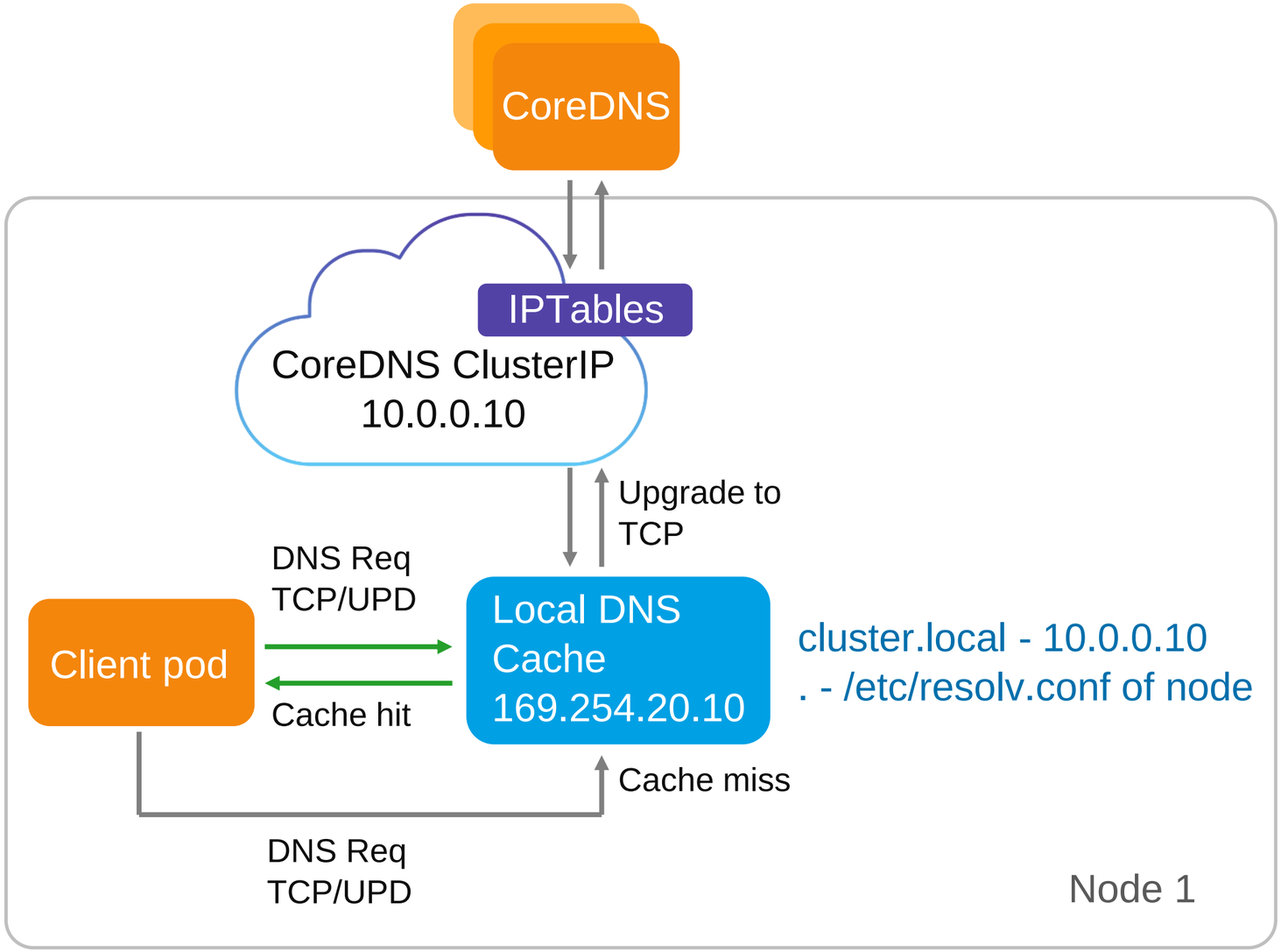
СЛЁРМ
+
Southbridge

DNS в Kubernetes. Публикация сервисов и приложений.

DNS


CoreDNS
IPTables
CoreDNS ClusterIP
10.0.0.10
Upgrade to
TCP
DNS Req
TCP/UPD
Client pod
Local DNS
Cache
169.254.20.10
cluster.local - 10.0.0.10
. - /etc/resolv.conf of node
Cache hit
Cache miss
DNS Req
TCP/UPD
Node 1

DNS

```
[root@master-1.slurm.io /]# kubectl get all -n kube-system | grep dns

NAME                                       READY   STATUS    RESTARTS   AGE
pod/coredns-56bc6b976d-drgrm               1/1     Running   0          95d
pod/coredns-56bc6b976d-r71dl               1/1     Running   0          95d
pod/dns-autoscaler-56c969bdb8-j4bwk        1/1     Running   0          95d
pod/nodelocaldns-4298p                     1/1     Running   1          95d
pod/nodelocaldns-8bgzg                     1/1     Running   0          95d
pod/nodelocaldns-csprl                     1/1     Running   0          95d
pod/nodelocaldns-fg2sl                     1/1     Running   1          95d
pod/nodelocaldns-gk6l7                     1/1     Running   0          95d
pod/nodelocaldns-j4jbd                     1/1     Running   0          95d
pod/nodelocaldns-rfqth                     1/1     Running   0          95d
pod/nodelocaldns-wdxz5                     1/1     Running   0          95d
pod/nodelocaldns-xc79w                     1/1     Running   0          95d

NAME                                       TYPE        CLUSTER-IP     EXTERNAL-IP   PORT(S)                  AGE
service/coredns                            ClusterIP   10.9.0.3         <none>      53/UDP,53/TCP,9153/TCP   95d

NAME                                       DESIRED   CURRENT   READY   UP-TO-DATE   AVAILABLE   NODE SELECTOR   AGE
daemonset.apps/nodelocaldns                9         9         9       9            9           <none>          95d

NAME                                       READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/coredns                    2/2     2            2           95d
deployment.apps/dns-autoscaler             1/1     1            1           95d

NAME                                       DESIRED   CURRENT   READY   AGE
replicaset.apps/coredns-56bc6b976d         2         2         2       95d
replicaset.apps/dns-autoscaler-56c969bdb8  1         1         1       95d
```

DNS

Внутри контейнера:

```
[root@centos-test /]# cat /etc/resolv.conf
nameserver 169.254.25.10
search default.svc.slurm.io svc.slurm.io slurm.io
options ndots:5
```

На ноде в Kubelet:

```
[root@master-1.slurm.io /]# cat /etc/kubernetes/kubelet.env | grep dns

--enforce-node-allocatable="" -cluster-dns=169.254.25.10 -cluster-domain=slurm.io
```

DNS

Внутри контейнера:

```
[root@centos-test /]# cat /etc/resolv.conf
nameserver 169.254.25.10
search default.svc.slurm.io svc.slurm.io slurm.io
options ndots:5
```

На ноде в Kubelet:

```
[root@master-1.slurm.io /]# cat /etc/kubernetes/kubelet.env | grep dns

--enforce-node-allocatable="" -cluster-dns=169.254.25.10 -cluster-domain=slurm.io
```

DNS

```
[root@centos-test /]# nslookup yandex.ru
Server:         169.254.25.10
Address:        169.254.25.10#53

Non-authoritative answer:
Name:   yandex.ru
Address: 77.88.55.66
Name:   yandex.ru
Address: 5.255.255.60
Name:   yandex.ru
Address: 77.88.55.70
Name:   yandex.ru
Address: 5.255.255.70

------------------------------------------------------------------------------------------------------
[root@centos-test /]# tcpdump -n -p -i eth0 port 53
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 262144 bytes

12:37:41.915647 IP 10.109.1.22.56099 > 169.254.25.10.domain: 8959+ A? yandex.ru.default.svc.slurm.io. (53)
12:37:41.917337 IP 169.254.25.10.domain > 10.109.1.22.56099: 8959 NXDomain*- 0/1/0 (146)
12:37:41.917544 IP 10.109.1.22.54469 > 169.254.25.10.domain: 62110+ A? yandex.ru.svc.slurm.io. (45)
12:37:41.918421 IP 169.254.25.10.domain > 10.109.1.22.54469: 62110 NXDomain*- 0/1/0 (138)
12:37:41.918575 IP 10.109.1.22.42564 > 169.254.25.10.domain: 58068+ A? yandex.ru.slurm.io. (41)
12:37:41.919348 IP 169.254.25.10.domain > 10.109.1.22.42564: 58068 NXDomain*- 0/1/0 (134)
12:37:41.919596 IP 10.109.1.22.41789 > 169.254.25.10.domain: 17670+ A? yandex.ru. (27)
12:37:42.064504 IP 169.254.25.10.domain > 10.109.1.22.41789: 17670 4/3/6 A 77.88.55.66, A 5.255.255.60, A 77.88.55.70, A 5.255.255.70 (460)
```

Способы публикации

Service: L3 OSI, NAT, kube-proxy

```
-A KUBE-MARK-MASQ -j MARK    --set-xmark 0x4000/0x4000
-A KUBE-NODEPORTS -p tcp -m comment --comment "s000000/np2:http"
-m tcp --dport 30029 -j KUBE-MARK-MASQ

-A KUBE-NODEPORTS -p tcp -m comment --comment "s000000/np2:http"
-m tcp --dport 30029 -j KUBE-SVC-2F3FOG2AWAH5Y5PC
```

Ingress: L7 OSI, HTTP и HTTPS, nginx, envoy, traefik, haproxy

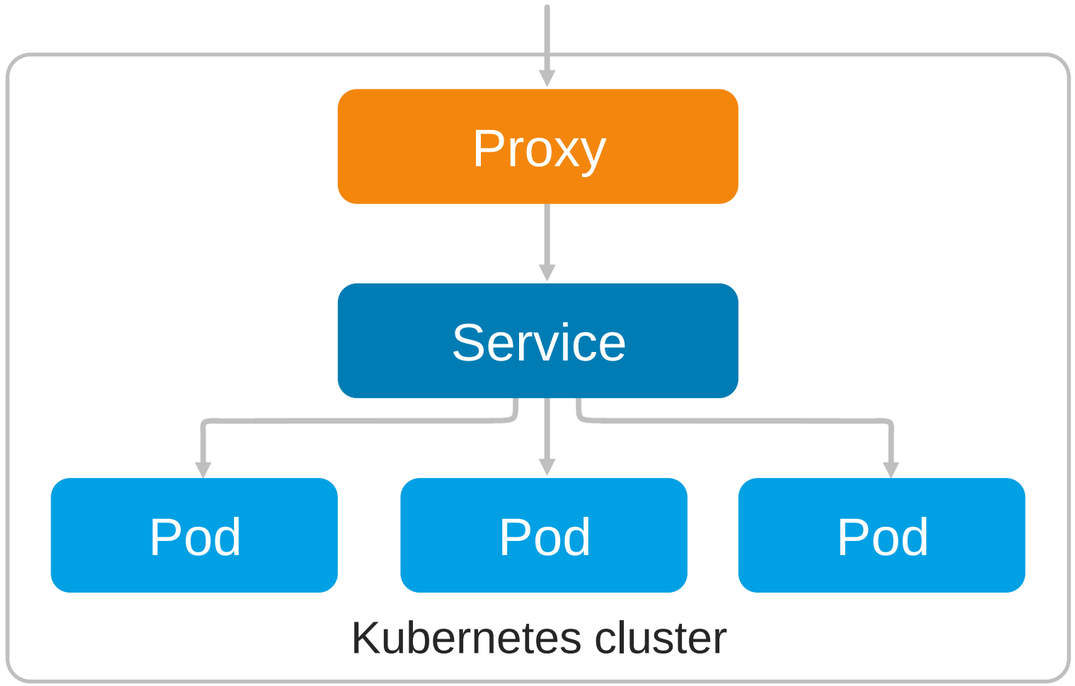
```
server {
  server_name slurm.io  ssl on;

  location {
     proxy_pass http://backend;
  }
}
```

Kubernetes Service

- ClusterIP
- NodePort
- LoadBalancer
- ExternalName
- ExternalIPs

ClusterIP

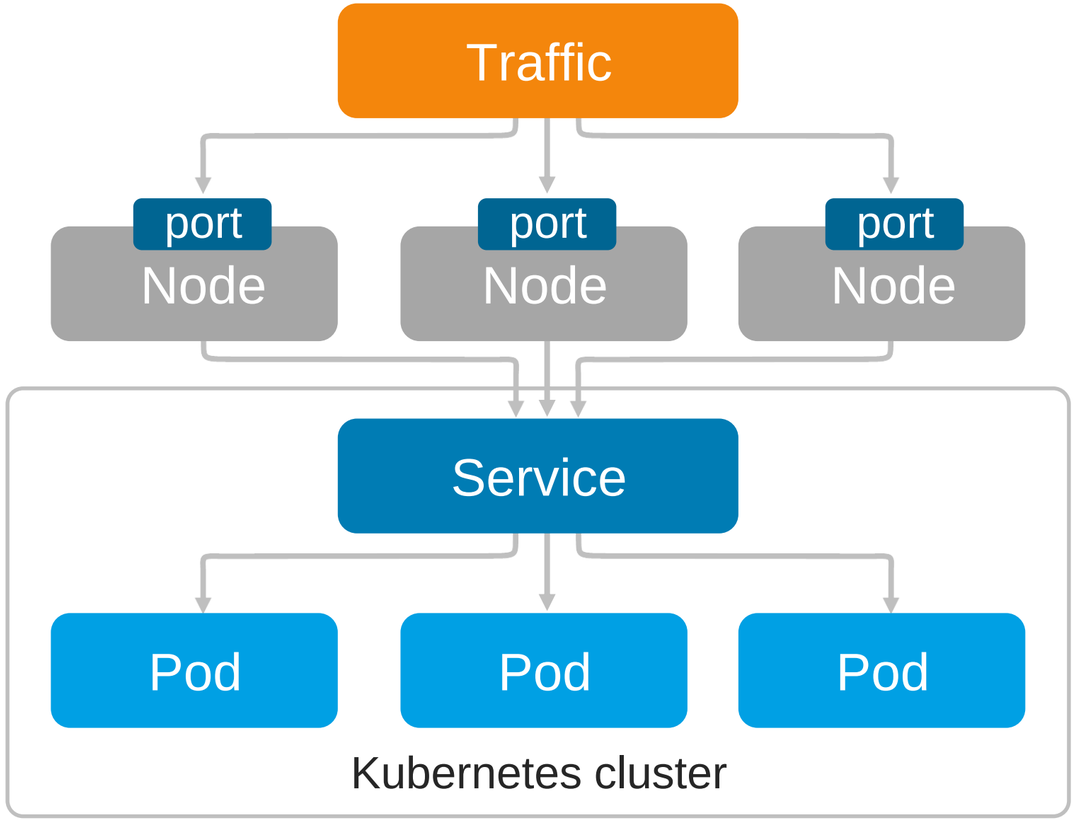
```yaml
apiVersion: v1
kind: Service
metadata:
  name: my-service
spec:
  selector:
    app: my-app
  type: ClusterIP
  ports:
  - name: http
    port: 80
    targetPort: 80
    protocol: TCP
```




```
kubectl proxy --port=8080  http://localhost:8080/api/v1/proxy/namespacesdefault/services/my-service:http/

kubectl port-forward service/my-service 10000:80
```

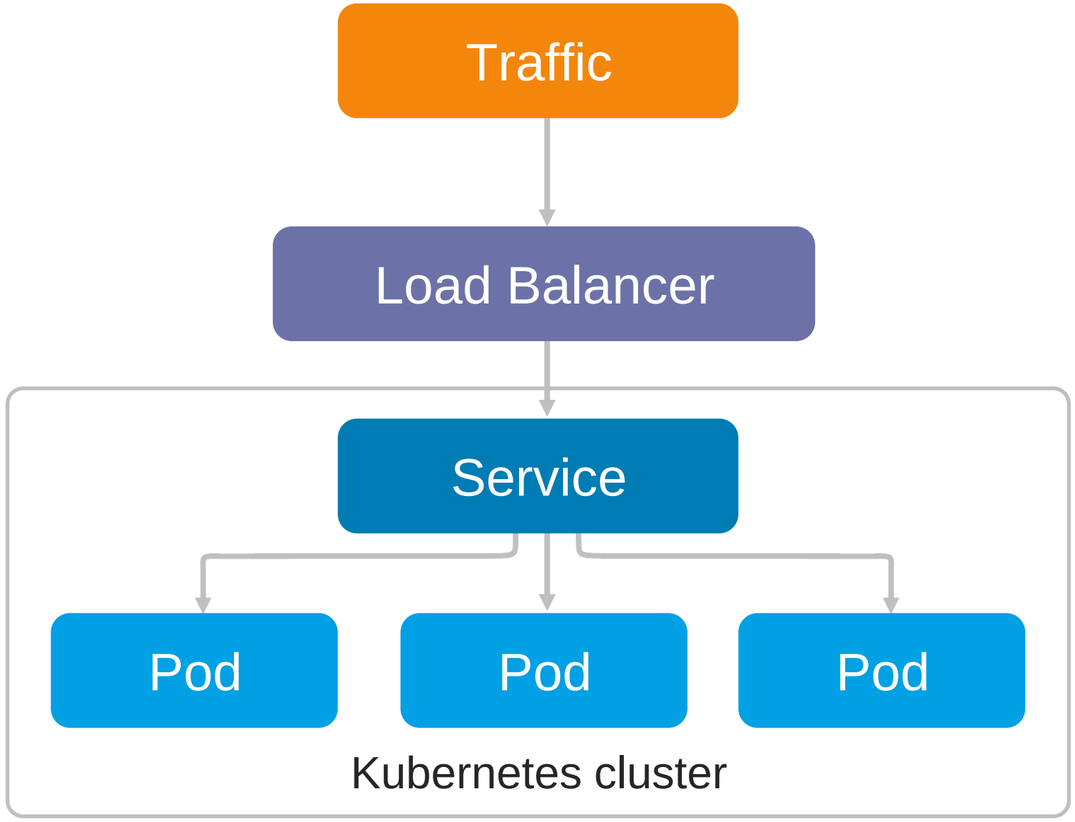
NodePort

```yaml
apiVersion: v1
kind: Service
metadata:
  name: my-service-np
spec:
  selector:
    app: my-app
  type: NodePort
  ports:
  - name: http
    port: 80
    targetPort: 80
    protocol: TCP
```



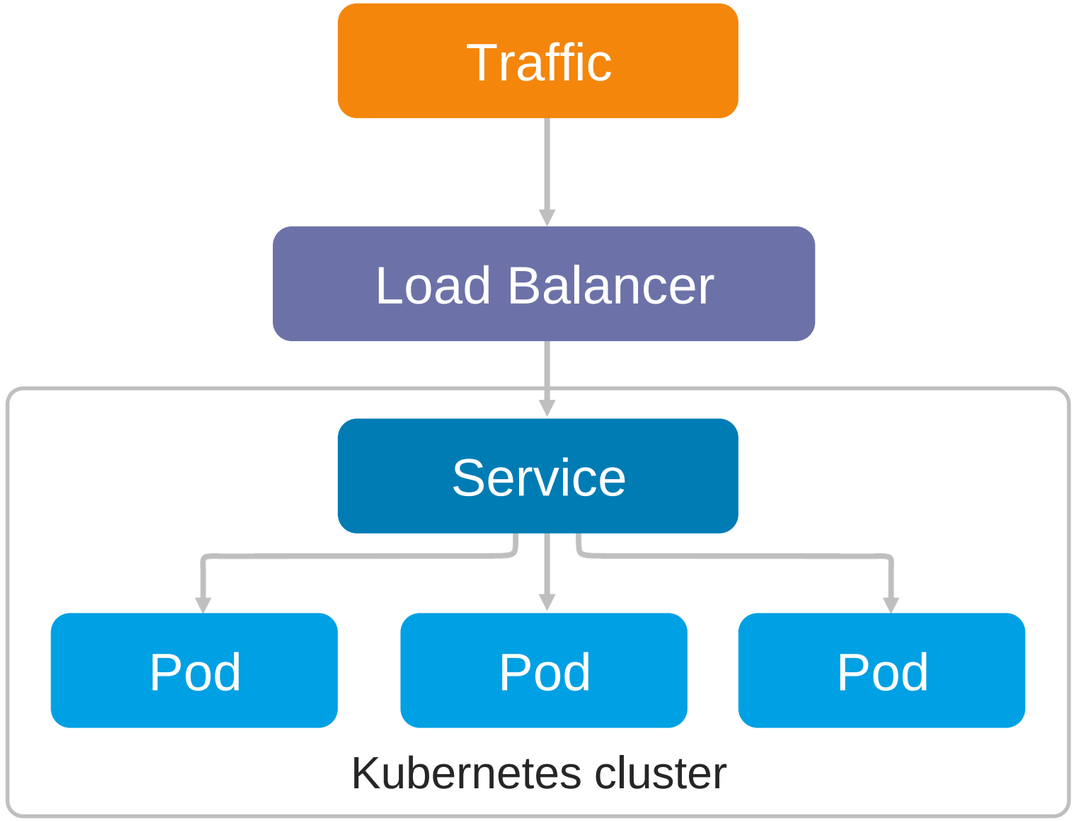
LoadBalancer

```yaml
apiVersion: v1
kind: Service
metadata:
  name: my-service-lb
spec:
  selector:
    app: my-app
  type: LoadBalancer
  ports:
  - name: http
    port: 80
    targetPort: 80
    protocol: TCP
```



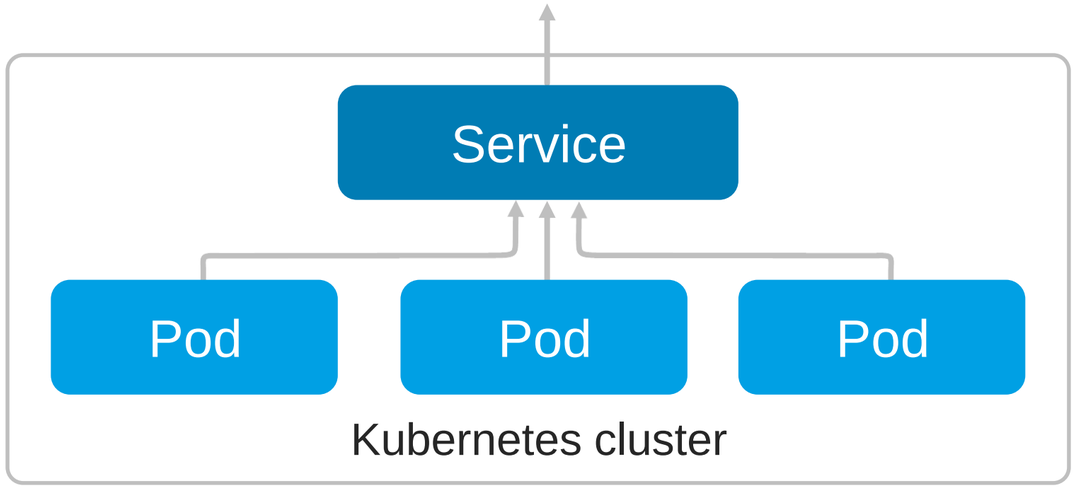
LoadBalancer static IP

```
apiVersion: v1
kind: Service
metadata:
  name: my-service-lb
spec:
  selector:
    app: my-app
  type: LoadBalancer
  loadBalancerIP:
  "1.1.1.1"
  ports:
  - port: 80
    targetPort: 80
```



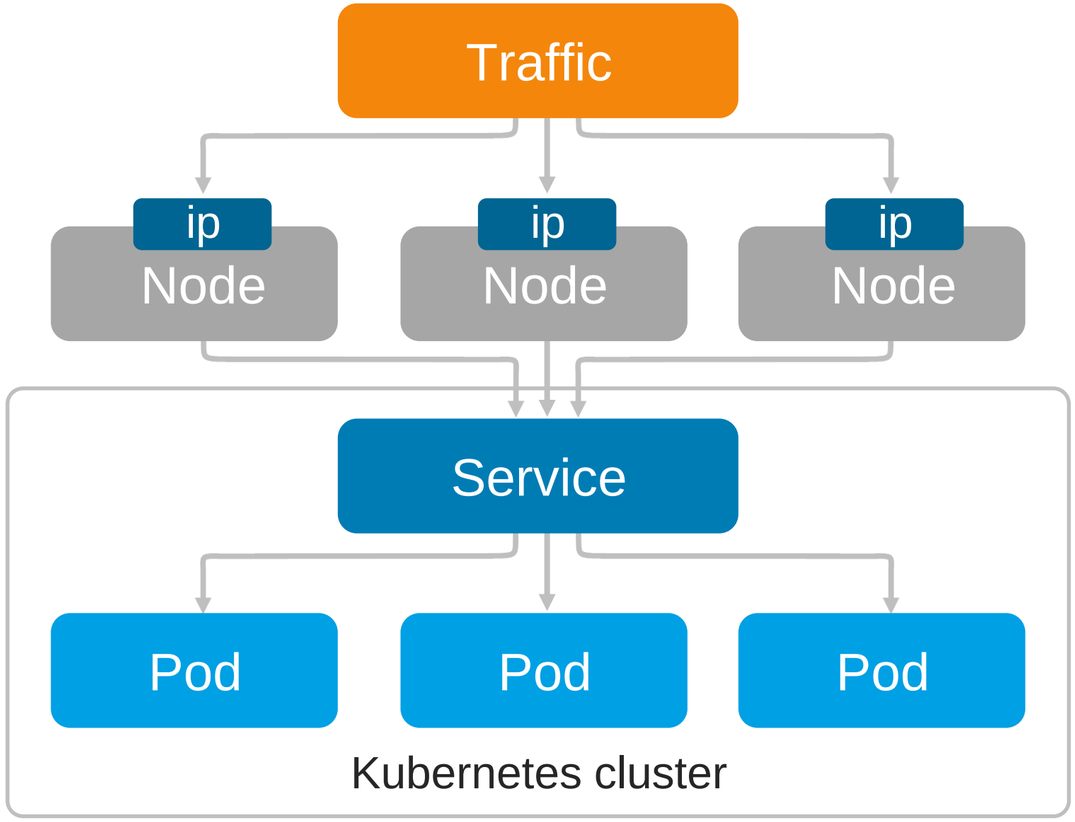
ExternalName

```
apiVersion: v1
kind: Service
metadata:
  name: my-service
spec:
  selector:
    app: my-app
  type: ExternalName
  externalName:
  my.database.
   example.com
```



ExternalIPs

```
apiVersion: v1
kind: Service
metadata:
  name: myservice
spec:
  selector:
    app: my-app
  ports:
  - name: http
    port: 80
    targetPort: 80
    protocol: TCP
  externalIPs:
  - 80.11.12.10
```



Kubernetes Service

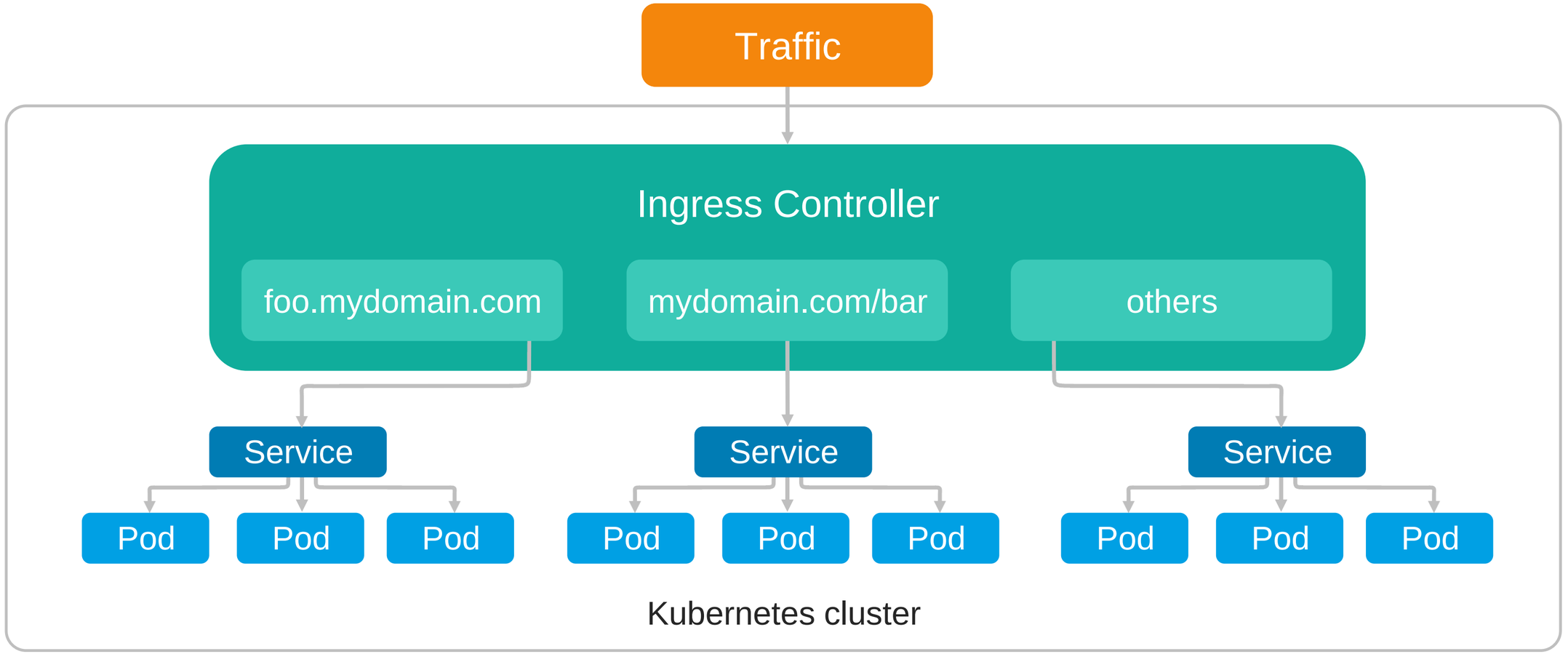
- ClusterIP
- NodePort
- LoadBalancer
- ExternalName
- ExternalIPs

Headless

```
apiVersion: v1
kind: Service
metadata:
  name: my-service
spec:
  selector:
    app: my-app
  ClusterIP: none
  ports:
  - name: http
    port: 80
    targetPort: 80
    protocol: TCP
```

```
⚡ kubectl exec pod-1 -- nslookup my-service
Server:      10.0.0.10
Address:    10.0.0.10#53

Name:    my-service.default.svc.cluster.local
Address: 10.0.12.5
Name:    my-service.default.svc.cluster.local
Address: 10.0.12.6
Name:    my-service.default.svc.cluster.local
Address: 10.0.12.7
```

Ingress


Traffic
Ingress Controller
foo.mydomain.com
mydomain.com/bar
others
Service
Service
Service
Pod
Pod
Pod
Pod
Pod
Pod
Pod
Pod
Pod
Kubernetes cluster

Ingress

```
apiVersion: extensions/v1beta1
kind: Ingress
metadata:
  name: my-ingress
  annotations:
     nginx.ingress.kubernetes.io/backend-protocol: "HTTPS"
spec:
  rules:
  - host: foo.mydomain.com
     http:
       paths:
       - backend:
           serviceName: foo
           servicePort: 8080
  - host: mydomain.com
     http:
       paths:
       - path: /bar/
         backend:
           serviceName: bar
           servicePort: 8080
```

HOST: foo.mydomain.com

HOST: mydomain.com
URL: /bar/

Ingress

```
apiVersion: extensions/v1beta1
kind: Ingress
metadata:
  name: my-ingress
  annotations:
    nginx.ingress.kubernetes.io/backend-protocol: "HTTPS"
spec:
  rules:
  - host: foo.mydomain.com
    http:
      paths:
      - backend:
          serviceName: foo
          servicePort: 8080
  - host: mydomain.com
    http:
      paths:
      - path: /bar/
        backend:
          serviceName: bar
          servicePort: 8080
```

HOST: foo.mydomain.com

HOST: mydomain.com
URL: /bar/

Ingress

Указываем сертификат в Ingress

```
apiVersion: extensions/v1beta1
kind: Ingress
metadata:
  name: tls-ingress
spec:
  tls:
  - hosts:
    - sslfoo.com
    secretName: secret-tls
```

Создаем секрет с сертификатом

```
apiVersion: v1
data:
  tls.crt: base64 encoded cert
  tls.key: base64 encoded key
kind: Secret
metadata:
  name: secret-tls
  namespace: default
type: kubernetes.io/tls
```

```
kubectl create secret tls ${CERT_NAME} --key ${KEY_FILE} --cert ${CERT_FILE}
```

Что мы выяснили:

- Какие способы публикации нашего приложения есть
- Какие типы сервисов есть и когда их применять
- Как работает DNS
- Как публиковать приложение через ингресс
- Как подключить HTTPS
- Что такое аннотация

Cert-manager

- Начинался как способ получить сертификат от LetsEncrypt

Cert-manager

- Начинался как способ получить сертификат от LetsEncrypt
- Автоматизирует получение SSL/TLS-сертификатов от различных удостоверяющих центров (LetsEncrypt, selfhosted, selfsigned)

Cert-manager

- Начинался как способ получить сертификат от LetsEncrypt
- Автоматизирует получение SSL/TLS-сертификатов от различных удостоверяющих центров (LetsEncrypt, selfhosted, selfsigned)
- Интегрируется с ингресс-контроллером

Cert-manager

- Начинался как способ получить сертификат от LetsEncrypt
- Автоматизирует получение SSL/TLS-сертификатов от различных удостоверяющих центров (LetsEncrypt, selfhosted, selfsigned)
- Интегрируется с ингресс-контроллером
- Автоматизирует продление сертификатов

Cert-manager

- Начинался как способ получить сертификат от LetsEncrypt
- Автоматизирует получение SSL/TLS-сертификатов от различных удостоверяющих центров (LetsEncrypt, selfhosted, selfsigned)
- Интегрируется с ингресс-контроллером
- Автоматизирует продление сертификатов
- CRD: Issuer, ClusterIssuer, Certificate

Cert-manager

- Начинался как способ получить сертификат от LetsEncrypt
- Автоматизирует получение SSL/TLS-сертификатов от различных удостоверяющих центров (LetsEncrypt, selfhosted, selfsigned)
- Интегрируется с ингресс-контроллером
- Автоматизирует продление сертификатов
- CRD: Issuer, ClusterIssuer, Certificate
- RBAC: certmanager.k8s.io

Cert-manager

```
kubectl apply --validate=false -f https://raw.githubusercontent.com/jetstack/cert-manager/release-0.11/deploy/manifests/00-crds.yaml

kubectl create namespace cert-manager

helm repo add jetstack https://charts.jetstack.io
helm repo update

helm install \
  --name cert-manager \
  --namespace cert-manager \
  --version v0.11.0 \
  --set ingressShim.defaultIssuerName=letsencrypt \
  --set ingressShim.defaultIssuerKind=ClusterIssuer \
  jetstack/cert-manager
```

Cert-manager

```
apiVersion: cert-manager.io/v1alpha2
kind: ClusterIssuer
metadata:
  name: letsencrypt  namespace: kube-system
spec:
  acme:
    # The ACME server URL
    server: https://acme-v02.api.letsencrypt.org/directory
    # Email address used for ACME registration
    email: letsencrypt@slurm.io
    # Name of a secret used to store the ACME account private key
    privateKeySecretRef:
      name: letsencrypt
    # Enable the HTTP-01 challenge provider
    solvers:
      - http01:
          ingress:
            class: nginx
```

Подключаем в Ingress

```
apiVersion: cert-manager.io/v1alpha2
kind: Certificate
metadata:
  name: hostname-ru
  namespace: default
spec:
  acme:
    config:
    - domains:
      - hostname.ru
      - www.hostname.ru
      http01:
        ingress: ""
        ingressClass: nginx
  secretName: hostname-ru-tls
  commonName: hostname.ru
  dnsNames:
  - hostname.ru
  - www.hostname.ru
```

```
  issuerRef:
      name: letsencrypt
      kind: ClusterIssuer
```

```
apiVersion: extensions/v1beta1
kind: Ingress
metadata:
  name:  tls-ingress
  annotations:
    kubernetes.io/tls-acme:
"true"
```

ИЛИ

```
    certmanager.k8s.io/cluster-
issuer:   letsencrypt
```

Что мы выяснили:

- Что такое Cert-manager и как он устанавливается
- Как настроить Cert-manager
- Как интегрировать его работу с Ingress-controller


СЛЁРМ
+
Southbridge
slurm.io