

# **Konténerek orkesztrálása**

Simon Csaba

# MOTIVÁCIÓ

# Motiváció - orkesztráció

- » Mi hiányzik egy teljes Docker rendszerhez?
  - » Orkesztráció
  - » Amit a felhők nyújtanak
  - » Cél: automatizált konténer telepítés és menedzsment multi-host környezetben (incl. skálázódás vezérlése)
- » Egyik megoldás: nyilvános felhőkben Docker
  - » Amazon Web Services, Google Cloud, Microsoft Azure
- » Másik megoldás: Docker + OpenStack
  - » OpenStack Magnum
- » Harmadik megoldás: Docker orkesztráció
  - » Apache Mesos (2010)
  - » Google Kubernetes (2014)
  - » Docker Swarm Mode (2016)

# Cloud Native Computing Foundation

- » Mikroszervíz ökoszisztéma
  - » Konténerek orkesztrálásával
  - » Google támogatja
  - » rkt-t javasolják a Docker helyett



About

Projects

Certification



gRPC



Sustaining and Inte  
Open Source Techn

---

# KUBERNETES

# Kubernetes - kezdetek

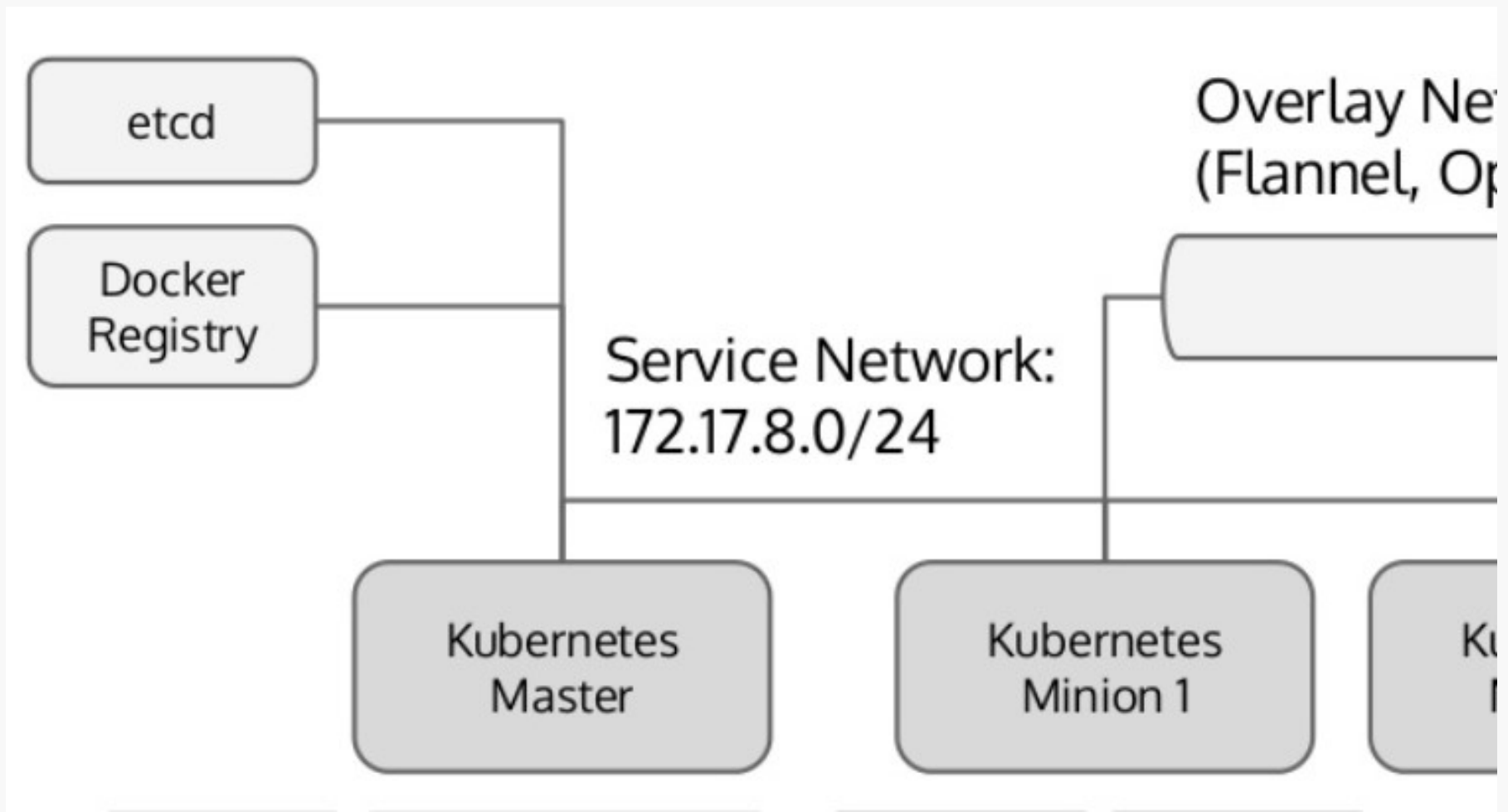
- » Deklaratív konténer csoport kezelés @ google
- » Konténer csoport: ugyanaz a hálózati névtér és kötetek (volume)

The screenshot shows a GitHub repository page for 'googlearchive / container-agent'. The left sidebar contains a list of recent commits: 'initial import' by proppy (May 16, 2014), 'Merge pull request #12 from jbeda' (Jun 8, 2014), and 'Add a deprecation notice' by thockin (Jun 20, 2014). The main content area shows the repository name, navigation tabs for Code, Issues (2), Pull requests (2), Pulse, and Graphs. The description reads: 'Simple agent for running containers based on a declarative manifest.' It also displays '13 commits' and '1 branch'. Below this, there's a section for 'Branch: master' with a 'New pull request' button, and buttons for 'New file' and 'Find file'. A pull request by 'elibixby' is visible, titled 'Merge pull request #19 from jonparrott/patch-1'. At the bottom, a file named 'container\_agent' is listed with its commit history.

## Kubernetes – main components

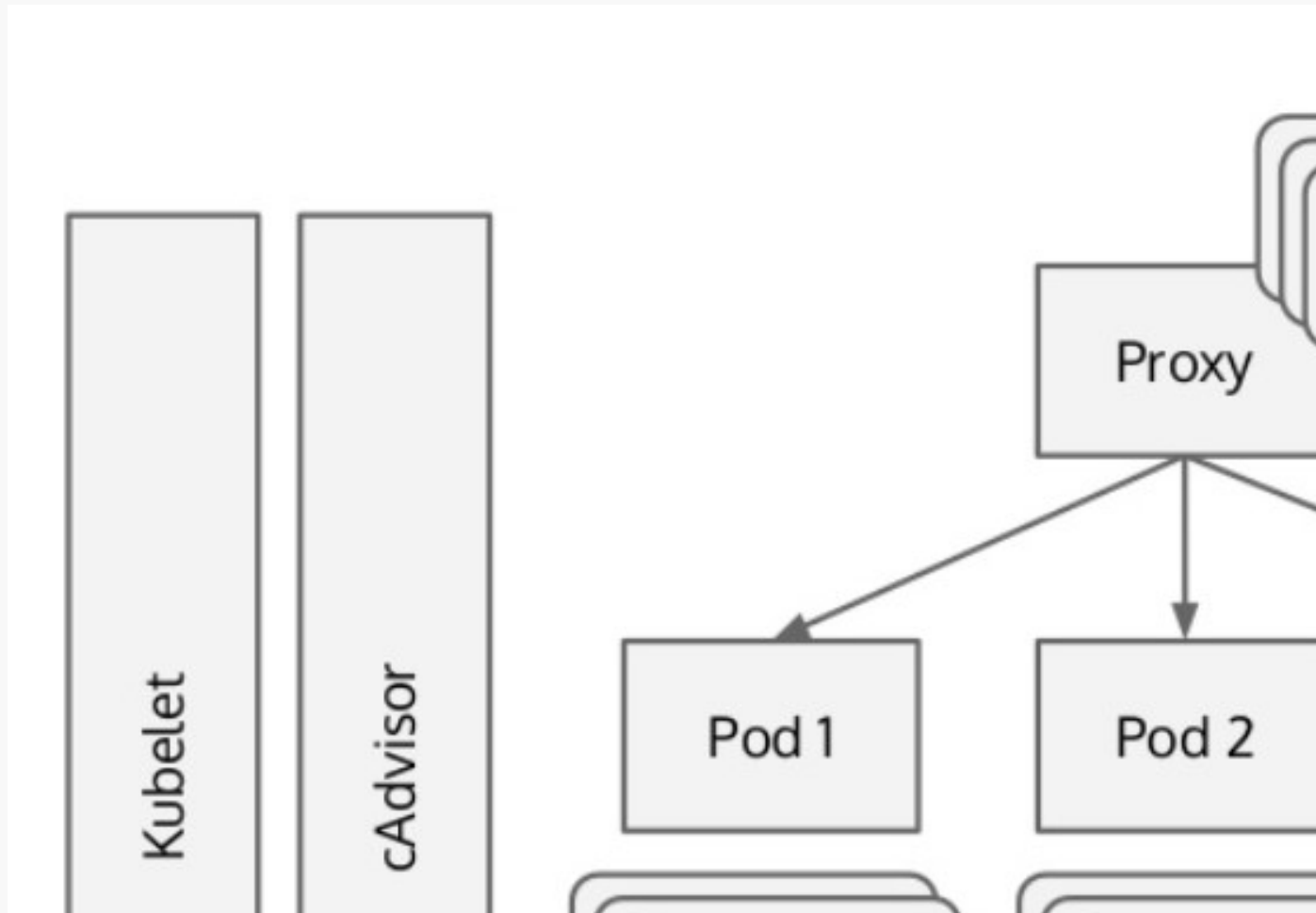
- **Pod** - A group of Containers
- **Labels** - Labels for identifying p
- **Kubelet** - Container Agent
- **Proxy** - A load balancer for Pod
- **etcd** - A metadata service
- **cAdvisor** - Container Advisor p  
usage/performance statistics  
**Replication Controller** - Manages

# Kubernetes deployment

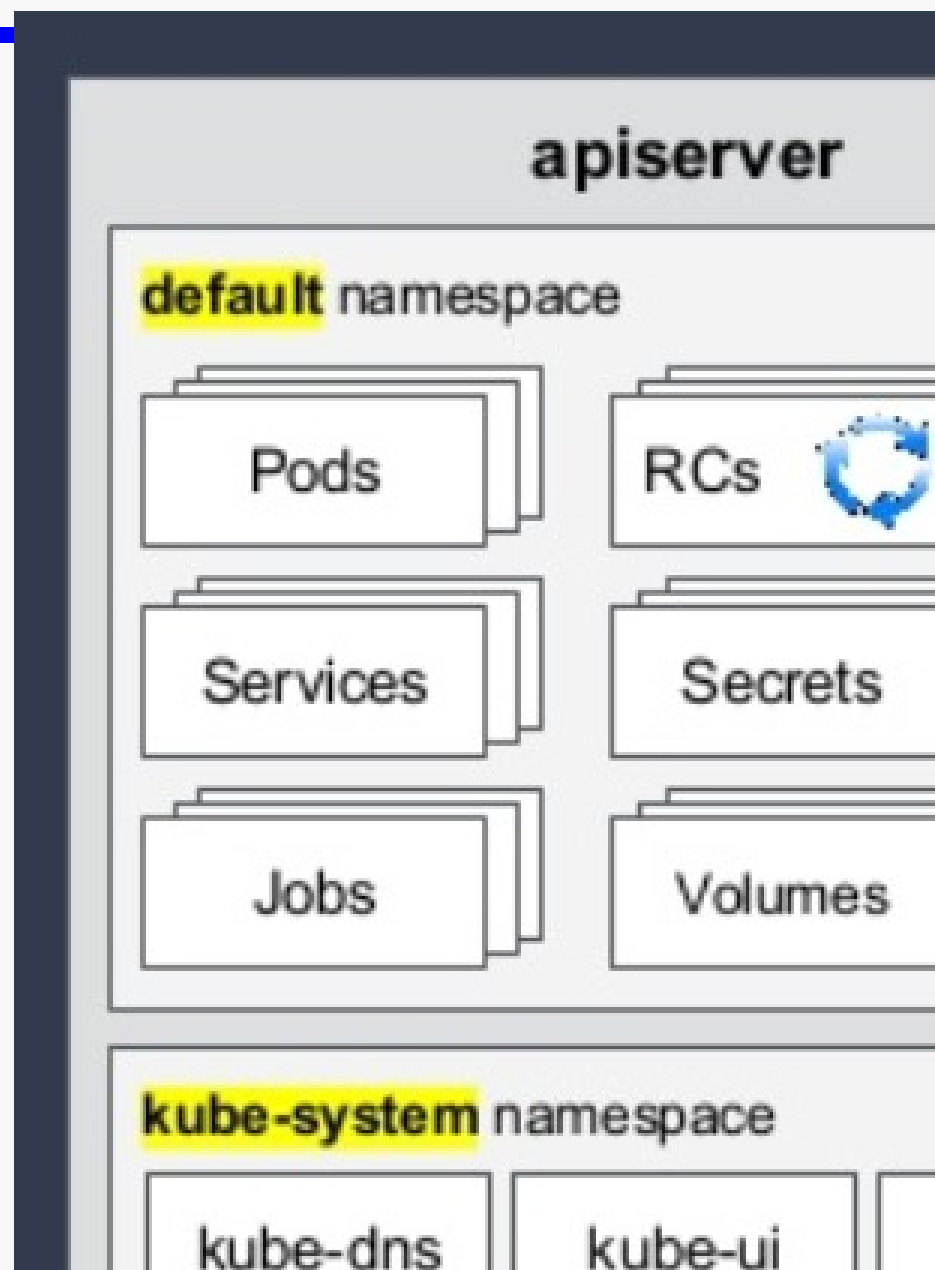




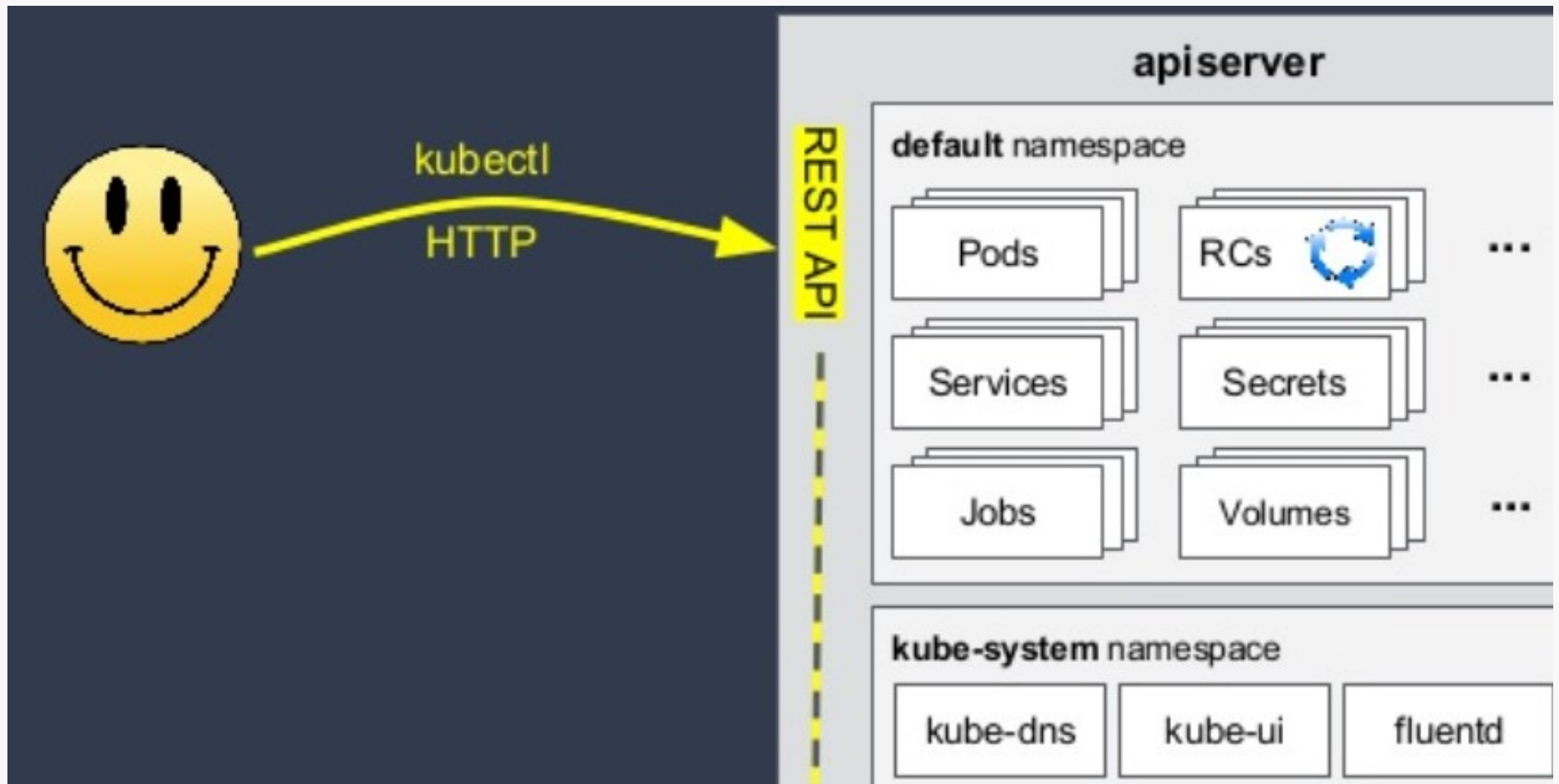
**Worker node = minion**



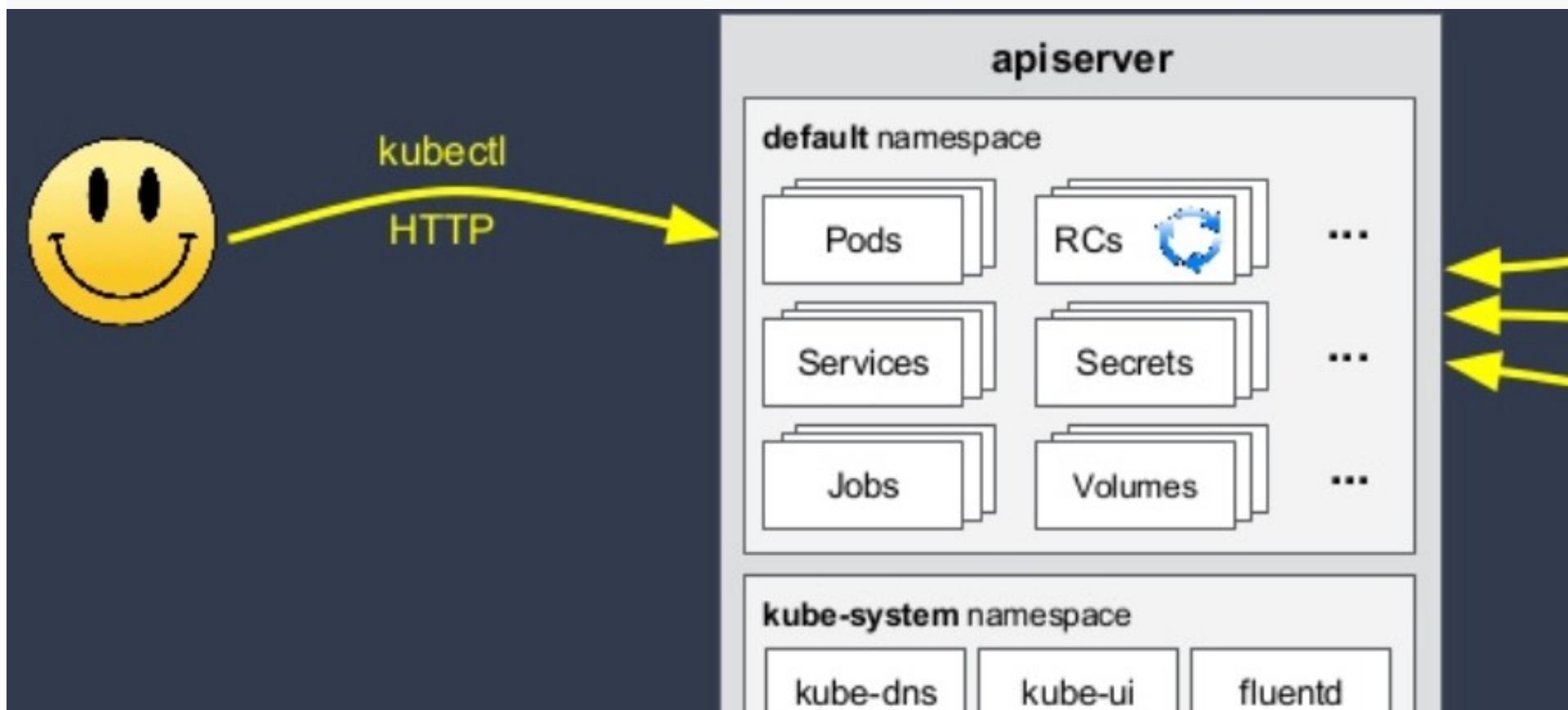
# API szerver



# API szerver kapcsolatai



# A kube-system szolgáltatásai

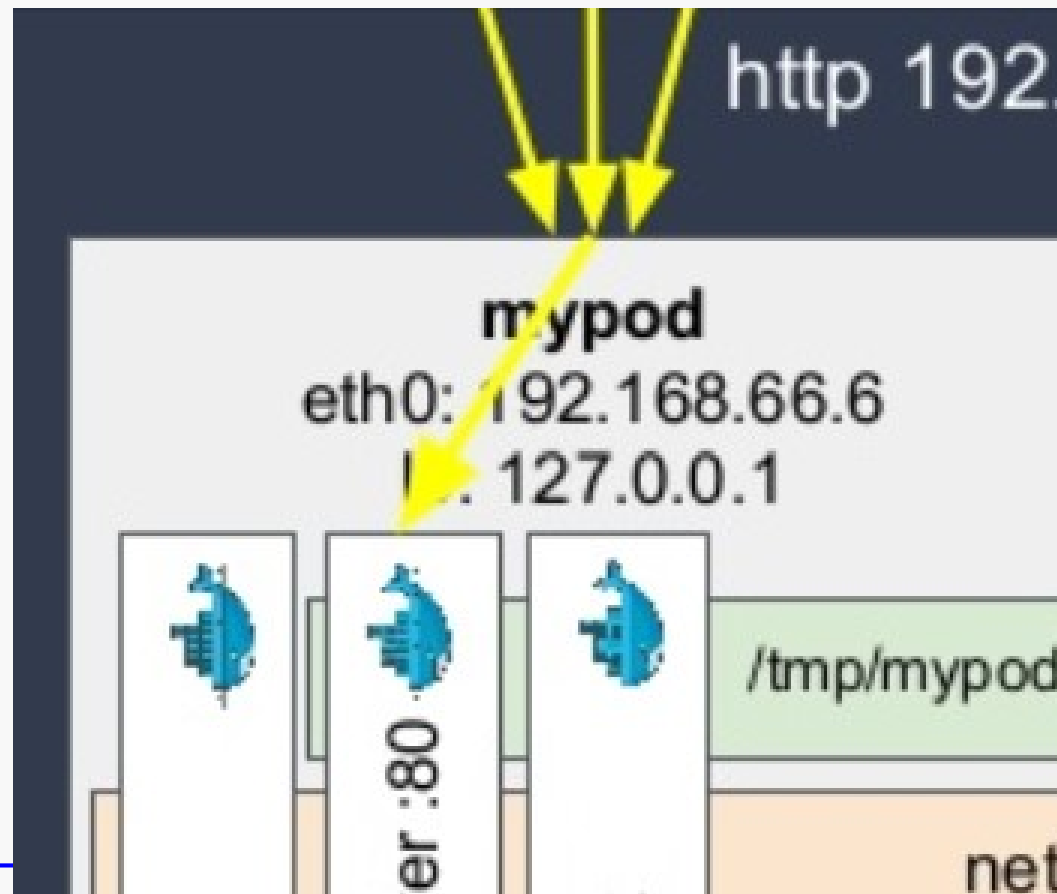


# Kubernetes network

- » At Pod level every container is in the same namespace
  - » Pro: can reach each other via localhost
  - » Consequence: mind the port assignment within a Pod (2 containers cannot use the same port)
- » Hosts must communicate with containers without NATs
- » Typical solutions:
  - » Flannel: own solution, flat overlay
  - » OVS: Open VSwitch – generic solution, widely used in the industry
  - » Lots of alternatives:  
<https://kubernetes.io/docs/concepts/cluster-administration/networking/#how-to-achieve-this>

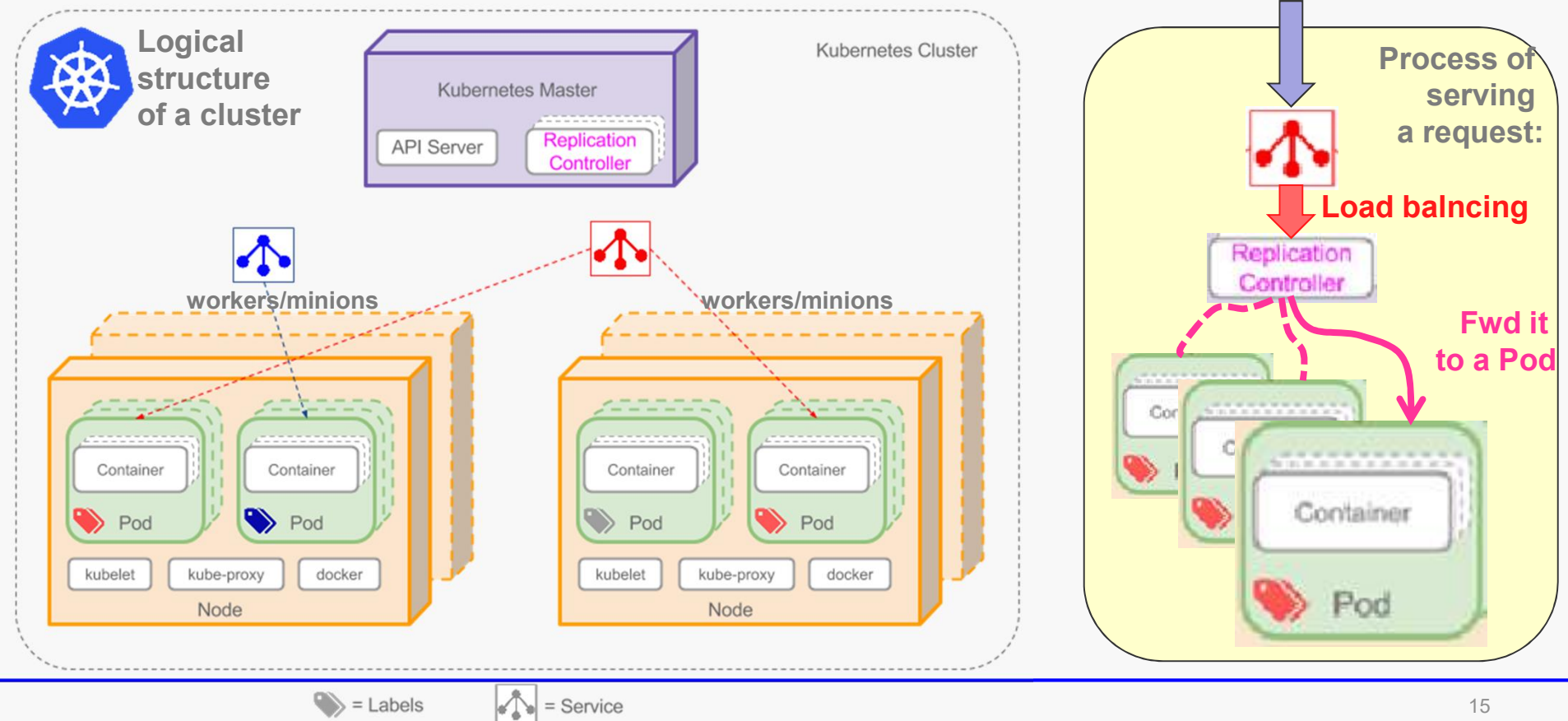
## Ha én (csak) Docker volnék...

- » Elvileg docker konténerekkel is meg lehet valósítani, azt, amit egy Kuberneteses pod tud...



# Logical structure of a Kubernetes cluster

- » Control by the master
- » Service offers access to users
  - » Handled by a load balancer (the Replication Controller)
  - » The request is answered by one Pod



# Demo

- » Kubernetes on-line demo
  - » Starting a Pod, handling in cli

<https://kubernetes.io/docs/tutorials/kubernetes-basics/>