CoreOS

About Me

CTO/CO-FOUNDER

systems engineer

@brandonphilips

github.com/philips

What is CoreOS?

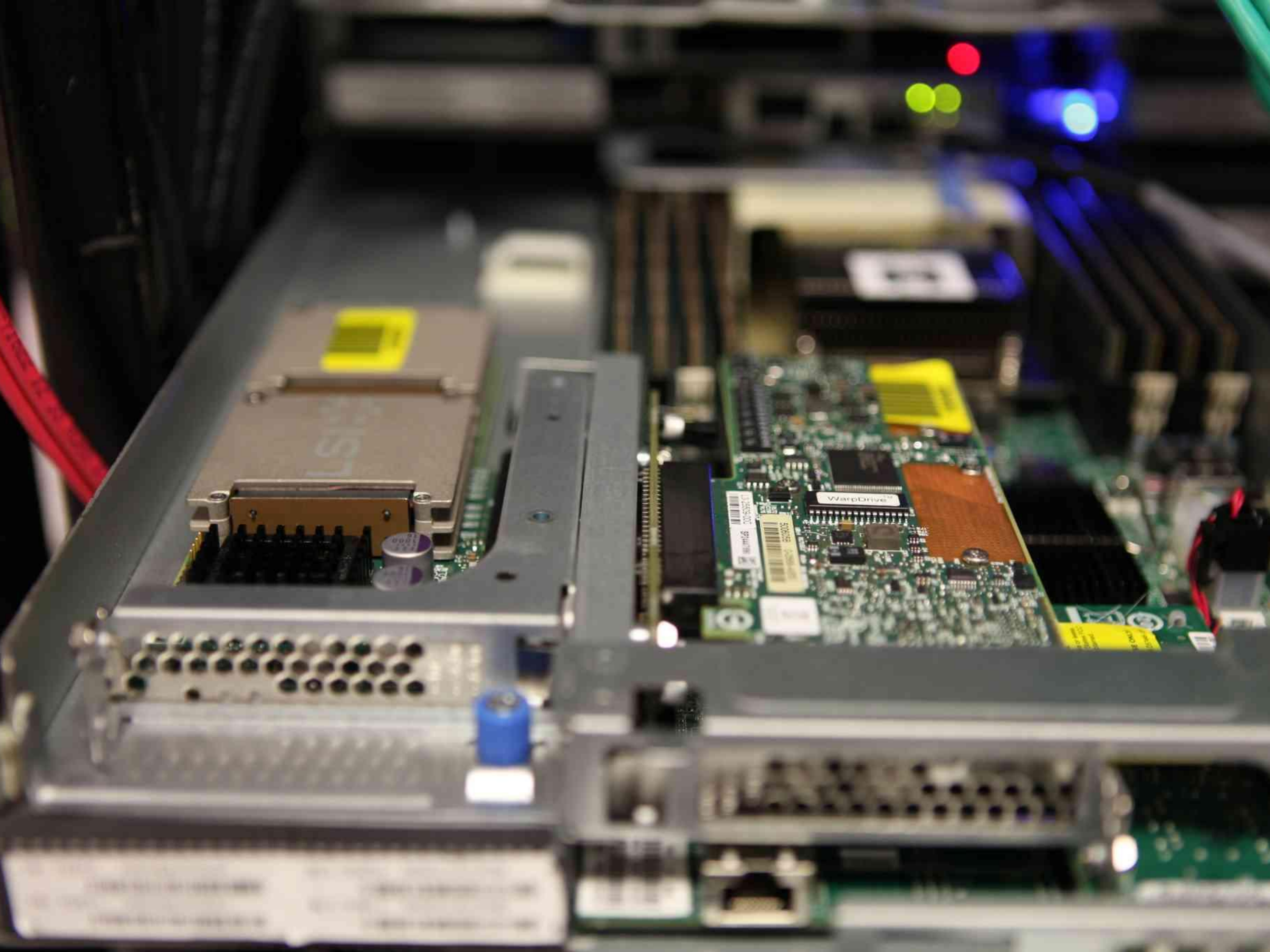
What is CoreOS?


amazon
web services™
EC2

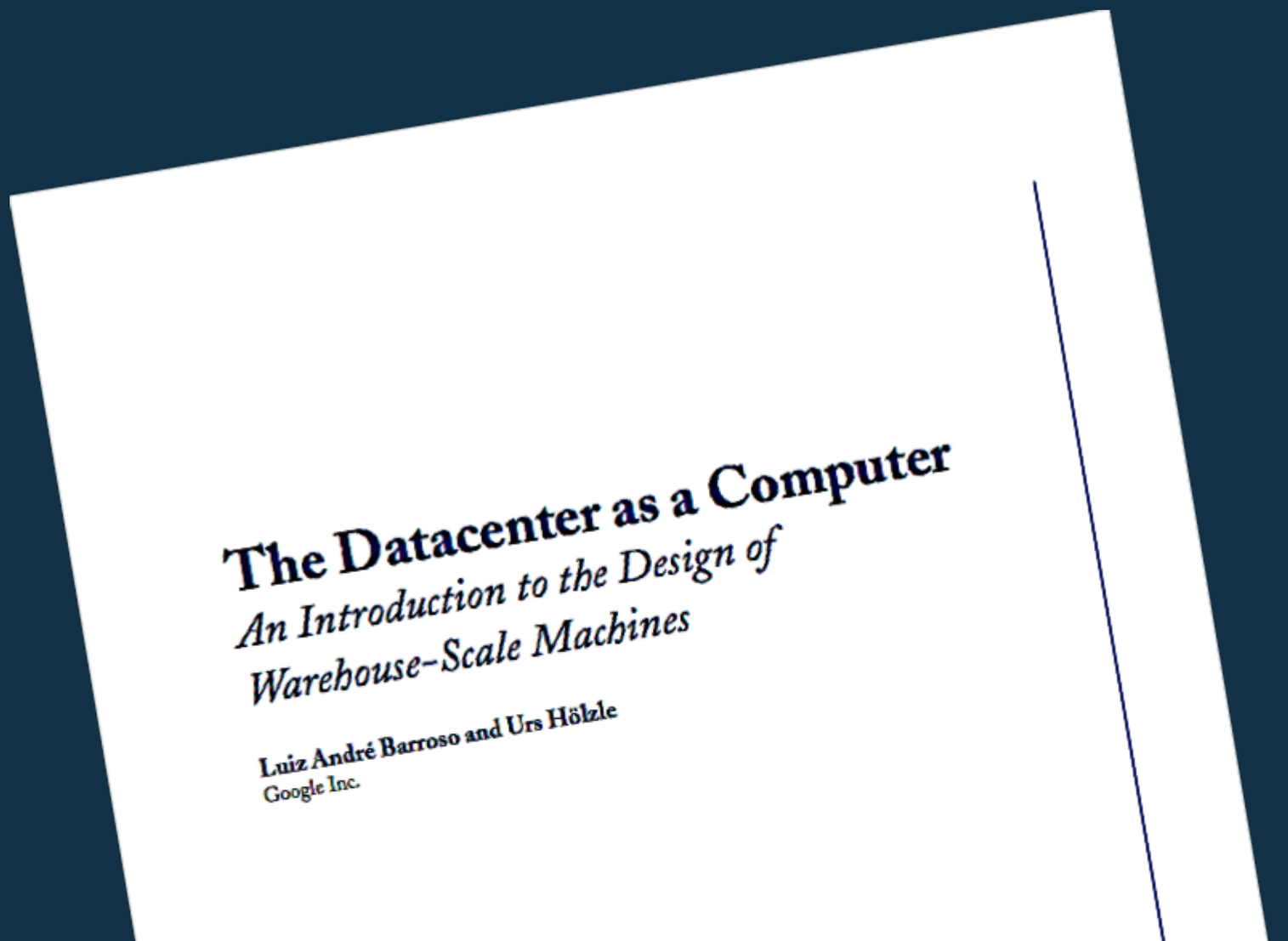

Google Cloud Platform


openstack™


RACKSPACE®
CLOUD SERVERS™


WarpDrive™

Why build CoreOS?



The Datacenter as a Computer

An Introduction to the Design of Warehouse-Scale Machines

Luiz André Barroso and Urs Hölzle
Google Inc.

minimal

reduce API contracts

kernel
systemd
etcd
ssh
docker

python
java
nginx
mysql
openssl

distro distro distro distro distro distro distro

app

kernel
systemd
etcd
ssh
docker

distro distro distro distro distro distro distro

python
java
nginx
mysql
openssl

app

kernel
systemd
etcd
ssh
docker

o distro distro distro distro distro distro distro distro distr

python
openssl-A
app1

java
openssl-B
app2

java
openssl-B
app3

CoreOS

o distro distro distro distro distro distro distro distro

python
openssl-A
app1

java
openssl-B
app2

java
openssl-B
app3

CoreOS

o distro distro distro distro distro distro distro distro

container

container

container

manual

updates

automatic

updates

automatic updates

auto updates

atomic with rollback

Data

Update

A

B

Data

A

B

0.0% Downloading

0.0% Downloaded

0.0% Installed

0.0% Instance Hold

100% Complete

CoreOS Applications Packages My Account

+ Create a New Group

CoreOS

Production

f698ffbe-7488-43cf-97a1-66fcf179609f

CHANNEL
Stable ›

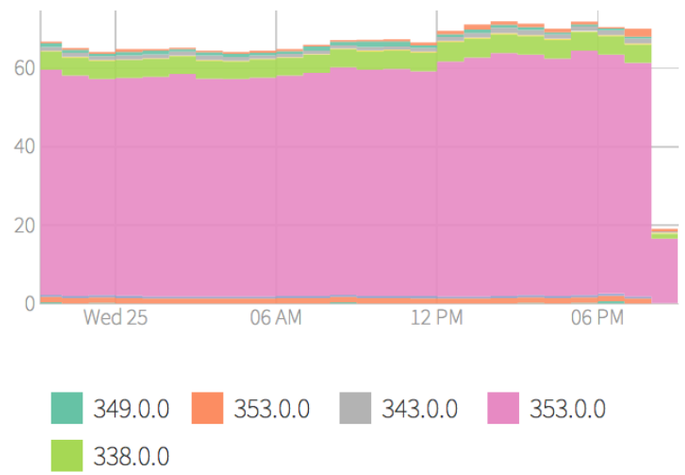
INSTANCES
69 instances

AUTOMATIC UPDATES
Enabled ›

UPDATE BEHAVIOR
10 updates allowed per 15 minutes ›

Version Breakdown

View All Graphs



Channels

View Packages

Alpha	361.0.0
Beta	357.0.0
Stable	353.0.0

+ Add New Channel

EC2 Autoscale

containers

run and isolate apps

pid ns

isolated pid 1

user ns

isolated uid 0

network ns

isolated netdev

mount ns

isolated /

cgroups

manage resources

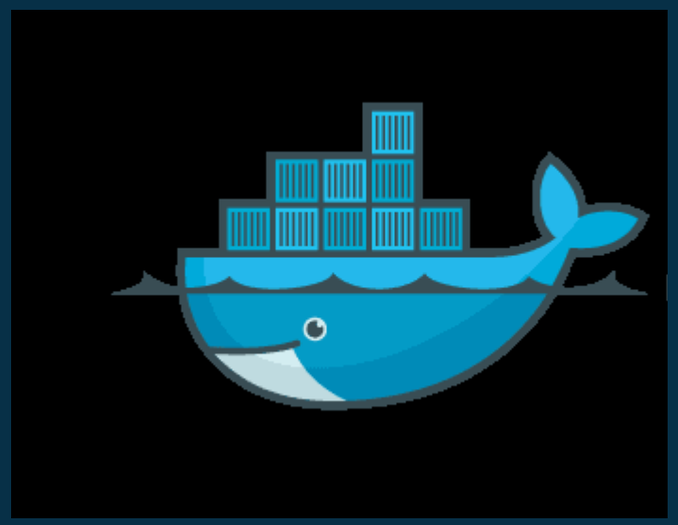
cgroups

count resources

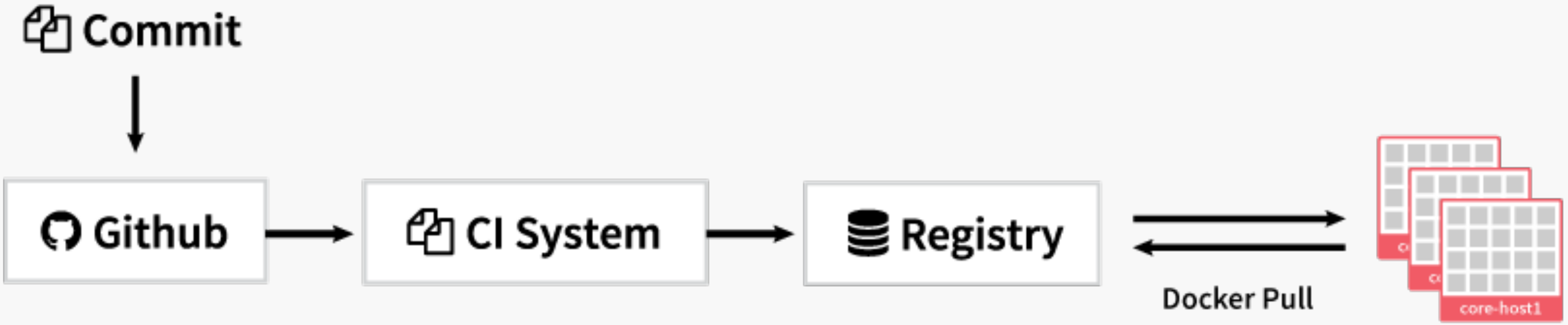
cAdvisor

cgroups

limit resources



docker engine


Commit
Github
CI System
Registry
Docker Pull
core-host1

configuration

machines and clusters

machine conf

get into the cluster

```
#cloud-config

users:
  - name: core
    coreos-ssh-import-github: philips

coreos:
  etcd:
   discovery: https://discovery.etcd.io/ba09c
units:
  - name: etcd.service
    command: start
  - name: fleet.service
    command: start
```

```
#cloud-config

users:
  - name: core
    coreos-ssh-import-github: philips

coreos:
  etcd:
   discovery: https://discovery.etcd.io/ba09c
units:
  - name: etcd.service
    command: start
  - name: fleet.service
    command: start
```

```
#cloud-config

users:
  - name: core
    coreos-ssh-import-github: philips

coreos:
  etcd:
   discovery: https://discovery.etcd.io/ba09c
units:
  - name: etcd.service
    command: start
  - name: fleet.service
    command: start
```

```
#cloud-config

users:
  - name: core
    coreos-ssh-import-github: philips

coreos:
  etcd:
   discovery: https://discovery.etcd.io/ba09c
units:
  - name: etcd.service
    command: start
  - name: fleet.service
    command: start
```

cluster conf

what is running

services

skydns, discoverd, confd

cluster conf

what should run

coordination

locksmith

scheduling

fleet, kubernetes

etcd

/etc

distributed

open source software
highly available and reliable
sequentially consistent
watchable
exposed via HTTP
runtime reconfigurable

-X GET

Get Wait

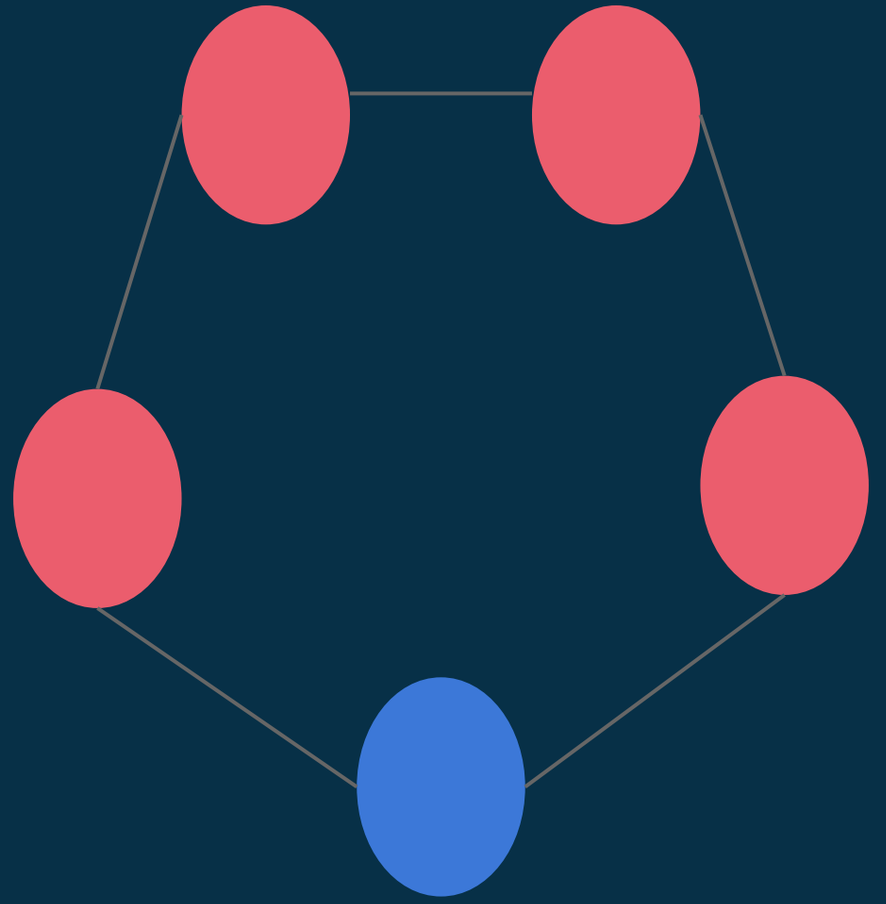
-X PUT

Put Create CAS

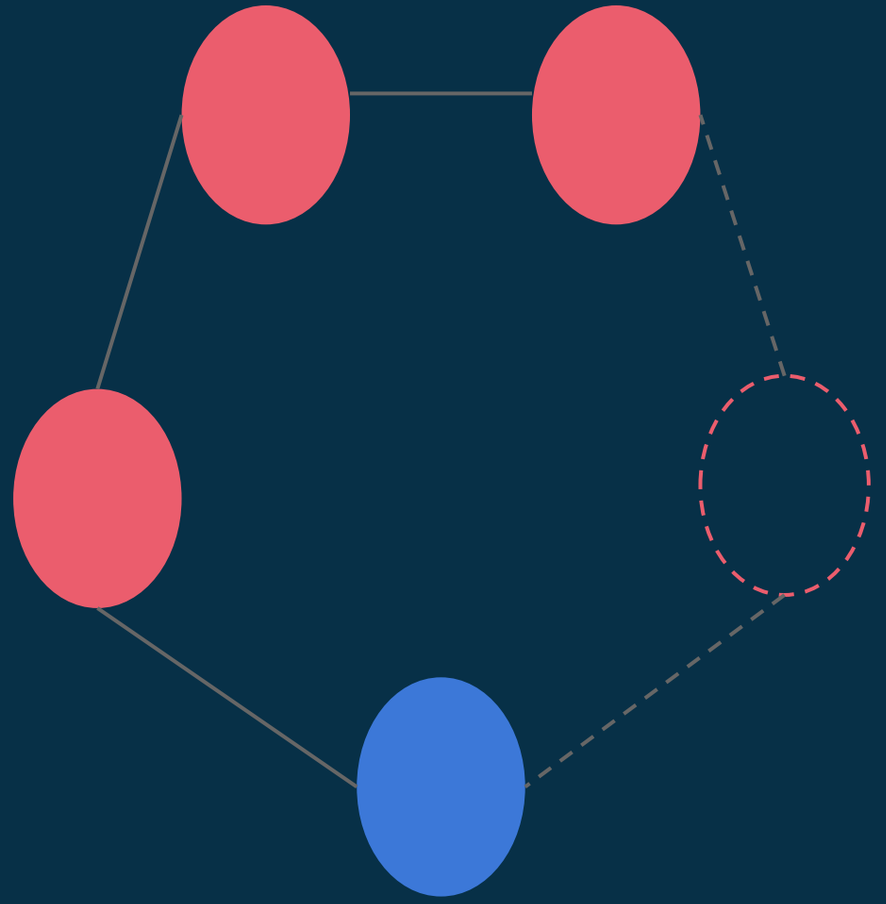
-X DELETE

Delete CAD

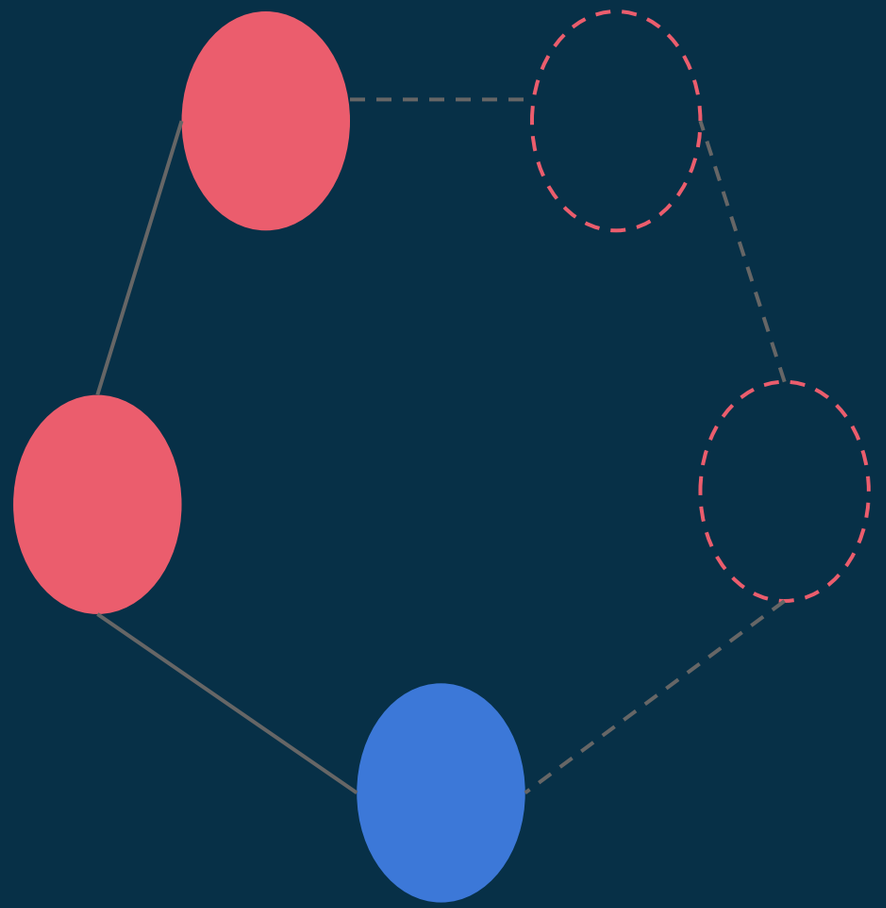
Available



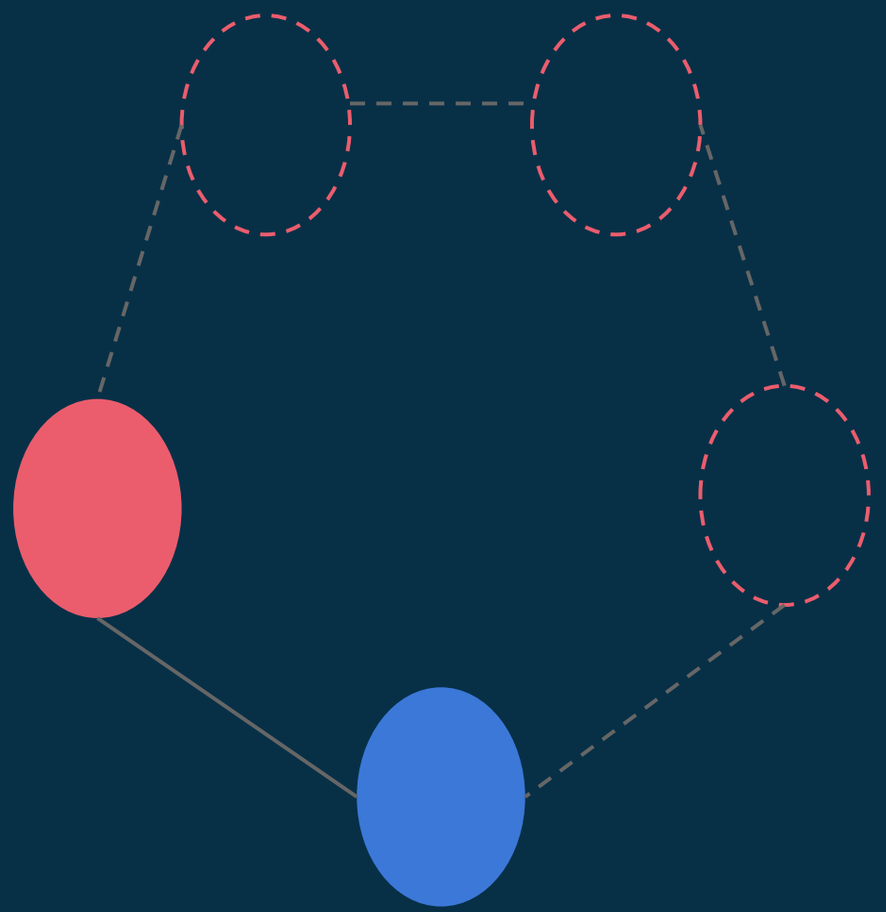
Available



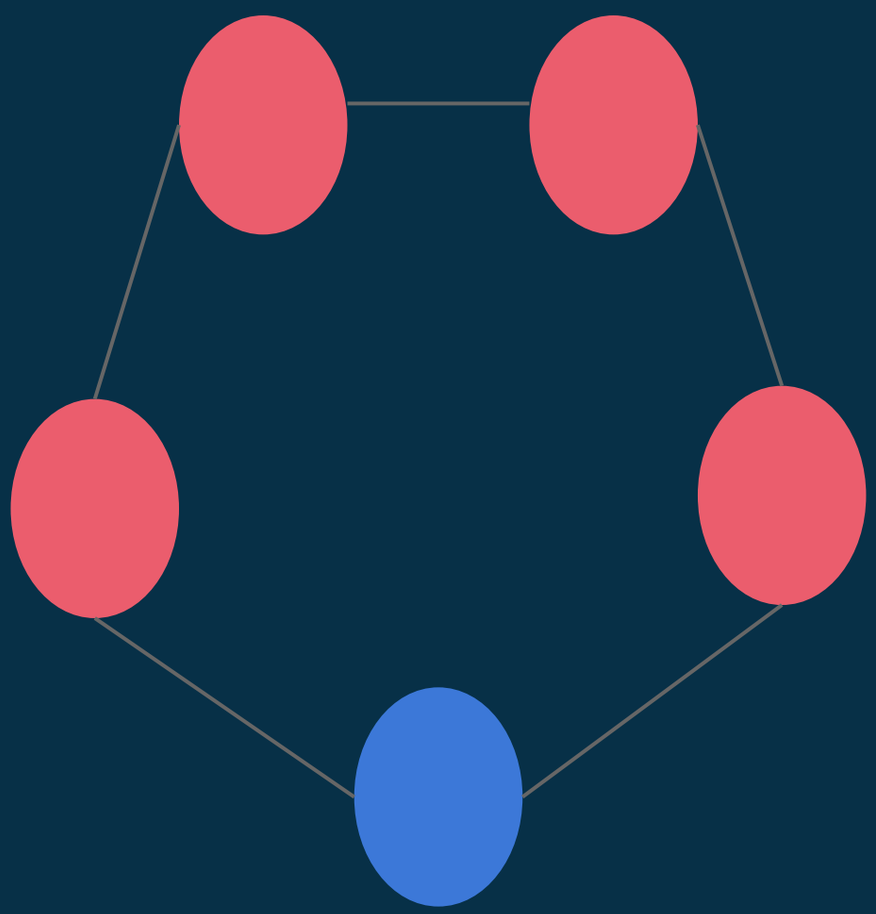
Available



Unavailable



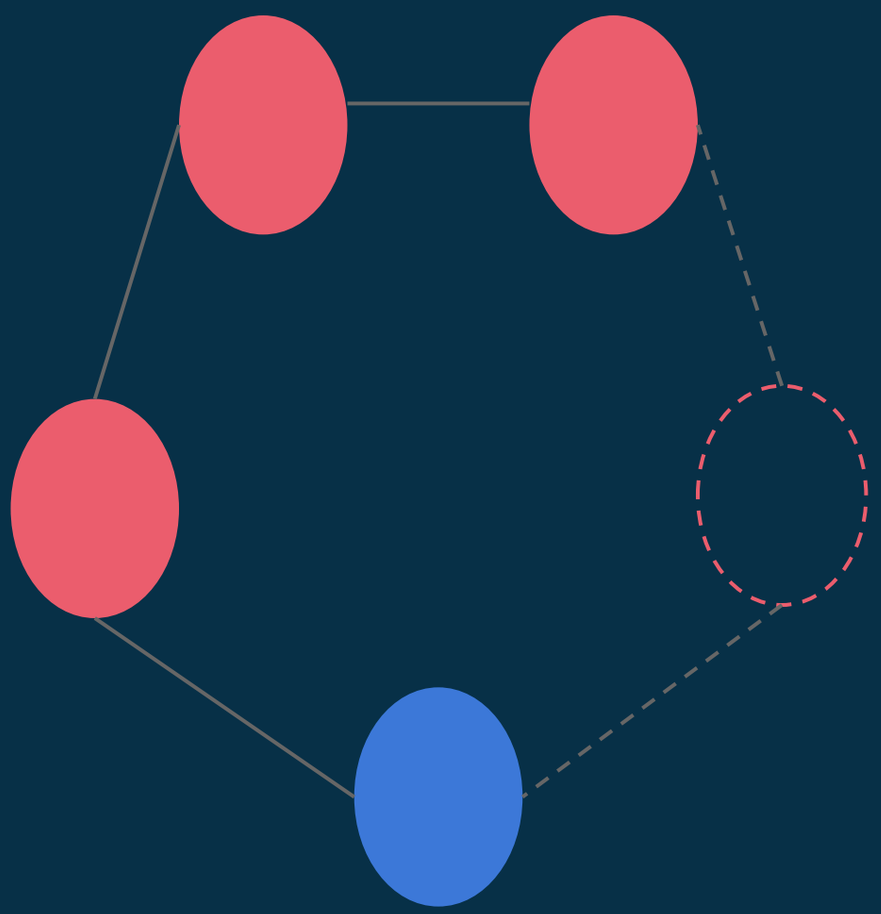
Available



Leader

Follower

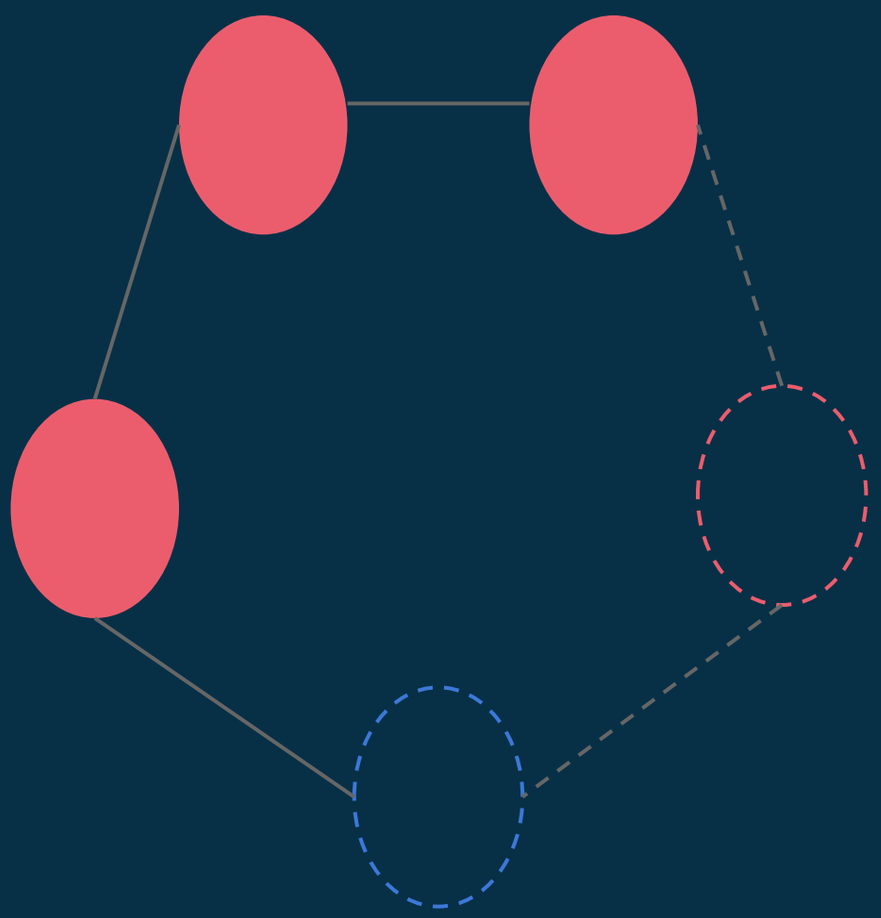
Available



Leader

Follower

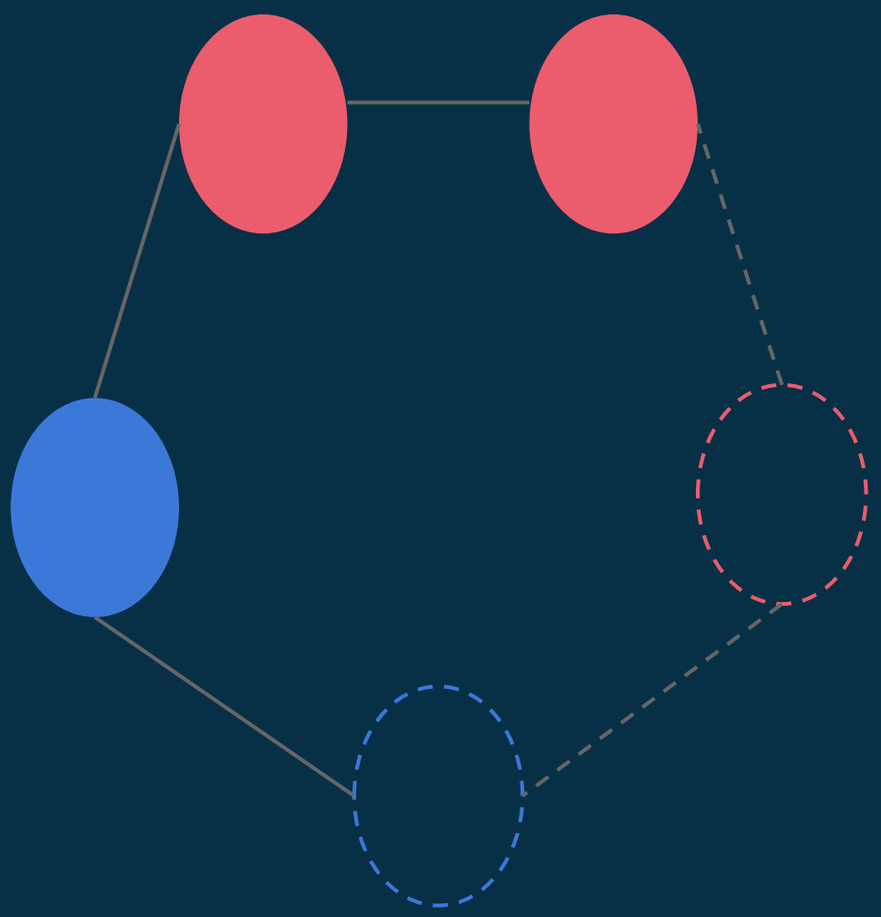
Temporarily Unavailable



Leader

Follower

Available



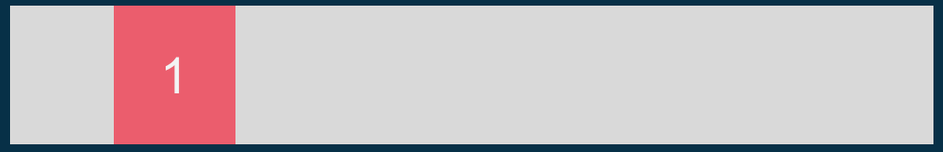
Leader

Follower

Sequential Consistency

Index Time

GET A @ index 2 -> 2



GET A @ index 2 -> blocking

GET A @ index 2 -> 2

etcd guarantees that a get at index X will always return the same result.

Avoid thinking in terms of real time because with network latency the result is always out-of-date.

Watchable Changes

HTTP Long-poll


```
> GET asdf?waitIndex=4&wait=true HTTP/1.1
> Accept: */*
>
< HTTP/1.1 200 OK
< Content-Type: application/json
< X-Etcd-Index: 3
< X-Raft-Index: 97
< X-Raft-Term: 0
<
BLOCK
```


```
> GET asdf?waitIndex=4&wait=true HTTP/1.1
> Accept: */*
>
< HTTP/1.1 200 OK
< Content-Type: application/json
< X-Etcd-Index: 3
< X-Raft-Index: 97
< X-Raft-Term: 0
<
{"action":"set","node":{"key":"/asdf","value":"foobar","
modifiedIndex":4,"createdIndex":4}}
```


```
> GET asdf?waitIndex=4&wait=true HTTP/1.1
> Accept: */*
>
< HTTP/1.1 200 OK
< Content-Type: application/json
< X-Etcd-Index: 4
< X-Raft-Index: 516
< X-Raft-Term: 0
<
{"action":"set","node":{"key":"/asdf","value":"foobar","
modifiedIndex":4,"createdIndex":4}}
```

scheduling

getting work to servers

You

Scheduler API

Scheduler

Machine(s)

```
$ cat foo.service
[Service]
ExecStart=/usr/bin/sleep 500
```

```
$ fleetctl start foo.service
Job foo.service launched on
e1cd2bcd.../172.17.8.101
```

```
while true {
  todo = diff(desState, curState)
  schedule(todo)
}
```

```
while true {
  todo = diff(desState, curState)
  schedule(todo)
}
```

```
while true {
  todo = diff(desState, curState)
  schedule(todo)
}
```

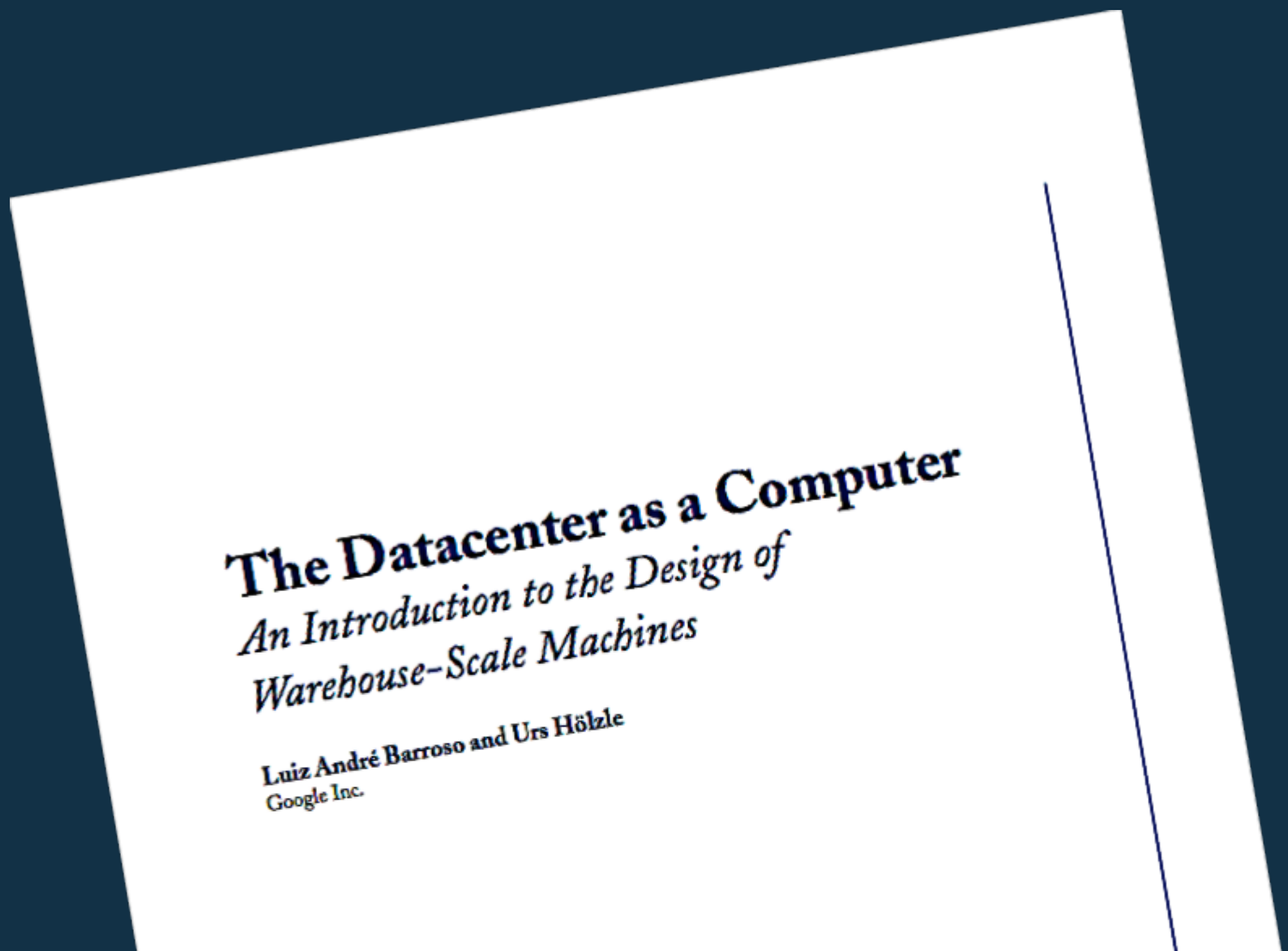
```
while true {
  todo = diff(desState, curState)
  schedule(todo)
}
```

Talk tomorrow!

fleet: systemd for clusters

2:30pm Mississippi

Why use CoreOS?



The Datacenter as a Computer

An Introduction to the Design of Warehouse-Scale Machines

Luiz André Barroso and Urs Hölzle
Google Inc.

is it ready?

stable released

the future

lots of work todo

CoreOS

DEMO?