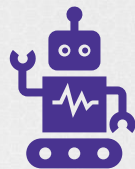# AUTHENTICATION

Admins

Developers

End Users
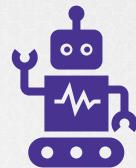
Bots

Accounts

Admins

Developers

User

Bots
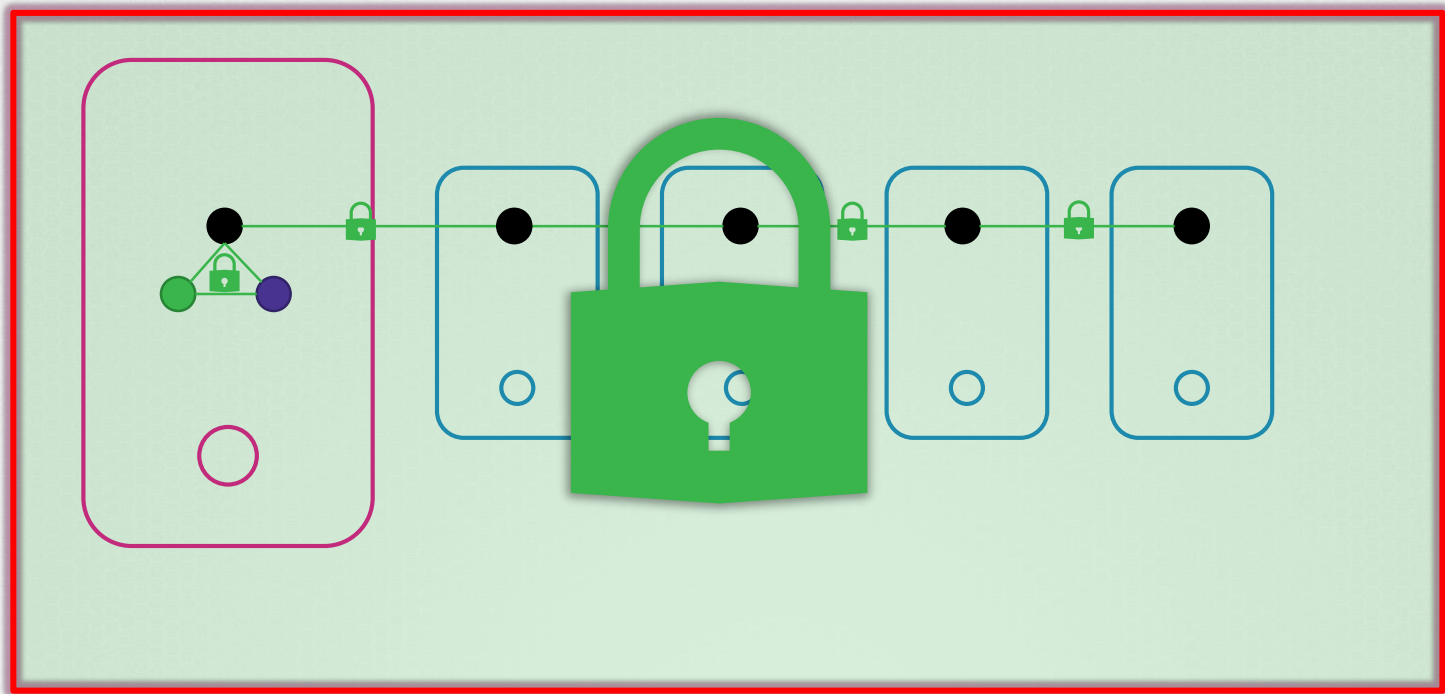
Service Accounts

Accounts

Admins

Developers

Bots

User

Service Accounts

```
kubectl create user user1
user1 Created
```

```
kubectl list users
Username
user1
user2
```

```
kubectl create serviceaccount sa1
Service Account sa1 Created
```
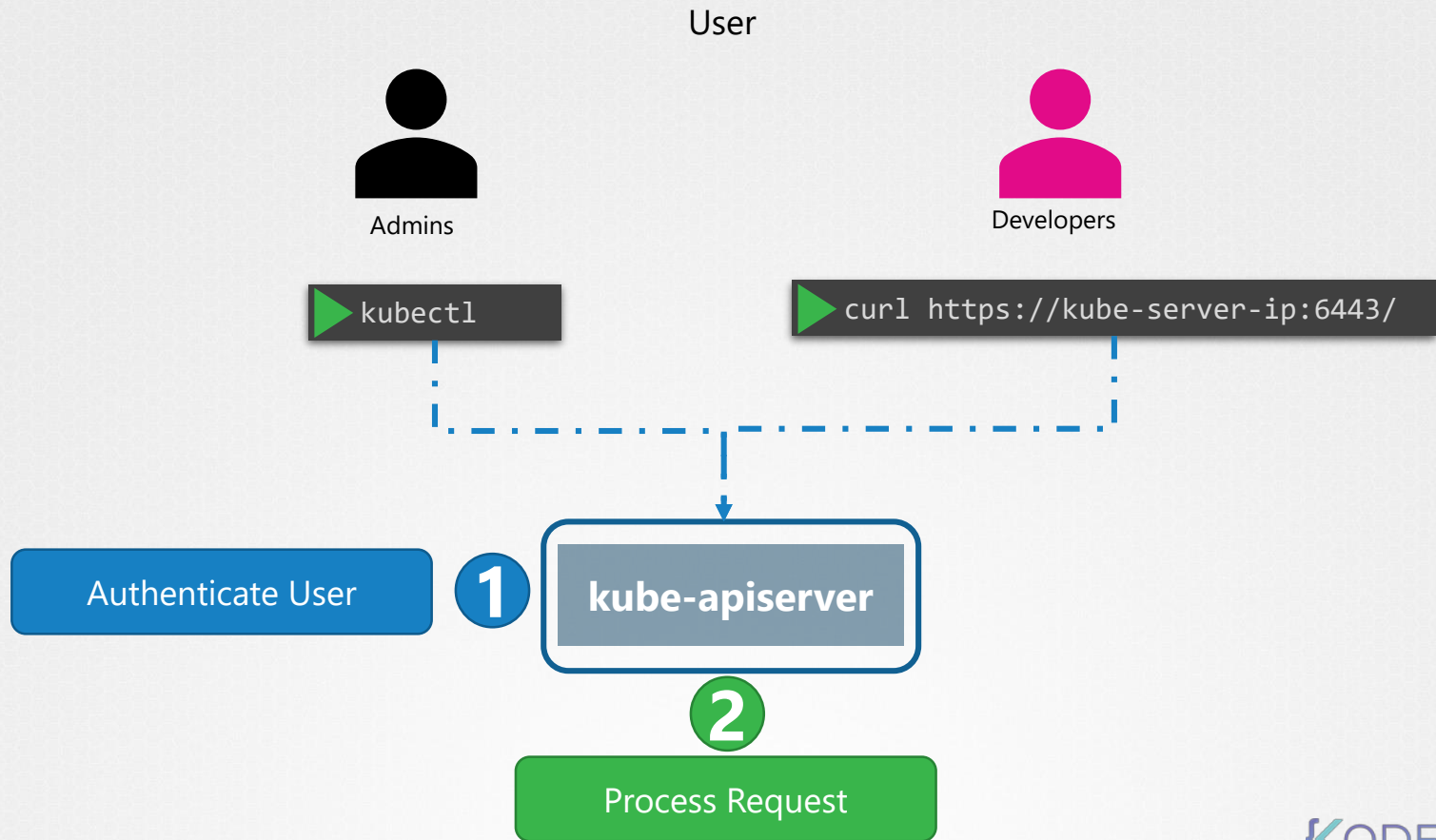
```
kubectl list serviceaccount
ServiceAccount
sa1
```

KODEKLOUD

Accounts

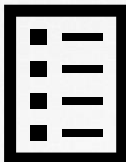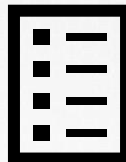User

Admins

Developers

```
> kubectl
```

```
> curl https://kube-server-ip:6443/
```

Authenticate User **1** kube-apiserver

2

Process Request

KODEKLOUD

Auth Mechanisms

kube-apiserver

Static Password File   Static Token File

Auth Mechanisms - Basic

--basic-auth-file

| kube-apiserver |

`--basic-auth-file=user-details.csv`

user-details.csv

```
password123,user1,u0001
password123,user2,u0002
password123,user3,u0003
password123,user4,u0004
password123,user5,u0005
```

kube-apiserver.service

```
ExecStart=/usr/local/bin/kube-apiserver \\
  --advertise-address=${INTERNAL_IP} \\
  --allow-privileged=true \\
  --apiserver-count=3 \\
  --authorization-mode=Node,RBAC \\
  --bind-address=0.0.0.0 \\
  --enable-swagger-ui=true \\
  --etcd-servers=https://127.0.0.1:2379 \\
  --event-ttl=1h \\
  --runtime-config=api/all \\
  --service-cluster-ip-range=10.32.0.0/24 \\
  --service-node-port-range=30000-32767 \\
  --v=2
```

Note: Showing fewer options for simplicity

KODEKLOUD

Authenticate User

Username, password

```
> curl -v -k https://master-node-ip:6443/api/v1/pods -u "user1:password123"
```

```
{
  "kind": "PodList",
  "apiVersion": "v1",
  "metadata": {
    "selfLink": "/api/v1/pods",
    "resourceVersion": "3594"
  },
  "items": [
    {
      "metadata": {
        "name": "nginx-64f497f8fd-krkg6",
        "generateName": "nginx-64f497f8fd-",
        "namespace": "default",
        "selfLink": "/api/v1/namespaces/default/pods/nginx-64f497f8fd-krkg6",
        "uid": "77dd7dfb-2914-11e9-b468-0242ac11006b",
        "resourceVersion": "3569",
        "creationTimestamp": "2019-02-05T07:05:49Z",
        "labels": {
          "pod-template-hash": "2090539498",
          "run": "nginx"
```

KLOUD

Auth Mechanisms - Basic

Static Password File

Static Token File

`user-details.csv`

```
password123,user1,u0001,group1
password123,user2,u0002,group1
password123,user3,u0003,group2
password123,user4,u0004,group2
password123,user5,u0005,group2
```

`user-token-details.csv`

```
KpjCVbI7rCFAHYPkByTIzRb7gu1cUc4B,user10,u0010,group1
rJjncHmvtXHc6MlWQddhtvNyyhgTdxSC,user11,u0011,group1
mjpOFIEiFOkL9toikaRNtt59ePtczZSq,user12,u0012,group2
PG41IXhs7QjqwWkmBkvgGT9glOyUqZij,user13,u0013,group2
```

`--token-auth-file=user-details.csv`

Token

```
curl -v -k https://master-node-ip:6443/api/v1/pods --header "Authorization: Bearer KpjCVbI7rCFAHYPkBzRb7gu1cUc4B"
```

Note: Showing fewer options for simplicity

KODEKLOUD

Note

- This is not a recommended authentication mechanism
- Consider volume mount while providing the auth file in a kubeadm setup
- Setup Role Based Authorization for the new users

Certificate (Public Key)
----
*.crt *.pem

server.crt
server.pem
client.crt
client.pem

Private Key
----
*.key *-key.pem

server.key
server-key.pem
client.key
client-key.pem

Public Key (Lock)

Private Key

User: John
Password: Pass123

User: John
Password: Pass123

XCVB: DKSJD
LKJSDFK: XZKJSDLF

XCVB: DKSJD
LKJSDFK: XZKJSDLF

KODEKLOUD

TLS CERTIFICATES

What Certificates?

kube-apiserver
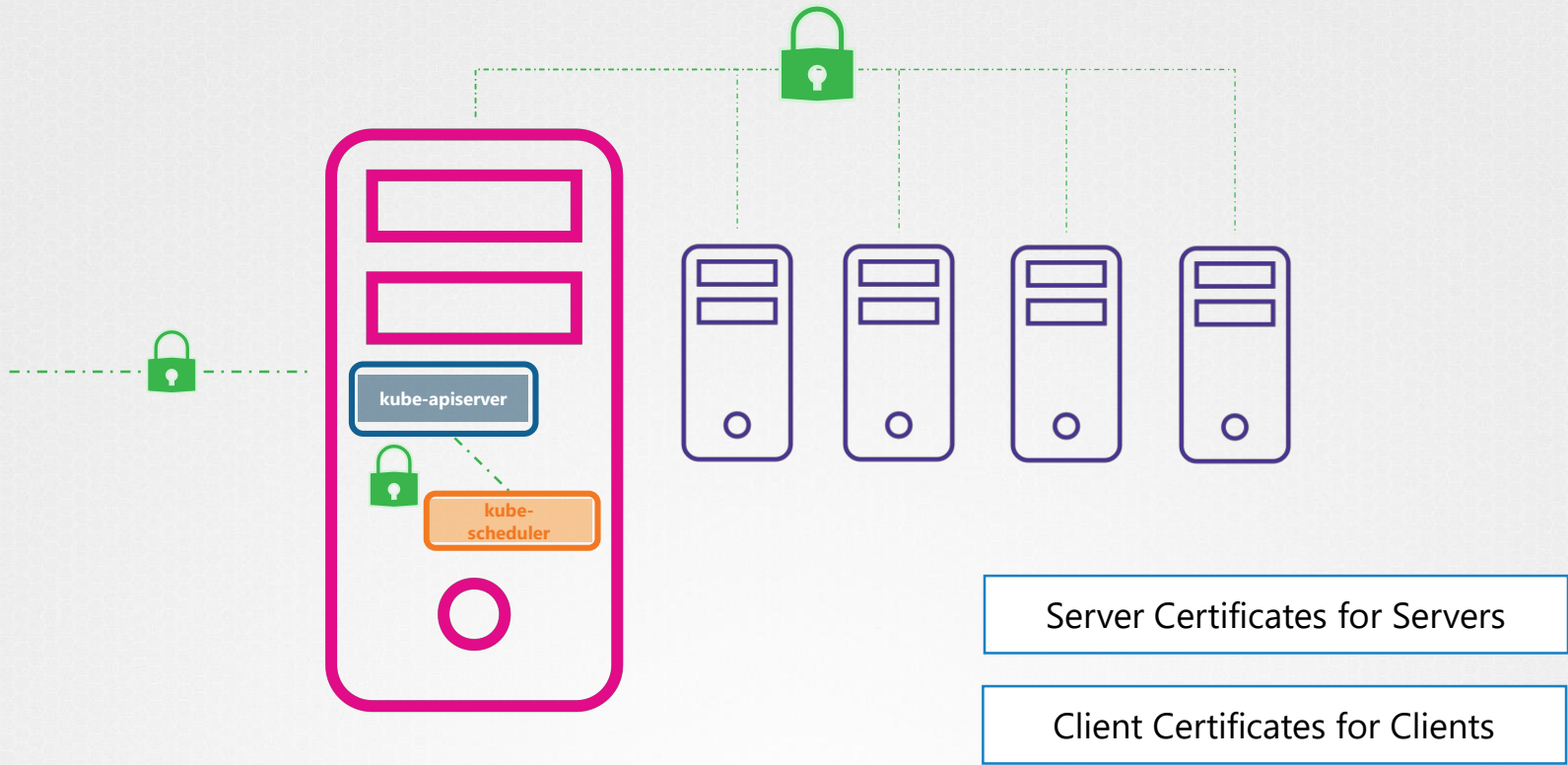
kube-scheduler

Server Certificates for Servers

Client Certificates for Clients

Server Certificates for Servers

KUBE-API SERVER

apiserver.crt apiserver.key

ETCD SERVER

etcdserver.crt etcdserver.key

KUBELET SERVER

kubelet.crt kubelet.key

KODEKLOUD
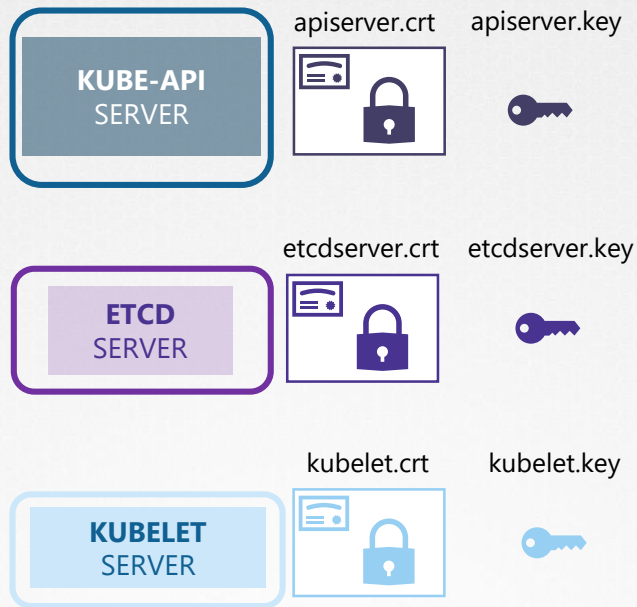
Client Certificates for Clients
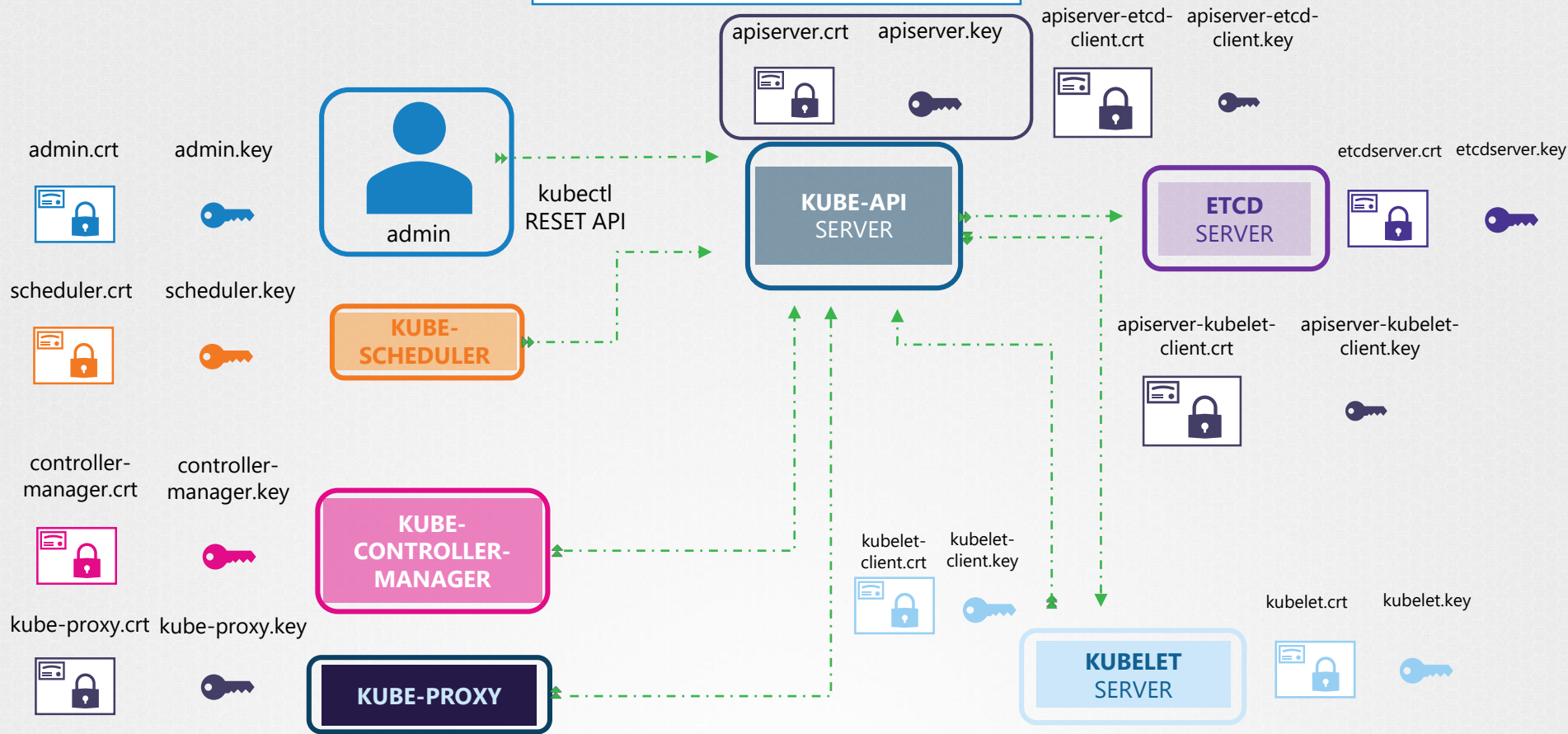
CERTIFICATE AUTHORITY (CA)

Client Certificates for Clients

admin.crt   admin.key

admin

scheduler.crt   scheduler.key

KUBE-SCHEDULER

controller-manager.crt   controller-manager.key

KUBE-CONTROLLER-MANAGER

kube-proxy.crt   kube-proxy.key

KUBE-PROXY

apiserver-kubelet-client.crt   apiserver-kubelet-client.key

KUBE-API SERVER

apiserver-etcd-client.crt   apiserver-etcd-client.key

KUBE-API SERVER

kubelet-client.crt   kubelet-client.key

KUBELET SERVER

Server Certificates for Servers

etcdserver.crt   etcdserver.key

ETCD SERVER

apiserver.crt   apiserver.key

KUBE-API SERVER

kubelet.crt   kubelet.key

KUBELET SERVER

KODEKLOUD

TLS CERTIFICATES

Generate Certificates

OPENSSL

CERTIFICATE AUTHORITY (CA)

(Important)

Generate Keys

🔑 ca.key

```
▶ openssl genrsa -out ca.key 2048
ca.key
```

Certificate Signing Request

🔒 ca.csr

Important

```
▶ openssl req -new -key ca.key -subj  "/CN=KUBERNETES-CA" -out ca.csr
ca.csr
```

Self Sign

Sign Certificates

Self Sign

🔒 ca.crt

```
▶ openssl x509 -req -in ca.csr -signkey ca.key -out ca.crt
ca.crt
```

KODEKLOUD

🔑 ca.key 🖼️🔒 ca.crt

Generate Keys

admin.key

🔑

```
▶ openssl genrsa -out admin.key 2048
admin.key
```

Certificate Signing Request

admin.csr

🔒

```
▶ openssl req -new -key admin.key -subj \
        "/CN=kube-admin" -out admin.csr
admin.csr
```

Important

Sign Certificates

Sign with CA

```
▶ openssl x509 -req -in admin.csr –CA ca.crt -CAkey ca.key -out admin.crt
```

02:13

NEW YORK
NY, US

✓

{KODE{KLOUD

ca.key
ca.crt

Generate Keys

scheduler.key

Certificate
Signing
Request

scheduler.csr

Sign
Certificates

scheduler.crt

CERTIFICATE

This Certificate Proudly Presented to

SYSTEM: KUBE-SCHEDULER

MIIDvDCCAqSgAwIBAgIUFZJ+94HWBrNF4jjZS6cVQg5d3pAwDQYJKoZIhvcNAQEL
BQAwZDELMAkGA1UEBhMCVVMxDzANBgNVBAgTBk9yZWdvbjERMA8GA1UEBxMIUG9y
dGxhbmQxETAPBgNVBAoTCFN5bWFudGGVjMQswCQYDVQQLEwJDQTERMA8GA1UEAxMI
U3ltYW50ZWMMwHhcNMTkwMjA4MDIxMzAwWhcNMjQwMjA3MDIxMzAwWjBkMQswCQYD
VQQGEwJVUzEPMA0GA1UECBMGT3J1Z29uMREwDwYDVQQHEwhQb3J0bGFuZDERMA8G

NEW YORK
NY, US

Issued by:

KUBE CONTROLLER MANGER

ca.key   ca.crt

Generate Keys

controller-manager.key

Certificate
Signing
Request

controller-manager.csr

Sign
Certificates

controller-manager.crt

CERTIFICATE

This Certificate Proudly Presented to

SYSTEM: KUBE-CONTOLLER-MANAGER

MIIDvDCCAqSgAwIBAgIUFZJ+94HWBrNF4jjZS6cVQg5d3pAwDQYJKoZIhvcNAQEL
BQAwZDELMAkGA1UEBhMCVVMxDzANBgNVBAgTBk9yZWdvbjERMA8GA1UEBxMIUG9y
dGxhbmQxETAPBgNVBAoTCFN5bWFudGVjMQswCQYDVQQLEwJDQTERMA8GA1UEAxMI
U3ltYW50ZWMwHhcNMTkwMjA4MDIxMzAwWhcNMjQwMjA3MDIxMzAwWjBkMQswCQYD
VQQGEwJVUzEPMA0GA1UECBMGT3J1Z29uMREwDwYDVQQHEwhQb3J0bGFuZDERMA8G

NEW YORK
NY, US

Issued by:

KODE KLOUD

ca.key   ca.crt

Client Certificates for Clients

admin.crt   admin.key

admin

scheduler.crt   scheduler.key

KUBE-SCHEDULER

controller-manager.crt   controller-manager.key

KUBE-CONTROLLER-MANAGER

kube-proxy.crt   kube-proxy.key

KUBE-PROXY

apiserver-kubelet-client.crt   apiserver-kubelet-client.key

apiserver-etcd-client.crt   apiserver-etcd-client.key

KUBE-API SERVER

kubelet-client.crt   kubelet-client.key

KUBELET SERVER

KODEKLOUD

Client Certificates for Clients

admin.crt   admin.key

admin

scheduler.crt   scheduler.key

KUBE-SCHEDULER

controller-manager.crt   controller-manager.key

KUBE-CONTROLLER-MANAGER

kube-proxy.crt   kube-proxy.key

KUBE-PROXY

ca.key   ca.crt

```
curl https://kube-apiserver:6443/api/v1/pods \
    --key admin.key --cert admin.crt
    --cacert ca.crt
```

```json
{
  "kind": "PodList",
  "apiVersion": "v1",
  "metadata": {
    "selfLink": "/api/v1/pods",
  },
  "items": []
}
```

kube-config.yaml

```yaml
apiVersion: v1
clusters:
- cluster:
    certificate-authority: ca.crt
    server: https://kube-apiserver:6443
  name: kubernetes
kind: Config
users:
- name: kubernetes-admin
  user:
    client-certificate: admin.crt
    client-key: admin.key
```

KODEKLOUD

ETCD SERVERS

etcdserver.crt   etcdserver.key

etcdpeer1.crt   etcdpeer1.key

ETCD SERVER

etcdpeer1.crt   etcdpeer1.key

ETCD PEER

etcdpeer2.crt   etcdpeer2.key

ETCD PEER

CERTIFICATE

This Certificate Proudly Presented to

ETCD-SERVER

MIIDvDCCAqSgAwIBAgIUFZJ+94HWBrNF4jjZS6cVQg5d3pAwDQYJKoZIhvcNAQEL
BQAwZDELMAkGA1UEBhMCVVMxDzANBgNVBAgTBk9yZWdvbjERMA8GA1UEBxMIUG9y
dGxhbmQxETAPBgNVBAoTCFN5bWFudGVjMQswCQYDVQQLEwJDQTERMA8GA1UEAxMI
U3ltYW50ZWMwHhcNMTkwMjA4MDIxMzAwWhcNMjQwMjA3MDIxMzAwWjBkMQswCQYD
VQQGEwJVUzEPMA0GA1UECBMGT3JlZ29uMREwDwYDVQQHEwhQb3J0bGFuZDERMA8G

NEW YORK
NY, US

Issued by:

CERTIFICATE

This Certificate Proudly Presented to

ETCD-PEER

NEW YORK
NY, US

KODE**K**LOUD

ETCD SERVERS


```
cat etcd.yaml

- etcd
  - --advertise-client-urls=https://127.0.0.1:2379
  - --key-file=/path-to-certs/etcdserver.key
  - --cert-file=/path-to-certs/etcdserver.crt
  - --client-cert-auth=true
  - --data-dir=/var/lib/etcd
  - --initial-advertise-peer-urls=https://127.0.0.1:2380
  - --initial-cluster=master=https://127.0.0.1:2380
  - --listen-client-urls=https://127.0.0.1:2379
  - --listen-peer-urls=https://127.0.0.1:2380
  - --name=master
  - --peer-cert-file=/path-to-certs/etcdpeer1.crt
  - --peer-client-cert-auth=true
  - --peer-key-file=/etc/kubernetes/pki/etcd/peer.key
  - --peer-trusted-ca-file=/etc/kubernetes/pki/etcd/ca.crt
  - --snapshot-count=10000
  - --trusted-ca-file=/etc/kubernetes/pki/etcd/ca.crt
```

etcdserver.crt   etcdserver.key

etcdpeer1.crt   etcdpeer1.key

ETCD SERVER

KUBE API SERVER

apiserver.crt   apiserver.key

KUBE-API SERVER

CERTIFICATE

kubernetes   10.96.0.1

kubernetes.default   172.17.0.87

This Certificate Proudly Presented to

kubernetes.default.svc

KUBE-API SERVER

kubernetes.default.svc.cluster.local

MIIDvDCCAqSgAwIBAgIUFZJ+94HWBrNF4jjZS6cVQg5d3pAwDQYJKoZIhvcNAQEL
BQAwZDELMAkGA1UEBhMCVVMxDzANBgNVBAgTBk9yZWdvbjERMA8GA1UEBxMIUG9y
dGxhbmQxETAPBgNVBAoTCFN5bWFudGVjMQswCQYDVQQLEwJDQTERMA8GA1UEAxMI
U3ltYW50ZWMMwHhcNMTkwMjA4MDIxMzAwWhcNMjQwMjA3MDIxMzAwWjBkMQswCQYD
VQQGEwJVUzEPMA0GA1UECBMGT3J1Z29uMREwDwYDVQQHEwhQb3J0bGFuZDERMA8G

NEW YORK
NY, US

Issued by:

✔

KODE KLOUD

KUBE API SERVER

apiserver.crt   apiserver.key

KUBE-API SERVER

```
openssl req -new -key apiserver.key -subj \
    "/CN=kube-apiserver" -out apiserver.csr –config openssl.cnf
```
```
apiserver.csr
```

openssl.cnf

```
[req]
req_extensions = v3_req
[ v3_req ]
basicConstraints = CA:FALSE
keyUsage = nonRepudiation,
subjectAltName = @alt_names
[alt_names]
DNS.1 = kubernetes
DNS.2 = kubernetes.default
DNS.3 = kubernetes.default.svc
DNS.4 = kubernetes.default.svc.cluster.local
IP.1 = 10.96.0.1
IP.2 = 172.17.0.87
```

```
openssl x509 -req -in apiserver.csr \
    -CA ca.crt -CAkey ca.key -out apiserver.crt
```
```
apiserver.crt
```

CERTIFICATE

kubernetes   10.96.0.1

kubernetes.default   172.17.0.87

kubernetes.default.svc

This Certificate Proudly Presented to

kubernetes.default.svc.cluster.local

KUBE-API SERVER

```
MIIDvDCCAqSgAwIBAgIUFZJ+94HWBrNF4jjZS6cVQg5d3pAwDQYJKoZIhvcNAQEL
BQAwZDELMAkGA1UEBhMCVVMxDzANBgNVBAgTBk9yZWdvbjERMA8GA1UEBxMIUG9y
dGxhbmQxETAPBgNVBAoTCFN5bWFudGVjMQswCQYDVQQLEwJDQTERMA8GA1UEAxMI
U3l tYW50ZWMwHhcNMTkwMjA4MDIxMzAwWhcNMjQwMjA3MDIxMzAwWjBkMQswCQYD
VQQGEwJVUzEPMA0GA1UECBMGT3JlZ29uMREwDwYDVQQHEwhQb3J0bGERMA8G
```

NEW YORK
NY, US

Issued by:

KODEKLOUD

KUBE API SERVER

apiserver.crt   apiserver.key

KUBE-API SERVER

apiserver-kubelet-client.crt   apiserver-kubelet-client.key

apiserver-etcd-client.crt   apiserver-etcd-client.key

KUBE-API SERVER

```
ExecStart=/usr/local/bin/kube-apiserver \\
  --advertise-address=${INTERNAL_IP} \\
  --allow-privileged=true \\
  --apiserver-count=3 \\
  --authorization-mode=Node,RBAC \\
  --bind-address=0.0.0.0 \\
  --enable-swagger-ui=true \\
  --etcd-cafile=/var/lib/kubernetes/ca.pem \\
  --etcd-certfile=/var/lib/kubernetes/apiserver-etcd-client.crt \\
  --etcd-keyfile=/var/lib/kubernetes/apiserver-etcd-client.key \\
  --etcd-servers=https://127.0.0.1:2379 \\
  --event-ttl=1h \\
  --kubelet-certificate-authority=/var/lib/kubernetes/ca.pem \\
  --kubelet-client-certificate=/var/lib/kubernetes/apiserver-etcd-client.crt \\
  --kubelet-client-key=/var/lib/kubernetes/apiserver-etcd-client.key \\
  --kubelet-https=true \\
  --runtime-config=api/all \\
  --service-account-key-file=/var/lib/kubernetes/service-account.pem \\
  --service-cluster-ip-range=10.32.0.0/24 \\
  --service-node-port-range=30000-32767 \\
  --client-ca-file=/var/lib/kubernetes/ca.pem \\
  --tls-cert-file=/var/lib/kubernetes/apiserver.crt \\
  --tls-private-key-file=/var/lib/kubernetes/apiserver.key \\
  --v=2
```

KUBECTL NODES (SERVER CERT)

kubelet.crt   kubelet.key

KUBELET SERVER

CERTIFICATE

node01

NEW YORK
NY, US

CERTIFICATE

node02

NEW YORK
NY, US

CERTIFICATE

node03

NEW YORK
NY, US

KUBELET

KUBELET

KUBELET

node01

node02

node03

kubelet-config.yaml (node01)

```yaml
kind: KubeletConfiguration
apiVersion: kubelet.config.k8s.io/v1beta1
authentication:
  x509:
    clientCAFile: "/var/lib/kubernetes/ca.pem"
authorization:
  mode: Webhook
clusterDomain: "cluster.local"
clusterDNS:
  - "10.32.0.10"
podCIDR: "${POD_CIDR}"
resolvConf: "/run/systemd/resolve/resolv.conf"
runtimeRequestTimeout: "15m"
tlsCertFile: "/var/lib/kubelet/kubelet-node01.crt"
tlsPrivateKeyFile: "/var/lib/kubelet/kubelet-node01.key"
```

KODEKLOUD

KUBECTL NODES (CLIENT CERT)

Kubelet-client.crt Kubelet-client.key

KUBELET SERVER

CERTIFICATE
Group:
SYSTEM:NODES
This Certificate Proudly Presented to
system:node:node01

NEW YORK
NY, US

CERTIFICATE
Group:
SYSTEM:NODES
This Certificate Proudly Presented to
system:node:node02

NEW YORK
NY, US

CERTIFICATE
Group:
SYSTEM:NODES
This Certificate Proudly Presented to
system:node:node03

NEW YORK
NY, US

KUBELET

KUBELET

KUBELET

node01

node02

node03

KODEKLOUD

TLS CERTIFICATES

View Certificate Details

"The Hard Way"

```
▶ cat /etc/systemd/system/kube-apiserver.service
```

```
[Service]
ExecStart=/usr/local/bin/kube-apiserver \\
  --advertise-address=172.17.0.32 \\
  --allow-privileged=true \\
  --apiserver-count=3 \\
  --authorization-mode=Node,RBAC \\
  --bind-address=0.0.0.0 \\
  --client-ca-file=/var/lib/kubernetes/ca.pem \\
  --enable-swagger-ui=true \\
  --etcd-cafile=/var/lib/kubernetes/ca.pem \\
  --etcd-certfile=/var/lib/kubernetes/kubernetes.pem \\
  --etcd-keyfile=/var/lib/kubernetes/kubernetes-key.pem \\
  --event-ttl=1h \\
  --kubelet-certificate-authority=/var/lib/kubernetes/ca.pem \\
  --kubelet-client-key=/var/lib/kubernetes/kubernetes-key.pem \\
  --kubelet-https=true \\
  --service-node-port-range=30000-32767 \\
  --tls-cert-file=/var/lib/kubernetes/kubernetes.pem \\
  --tls-private-key-file=/var/lib/kubernetes/kubernetes-key.pem
  --v=2
```

kubeadm

```
▶ cat /etc/kubernetes/manifests/kube-apiserver.yaml
```

```
spec:
  containers:
  - command:
    - kube-apiserver
    - --authorization-mode=Node,RBAC
    - --advertise-address=172.17.0.32
    - --allow-privileged=true
    - --client-ca-file=/etc/kubernetes/pki/ca.crt
    - --disable-admission-plugins=PersistentVolumeLabel
    - --enable-admission-plugins=NodeRestriction
    - --enable-bootstrap-token-auth=true
    - --etcd-cafile=/etc/kubernetes/pki/etcd/ca.crt
    - --etcd-certfile=/etc/kubernetes/pki/apiserver-etcd-client.cr
    - --etcd-keyfile=/etc/kubernetes/pki/apiserver-etcd-client.key
    - --etcd-servers=https://127.0.0.1:2379
    - --insecure-port=0
    - --kubelet-client-certificate=/etc/kubernetes/pki/apiserver-k
    - --kubelet-client-key=/etc/kubernetes/pki/apiserver-kubelet-c
    - --kubelet-preferred-address-types=InternalIP,ExternalIP,Host
    - --proxy-client-cert-file=/etc/kubernetes/pki/front-proxy-cli
    - --proxy-client-key-file=/etc/kubernetes/pki/front-proxy-clie
    - --requestheader-allowed-names=front-proxy-client
```

kubeadm

Component	Type	Certificate Path	CN Name	ALT Names	Organization	Issuer	Expiration
kube-apiserver	Server						
kube-apiserver	Server						
kube-apiserver	Server						
kube-apiserver	Client (Kubelet)						
kube-apiserver	Client (Kubelet)						
kube-apiserver	Client (Etcd)						
kube-apiserver	Client (Etcd)						
kube-apiserver	Client (Etcd)						

```
▶  cat /etc/kubernetes/manifests/kube-apiserver.yaml

spec:
  containers:
  - command:
    - kube-apiserver
    - --authorization-mode=Node,RBAC
    - --advertise-address=172.17.0.32
    - --allow-privileged=true
    - --client-ca-file=/etc/kubernetes/pki/ca.crt
    - --disable-admission-plugins=PersistentVolumeLabel
    - --enable-admission-plugins=NodeRestriction
    - --enable-bootstrap-token-auth=true
    - --etcd-cafile=/etc/kubernetes/pki/etcd/ca.crt
    - --etcd-certfile=/etc/kubernetes/pki/apiserver-etcd-client.crt
    - --etcd-keyfile=/etc/kubernetes/pki/apiserver-etcd-client.key
    - --etcd-servers=https://127.0.0.1:2379
    - --insecure-port=0
    - --kubelet-client-certificate=/etc/kubernetes/pki/apiserver-kubelet-client.crt
    - --kubelet-client-key=/etc/kubernetes/pki/apiserver-kubelet-client.key
    - --kubelet-preferred-address-types=InternalIP,ExternalIP,Hostname
    - --proxy-client-cert-file=/etc/kubernetes/pki/front-proxy-client.crt
    - --proxy-client-key-file=/etc/kubernetes/pki/front-proxy-client.key
    - --secure-port=6443
    - --service-account-key-file=/etc/kubernetes/pki/sa.pub
    - --service-cluster-ip-range=10.96.0.0/12
    - --tls-cert-file=/etc/kubernetes/pki/apiserver.crt
    - --tls-private-key-file=/etc/kubernetes/pki/apiserver.key
```

View Certificate

/etc/kubernetes/pki/apiserver.crt

```
openssl x509 -in /etc/kubernetes/pki/apiserver.crt -text -noout
```

```
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number: 3147495682089747350 (0x2bae26a58f090396)
    Signature Algorithm: sha256WithRSAEncryption
        Issuer: CN=kubernetes
        Validity
            Not Before: Feb 11 05:39:19 2019 GMT
            Not After : Feb 11 05:39:20 2020 GMT
        Subject: CN=kube-apiserver
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
                Public-Key: (2048 bit)
                Modulus:
                    00:d9:69:38:80:68:3b:b7:2e:9e:25:00:e8:fd:01:

                Exponent: 65537 (0x10001)
        X509v3 extensions:
            X509v3 Key Usage: critical
                Digital Signature, Key Encipherment
            X509v3 Extended Key Usage:
                TLS Web Server Authentication
            X509v3 Subject Alternative Name:
                DNS:master, DNS:kubernetes, DNS:kubernetes.default,
DNS:kubernetes.default.svc, DNS:kubernetes.default.svc.cluster.local, IP
Address:10.96.0.1, IP Address:172.17.0.27
```

KODEKLOUD

kubeadm

Certificate Path	CN Name	ALT Names	Organization	Issuer	Expiration
/etc/kubernetes/pki/apiserver.crt	kube-apiserver	DNS:master DNS:kubernetes DNS:kubernetes.default DNS:kubernetes.default.svc IP Address:10.96.0.1 IP Address:172.17.0.27		kubernetes	Feb 11 05:39:20 2020
/etc/kubernetes/pki/apiserver.key					
/etc/kubernetes/pki/ca.crt	kubernetes			kubernetes	Feb 8 05:39:19 2029
/etc/kubernetes/pki/apiserver-kubelet-client.crt	kube-apiserver-kubelet-client		system:masters	kubernetes	Feb 11 05:39:20 2020
/etc/kubernetes/pki/apiserver-kubelet-client.key					
/etc/kubernetes/pki/apiserver-etcd-client.crt	kube-apiserver-etcd-client		system:masters	self	Feb 11 05:39:22 2020
/etc/kubernetes/pki/apiserver-etcd-client.key					
/etc/kubernetes/pki/etcd/ca.crt	kubernetes			kubernetes	Feb 8 05:39:21 2017

KODEKLOUD

Default CN	Parent CA	O (in Subject)	kind	hosts (SAN)
kube-etcd	etcd-ca		server, client [1][etcdbug]	`localhost`, `127.0.0.1`
kube-etcd-peer	etcd-ca		server, client	`<hostname>`, `<Host_IP>`, `localhost`, `127.0.0.1`
kube-etcd-healthcheck-client	etcd-ca		client	
kube-apiserver-etcd-client	etcd-ca	system:masters	client	
kube-apiserver	kubernetes-ca		server	`<hostname>`, `<Host_IP>`, `<advertise_IP>`, `[1]`
kube-apiserver-kubelet-client	kubernetes-ca	system:masters	client	
front-proxy-client	kubernetes-front-proxy-ca		client	

Default CN	recommend key path	recommended cert path	command	key argument	cert argument
etcd-ca		etcd/ca.crt	kube-apiserver		−etcd-cafile
etcd-client	apiserver-etcd-client.key	apiserver-etcd-client.crt	kube-apiserver	−etcd-keyfile	−etcd-certfile
kubernetes-ca		ca.crt	kube-apiserver		−client-ca-file
kube-apiserver	apiserver.key	apiserver.crt	kube-apiserver	−tls-private-key-file	−tls-cert-file
apiserver-kubelet-client		apiserver-kubelet-client.crt	kube-apiserver		−kubelet-client-certificate
front-proxy-ca		front-proxy-ca.crt	kube-apiserver		−requestheader-client-ca-file
front-proxy-client	front-proxy-client.key	front-proxy-client.crt	kube-apiserver	−proxy-client-key-file	−proxy-client-cert-file
etcd-ca		etcd/ca.crt	etcd		−trusted-ca-file, −peer-trusted-ca-file
kube-etcd	etcd/server.key	etcd/server.crt	etcd	−key-file	−cert-file
kube-etcd-peer	etcd/peer.key	etcd/peer.crt	etcd	−peer-key-file	−peer-cert-file
etcd-ca		etcd/ca.crt	etcdctl[2]		−cacert
kube-etcd-healthcheck-client	etcd/healthcheck-client.key	etcd/healthcheck-client.crt	etcdctl[2]	−key	−cert

Inspect Service Logs

```
journalctl -u etcd.service -l
```

```
2019-02-13 02:53:28.144631 I | etcdmain: etcd Version: 3.2.18
2019-02-13 02:53:28.144680 I | etcdmain: Git SHA: eddf599c6
2019-02-13 02:53:28.144684 I | etcdmain: Go Version: go1.8.7
2019-02-13 02:53:28.144688 I | etcdmain: Go OS/Arch: linux/amd64
2019-02-13 02:53:28.144692 I | etcdmain: setting maximum number of CPUs to 4, total number of available CPUs is 4
2019-02-13 02:53:28.144734 N | etcdmain: the server is already initialized as member before, starting as etcd
member...
2019-02-13 02:53:28.146625 I | etcdserver: name = master
2019-02-13 02:53:28.146637 I | etcdserver: data dir = /var/lib/etcd
2019-02-13 02:53:28.146642 I | etcdserver: member dir = /var/lib/etcd/member
2019-02-13 02:53:28.146645 I | etcdserver: heartbeat = 100ms
2019-02-13 02:53:28.146648 I | etcdserver: election = 1000ms
2019-02-13 02:53:28.146651 I | etcdserver: snapshot count = 10000
2019-02-13 02:53:28.146677 I | etcdserver: advertise client URLs = 2019-02-13 02:53:28.185353 I | etcdserver/api:
enabled capabilities for version 3.2
2019-02-13 02:53:28.185588 I | embed: ClientTLS: cert = /etc/kubernetes/pki/etcd/server.crt, key =
/etc/kubernetes/pki/etcd/server.key, ca = , trusted-ca = /etc/kubernetes/pki/etcd/old-ca.crt, client-cert-auth =
true
2019-02-13 02:53:30.080017 I | embed: ready to serve client requests
2019-02-13 02:53:30.080130 I | etcdserver: published {Name:master ClientURLs:[https://127.0.0.1:2379]} to cluster
c9be114fc2da2776
2019-02-13 02:53:30.080281 I | embed: serving client requests on 127.0.0.1:2379
WARNING: 2019/02/13 02:53:30 Failed to dial 127.0.0.1:2379: connection error: desc = "transport: authentication
handshake failed: remote error: tls: bad certificate"; please retry.
```

KODEKLOUD

View Logs

```
kubectl logs etcd-master
```

```
2019-02-13 02:53:28.144631 I | etcdmain: etcd Version: 3.2.18
2019-02-13 02:53:28.144680 I | etcdmain: Git SHA: eddf599c6
2019-02-13 02:53:28.144684 I | etcdmain: Go Version: go1.8.7
2019-02-13 02:53:28.144688 I | etcdmain: Go OS/Arch: linux/amd64
2019-02-13 02:53:28.144692 I | etcdmain: setting maximum number of CPUs to 4, total number of available CPUs is 4
2019-02-13 02:53:28.144734 N | etcdmain: the server is already initialized as member before, starting as etcd member...
2019-02-13 02:53:28.146625 I | etcdserver: name = master
2019-02-13 02:53:28.146637 I | etcdserver: data dir = /var/lib/etcd
2019-02-13 02:53:28.146642 I | etcdserver: member dir = /var/lib/etcd/member
2019-02-13 02:53:28.146645 I | etcdserver: heartbeat = 100ms
2019-02-13 02:53:28.146648 I | etcdserver: election = 1000ms
2019-02-13 02:53:28.146651 I | etcdserver: snapshot count = 10000
2019-02-13 02:53:28.146677 I | etcdserver: advertise client URLs = 2019-02-13 02:53:28.185353 I | etcdserver/api:
enabled capabilities for version 3.2
2019-02-13 02:53:28.185588 I | embed: ClientTLS: cert = /etc/kubernetes/pki/etcd/server.crt, key =
/etc/kubernetes/pki/etcd/server.key, ca = , trusted-ca = /etc/kubernetes/pki/etcd/old-ca.crt, client-cert-auth =
true
2019-02-13 02:53:30.080017 I | embed: ready to serve client requests
2019-02-13 02:53:30.080130 I | etcdserver: published {Name:master ClientURLs:[https://127.0.0.1:2379]} to cluster
c9be114fc2da2776
2019-02-13 02:53:30.080281 I | embed: serving client requests on 127.0.0.1:2379
WARNING: 2019/02/13 02:53:30 Failed to dial 127.0.0.1:2379: connection error: desc = "transport: authentication
handshake failed: remote error: tls: bad certificate"; please retry.
```

View Logs

```
▶ docker ps -a
```

```
CONTAINER ID                        STATUS                  NAMES
23482a09f25b        Up 12 minutes           k8s_kube-apiserver_kube-apiserver-master_kube-system_8758a3d10776bb527e043f
b9bf77348c96        Up 18 minutes           k8s_etcd_etcd-master_kube-system_2cc1c8a24b68ab9b46bca47e153e74c6_0
87fc69913973        Up 18 minutes           k8s_POD_etcd-master_kube-system_2cc1c8a24b68ab9b46bca47e153e74c6_0
fda322157b86        Exited (255) 18 minutes ago    k8s_kube-apiserver_kube-apiserver-master_kube-system_8758a3d10776bb527e043f
0794bdfd57d8        Up 40 minutes           k8s_kube-scheduler_kube-scheduler-master_kube-system_009228e74aef4d7babd796
00f3f95d2102        Up 40 minutes           k8s_kube-controller-manager_kube-controller-manager-master_kube-system_ac1
b8e6a0e173dd        Up About an hour        k8s_weave_weave-net-8dzwb_kube-system_22cd7993-2f2d-11e9-a2a6-0242ac110021
18e47bad320e        Up About an hour        k8s_weave-npc_weave-net-8dzwb_kube-system_22cd7993-2f2d-11e9-a2a6-0242ac110
4d087daf0380        Exited (1) About an hour ago    k8s_weave_weave-net-8dzwb_kube-system_22cd7993-2f2d-11e9-a2a6-0242ac110021
e923140101a3        Up About an hour        k8s_kube-proxy_kube-proxy-cdmlz_kube-system_22cd267f-2f2d-11e9-a2a6-0242ac1
e0db7e63d18e        Up About an hour        k8s_POD_weave-net-8dzwb_kube-system_22cd7993-2f2d-11e9-a2a6-0242ac110021_0
74c257366f65        Up About an hour        k8s_POD_kube-proxy-cdmlz_kube-system_22cd267f-2f2d-11e9-a2a6-0242ac110021_0
8f514eac9d04        Exited (255) 40 minutes ago    k8s_kube-controller-manager_kube-controller-manager-master_kube-system_ac1
b39c5c594913        Exited (1) 40 minutes ago    k8s_kube-scheduler_kube-scheduler-master_kube-system_009228e74aef4d7babd796
3aefcb20ed30        Up 2 hours              k8s_POD_kube-apiserver-master_kube-system_8758a3d10776bb527e043fccfc835986_
576c8a273b50        Up 2 hours              k8s_POD_kube-controller-manager-master_kube-system_ac1d4c5ae0fbe553b664a6c2
4b3c5f34efde        Up 2 hours              k8s_POD_kube-scheduler-master_kube-system_009228e74aef4d7babd7968782118d5e_
```

View Logs

```
▶  docker logs 87fc
```

```
2019-02-13 02:53:28.144631 I | etcdmain: etcd Version: 3.2.18
2019-02-13 02:53:28.144680 I | etcdmain: Git SHA: eddf599c6
2019-02-13 02:53:28.144684 I | etcdmain: Go Version: go1.8.7
2019-02-13 02:53:28.144688 I | etcdmain: Go OS/Arch: linux/amd64
2019-02-13 02:53:28.144692 I | etcdmain: setting maximum number of CPUs to 4, total number of available CPUs is 4
2019-02-13 02:53:28.144734 N | etcdmain: the server is already initialized as member before, starting as etcd
member...
2019-02-13 02:53:28.146625 I | etcdserver: name = master
2019-02-13 02:53:28.146637 I | etcdserver: data dir = /var/lib/etcd
2019-02-13 02:53:28.146642 I | etcdserver: member dir = /var/lib/etcd/member
2019-02-13 02:53:28.146645 I | etcdserver: heartbeat = 100ms
2019-02-13 02:53:28.146648 I | etcdserver: election = 1000ms
2019-02-13 02:53:28.146651 I | etcdserver: snapshot count = 10000
2019-02-13 02:53:28.146677 I | etcdserver: advertise client URLs = 2019-02-13 02:53:28.185353 I | etcdserver/api:
enabled capabilities for version 3.2
2019-02-13 02:53:28.185588 I | embed: ClientTLS: cert = /etc/kubernetes/pki/etcd/server.crt, key =
/etc/kubernetes/pki/etcd/server.key, ca = , trusted-ca = /etc/kubernetes/pki/etcd/old-ca.crt, client-cert-auth =
true
2019-02-13 02:53:30.080017 I | embed: ready to serve client requests
2019-02-13 02:53:30.080130 I | etcdserver: published {Name:master ClientURLs:[https://127.0.0.1:2379]} to cluster
c9be114fc2da2776
2019-02-13 02:53:30.080281 I | embed: serving client requests on 127.0.0.1:2379
WARNING: 2019/02/13 02:53:30 Failed to dial 127.0.0.1:2379: connection error: desc = "transport: authentication
handshake failed: remote error: tls: bad certificate"; please retry.
```

KODEKLOUD

TLS CERTIFICATES

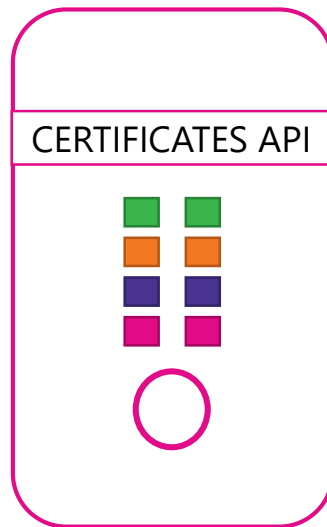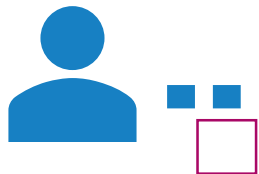## Certificate Workflow & API
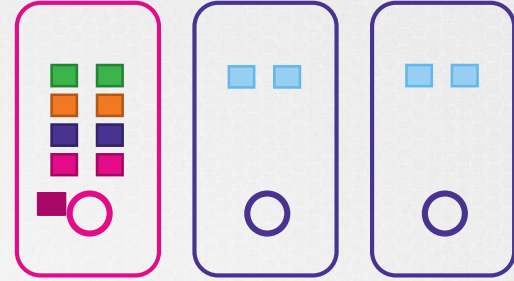
CERTIFICATE AUTHORITY (CA)
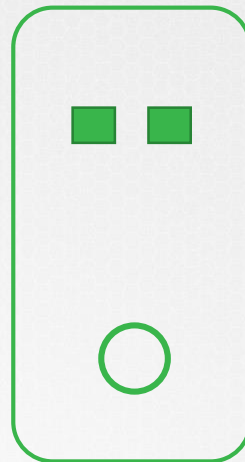
1. Create CertificateSigningRequest Object

2. Review Requests

3. Approve Requests

4. Share Certs to Users
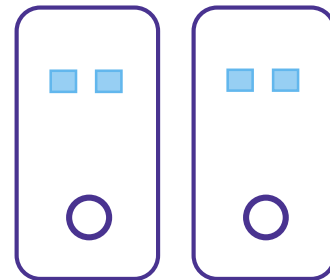
CERTIFICATES API

```
▶ openssl genrsa -out jane.key 2048
jane.key
```

```
▶ openssl req -new -key jane.key -subj  "/CN=jane" -out jane.csr
jane.csr


-----BEGIN CERTIFICATE REQUEST-----
MIICWDCCAUACAQAwEzERMA8GA1UEAwwIbmV3LXVzZXIwggEiMA0GCSqGSIb3DQEB
AQUAA4IBDwAwggEKAoIBAQDO0WJW+DXsAJSIrjpNo5vRIBplnzg+6xc9+UVwkKi0
LfC27t+1eEnON5Muq99NevmMEOnrDUO/thyVqP2w2XNIDRXjYyF40FbmD+5zWyCK
9w0BAQsFAAOCAQEAS9iS6C1uxTuf5BBYSU7QFQHUzalNxAdYsaORRQNwHZwHqGi4
hOK4a2zyNyi44OOijyaD6tUW8DSxkr8BLK8Kg3srREtJql5rLZy9LRVrsJghD4gY
P9NL+aDRSxROVSqBaB2nWeYpM5cJ5TF53lesNSNMLQ2++RMnjDQJ7juPEic8/dhk
Wr2EUM6UawzykrdHImwTv2mlMY0R+DNtV1Yie+0H9/YElt+FSGjh5L5YUvI1Dqiy
4l3E/y3qL71WfAcuH3OsVpUUnQISMdQs0qWCsbE56CC5DhPGZIpUbnKUpAwka+8E
vwQ07jG+hpknxmuFAeXxgUwodALaJ7ju/TDIcw==
-----END CERTIFICATE REQUEST-----
```

Name of User

jane.csr

-----BEGIN CERTIFICATE REQUEST-----
MIICWDCCAUACAQAwEzERMA8GA1UEAwwIbmV3LXVzZXIwggEiMA0GCSqGSIb3DQEB
AQUAA4IBDwAwggEKAoIBAQDO0WJW+DXsAJSIrjpNo5vRIBplnzg+6xc9+UVwkKi0
LfC27t+1eEnON5Muq99NevmMEOnrDUO/thyVqP2w2XNIDRXjYyF40FbmD+5zWyCK
9w0BAQsFAAOCAQEAS9iS6C1uxTuf5BBYSU7QFQHUzalNxAdYsaORRQNwHZwHqGi4
hOK4a2zyNyi44OOijyaD6tUW8DSxkr8BLK8Kg3srREtJql5rLZy9LRVrsJghD4gY
P9NL+aDRSxROVSqBaB2nWeYpM5cJ5TF53lesNSNMLQ2++RMnjDQJ7juPEic8/dhk
Wr2EUM6UawzykrdHImwTv2mlMY0R+DNtV1Yie+0H9/YElt+FSGjh5L5YUvI1Dqiy
4l3E/y3qL71WfAcuH3OsVpUUnQISMdQs0qWCsbE56CC5DhPGZIpUbnKUpAwka+8E
vwQ07jG+hpknxmuFAeXxgUwodALaJ7ju/TDIcw==
-----END CERTIFICATE REQUEST-----

jane-csr.yaml

```yaml
apiVersion: certificates.k8s.io/v1beta1
kind: CertificateSigningRequest
metadata:
  name: jane
spec:
  groups:
  - system:authenticated
  usages:
  - digital signature
  - key encipherment
  - server auth
  request:
```

cat jane.csr | base64

LS0tLS1CRUdJTiBDRVJUSUZJQ0FURSBSRVFVRVNULS0
tLS0KTU1JQ1dEQ0NBVUFDQVFBd0V6RVJNQThHQTFVRU
F3d01ibVYzTFhWelZYSXdnZ0VpTUEwR0NTSWIzR
FFFQgpBVVVBQTRJQkR3QXdnZ0VLQW9JQkFRRE8wV0pX
K0RYc0FKU01yanBObzV2Uk1CcGxuemcrNnhjOStVVnd
rS2kwTFk1TXVxOTlOZXZtTUVPbnJ
JnhjOStVVndrS2kwCkxmQzI3dCsxZUVuT0
41TXVxOTlOZXZtTUVPbnJ

```
kubectl get csr

NAME              AGE       REQUESTOR              CONDITION
jane              10m       admin@example.com        Pending
```

```
kubectl certificate approve jane

jane approved!
```

kubectl get csr jane -o yaml

```
apiVersion: certificates.k8s.io/v1beta1
kind: CertificateSigningRequest
metadata:
  creationTimestamp: 2019-02-13T16:36:43Z
  name: new-user
spec:
  groups:
  - system:masters
  - system:authenticated
  usages:
  - digital signature
  - key encipherment
  - server auth
  username: kubernetes-admin
status:
  certificate:
  LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0tCk1JSURDakNDQWZLZ0F3SUJBZ0lVRmwy
  Q2wxYXoxaWl5M3JNVisreFRYQUowU3dnd0RRWUpLb1pJaHZjTkFRRUwKQlFBd0ZURVRN
  QkVHQTFFVUF4TUthM1ZpWlhKdVpYUmxjekFlRncweE9UQXlNVE14TmpNeU1EQmFGd1dn
  Y0ZFDl2ajNuSXY3eFdDDS1NIRm5sU041c0t5Z0VxUkwzTFM5V29GelhHZDdWCmlEZ2FO
  MVVRMFBXTVhjN09FVnVjSWc1Yk4weEVHTkvWRU5tdUlBNlZZWeHVjS1h6aG9ldDY0MEd1
  MGU0YXFKWVIKWmVMVMbjBvRTFCY3dod2xic0I1ND0KLS0tLS1FTkQgQ0VSVElGSUNBVEUt
  LS0tLQo=
  conditions:
  - lastUpdateTime: 2019-02-13T16:37:21Z
    message: This CSR was approved by kubectl certificate approve.
    reason: KubectlApprove
    type: Approved
```

echo "LS0…Qo=" | base64 --decode

```
-----BEGIN CERTIFICATE -----
MIICWDCCAUACAQAwEzERMA8GA1UEAwwIbmV3LXVzZXIwgg
AQUAA4IBDwAwggEKAoIBAQDO0WJW+DXsAJSIrjpNo5vRIB
LfC27t+1eEnON5Muq99NevmMEOnrDUO/thyVqP2w2XNIDR
y3BihhB93MJ7Oql3UTvZ8TELqyaDknRl/jv/SxgXkok0AB
IF5nxAttMVkDPQ7NbeZRG43b+QWlVGR/z6DWOfJnbfezOt
EcCXAwqChjBLkz2BHPR4J89D6Xb8k39pu6jpyngV6uP0tI
j2qEL+hZEWkkFz80lNNtyT5LxMqENDCnIgwC4GZiRGbrAg
9w0BAQsFAAOCAQEAS9iS6C1uxTuf5BBYSU7QFQHUzalNxA
hOK4a2zyNyi44OOijyaD6tUW8DSxkr8BLK8Kg3srREtJql
P9NL+aDRSxROVSqBaB2nWeYpM5cJ5TF53lesNSNMLQ2++R
Wr2EUM6UawzykrdHImwTv2mlMY0R+DNtV1Yie+0H9/YElt
4l3E/y3qL71WfAcuH3OsVpUUnQISMdQs0qWCsbE56CC5Dh
vwQ07jG+hpknxmuFAeXxgUwodALaJ7ju/TDIcw==
-----END CERTIFICATE -----
```

KODEKLOUD
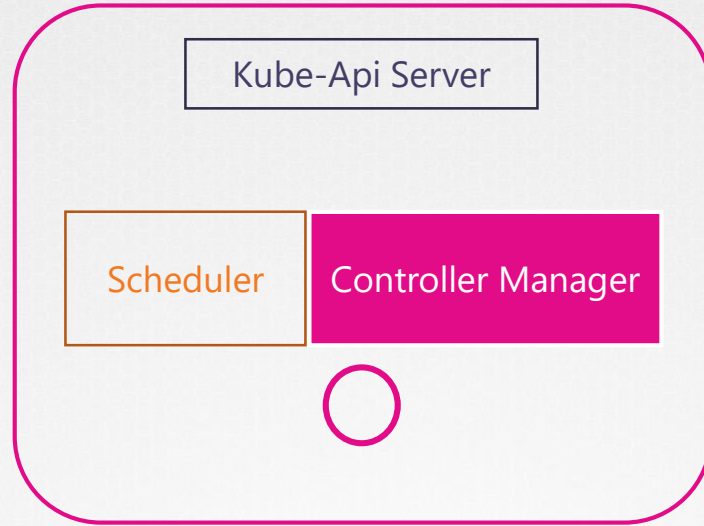
```
▶   cat /etc/kubernetes/manifests/kube-controller-manager.yaml
spec:
  containers:
  - command:
    - kube-controller-manager
    - --address=127.0.0.1
    - --cluster-signing-cert-file=/etc/kubernetes/pki/ca.crt
    - --cluster-signing-key-file=/etc/kubernetes/pki/ca.key
    - --controllers=*,bootstrapsigner,tokencleaner
    - --kubeconfig=/etc/kubernetes/controller-manager.conf
    - --leader-elect=true
    - --root-ca-file=/etc/kubernetes/pki/ca.crt
    - --service-account-private-key-file=/etc/kubernetes/pki/sa.key
    - --use-service-account-credentials=true
```

KODEKLOUD

Provision Certificate Authority

kube-apiserver

ca-key.pem

kubernetes-key.pem

ca.pem

kubernetes.pem

admin user

kube-controller-manager

kube-scheduler

Kube-proxy

kubelet

admin-key.pem

kcm-key.pem

Kube-scheduler-key.pem

kube-proxy-key.pem

worker01-key.pem

admin.pem

kcm.pem

Kube-scheduler.pem

kube-proxy.pem

worker01.pem

worker02-key.pem

worker02.pem

Default CN	Parent CA	O (in Subject)	kind	hosts (SAN)
kube-etcd	etcd-ca		server, client [1][etcdbug]	`localhost`, `127.0.0.1`
kube-etcd-peer	etcd-ca		server, client	`<hostname>`, `<Host_IP>`, `localhost`, `127.0.0.1`
kube-etcd-healthcheck-client	etcd-ca		client	
kube-apiserver-etcd-client	etcd-ca	system:masters	client	
kube-apiserver	kubernetes-ca		server	`<hostname>`, `<Host_IP>`, `<advertise_IP>`, `[1]`
kube-apiserver-kubelet-client	kubernetes-ca	system:masters	client	
front-proxy-client	kubernetes-front-proxy-ca		client	

Default CN	recommend key path	recommended cert path	command	key argument	cert argument
etcd-ca		etcd/ca.crt	kube-apiserver		−etcd-cafile
etcd-client	apiserver-etcd-client.key	apiserver-etcd-client.crt	kube-apiserver	−etcd-keyfile	−etcd-certfile
kubernetes-ca		ca.crt	kube-apiserver		−client-ca-file
kube-apiserver	apiserver.key	apiserver.crt	kube-apiserver	−tls-private-key-file	−tls-cert-file
apiserver-kubelet-client		apiserver-kubelet-client.crt	kube-apiserver		−kubelet-client-certificate
front-proxy-ca		front-proxy-ca.crt	kube-apiserver		−requestheader-client-ca-file
front-proxy-client	front-proxy-client.key	front-proxy-client.crt	kube-apiserver	−proxy-client-key-file	−proxy-client-cert-file
etcd-ca		etcd/ca.crt	etcd		−trusted-ca-file, −peer-trusted-ca-file
kube-etcd	etcd/server.key	etcd/server.crt	etcd	−key-file	−cert-file
kube-etcd-peer	etcd/peer.key	etcd/peer.crt	etcd	−peer-key-file	−peer-cert-file
etcd-ca		etcd/ca.crt	etcdctl[2]		−cacert
kube-etcd-healthcheck-client	etcd/healthcheck-client.key	etcd/healthcheck-client.crt	etcdctl[2]	−key	−cert

Configure certificates for user accounts

You must manually configure these administrator account and service accounts:

filename	credential name	Default CN	O (in Subject)
admin.conf	default-admin	kubernetes-admin	system:masters
kubelet.conf	default-auth	system:node: **<nodeName>** (see note)	system:nodes
controller-manager.conf	default-controller-manager	system:kube-controller-manager	
scheduler.conf	default-manager	system:kube-scheduler	

Component	Certificate	Purpose
API server	Cluster CA	Authenticate clients, TLS
API server	Etcd CA	Etcd server authentication
API server	Etcd client cert	Etcd client authentication
API server	Serving certificate	Serving API over HTTPS
API server	Kubelet client cert	Authenticating against Kubelet
Controller Manager	Client certificate	Authenticating against API server
Controller Manager	Cluster CA	Embedding in service account secrets
Scheduler	Client certificate	Authenticating against API server
Kubelet	Serving certificate	Serving API over HTTPS
Kubelet	Client certificate	Authenticating against API server
Kubelet	Cluster CA	Authenticating clients
Kube Proxy	Client certificate	Authenticating against API server

```
- kube-apiserver
  - --authorization-mode=Node,RBAC
  - --advertise-address=172.17.0.18
  - --allow-privileged=true
  - --client-ca-file=/etc/kubernetes/pki/ca.crt
  - --disable-admission-plugins=PersistentVolumeLabel
  - --enable-admission-plugins=NodeRestriction
  - --enable-bootstrap-token-auth=true
  - --etcd-cafile=/etc/kubernetes/pki/etcd/ca.crt
  - --etcd-certfile=/etc/kubernetes/pki/apiserver-etcd-client.crt
  - --etcd-keyfile=/etc/kubernetes/pki/apiserver-etcd-client.key
  - --etcd-servers=https://127.0.0.1:2379
  - --insecure-port=0
  - --kubelet-client-certificate=/etc/kubernetes/pki/apiserver-kubelet-client.crt
  - --kubelet-client-key=/etc/kubernetes/pki/apiserver-kubelet-client.key
  - --kubelet-preferred-address-types=InternalIP,ExternalIP,Hostname
  - --proxy-client-cert-file=/etc/kubernetes/pki/front-proxy-client.crt
  - --proxy-client-key-file=/etc/kubernetes/pki/front-proxy-client.key
  - --requestheader-allowed-names=front-proxy-client
  - --requestheader-client-ca-file=/etc/kubernetes/pki/front-proxy-ca.crt
  - --requestheader-extra-headers-prefix=X-Remote-Extra-
  - --requestheader-group-headers=X-Remote-Group
  - --requestheader-username-headers=X-Remote-User
  - --secure-port=6443
  - --service-account-key-file=/etc/kubernetes/pki/sa.pub
  - --service-cluster-ip-range=10.96.0.0/12
  - --tls-cert-file=/etc/kubernetes/pki/apiserver.crt
  - --tls-private-key-file=/etc/kubernetes/pki/apiserver.key
```

```
openssl genrsa -out old-ca.key 2048
openssl req -new -key old-ca.key -subj "/CN=old-ca" -out old-ca.csr
openssl x509 -req -in old-ca.csr -signkey old-ca.key -out old-ca.crt -days 365


openssl x509 -req -in ca.csr -signkey ca.key -out server.crt -days 365


openssl req -new -key apiserver-kubelet-client.key -out apiserver-kubelet-client.csr -subj "/CN=kube-apiserver-kubelet-client/O=system:masters"


openssl req -new -key apiserver-kubelet-client.key -out apiserver-kubelet-client.csr -subj "/CN=kube-apiserver-kubelet-client/O=system:masters"
openssl x509 -req -in apiserver-kubelet-client.csr -CA /root/new-ca/old-ca.crt -CAkey /root/new-ca/old-ca.key -CAcreateserial -out apiserver-kubelet-client-new.crt -days 365


openssl req -new -key apiserver-etcd-client.key -out apiserver-etcd-client.csr -subj "/CN=kube-apiserver-etcd-client/O=system:masters"
openssl x509 -req -in apiserver-etcd-client.csr -CA /root/new-ca/old-ca.crt -CAkey /root/new-ca/old-ca.key -CAcreateserial -out apiserver-etcd-client-new.crt -days 365


openssl req -new -key apiserver-etcd-client.key -out apiserver-etcd-client.csr -subj "/CN=kube-apiserver-etcd-client/O=system:masters"
openssl x509 -req -in apiserver-etcd-client.csr -CA /root/new-ca/old-ca.crt -CAkey /root/new-ca/old-ca.key -CAcreateserial -out apiserver-etcd-client-new.crt -days 365


openssl req -new -key /etc/kubernetes/pki/apiserver-etcd-client.key -out apiserver-etcd-client.csr -subj "/CN=kube-apiserver-etcd-client/O=system:masters"

openssl x509 -req -in apiserver-etcd-client.csr -CA ca.crt -CAkey ca.key -CAcreateserial -out apiserver-etcd-client.crt -days -10

openssl x509 -req -in apiserver-etcd-client.csr -CA ca.crt -CAkey ca.key -CAcreateserial -out apiserver-etcd-client.crt -startdate 190101010101Z

20170101000000Z
200801010000Z


"openssl", "req", "-new", "-key" ,"/etc/kubernetes/pki/apiserver-etcd-client.key", "-out", "/etc/kubernetes/pki/apiserver-etcd-client.csr", "-subj", "/CN=kube-apiserver-etcd-client/O=system:masters"

"openssl", "x509", "-req", "-in", "/etc/kubernetes/pki/apiserver-etcd-client.csr", "-CA", "/etc/kubernetes/pki/etcd/ca.crt", "-CAkey", "/etc/kubernetes/pki/etcd/ca.key", "-CAcreateserial", "-out", "/etc/kubernetes/pki/apiserver-etcd-client.crt"

openssl x509 -req -in /etc/kubernetes/pki/apiserver-etcd-client.csr -CA /etc/kubernetes/pki/etcd/ca.crt -CAkey /etc/kubernetes/pki/etcd/ca.key -CAcreateserial -out /etc/kubernetes/pki/apiserver-etcd-client.crt -days 100


openssl x509 -req -in apiserver.csr -CA ca.crt -CAkey ca.key -CAcreateserial -out apiserver.crt
```

Security
KUBECONFIG

```
curl https://my-kube-playground:6443/api/v1/pods \
    --key admin.key
    --cert admin.crt
    --cacert ca.crt
```

```
{
  "kind": "PodList",
  "apiVersion": "v1",
  "metadata": {
    "selfLink": "/api/v1/pods",
  },
  "items": []
}
```

```
kubectl get pods
      --server my-kube-playground:6443
      --client-key admin.key
      --client-certificate admin.crt
      --certificate-authority ca.crt
```
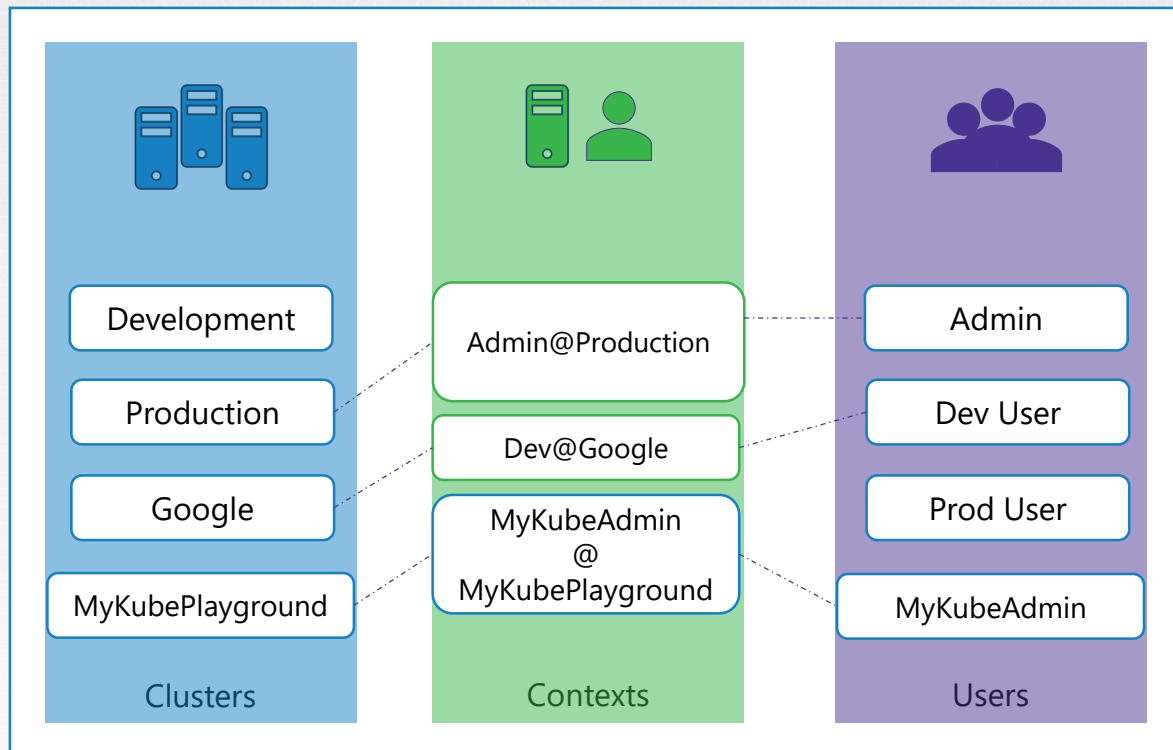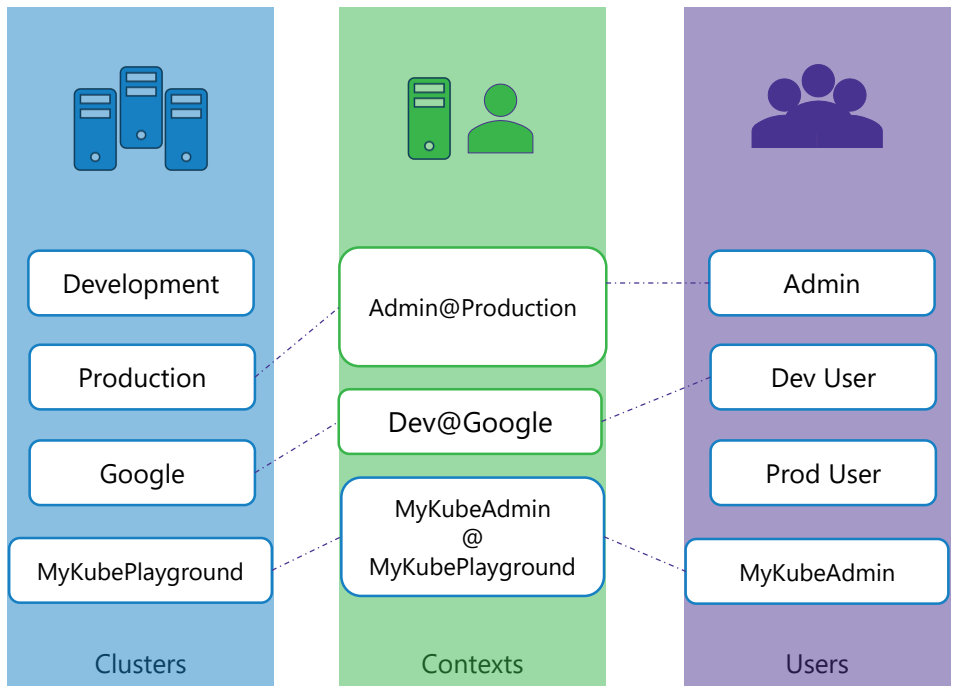```
No resources found.
```

$HOME/.kube/config

config

## KubeConfig File

```
--server my-kube-playground:6443
--client-key admin.key
--client-certificate admin.crt
--certificate-authority ca.crt
```

▶ kubectl get pods
    --kubeconfig config

No resources found.

KubeConfig File

$HOME/.kube/config

```
--server my-kube-playground:6443
--client-key admin.key
--client-certificate admin.crt
--certificate-authority ca.crt
```


Clusters	Contexts	Users
Development	Admin@Production	Admin
Production	Dev@Google	Dev User
Google	MyKubeAdmin @ MyKubePlayground	Prod User
MyKubePlayground		MyKubeAdmin

KODEKLOUD

KubeConfig File

```
apiVersion: v1
kind: Config

clusters:

- name: my-kube-playground
  cluster:
    certificate-authority: ca.crt
    server: https://my-kube-playground:6443


contexts:

- name: my-kube-admin@my-kube-playground
  context:
    cluster:
    user:


users:

- name: my-kube-admin
  user:
    client-certificate: admin.crt
    client-key: admin.key
```

$HOME/.kube/config

Clusters	Contexts	Users
Development	Admin@Production	Admin
Production	Dev@Google	Dev User
Google		Prod User
MyKubePlayground	MyKubeAdmin @ MyKubePlayground	MyKubeAdmin

KODEKLOUD

KubeConfig File

```
apiVersion: v1
kind: Config

current-context: dev-user@google

clusters:
- name: my-kube-playground    (values hidden...)
- name: development
- name: production
- name: google

contexts:
- name: my-kube-admin@my-kube-playground
- name: dev-user@google
- name: prod-user@production

users:
- name: my-kube-admin
- name: admin
- name: dev-user
- name: prod-user
```

$HOME/.kube/config

| Development |
| Production |
| Google |
| MyKubePlayground |

Clusters

| Admin@Production |
| Dev@Google |
| MyKubeAdmin @ MyKubePlayground |

Contexts

| Admin |
| Dev User |
| Prod User |
| MyKubeAdmin |

Users

KODEKLOUD
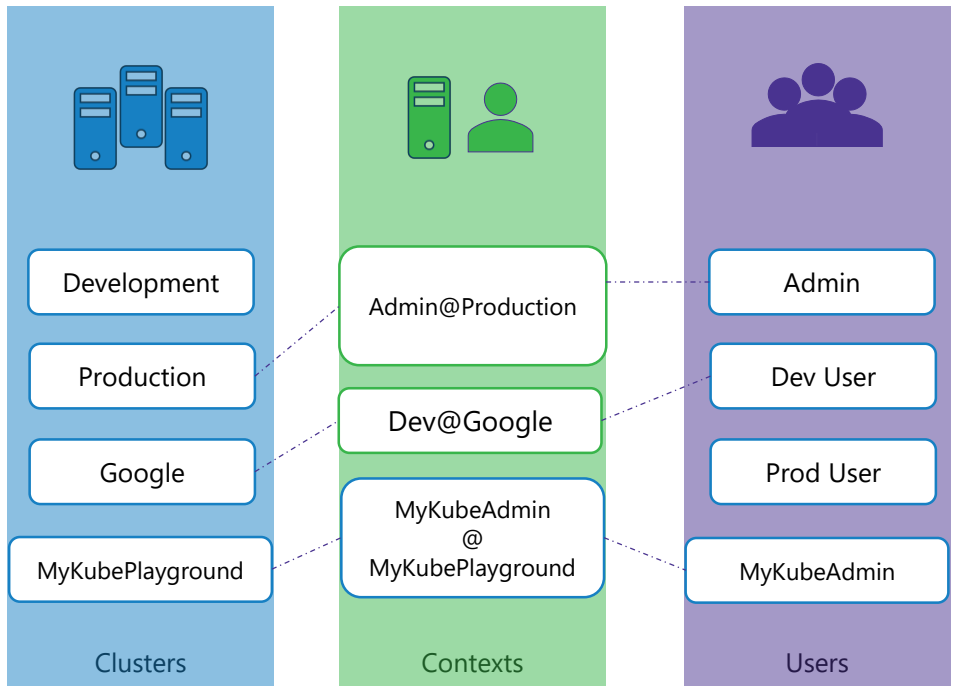
Kubectl config

```
▶  kubectl config view
```

```
apiVersion: v1

kind: Config
current-context: kubernetes-admin@kubernetes

clusters:
- cluster:
    certificate-authority-data: REDACTED
    server: https://172.17.0.5:6443
  name: kubernetes

contexts:
- context:
    cluster: kubernetes
    user: kubernetes-admin
  name: kubernetes-admin@kubernetes


users:
- name: kubernetes-admin
  user:
    client-certificate-data: REDACTED
    client-key-data: REDACTED
```

```
▶  kubectl config view –kubeconfig=my-custom-config
```

```
apiVersion: v1

kind: Config
current-context: my-kube-admin@my-kube-playground


clusters:
- name: my-kube-playground
- name: development
- name: production


contexts:
- name: my-kube-admin@my-kube-playground
- Name: prod-user@production


users:
- name: my-kube-admin
- name: prod-user
```

Kubectl config

```
apiVersion: v1

kind: Config
current-context: my-kube-admin@my-kube-playground


clusters:
- name: my-kube-playground
- name: development
- name: production


contexts:
-  name: my-kube-admin@my-kube-playground
-  Name: prod-user@production


users:
- name: my-kube-admin
- name: prod-user
```

```
apiVersion: v1

kind: Config
current-context: prod-user@production


clusters:
- name: my-kube-playground
- name: development
- name: production


contexts:
-  name: my-kube-admin@my-kube-playground
-  Name: prod-user@production


users:
- name: my-kube-admin
- name: prod-user
```

KODEKLOUD

Kubectl config

```
kubectl config -h

Available Commands:
  current-context Displays the current-context
  delete-cluster  Delete the specified cluster from the kubeconfig
  delete-context  Delete the specified context from the kubeconfig
  get-clusters    Display clusters defined in the kubeconfig
  get-contexts    Describe one or many contexts
  rename-context  Renames a context from the kubeconfig file.
  set             Sets an individual value in a kubeconfig file
  set-cluster     Sets a cluster entry in kubeconfig
  set-context     Sets a context entry in kubeconfig
  set-credentials Sets a user entry in kubeconfig
  unset           Unsets an individual value in a kubeconfig file
  use-context     Sets the current-context in a kubeconfig file
  view            Display merged kubeconfig settings or a specified kubeconfig file
```

KODEKLOUD

Namespaces

```
apiVersion: v1
kind: Config

clusters:
- name: production
  cluster:
    certificate-authority: ca.crt
    server: https://172.17.0.51:6443

contexts:
- name: admin@production
  context:
    cluster: production
    user: admin
    namespace: finance

users:
- name: admin
  user:
    client-certificate: admin.crt
    client-key: admin.key
```
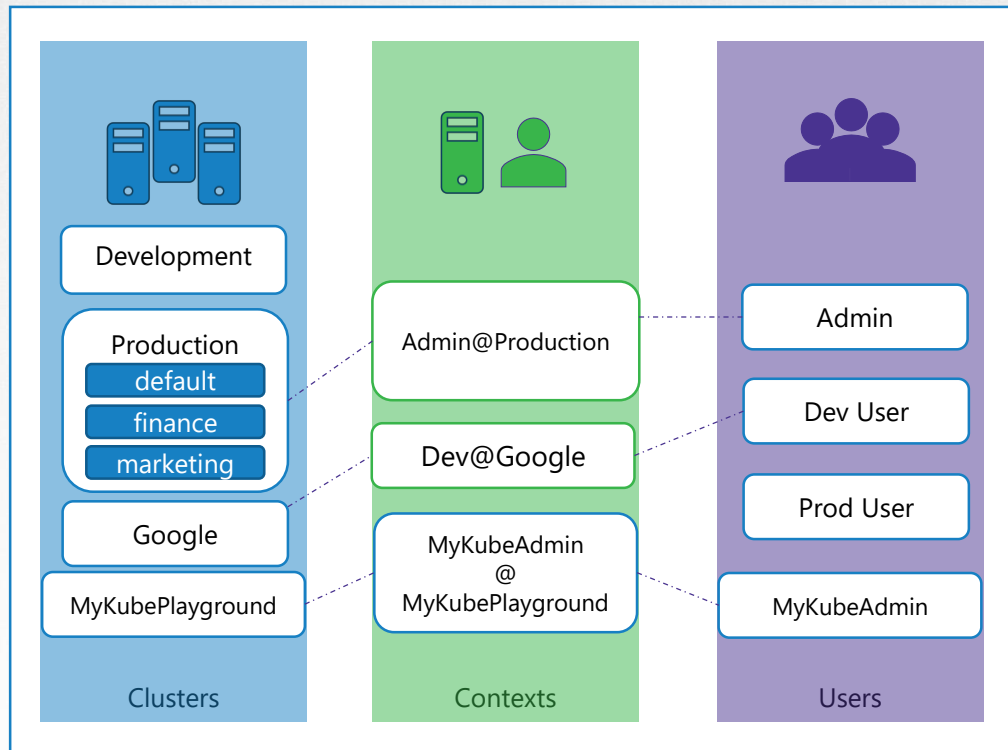
Certificates in KubeConfig

```yaml
apiVersion: v1
kind: Config

clusters:
- name: production
  cluster:
    certificate-authority: /etc/kubernetes/pki/ca.crt
    server: https://172.17.0.51:6443

contexts:
- name: admin@production
  context:
    cluster: production
    user: admin
    namespace: finance

users:
- name: admin
  user:
    client-certificate: /etc/kubernetes/pki/users/admin.crt
    client-key: /etc/kubernetes/pki/users/admin.key
```

Certificates in KubeConfig

```yaml
apiVersion: v1
kind: Config

clusters:
- name: production
  cluster:
    certificate-authority: /etc/kubernetes/pki/ca.crt

    certificate-authority-data:
```

-----BEGIN CERTIFICATE -----
MIICWDCCAUACAQAwEzERMA8GA1UEAwwIbmV3LXVzZXIIwggEiMA0GC
AQUAA4IBDwAwggEKAoIBAQDO0WJW+DXsAJSIrjpNo5vRIBplnzg+6
LfC27t+1eEnON5Muq99NevmMEOnrDUO/thyVqP2w2XNIDRXjYyF40
y3BihhB93MJ7Oql3UTvZ8TELqyaDknRl/jv/SxgXkok0ABUTpWMx4
IF5nxAttMVkDPQ7NbeZRG43b+QWlVGR/z6DWOfJnbfezOtaAydGLT
EcCXAwqChjBLkz2BHPR4J89D6Xb8k39pu6jpyngV6uP0tIbOzpqNv
j2qEL+hZEWkkFz80lNNtyT5LxMqENDCnIgwC4GZiRGbrAgMBAAGgA
9w0BAQsFAAOCAQEAS9iS6C1uxTuf5BBYSU7QFQHUzalNxAdYsaORR
hOK4a2zyNyi44OOijyaD6tUW8DSxkr8BLK8Kg3srREtJql5rLZy9L
P9NL+aDRSxROVSqBaB2nWeYpM5cJ5TF53lesNSNMLQ2++RMnjDQJ7
Wr2EUM6UawzykrdHImwTv2mlMY0R+DNtV1Yie+0H9/YElt+FSGjh5
4l3E/y3qL71WfAcuH3OsVpUUnQISMdQs0qWCsbE56CC5DhPGZIpUb
vwQ07jG+hpknxmuFAeXxgUwodALaJ7ju/TDIcw==
-----END CERTIFICATE -----

```
▶ cat ca.crt | base64
```

LS0tLS1CRUdJTiBDRVJUSUZJQ0FURSBSRVFVRVN
tLS0KTUlJQ1dEQ0NBVUFDQVFBd0V6RVJNQThHQT
F3d0libVYzTFhWelpYSXdnZ0VpTUEwR0NTcUdTI
FFFQgpBUVVBQTRJQkR3QXdnZ0VLQW9JQkFRRE8w
K0RYc0FFU0lyanBObzV2UklCcGxuemcrNnhpOSt
rS2kwCkxmQzI3dCsxZUVuT041TXVxOTlOZXZtTU
J

API Groups

Pre-Requisite

```
curl https://kube-master:6443/version
```

```
{
  "major": "1",
  "minor": "13",
  "gitVersion": "v1.13.0",
  "gitCommit": "ddf47ac13c1a9483ea035a79cd7c10005ff21a6d",
  "gitTreeState": "clean",
  "buildDate": "2018-12-03T20:56:12Z",
  "goVersion": "go1.11.2",
  "compiler": "gc",
  "platform": "linux/amd64"
}
```

```
curl https://kube-master:6443/api/v1/pods
```

```
{
  "kind": "PodList",
  "apiVersion": "v1",
  "metadata": {
    "selfLink": "/api/v1/pods",
    "resourceVersion": "153068"
  },
  "items": [
    {
      "metadata": {
        "name": "nginx-5c7588df-ghsbd",
        "generateName": "nginx-5c7588df-",
        "namespace": "default",
        "creationTimestamp": "2019-03-20T10:57:48Z",
        "labels": {
          "app": "nginx",
          "pod-template-hash": "5c7588df"
        },
        "ownerReferences": [
          {
            "apiVersion": "apps/v1",
            "kind": "ReplicaSet",
            "name": "nginx-5c7588df",
            "uid": "398ce179-4af9-11e9-beb6-020d3114c7a7",
            "controller": true,
            "blockOwnerDeletion": true
          }
        ]
      },
```

/metrics /healthz /version /api /apis /logs

core

named

/api

/apis

Pod v1 core

[kubectl example]   [curl example]

Group	Version
core	v1

⚠️ **Warning:**

It is recommended that users create Pods only through a Controller, and not directly. See Controllers: Deploy

ℹ️ Appears In:

- PodList [core/v1]

Field	Description
apiVersion string	APIVersion defines the versioned schema of this representation of an object. Servers should c https://git.k8s.io/community/contributors/devel/api-conventions.md#resources

```
 ▶  curl http://localhost:6443 -k
{
  "paths": [
    "/api",
    "/api/v1",
    "/apis",
    "/apis/",
    "/healthz",
    "/logs",
    "/metrics",
    "/openapi/v2",
    "/swagger-2.0.0.json",
```

```
 ▶  curl http://localhost:6443/apis -k | grep "name"
        "name": "extensions",
        "name": "apps",
        "name": "events.k8s.io",
        "name": "authentication.k8s.io",
        "name": "authorization.k8s.io",
        "name": "autoscaling",
        "name": "batch",
        "name": "certificates.k8s.io",
        "name": "networking.k8s.io",
        "name": "policy",
        "name": "rbac.authorization.k8s.io",
        "name": "storage.k8s.io",
        "name": "admissionregistration.k8s.io",
        "name": "apiextensions.k8s.io",
        "name": "scheduling.k8s.io",
```

Kube ApiServer

```
curl http://localhost:6443 -k
```

```
{
  "kind": "Status",
  "apiVersion": "v1",
  "metadata": {

  },
  "status": "Failure",
  "message": "forbidden: User \"system:anonymous\" cannot get path \"/\"",
  "reason": "Forbidden",
  "details": {

  },
  "code": 403
}
```
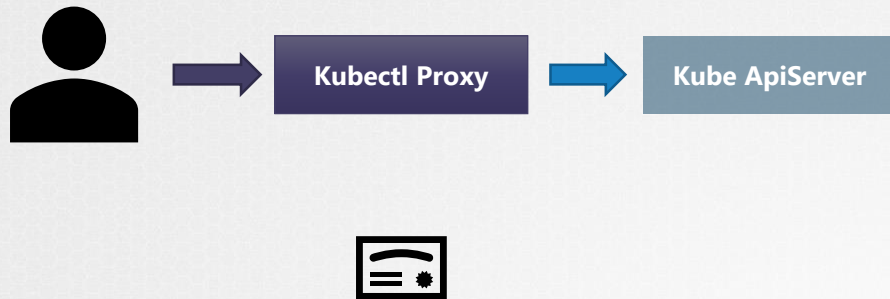
```
curl http://localhost:6443 –k
        --key admin.key
        --cert admin.crt
        --cacert ca.crt
```

```
{
  "paths": [
    "/api",
    "/api/v1",
    "/apis",
    "/apis/",
    "/healthz",
    "/logs",
    "/metrics"
```

kubectl proxy


```
▶ kubectl proxy
Starting to serve on 127.0.0.1:8001
```

```
▶ curl http://localhost:8001 -k
{
  "paths": [
    "/api",
    "/api/v1",
    "/apis",
    "/apis/",
    "/healthz",
    "/logs",
    "/metrics",
    "/openapi/v2",
    "/swagger-2.0.0.json",
```

Important

With this we need not
↑ to Provide Certificates
manually

Kube proxy ≠ **Kube**ctl **proxy**

KODEKLOUD

Key Takeaways

named

/apis

API Groups

/apps /extensions /networking.k8s.io /storage.k8s.io /authentication.k8s.io /certificates.k8s.io

/v1 /v1 /v1

Resources

/deployments
/replicasets
/statefulsets

/networkpolicies

/certificatesigningrequests

Verbs

list
get
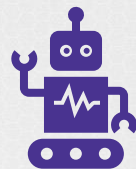create
delete
update
watch

KODEKLOUD

AUTHORIZATION

Why Authorization?

Admins

Developers

Bots

```
▶ kubectl get pods
NAME      STATUS  ROLES   AGE    VERSION
worker-1  Ready   <none>  5d21h  v1.13.0
worker-2  Ready   <none>  5d21h  v1.13.0
```

```
▶ kubectl get pods
NAME      STATUS  ROLES   AGE    VERSION
worker-1  Ready   <none>  5d21h  v1.13.0
worker-2  Ready   <none>  5d21h  v1.13.0
```

```
▶ kubectl get pods
Error from server (Forbidden): nodes
"worker-1" is forbidden: User "Bot-1"
delete resource "nodes"
```

```
▶ kubectl get nodes
NAME      STATUS  ROLES   AGE    VERSION
worker-1  Ready   <none>  5d21h  v1.13.0
worker-2  Ready   <none>  5d21h  v1.13.0
```

```
▶ kubectl get nodes
NAME      STATUS  ROLES   AGE    VERSION
worker-1  Ready   <none>  5d21h  v1.13.0
worker-2  Ready   <none>  5d21h  v1.13.0
```

```
▶ kubectl get nodes
Error from server (Forbidden): nodes
"worker-1" is forbidden: User "Bot-1"
delete resource "nodes"
```

```
▶ kubectl delete node worker-2
Node worker-2 Deleted!
```

```
▶ kubectl delete node worker-2
Error from server (Forbidden): nodes
"worker-1" is forbidden: User "developer"
cannot delete resource "nodes"
```

```
▶ kubectl delete node worker
Error from server (Forbidden): nodes
"worker-1" is forbidden: User "Bot-1"
delete resource "nodes"
```
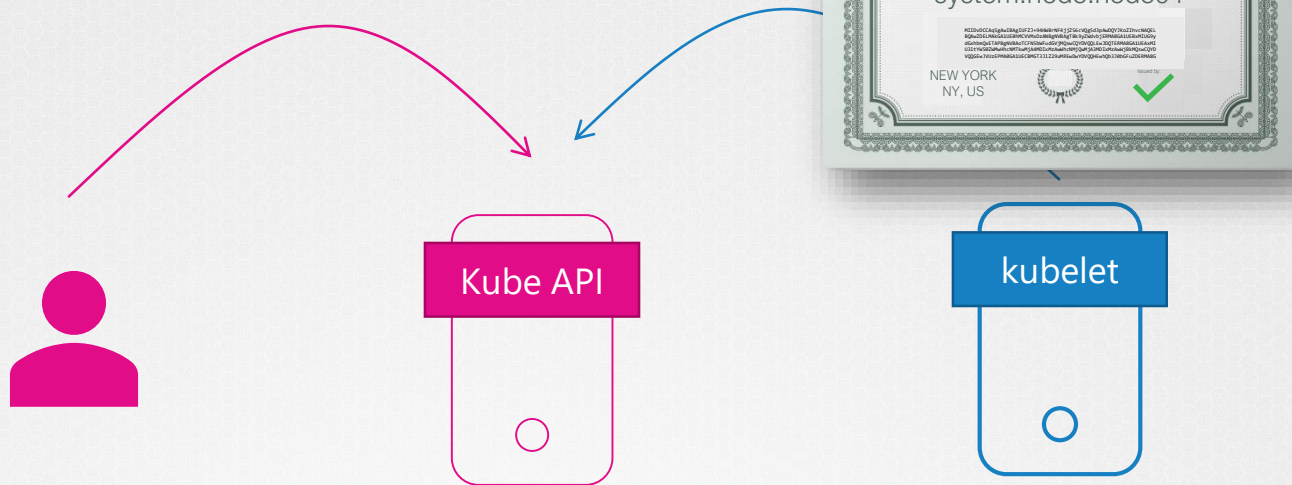
Authorization Mechanisms

Node   ABAC   RBAC   Webhook

Node Authorizer

CERTIFICATE

Group:
SYSTEM:NODES

This Certificate Proudly Presented to

system:node:node01

NEW YORK
NY, US

Kube API

kubelet

- Read
  - Services
  - Endpoints
  - Nodes
  - Pods
- Write
  - Node status
  - Pod status
  - events

KODEKLOUD

ABAC

dev-user

→

- ✓ Can view PODs
- ✓ Can create PODs
- ✓ Can Delete PODs

{"kind": "Policy", "spec": {"user": "dev-user", "namespace": "*", "resource": "pods", "apiGroup": "*"}}

ABAC

- ✓ Can view PODs
- ✓ Can create PODs
- ✓ Can Delete PODs

dev-user

- ✓ Can view PODs
- ✓ Can create PODs
- ✓ Can Delete PODs

dev-user-2

- ✓ Can view PODs
- ✓ Can create PODs
- ✓ Can Delete PODs

dev-users

- ✓ Can view CSR
- ✓ Can approve CSR

security-1

{"kind": "Policy", "spec": {"user": "dev-user", "namespace": "*", "resource": "pods", "apiGroup": "*"}}
{"kind": "Policy", "spec": {"user": "dev-user-2", "namespace": "*", "resource": "pods", "apiGroup": "*"}}
{"kind": "Policy", "spec": {"group": "dev-users", "namespace": "*", "resource": "pods", "apiGroup": "*"}}
{"kind": "Policy", "spec": {"user": "security-1", "namespace": "*", "resource": "csr", "apiGroup": "*"}}

RBAC

Webhook

Kube API

Open Policy Agent

User **dev-user** requested read access to **Pods**. Should I allow?

I checked. Yes!

Authorization Mode

NODE	ABAC	RBAC	WEBHOOK

AlwaysAllow	AlwaysDeny
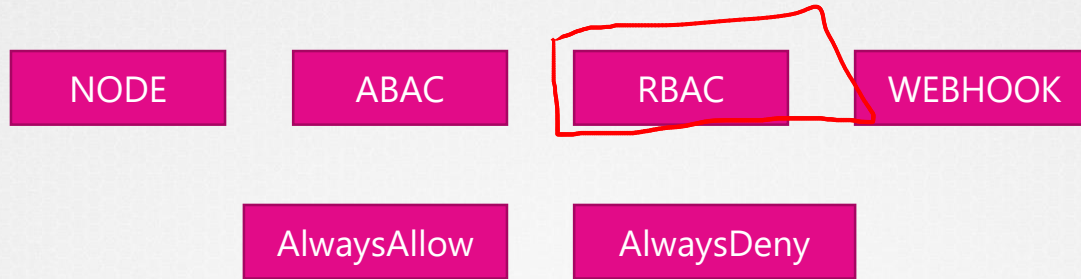
Authorization Mode

AlwaysAllow	NODE	ABAC	RBAC	WEBHOOK	AlwaysDeny

```
ExecStart=/usr/local/bin/kube-apiserver \\
  --advertise-address=${INTERNAL_IP} \\
  --allow-privileged=true \\
  --apiserver-count=3 \\
  --authorization-mode=Node,RBAC,Webhook \\
  --bind-address=0.0.0.0 \\
  --enable-swagger-ui=true \\
  --etcd-cafile=/var/lib/kubernetes/ca.pem \\
  --etcd-certfile=/var/lib/kubernetes/apiserver-etcd-client.crt \\
  --etcd-keyfile=/var/lib/kubernetes/apiserver-etcd-client.key \\
  --etcd-servers=https://127.0.0.1:2379 \\
  --event-ttl=1h \\
  --kubelet-certificate-authority=/var/lib/kubernetes/ca.pem \\
  --kubelet-client-certificate=/var/lib/kubernetes/apiserver-etcd-client.crt \\
  --kubelet-client-key=/var/lib/kubernetes/apiserver-etcd-client.key \\
  --service-node-port-range=30000-32767 \\
  --client-ca-file=/var/lib/kubernetes/ca.pem \\
  --tls-cert-file=/var/lib/kubernetes/apiserver.crt \\
  --tls-private-key-file=/var/lib/kubernetes/apiserver.key \\
  --v=2
```

Authorization Mode
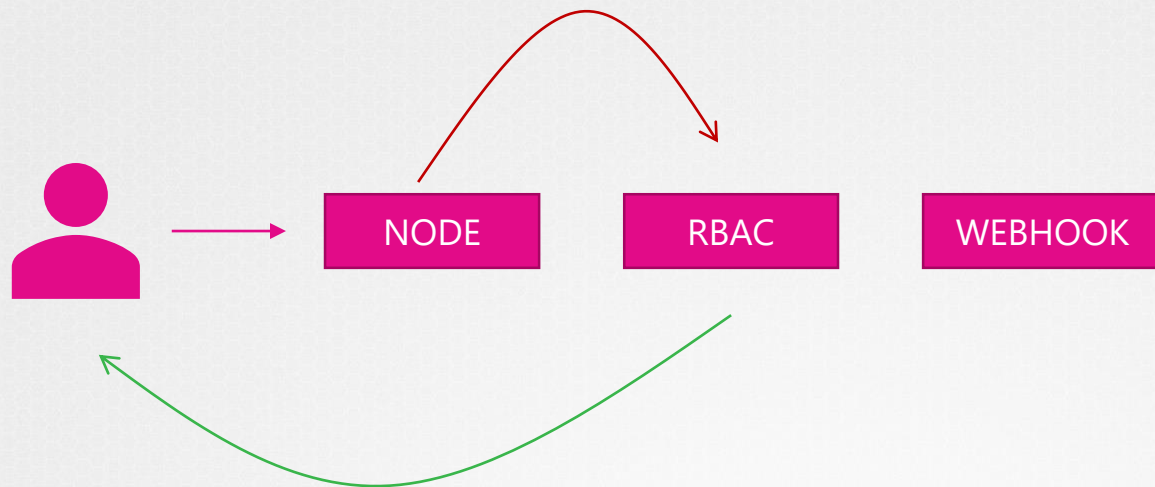

```
ExecStart=/usr/local/bin/kube-apiserver \\
    --advertise-address=${INTERNAL_IP} \\
    --allow-privileged=true \\
    --apiserver-count=3 \\
    --authorization-mode=Node,RBAC,Webhook \\
    --bind-address=0.0.0.0 \\
```

RBAC

RBAC

Developer

- ✓ Can view PODs
- ✓ Can create PODs
- ✓ Can Delete PODs
- ✓ Can Create ConfigMaps

developer-role.yaml

```yaml
apiVersion: rbac.authorization.k8s.io/v1
kind: Role
metadata:
  name: developer
rules:
- apiGroups: [""]
  resources: ["pods"]
  verbs: ["list", "get", "create", "update", "delete"]

- apiGroups: [""]
  resources: ["ConfigMap"]
  verbs: ["create"]
```

```
▶ kubectl create -f developer-role.yaml
```

KODEKLOUD

RBAC

dev-user → Developer

- ✓ Can view PODs
- ✓ Can create PODs
- ✓ Can Delete PODs
- ✓ Can Create ConfigMaps

Namespace: default

```
developer-role.yaml
apiVersion: rbac.authorization.k8s.io/v1
kind: Role
metadata:
  name: developer
rules:
- apiGroups: [""]
  resources: ["pods"]
  verbs: ["list", "get", "create", "update", "de
- apiGroups: [""]
  resources: ["ConfigMap"]
  verbs: ["create"]
```

```
devuser-developer-binding.yaml
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  name: devuser-developer-binding
subjects:
- kind: User
  name: dev-user
  apiGroup: rbac.authorization.k8s.io
roleRef:
  kind: Role
  name: developer
  apiGroup: rbac.authorization.k8s.io
```

```
> kubectl create -f devuser-developer-binding.yaml
```

View RBAC

```
kubectl get roles
```
```
NAME        AGE
developer   4s
```

```
kubectl get rolebindings
```
```
NAME                       AGE
devuser-developer-binding  24s
```

```
kubectl describe role developer
```
```
Name:         developer
Labels:       <none>
Annotations:  <none>
PolicyRule:
  Resources   Non-Resource URLs   Resource Names   Verbs
  ---------   -----------------   --------------   -----
  ConfigMap   []                  []               [create]
  pods        []                  []               [get watch list create delete]
```

View RBAC

```
kubectl describe rolebinding devuser-developer-binding
```
```
Name:          devuser-developer-binding
Labels:        <none>
Annotations:   <none>
Role:
  Kind:  Role
  Name:  developer
Subjects:
  Kind  Name      Namespace
  ----  ----      ---------
  User  dev-user
```

Check Access (Import)

```
kubectl auth can-i create deployments
yes
```

```
kubectl auth can-i delete nodes
no
```

```
kubectl auth can-i create deployments --as dev-user
no
```

```
kubectl auth can-i create pods --as dev-user
yes
```

```
kubectl auth can-i create pods --as dev-user --namespace test
no
```

KODEKLOUD

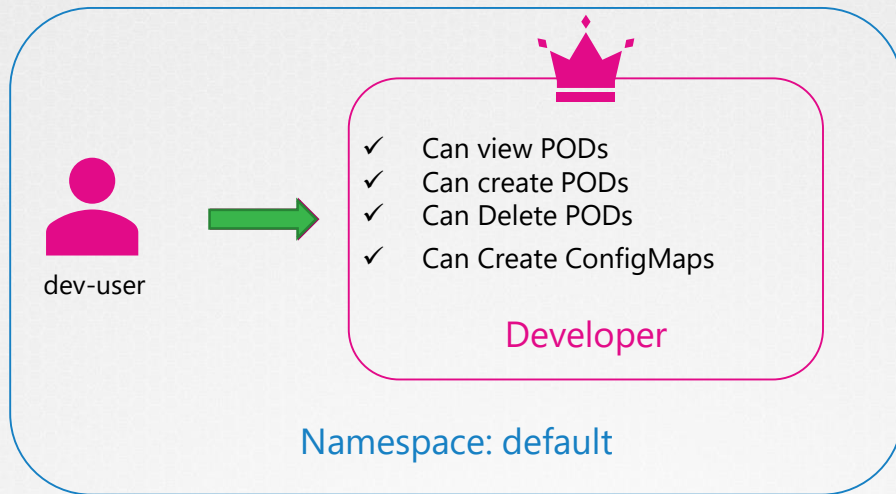# Resource Names

blue   green   orange
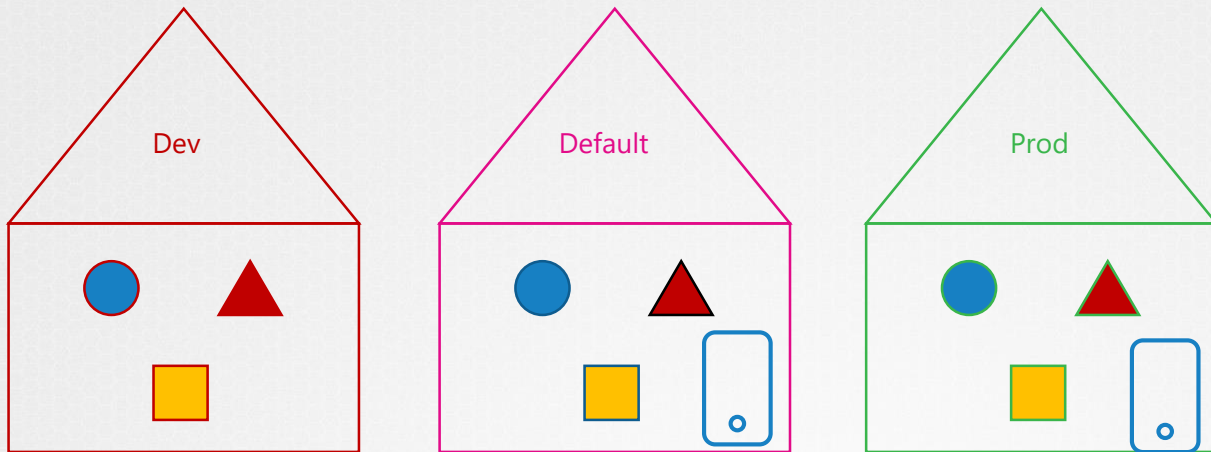
purple   pink

developer-role.yaml

```yaml
apiVersion: rbac.authorization.k8s.io/v1
kind: Role
metadata:
  name: developer
rules:
- apiGroups: [""]
  resources: ["pods"]
  verbs: ["get", "create", "update"]
  resourceNames: ["blue", "orange"]
```
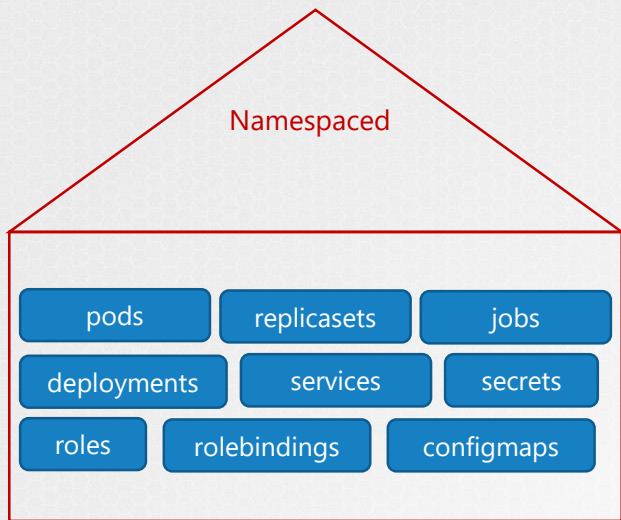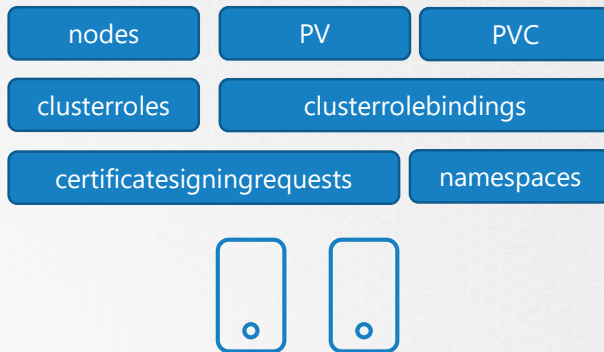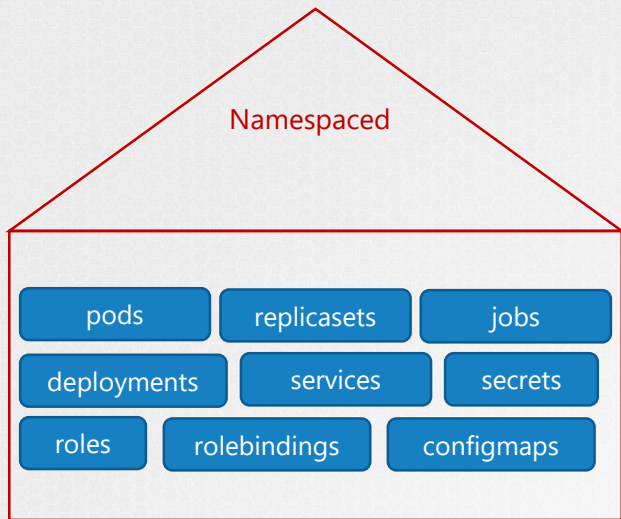
KODEKLOUD

Cluster Roles

Roles

dev-user

- ✓ Can view PODs
- ✓ Can create PODs
- ✓ Can Delete PODs
- ✓ Can Create ConfigMaps

Developer

Namespace: default

Namespace

Namespace

Namespaced

pods | replicasets | jobs
deployments | services | secrets
roles | rolebindings | configmaps

Cluster Scoped

nodes | PV | PVC
clusterroles | clusterrolebindings
certificatesigningrequests | namespaces

```
kubectl api-resources --namespaced=true
```

```
kubectl api-resources --namespaced=false
```

Namespace

Namespaced

pods	replicasets	jobs
deployments	services	secrets
roles	rolebindings	configmaps

Cluster Scoped

nodes	PV	PVC
clusterroles	clusterrolebindings	
certificatesigningrequests	namespaces	

KODEKLOUD

👑 clusterroles

Cluster Admin
- ✓ Can view Nodes
- ✓ Can create Nodes
- ✓ Can delete Nodes

Storage Admin
- ✓ Can view PVs
- ✓ Can create PVs
- ✓ Can delete PVCs

cluster-admin-role.yaml

```yaml
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: cluster-administrator
rules:
- apiGroups: [""]
  resources: ["nodes"]
  verbs: ["list", "get", "create", "delete"]
```

```
▶ kubectl create -f cluster-admin-role.yaml
```

KODEKLOUD

clusterrolebinding

cluster-admin

Can view Nodes
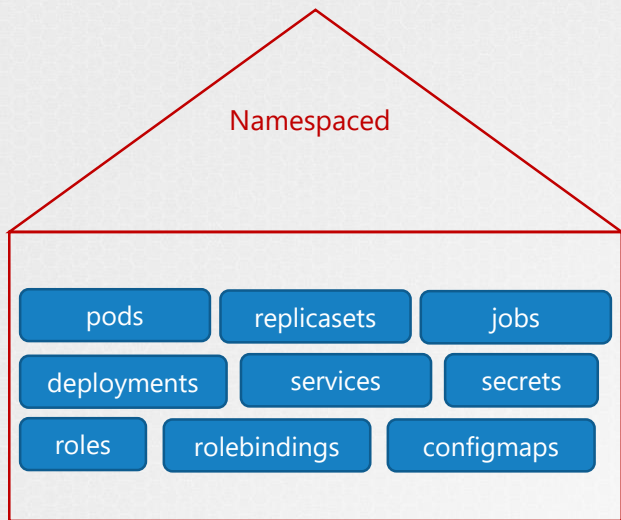Can create Nodes
Can delete Nodes

Cluster Admin

cluster-admin-role.yaml

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: cluster-administrator
rules:
- apiGroups: [""]
  resources: ["nodes"]
  verbs: ["list", "get", "create", "delete"]
```

cluster-admin-role-binding.yaml

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: cluster-admin-role-binding
subjects:
- kind: User
  name: cluster-admin
  apiGroup: rbac.authorization.k8s.io
roleRef:
  kind: ClusterRole
  name: cluster-administrator
  apiGroup: rbac.authorization.k8s.io
```

```
> kubectl create –f cluster-admin-role-binding.yaml
```

Cluster Roles

Namespaced

pods	replicasets	jobs
deployments	services	secrets
roles	rolebindings	configmaps

Cluster Scoped

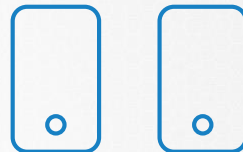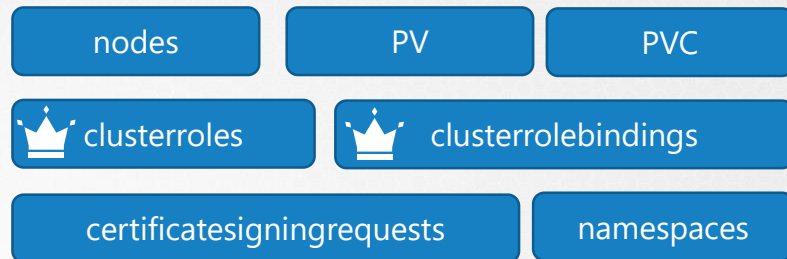|		
nodes	PV	PVC
clusterroles	clusterrolebindings	
certificatesigningrequests	namespaces	

Image Security

Image

nginx-pod.yaml

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: nginx-pod
spec:
  containers:
  - name: nginx
    image: nginx
```

Image

image: docker.io/nginx/nginx

Registry   User/   Image/
                Account   Repository

gcr.io/ kubernetes-e2e-test-images/dnsutils

Private Repository

```
▶ docker login private-registry.io
```

```
Login with your Docker ID to push and pull images from Docker Hub. If you don't have a Docker ID, head over to
https://hub.docker.com to create one.
Username: registry-user
Password:
WARNING! Your password will be stored unencrypted in /home/vagrant/.docker/config.json.

Login Succeeded
```

```
▶ docker run private-registry.io/apps/internal-app
```

Private Repository

```
docker login private-registry.io
```

```
docker run private-registry.io/apps/internal-app
```

nginx-pod.yaml

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: nginx-pod
spec:
  containers:
  - name: nginx
    image:
  imagePullSecrets:
  - name: regcred
```

```
kubectl create secret docker-registry regcred \
    --docker-server= private-registry.io    \
    --docker-username= registry-user  \
    --docker-password= registry-password  \
    --docker-email= registry-user@org.com
```

KODEKLOUD