讲师介绍

李振良（阿良）

资深运维工程师，8年运维实战经验，51CTO知名博主。曾就职在IDC，大数据，金融行业，现任职奇虎360公司。曾主导自动化运维与K8S容器平台建设，现管理近800台服务器，主要负责360浏览器业务与容器化迁移。

技术博客：http://blog.51cto.com/lizhenliang

DevOps技术栈

专注于分享DevOps工具链及经验总结

运维 开发 容器

架构 职场 面试

微信扫一扫关注

阿良微信

Kubernetes 概述

1. Kubernetes是什么
2. Kubernetes特性
3. Kubernetes集群架构与组件
4. Kubernetes核心概念

Kubernetes是什么

- Kubernetes是Google在2014年开源的一个容器集群管理系统，Kubernetes简称K8S。
- K8S用于容器化应用程序的部署，扩展和管理。
- K8S提供了容器编排，资源调度，弹性伸缩，部署管理，服务发现等一系列功能。
- Kubernetes目标是让部署容器化应用简单高效。

官方网站：http://www.kubernetes.io

Kubernetes特性

- **自我修复**

在节点故障时重新启动失败的容器，替换和重新部署，保证预期的副本数量；杀死健康检查失败的容器，并且在未准备好之前不会处理客户端请求，确保线上服务不中断。

- **弹性伸缩**

使用命令、UI或者基于CPU使用情况自动快速扩容和缩容应用程序实例，保证应用业务高峰并发时的高可用性；业务低峰时回收资源，以最小成本运行服务。

- **自动部署和回滚**

K8S采用滚动更新策略更新应用，一次更新一个Pod，而不是同时删除所有Pod，如果更新过程中出现问题，将回滚更改，确保升级不受影响业务。

- **服务发现和负载均衡**

K8S为多个容器提供一个统一访问入口（内部IP地址和一个DNS名称），并且负载均衡关联的所有容器，使得用户无需考虑容器IP问题。

- **机密和配置管理**

管理机密数据和应用程序配置，而不需要把敏感数据暴露在镜像里，提高敏感数据安全性。并可以将一些常用的配置存储在K8S中，方便应用程序使用。

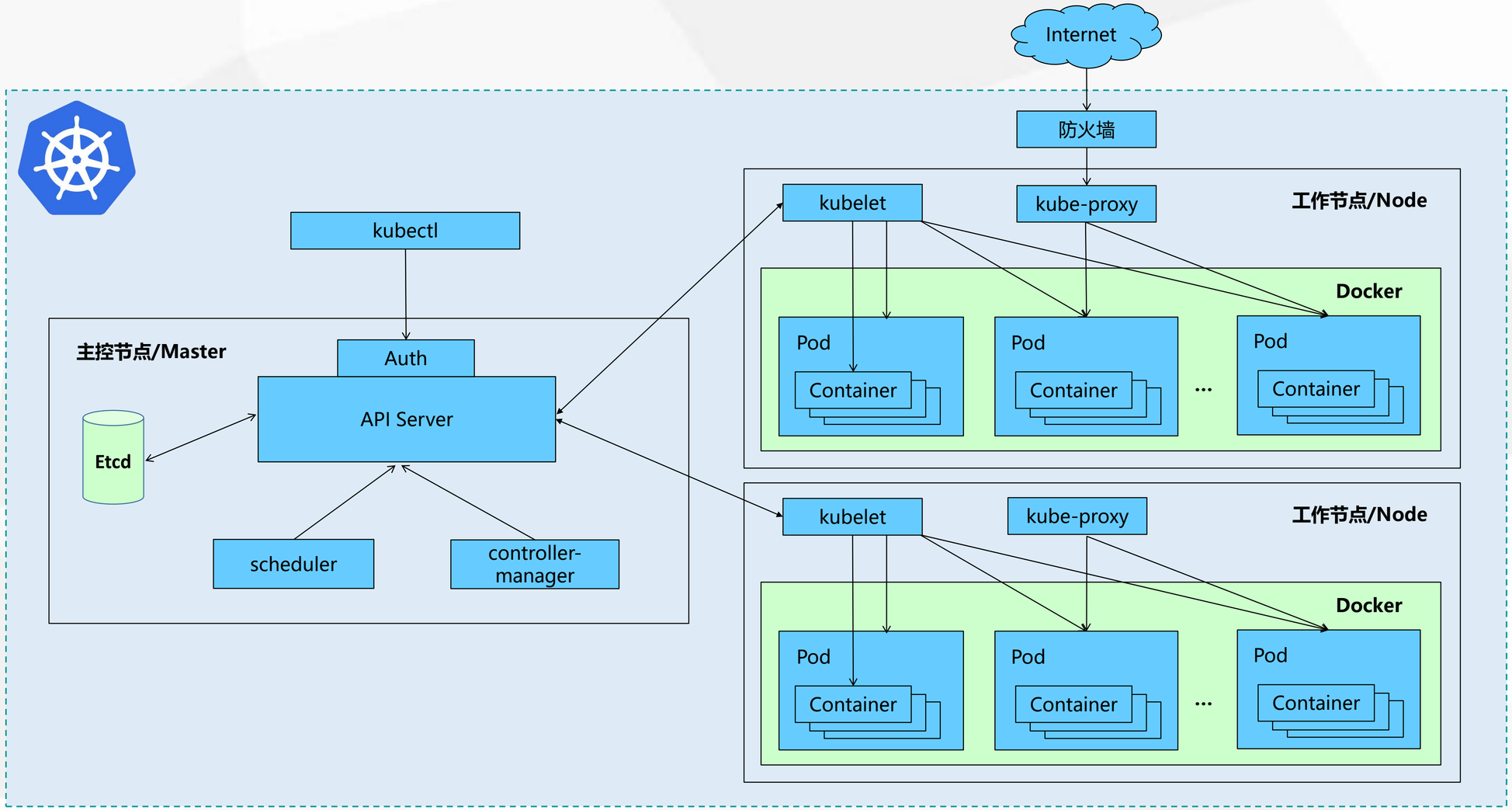
- **存储编排**

挂载外部存储系统，无论是来自本地存储，公有云（如AWS），还是网络存储（如NFS、GlusterFS、Ceph）都作为集群资源的一部分使用，极大提高存储使用灵活性。

- **批处理**

提供一次性任务，定时任务；满足批量数据处理和分析的场景。

Kubernetes集群架构与组件



Kubernetes集群架构与组件

Master组件

- **kube-apiserver**

Kubernetes API，集群的统一入口，各组件协调者，以RESTful API提供接口服务，所有对象资源的增删改查和监听操作都交给APIServer处理后再提交给Etcd存储。

- **kube-controller-manager**

处理集群中常规后台任务，一个资源对应一个控制器，而ControllerManager就是负责管理这些控制器的。

- **kube-scheduler**

根据调度算法为新创建的Pod选择一个Node节点，可以任意部署,可以部署在同一个节点上,也可以部署在不同的节点上。

- **etcd**

分布式键值存储系统。用于保存集群状态数据，比如Pod、Service等对象信息。

Node组件

- **kubelet**

kubelet是Master在Node节点上的Agent，管理本机运行容器的生命周期，比如创建容器、Pod挂载数据卷、下载secret、获取容器和节点状态等工作。kubelet将每个Pod转换成一组容器。

- **kube-proxy**

在Node节点上实现Pod网络代理，维护网络规则和四层负载均衡工作。

- **docker或rocket**

容器引擎，运行容器。

Kubernetes核心概念

- **Pod**
  - 最小部署单元
  - 一组容器的集合
  - 一个Pod中的容器共享网络命名空间
  - Pod是短暂的
- **Controllers**
  - ReplicaSet： 确保预期的Pod副本数量
  - Deployment： 无状态应用部署
  - StatefulSet： 有状态应用部署
  - DaemonSet： 确保所有Node运行同一个Pod
  - Job： 一次性任务
  - Cronjob： 定时任务

  更高级层次对象，部署和管理Pod
- **Service**
  - 防止Pod失联
  - 定义一组Pod的访问策略

- **Label：** 标签，附加到某个资源上，用于关联对象、查询和筛选
- **Namespace：** 命名空间，将对象逻辑上隔离

搭建一个 Kubernetes 集群

1. 官方提供的三种部署方式
2. Kubernetes集群环境规划
3. HTTPS证书介绍
4. Etcd数据库集群部署
5. Node安装Docker
6. Flannel容器集群网络部署
7. 部署Master组件
8. 部署Node组件
9. 部署一个测试示例

官方提供的三种部署方式

- **minikube**

Minikube是一个工具，可以在本地快速运行一个单点的Kubernetes，仅用于尝试Kubernetes或日常开发的用户使用。

部署地址：https://kubernetes.io/docs/setup/minikube/

- **kubeadm**

Kubeadm也是一个工具，提供kubeadm init和kubeadm join，用于快速部署Kubernetes集群。

部署地址：https://kubernetes.io/docs/reference/setup-tools/kubeadm/kubeadm/

- **二进制包**

推荐，从官方下载发行版的二进制包，手动部署每个组件，组成Kubernetes集群。

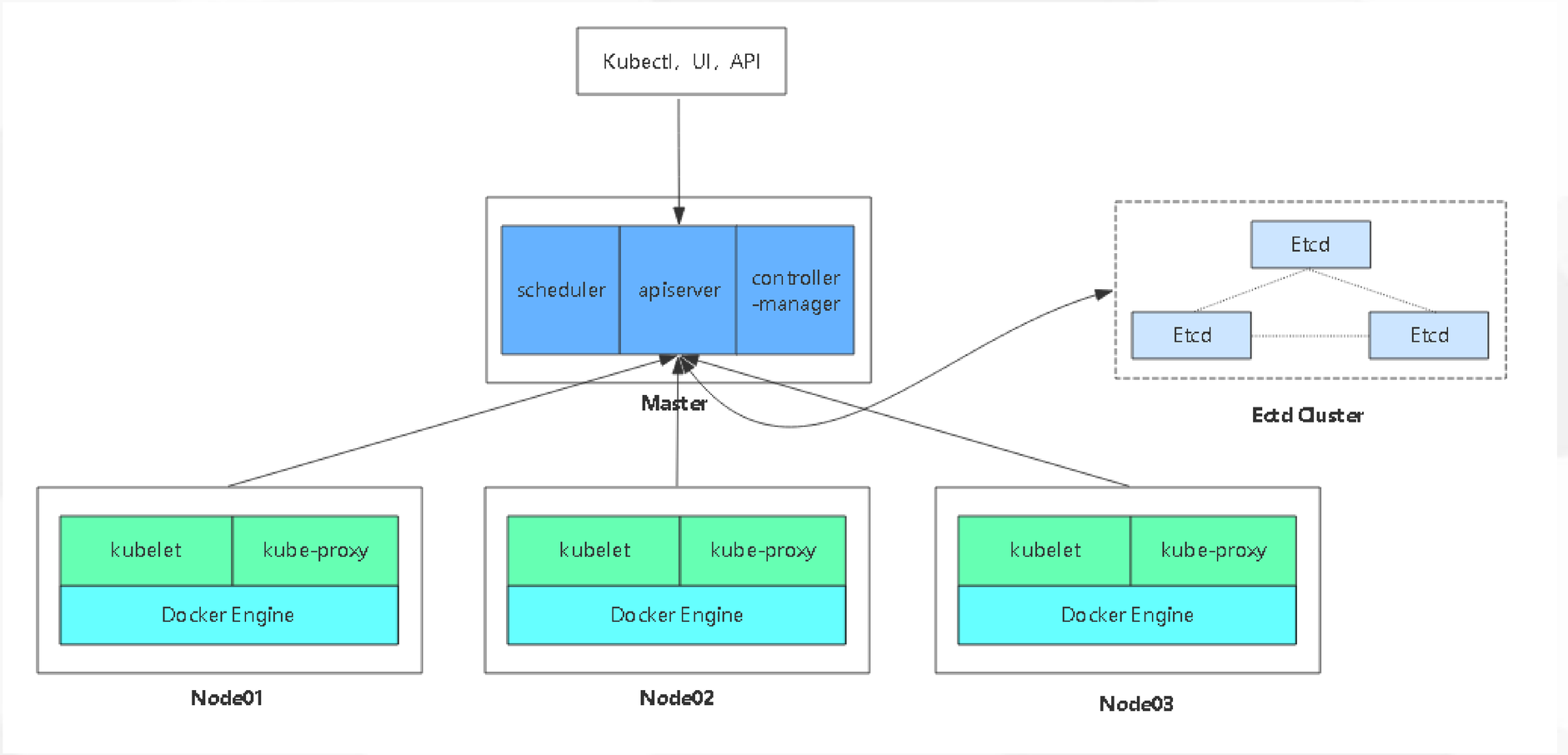
下载地址：https://github.com/kubernetes/kubernetes/releases

Kubernetes集群环境规划

软件	版本
Linux操作系统	CentOS7.5_x64
Kubernetes	1.13
Docker	18.xx-ce
Etcd	3.x
Flannel	0.10

角色	IP	组件	推荐配置
k8s-master	192.168.31.63	kube-apiserver kube-controller-manager kube-scheduler etcd	CPU：2C+ 内存：2G+
k8s-node01	192.168.31.65	kubelet kube-proxy docker flannel etcd	
k8s-node02	192.168.31.66	kubelet kube-proxy docker flannel etcd	

Kubernetes集群环境规划



自签SSL证书

组件	使用的证书
etcd	ca.pem，server.pem，server-key.pem
flannel	ca.pem，server.pem，server-key.pem
kube-apiserver	ca.pem，server.pem，server-key.pem
kubelet	ca.pem，ca-key.pem
kube-proxy	ca.pem，kube-proxy.pem，kube-proxy-key.pem
kubectl	ca.pem，admin.pem，admin-key.pem

Etcd数据库集群部署

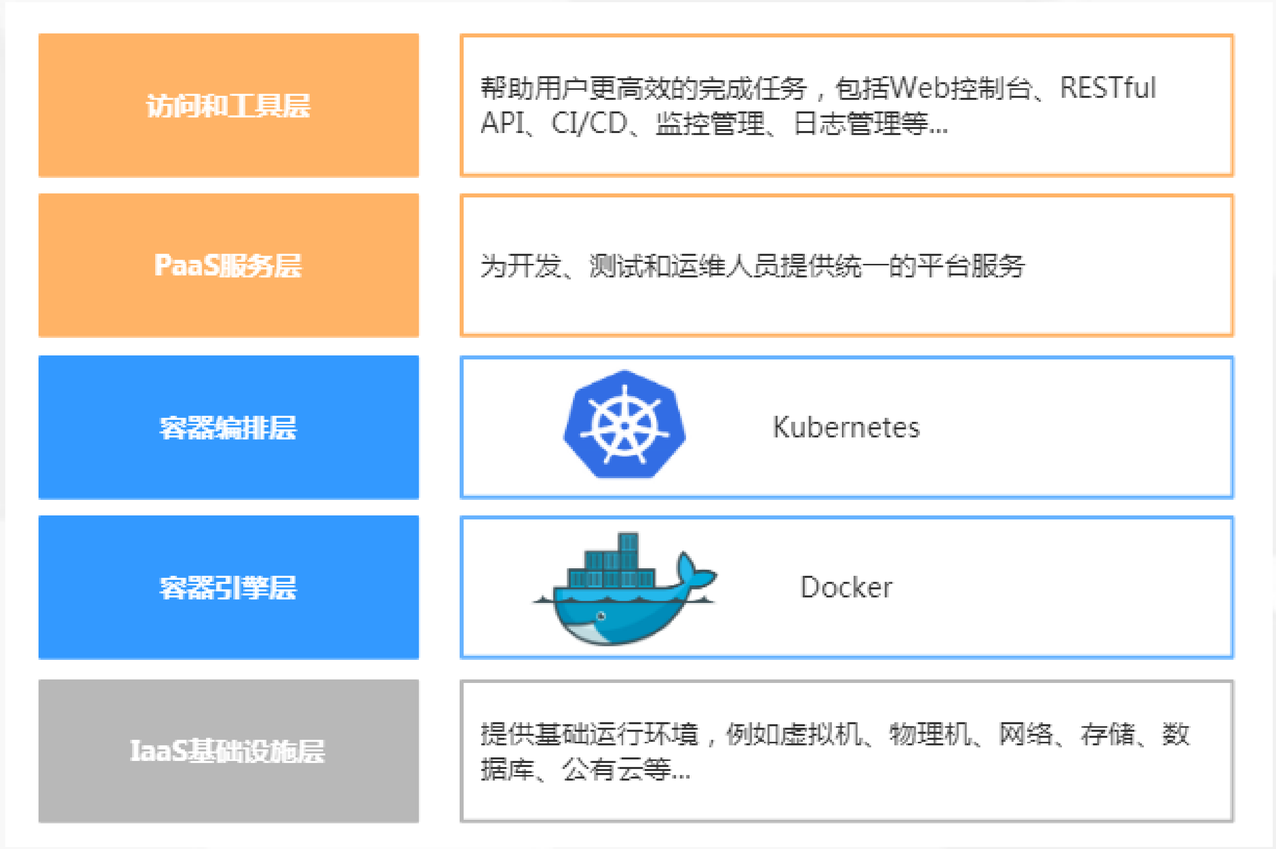
- **二进制包下载地址**

https://github.com/etcd-io/etcd/releases

- **查看集群状态**

```
/opt/etcd/bin/etcdctl \
--ca-file=ca.pem --cert-file=server.pem --key-file=server-key.pem \
--endpoints="https://192.168.0.x:2379,https://192.168.0.x:2379,https://192.168.0.x:2379" \
cluster-health
```

Node安装Docker



Flannel容器集群网络部署

1. 写入分配的子网段到etcd，供flanneld使用

```
/opt/etcd/bin/etcdctl \
--ca-file=ca.pem --cert-file=server.pem --key-file=server-key.pem \
--endpoints="https://192.168.0.x:2379,https://192.168.0.x:2379,https://192.168.0.x:2379" \
set /coreos.com/network/config  '{ "Network": "172.17.0.0/16", "Backend": {"Type": "vxlan"}}'
```

2. 下载二进制包

https://github.com/coreos/flannel/releases

3. 部署与配置Flannel

4. systemd管理Flannel

5. 配置Docker使用Flannel生成的子网

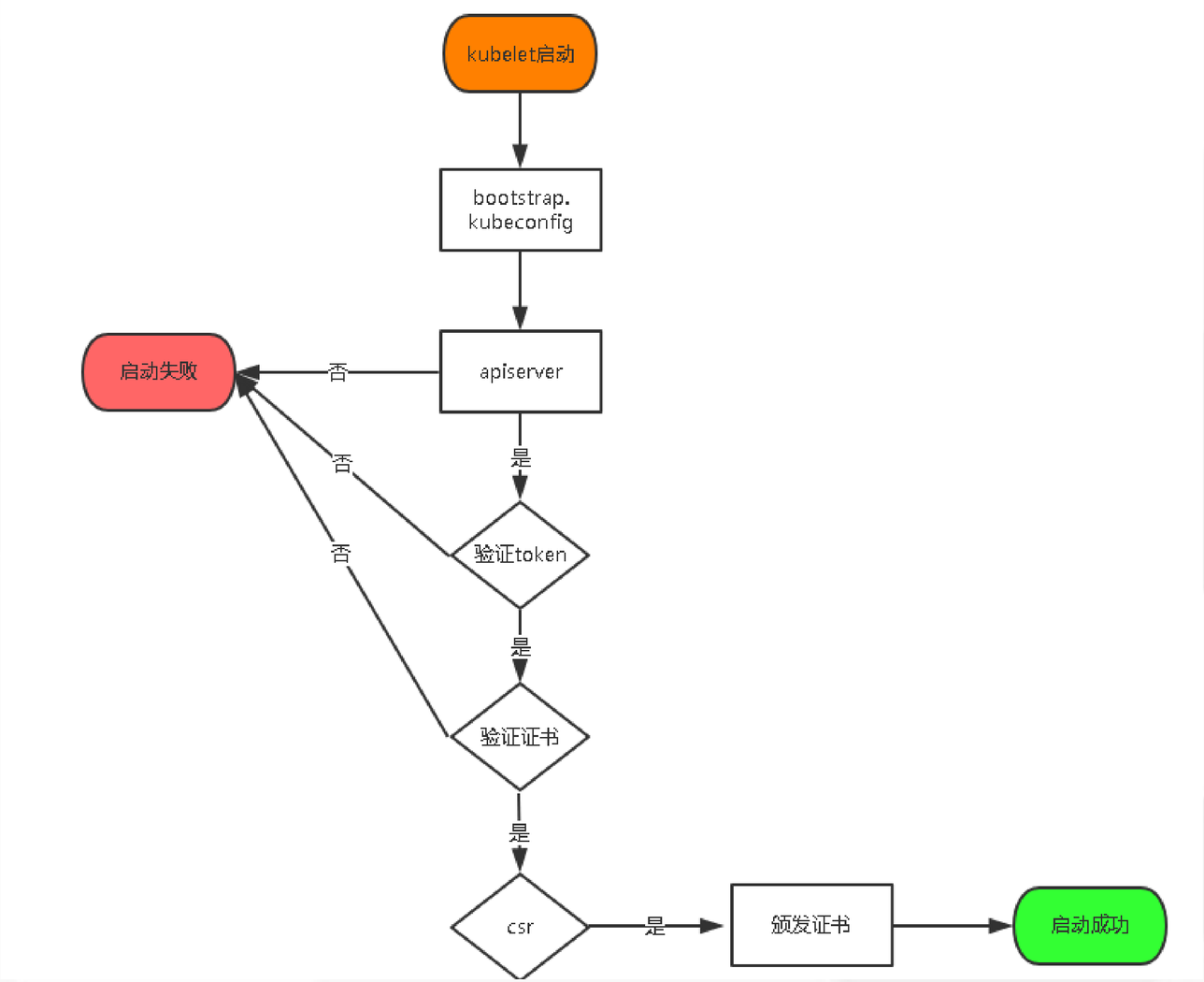
6. 启动Flannel

部署Master组件

1. kube-apiserver
2. kube-controller-manager
3. kube-scheduler

配置文件 -> systemd管理组件 -> 启动

部署Node组件



部署Node组件

1. 将kubelet-bootstrap用户绑定到系统集群角色

```
kubectl create clusterrolebinding kubelet-bootstrap \
--clusterrole=system:node-bootstrapper \
--user=kubelet-bootstrap
```

2. 创建kubeconfig文件

3. 部署kubelet，kube-proxy组件

部署一个测试示例

```
# kubectl run nginx --image=nginx --replicas=3
# kubectl get pod
# kubectl expose deployment nginx --port=88 --target-port=80 --type=NodePort
# kubectl get svc nginx
```

谢谢

DevOps技术栈

专注于分享DevOps工具链及经验总结

运维 开发 容器

架构 职场 面试

微信扫一扫关注

阿良微信