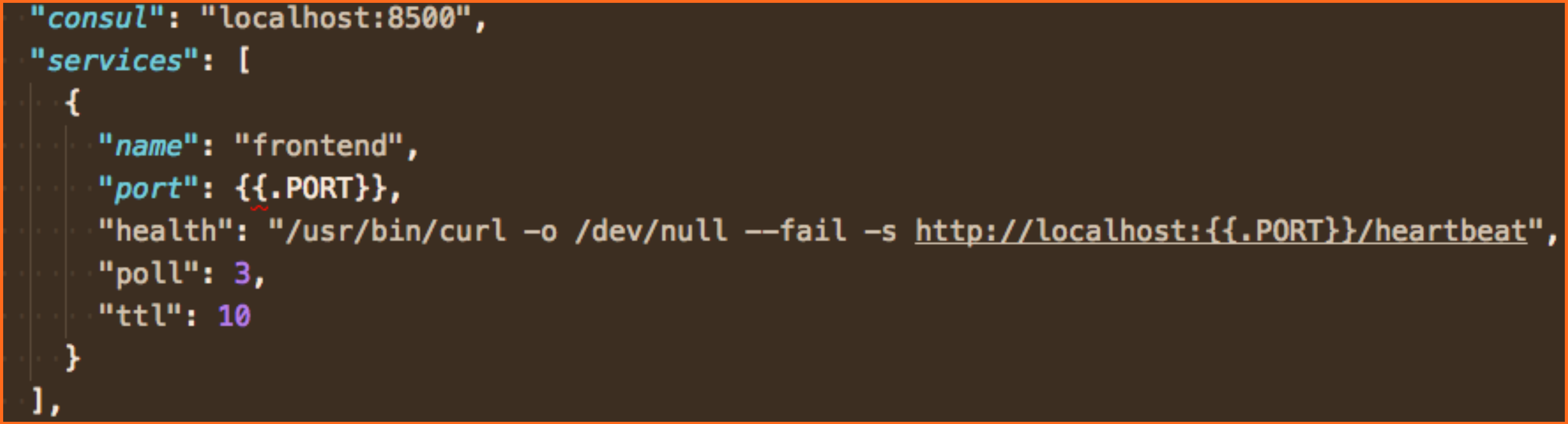
- Self-aware, self-operating containers
- Supports Consul and etcd
- Portable, works anywhere docker does
- Open-source, free: joyent/containerpilot

TRITON

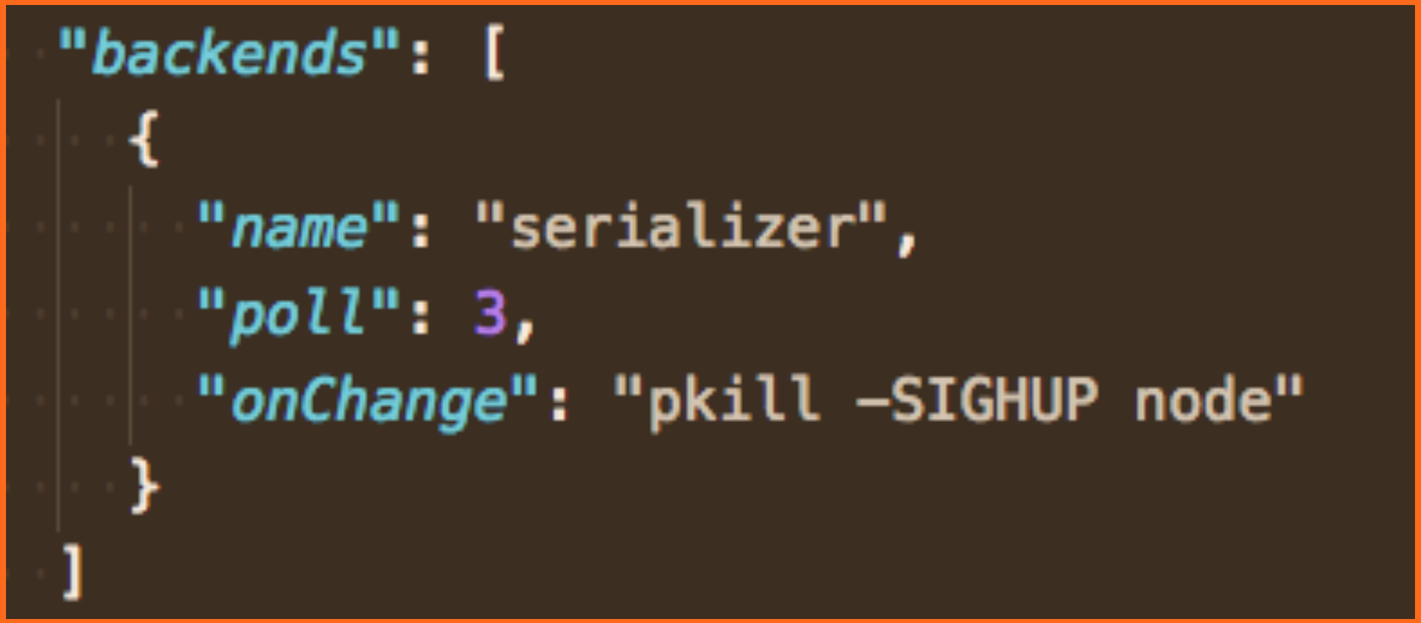
ContainerPilot



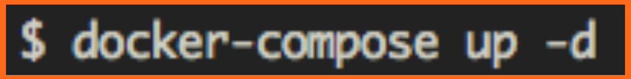
```
FROM node:6.9.1-alpine

# Install Consul
# curl && unzip package to /usr/local/bin

# Install ContainerPilot
# curl and untar to /bin

# Copy ContainerPilot configuration

# Install our application

CMD ["/bin/containerpilot", "node", "/opt/app/"]
```



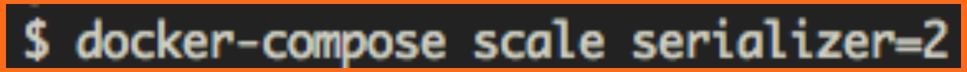
```
  "consul": "localhost:8500",
  "services": [
    {
      "name": "frontend",
      "port": {{.PORT}},
      "health": "/usr/bin/curl -o /dev/null --fail -s http://localhost:{{.PORT}}/heartbeat",
      "poll": 3,
      "ttl": 10
    }
  ],
```



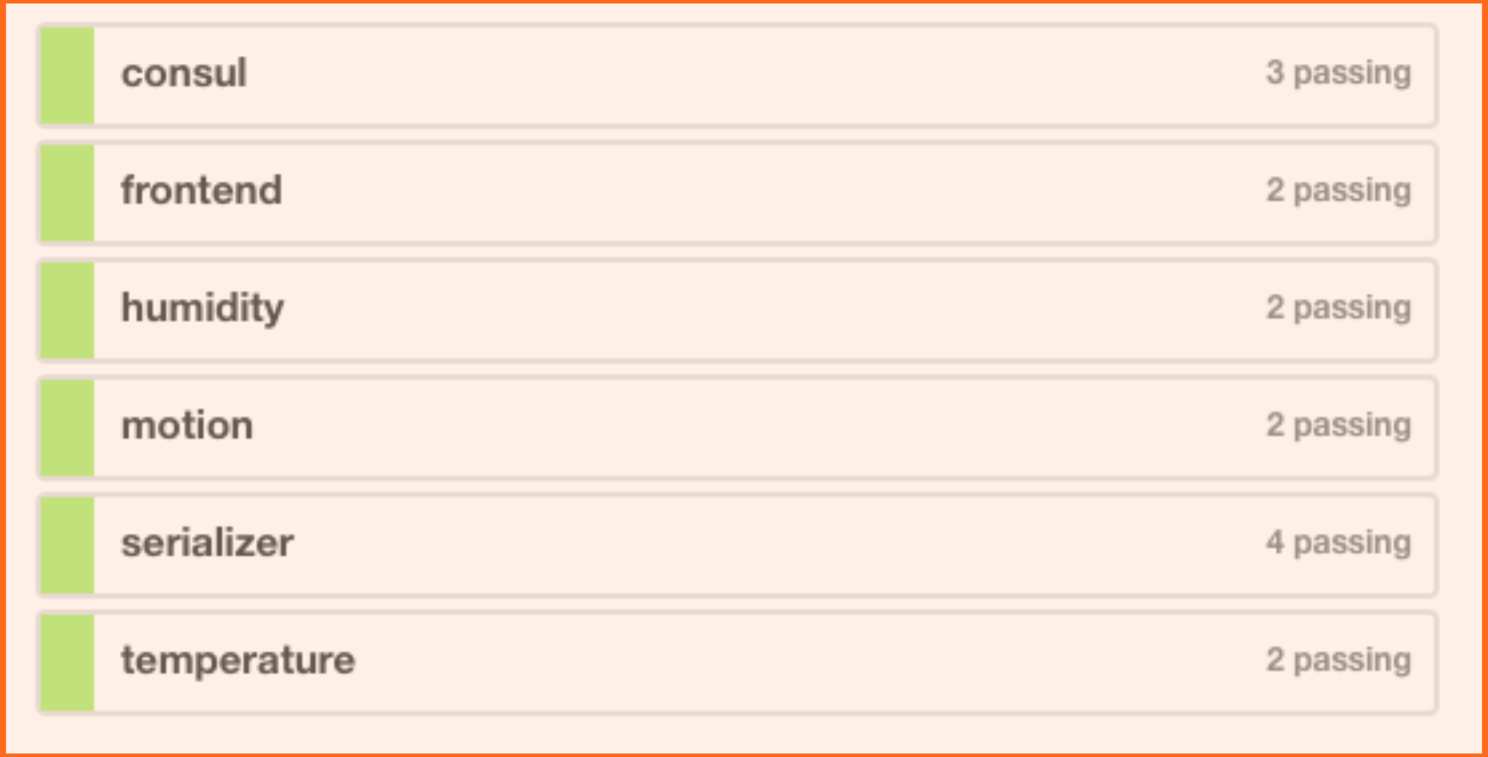
```
"backends": [
  {
    "name": "serializer",
    "poll": 3,
    "onChange": "pkill -SIGHUP node"
  }
]
```



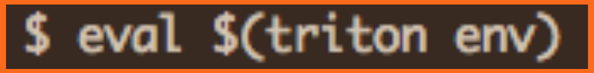
```
$ docker-compose up -d
```



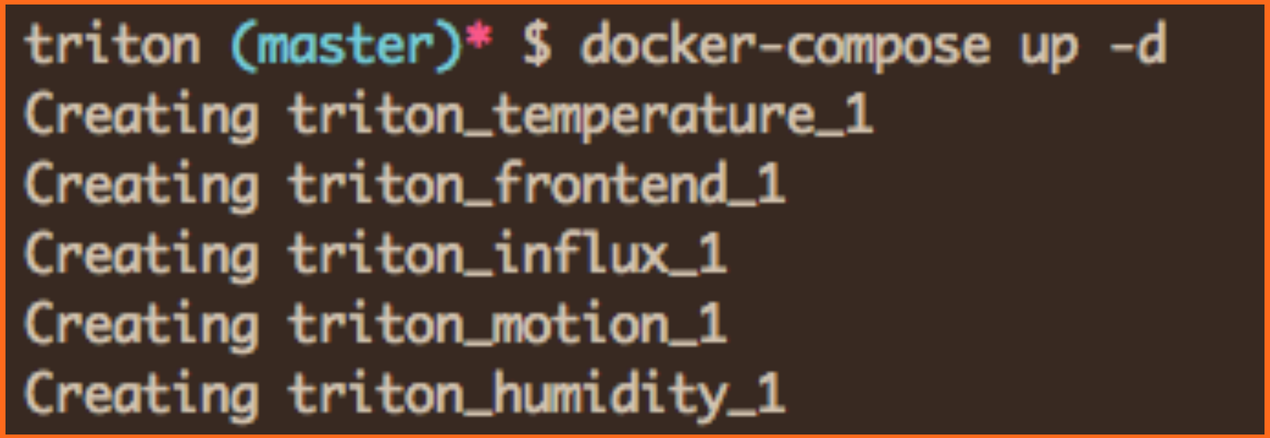
```
$ docker-compose scale serializer=2
```



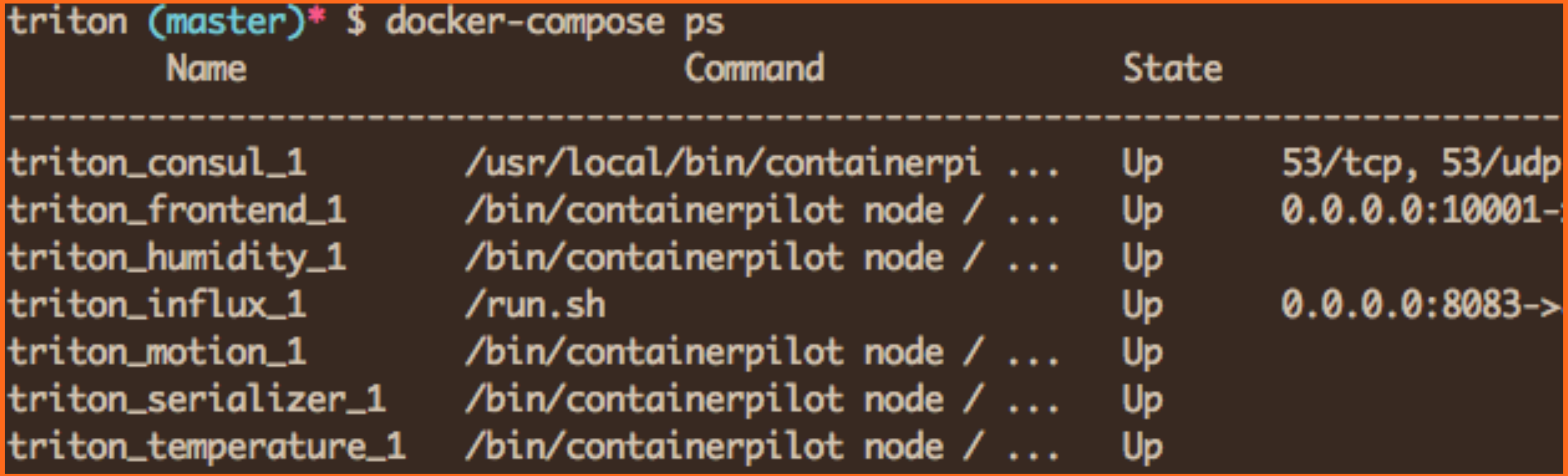
consul	3 passing
frontend	2 passing
humidity	2 passing
motion	2 passing
serializer	4 passing
temperature	2 passing



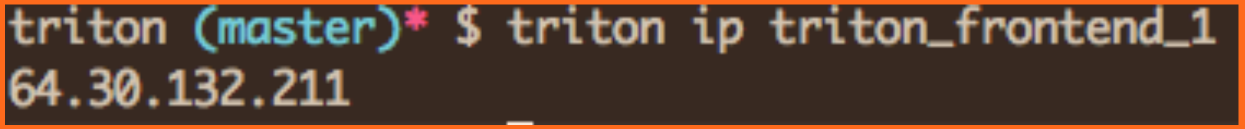
```
$ eval $(triton env)
```



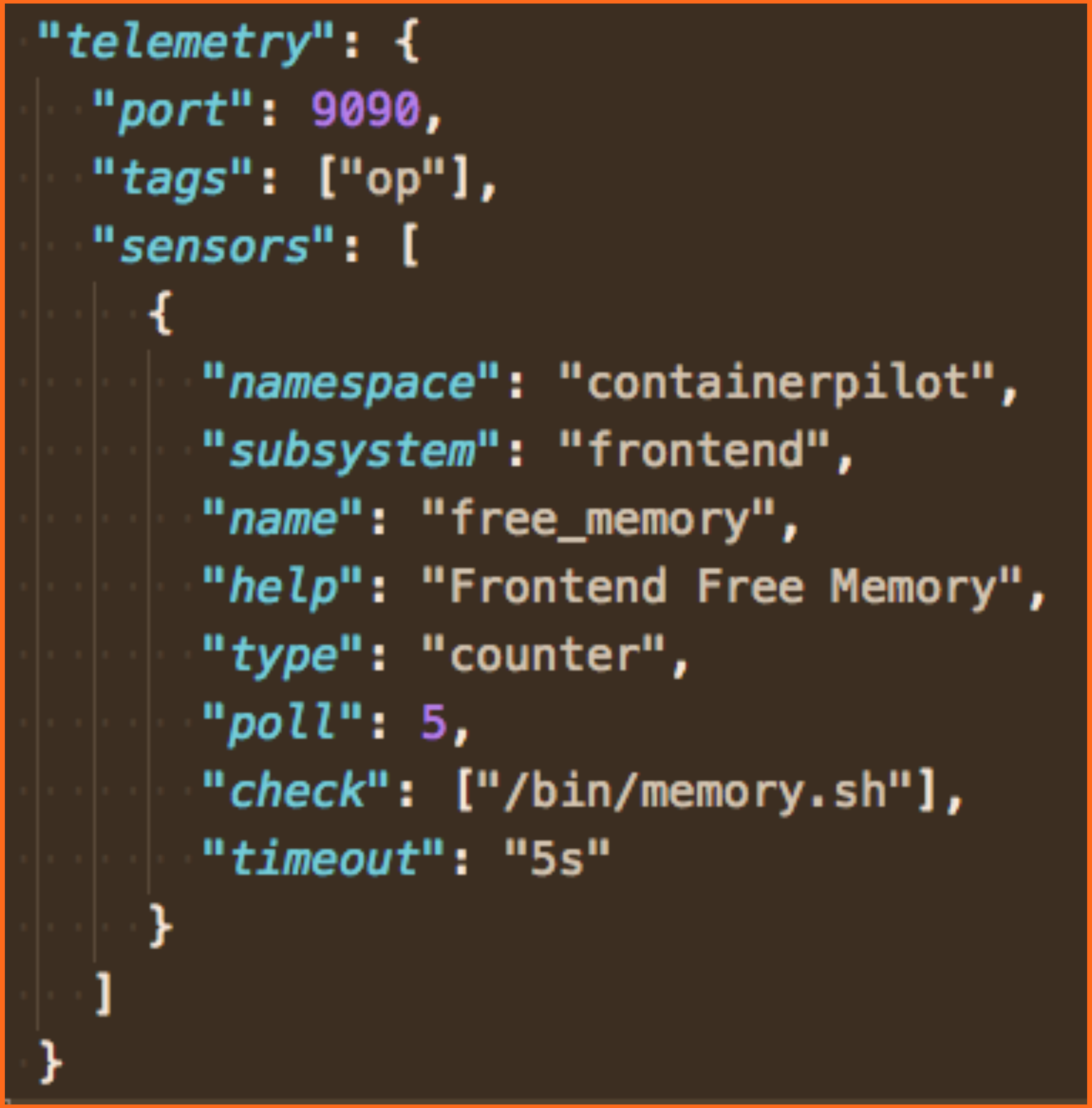
```
triton (master)* $ docker-compose up -d
Creating triton_temperature_1
Creating triton_frontend_1
Creating triton_influx_1
Creating triton_motion_1
Creating triton_humidity_1
```



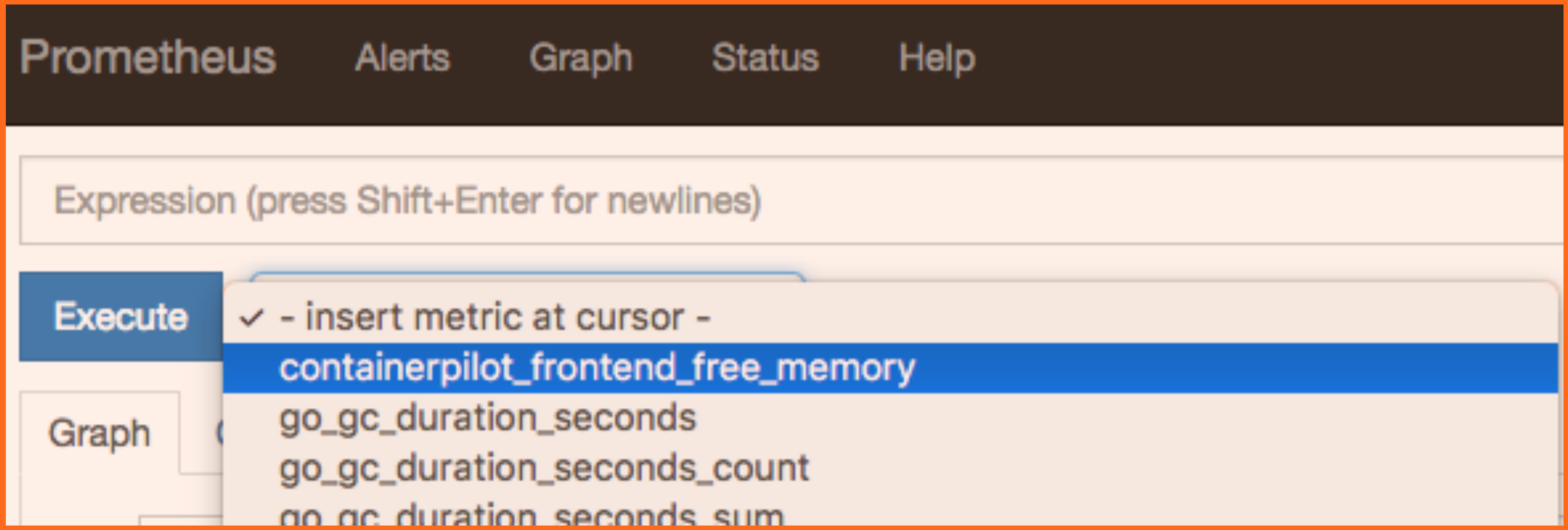
```
triton (master)* $ docker-compose ps
        Name                          Command                   State
-------------------------------------------------------------------------------
triton_consul_1         /usr/local/bin/containerpi ...   Up      53/tcp, 53/udp
triton_frontend_1       /bin/containerpilot node / ...   Up      0.0.0.0:10001-
triton_humidity_1       /bin/containerpilot node / ...   Up
triton_influx_1         /run.sh                          Up      0.0.0.0:8083->
triton_motion_1         /bin/containerpilot node / ...   Up
triton_serializer_1     /bin/containerpilot node / ...   Up
triton_temperature_1    /bin/containerpilot node / ...   Up
```

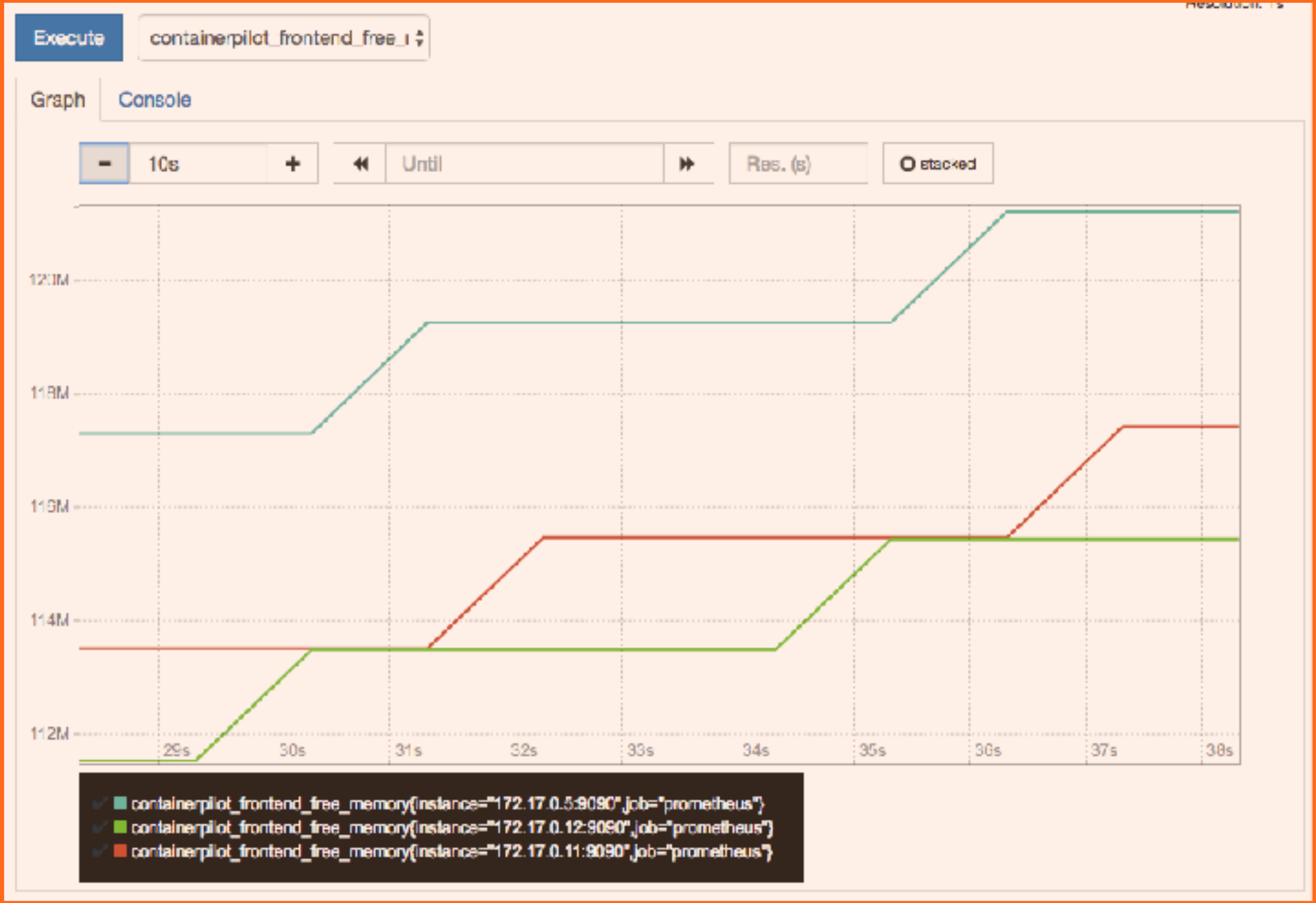


```
triton (master)* $ triton ip triton_frontend_1
64.30.132.211
```

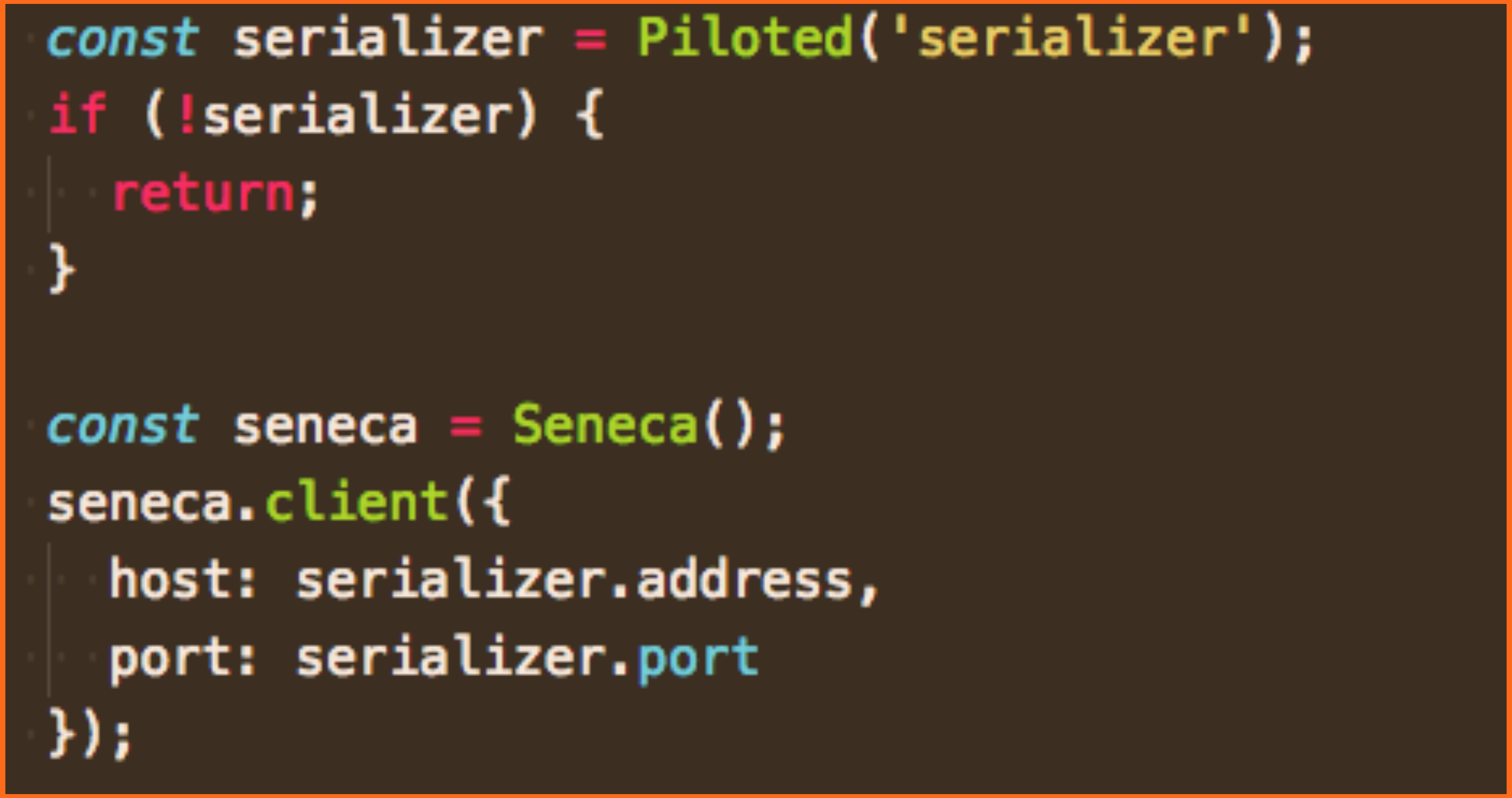


```
"telemetry": {
  "port": 9090,
  "tags": ["op"],
  "sensors": [
    {
      "namespace": "containerpilot",
      "subsystem": "frontend",
      "name": "free_memory",
      "help": "Frontend Free Memory",
      "type": "counter",
      "poll": 5,
      "check": ["/bin/memory.sh"],
      "timeout": "5s"
    }
  ]
}
```


Prometheus
Alerts
Graph
Status
Help
Expression (press Shift+Enter for newlines)
Execute
✓ - insert metric at cursor -
containerpilot_frontend_free_memory
go_gc_duration_seconds
go_gc_duration_seconds_count
go_gc_duration_seconds_sum
Graph


Resolution: 1s
Execute
containerpilot_frontend_free_r
Graph
Console
-
10s
+
Until
Res. (s)
○ stacked
120M
118M
116M
114M
112M
29s
30s
31s
32s
33s
34s
35s
36s
37s
38s
✓ 🟩 containerpilot_frontend_free_memory{instance="172.17.0.5:9090",job="prometheus"}
✓ 🟩 containerpilot_frontend_free_memory{instance="172.17.0.12:9090",job="prometheus"}
✓ 🟥 containerpilot_frontend_free_memory{instance="172.17.0.11:9090",job="prometheus"}

Node.js Modules



```
const serializer = Piloted('serializer');
if (!serializer) {
  return;
}

const seneca = Seneca();
seneca.client({
  host: serializer.address,
  port: serializer.port
});
```

ContainerPilot Enables

- Developers to move quickly
- Microservices are more resilient
- Observable, less complex, easier to diagnose issues (no load balancer)

Future Work

- Hiring Node.js core Engineer (joyent.com/careers)
- OpenTracing added to Triton (RFD 35)
- Container scheduler service on Triton (RFD 36)
- Much more, visit joyent/rfd for everything!

Joyent®

Thanks Y’all

Slides & Links: jsgeek.com/ni