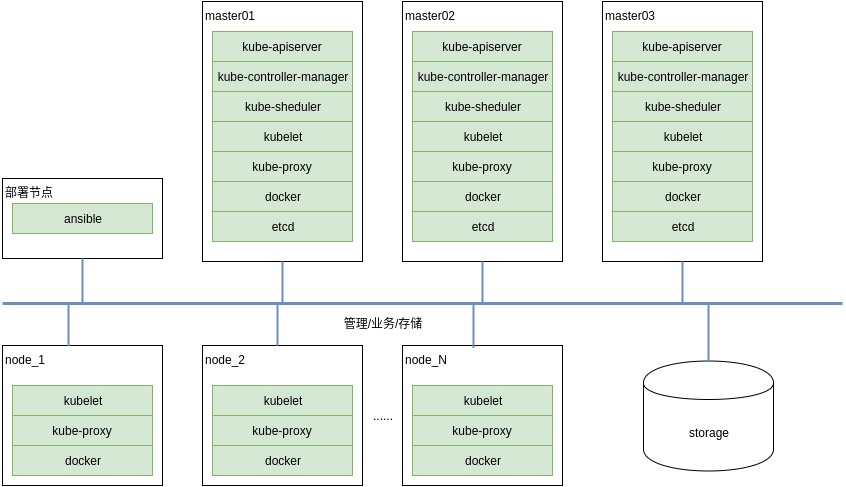
# kubernetes集群部分设计文档

## 功能点及部署架构



集群部署节点分为3中角色：部署节点，master，node。

* 部署节点：需要安装ansible，可以多个kubernetes共用，通过切换配置文件来部署不同的集群。
* master：作为kubernetes的控制节点，上面需要部署kube-apiserver，kube-controller-manager， kube-scheduler，kubelet，kube-proxy，dockder服务，etcd可以部署在master节点上，也可以单独部署。初期部署在master节点上，随着集群规模增大，可以将etcd单独部署，上SSD盘，将Kubernetes Events 存储在一个单独的 etcd 集群里等一系列优化。
* node：业务Pod的负载节点，上面需要部署kubelet，kube-proxy，dockder服务。
* 共享存储可以是nfs，ceph，glusterfs等。根据后端存储不同，可能需要安装不同的provider。
* 目前管理/业务/存储网络走的是同一个物理网口，如果业务量大，可以考虑隔离。

## 参数配置项

以下描述常用的各个服务常用参数，全部的参数含义可以参见官方文档。

[https://kubernetes.io/docs/reference/command-line-tools-reference/<组件名>/](https://kubernetes.io/docs/reference/command-line-tools-reference/kube-apiserver/)

### kube-apiserver参数

|  |  |  |
| --- | --- | --- |
| 项目 | 说明 | 示例 |
| advertise-address | 向集群成员通知apiserver消息的IP地址。这个地址必须能够被集群中其他成员访问。如果IP地址为空，将会使用--bind-address，如果未指定--bind-address，将会使用主机的默认接口地址。 | 192.168.3.142 |
| allow-privileged | 如果为true,将允许特权容器. | true |
| apiserver-count | 集群中运行的apiserver数量，必须为正数。（默认值1） | 3 |
| audit-log-maxage | 基于文件名中的时间戳，旧审计日志文件的最长保留天数。 | 30 |
| audit-log-maxbackup | 旧审计日志文件的最大保留个数. | 1 |
| audit-log-maxsize | 审计日志被轮转前的最大兆字节数。 | 100 |
| audit-log-path | 如果设置该值，所有到apiserver的请求都将会被记录到这个文件。'-'表示记录到标准输出。 | /var/log/audit/kube-apiserver-audit.log |
| audit-policy-file | 定义审计策略配置的文件的路径。需要打开'AdvancedAuditing'特性开关。AdvancedAuditing需要一个配置来启用审计功能。 | /etc/kubernetes/audit-policy/apiserver-audit-policy.yaml |
| authorization-mode | 在安全端口上进行权限验证的插件的顺序列表。以逗号分隔的列表，包括：AlwaysAllow,AlwaysDeny,ABAC,Webhook,RBAC,Node.（默认值"AlwaysAllow"） | Node,RBAC |
| bind-address | 监听--seure-port的IP地址。被关联的接口必须能够被集群其它节点和CLI/web客户端访问。如果为空，则将使用所有接口（0.0.0.0）。（默认值0.0.0.0） | 0.0.0.0 |
| client-ca-file | 如果设置此标志，对于任何请求，如果存包含client-ca-file中的authorities签名的客户端证书，将会使用客户端证书中的CommonName对应的身份进行认证。 | /etc/kubernetes/ssl/ca.crt |
| enable-admission-plugins | 设定启用的权限相关的插件的列表，以逗号分隔。该列表中的插件顺序不影响结果。该列表中的插件会与默认启用的插件一起启用。 | NodeRestriction |
| enable-bootstrap-token-auth | 启用此选项以允许'kube-system'命名空间中的'bootstrap.kubernetes.io/token'类型密钥可以被用于TLS的启动认证。 | true |
| endpoint-reconciler-type | 设置启用的终端协调器（可选：master-count、lease、none，默认：lease）。 | lease |
| etcd-cafile | 用于保护etcd通信的SSLCA文件。 | /etc/ssl/etcd/ssl/ca.pem |
| etcd-certfile | 用于保护etcd通信的的SSL证书文件。 | /etc/ssl/etcd/ssl/node-test-master01.pem |
| etcd-keyfile | 用于保护etcd通信的SSL密钥文件. | /etc/ssl/etcd/ssl/node-test-master01-key.pem |
| etcd-servers | 连接的etcd服务器列表,形式为（scheme://ip:port)，使用逗号分隔。 | https://192.168.3.142:2379,https://192.168.3.143:2379,https://192.168.3.144:2379 |
| insecure-port | 用于监听不安全和为认证访问的端口。这个配置假设你已经设置了防火墙规则，使得这个端口不能从集群外访问。对集群的公共地址的443端口的访问将被代理到这个端口。默认设置中使用nginx实现。（默认值8080） | 0 |
| kubelet-client-certificate | 用于TLS的客户端证书文件路径。 | /etc/kubernetes/ssl/apiserver-kubelet-client.crt |
| kubelet-client-key | 用于TLS的客户端证书密钥文件路径. | /etc/kubernetes/ssl/apiserver-kubelet-client.key |
| kubelet-preferred-address-types | 用于kubelet连接的首选NodeAddressTypes列表。(默认值[Hostname,InternalDNS,InternalIP,ExternalDNS,ExternalIP]) | InternalDNS,InternalIP,Hostname,ExternalDNS,ExternalIP |
| proxy-client-cert-file | 当必须调用外部程序时，用于证明aggregator或者kube-apiserver的身份的客户端证书。包括代理到用户api-server的请求和调用webhook准入控制插件的请求。它期望这个证书包含一个来自于CA中的--requestheader-client-ca-file标记的签名。该CA在kube-system命名空间的'extension-apiserver-authentication'configmap中发布。从Kube-aggregator收到调用的组件应该使用该CA进行他们部分的双向TLS验证。 | /etc/kubernetes/ssl/front-proxy-client.crt |
| proxy-client-key-file | 当必须调用外部程序时，用于证明aggregator或者kube-apiserver的身份的客户端证书密钥。包括代理到用户api-server的请求和调用webhook准入控制插件的请求。 | /etc/kubernetes/ssl/front-proxy-client.key |
| requestheader-allowed-names | 使用--requestheader-username-headers指定的，允许在头部提供用户名的客户端证书通用名称列表。如果为空，任何通过--requestheader-client-ca-file中authorities验证的客户端证书都是被允许的。 | front-proxy-client |
| requestheader-client-ca-file | 在信任请求头中以--requestheader-username-headers指示的用户名之前，用于验证接入请求中客户端证书的根证书捆绑。 | /etc/kubernetes/ssl/front-proxy-ca.crt |
| requestheader-extra-headers-prefix | 用于检查的请求头的前缀列表。建议使用X-Remote-Extra-。 | X-Remote-Extra- |
| requestheader-group-headers | 用于检查群组的请求头列表。建议使用X-Remote-Group. | X-Remote-Group |
| requestheader-username-headers | 用于检查用户名的请求头列表。建议使用X-Remote-User。 | X-Remote-User |
| runtime-config | 传递给apiserver用于描述运行时配置的键值对集合。 |  |
| secure-port | 用于监听具有认证授权功能的HTTPS协议的端口。如果为0，则不会监听HTTPS协议。（默认值6443) | 6443 |
| Service-account-key-file | 包含PEM加密的x509RSA或ECDSA私钥或公钥的文件，用于验证ServiceAccount令牌。如果设置该值，--tls-private-key-file将会被使用。指定的文件可以包含多个密钥，并且这个标志可以和不同的文件一起多次使用。 | /etc/kubernetes/ssl/sa.pub |
| Service-cluster-ip-range | CIDR表示的IP范围，服务的clusterip将从中分配。一定不要和分配给nodes和Pods的IP范围产生重叠。 | 10.14.0.0/15 |
| Service-node-port-range | 设定一个端口的范围，用来保留 NodePort 可见的服务（默认值：30000-32767）。两端均包含在内。 | 30000-32767 |
| storage-backend | 持久化存储后端。选项为:'etcd3'(默认),'etcd2'. | etcd3 |
| tls-cert-file | 包含用于HTTPS的默认x509证书的文件。（如果有CA证书，则附加于server证书之后）。如果启用了HTTPS服务，并且没有提供--tls-cert-file和--tls-private-key-file，则将为公共地址生成一个自签名的证书和密钥并保存于/var/run/kubernetes目录。 | /etc/kubernetes/ssl/apiserver.crt |
| tls-private-key-file | 包含匹配--tls-cert-file的x509证书私钥的文件。 | /etc/kubernetes/ssl/apiserver.key |

### kube-controller-manager参数

|  |  |  |
| --- | --- | --- |
| 项目 | 说明 | 示例 |
| allocate-node-cidrs | 是否应在云提供商上分配和设置Pod的CIDR。 | true |
| authentication-kubeconfig | kubeconfig文件指向有权创建tokenaccessreviews.authentication.k8s.io的kubernetes服务器。这是可选的。如果是空的，所有令牌请求都被视为匿名，并且不会在集群中查找客户端CA | /etc/kubernetes/controller-manager.conf |
| authorization-kubeconfig | kubeconfig文件指向 'core' kubernetes服务器。 这是可选的。 如果是空的，所有未经授权的请求将被拒绝。 | /etc/kubernetes/controller-manager.conf |
| bind-address | 监听--seure-port的IP地址。被关联的接口必须能够被集群其它节点和CLI/web客户端访问。如果为空，则将使用所有接口（0.0.0.0）。（默认值0.0.0.0） | 0.0.0.0 |
| client-ca-file | 如果设置此标志，对于任何请求，如果存包含client-ca-file中的authorities签名的客户端证书，将会使用客户端证书中的CommonName对应的身份进行认证。 | /etc/kubernetes/ssl/ca.crt |
| cluster-cidr | 集群中Pod的CIDR范围。要求--allocate-node-cidrs为true | 10.32.0.0/13 |
| cluster-signing-cert-file | 一个PEM编码的有X509 CA证书的文件，用于在集群内发布证书 | /etc/kubernetes/ssl/ca.crt |
| cluster-signing-key-file | 一个PEM编码的有RSA或ECDSA私钥的文件，用于对集群内的证书进行签名 | /etc/kubernetes/ssl/ca.key |
| configure-cloud-routes | 是否应在云提供商上配置allocate-node-cidrs分配的CIDR | false |
| controllers | 要启用的控制器列表。 '\*'启用所有默认控制器，'foo'启用名为'foo'的控制器，'-foo'禁用名为 foo 的控制器 | \*,bootstrapsigner,tokencleaner |
| kubeconfig | 包含授权和master位置信息的kubeconfig文件的路径。 | /etc/kubernetes/controller-manager.conf |
| leader-elect | 默认:true在执行主循环之前，启动leader选举客户端并获得领导能力。在运行复制组件以实现高可用性时启用此选项。 | true |
| node-cidr-mask-size | 集群中节点cidr的掩码大小。 | 24 |
| node-monitor-grace-period | 在标记节点不健康之前，允许运行节点不响应的时间。必须是kubelet的 nodeStatusUpdateFrequency 的 N 倍，N表示允许kubelet发布节点状态的重试次数。 | 40s |
| node-monitor-period | 在NodeController中同步NodeStatus的时间间隔。 | 5s |
| Pod-eviction-timeout | 删除故障节点上的Pod的宽限期。 | 5m0s |
| requestheader-client-ca-file | 在信任请求头中以--requestheader-username-headers指示的用户名之前，用于验证接入请求中客户端证书的根证书捆绑。 | /etc/kubernetes/ssl/front-proxy-ca.crt |
| root-ca-file | 根 CA 证书文件路径，将被用于 Service Account 的 token secret 中 | /etc/kubernetes/ssl/ca.crt |
| Service-account-private-key-file | 用于签署 Service account tokens 的 PEM 编码的RSA或ECDSA密钥文件 | /etc/kubernetes/ssl/sa.key |
| use-Service-account-credentials | 如果为true，为每个控制器使用单个Service account | true |

### kube-scheduler参数

|  |  |  |
| --- | --- | --- |
| 项目 | 说明 | 示例 |
| bind-address | 监听--seure-port的IP地址。被关联的接口必须能够被集群其它节点和CLI/web客户端访问。如果为空，则将使用所有接口（0.0.0.0）。（默认值0.0.0.0） | 0.0.0.0 |
| kubeconfig | 包含授权和master位置信息的kubeconfig文件的路径。 | /etc/kubernetes/scheduler.conf |
| leader-elect | 默认:true在执行主循环之前，启动leader选举客户端并获得领导能力。在运行复制组件以实现高可用性时启用此选项。 | true |

### kubelet参数

|  |  |  |
| --- | --- | --- |
| 项目 | 说明 | 示例 |
| bootstrap-kubeconfig | 如果 --kubeconfig 指定的文件不存在，kubelet 使用该文件中的用户名和 token 向 kube-apiserver 发送 TLS Bootstrapping 请求；管理员通过了 CSR 请求后，kubelet 自动在 --cert-dir 目录创建证书和私钥文件(kubelet-client.crt 和 kubelet-client.key)，然后写入 --kubeconfig 文件； | /etc/kubernetes/bootstrap-kubelet.conf |
| kubeconfig | kubeconfig文件的路径，指定如何连接到API server。 | /etc/kubernetes/kubelet.conf |
| authentication-token-webhook | 使用 TokenReview API 确定承载令牌的身份验证。 |  |
| enforce-node-allocatable | kubelet强制执行的节点可分配强制执行级别的逗号分隔列表。可选项是'none'，'Pods'，'system-reserved'和'kube-reserved'。如果指定后两个选项，则还必须分别设置'--system-reserved-cgroup'和'--kube-reserved-cgroup'。如果指定'none'，不用设置其他选项。 |  |
| client-ca-file | 启用客户端证书认证。该参数引用的文件中必须包含一个或多个证书颁发机构，用于验证提交给该组件的客户端证书。如果客户端证书已验证，则用其中的 Common Name 作为请求的用户名 | /etc/kubernetes/ssl/ca.crt |
| rotate-certificates | 证书过期时，从kube-apiserver请求新证书来自动更新kubelet客户端证书。 |  |
| Pod-manifest-path | 包含要运行的静态Pod文件的路径，或单个静态Pod文件的路径。以点开头的文件将被忽略。 | /etc/kubernetes/manifests |
| Pod-infra-container-image | 每个Pod中的network / ipc命名空间容器会使用的映像。此参数仅在container-runtime设置为docker时有效。 | gcr.azk8s.cn/google\_containers/pause-amd64:3.1 |
| node-status-update-frequency | 指定kubelet将节点状态发布到master的频率。该参数必须与nodecontroller中的nodeMonitorGracePeriod一起使用。 | 10s |
| cgroup-driver | kubelet 用于操作主机上的cgroup的驱动程序。可能的值：'cgroupfs'，'systemd' | cgroupfs |
| max-Pods | 可以在此Kubelet上运行的Pod数。 | 110 |
| anonymous-auth | 启用对访问 API server 安全端口的匿名请求。没有被其他身份验证方法拒绝的请求被视为匿名请求。匿名请求在系统中的用户名：anonymous，组名：unauthenticated。 | false |
| read-only-port | Kubelet的只读端口，无需认证/授权即可服务（设置为0以禁用） | 0 |
| fail-swap-on | 如果在节点上启用swap，则使Kubelet无法启动。 | TRUE |
| dynamic-config-dir | Kubelet将使用此目录检查下载的配置并跟踪配置运行状况。如果该目录尚不存在，Kubelet将创建该目录。路径可以是绝对的或相对的，相对路径从Kubelet的当前工作目录开始。提供此参数会启用动态Kubelet配置。必须启用DynamicKubeletConfig功能门才能传递此参数，该功能是测试版。 | /etc/kubernetes/dynamic\_kubelet\_dir |
| runtime-cgroups | 要创建和运行runtime里的cgroup的可选名称。 | /systemd/system.slice |
| kubelet-cgroups | 用于创建和运行Kubelet的cgroups的可选名称。 | /systemd/system.slice |
| cluster-dns | DNS服务器IP地址列表。对于具有“dnsPolicy = ClusterFirst”的Pod，此值用于容器DNS服务器。注意：列表中的所有DNS服务器必须提供相同的记录集，否则集群的名称解析可能无法正常工作。无法保证可以联系哪个DNS服务器进行名称解析。 | 169.254.25.10 |
| cluster-domain | 集群的域名。如果设置，除了主机的搜索域之外，kubelet还将配置所有容器以搜索此域 | k8s.sh |
| resolv-conf | 用作容器DNS解析配置的基础解析器配置文件。 | /etc/resolv.conf |
| kube-reserved | 一组ResourceName = ResourceQuantity 对（例如cpu = 200m，memory = 500Mi，ephemeral-storage = 1Gi）， 表示为kubernetes系统组件保留的资源。前支持根文件系统的cpu，内存和本地临时存储。 | cpu=200m,memory=512M |
| node-labels | 在集群中注册节点时要添加的标签。 |  |

### kube-proxy参数

|  |  |  |
| --- | --- | --- |
| 项目 | 说明 | 示例 |
| bind-address | kube-proxy 提供服务的 IP 地址 | 0.0.0.0 |
| cluster-cidr | 集群中 Pod 的CIDR范围。 | 10.32.0.0/13 |
| config-sync-period | 刷新 apiserver 的配置的频率，必须大于0. | 15m0s |
| conntrack-max-per-core | 每个CPU核跟踪的最大NAT连接数（0表示按原样保留限制并忽略conntrack-min） | 32768 |
| conntrack-min | 要分配的最小conntrack条目数，不管conntrack-max-per-core是多少 | 131072 |
| conntrack-tcp-timeout-close-wait | CLOSE\_WAIT状态下TCP连接的NAT超时 | 1h0m0s |
| conntrack-tcp-timeout-established | TCP连接的空闲超时（0保持原样） | 24h0m0s |
| healthz-bind-address | 健康检查服务器提供服务的IP地址及端口（对于所有IPv4接口设置为0.0.0.0，对于所有IPv6接口设置为::） | 0.0.0.0 |
| healthz-port | 健康检查服务器提供服务的端口 | 10256 |
| hostname-override | 如果非空，则使用此字符串作为标识而不是实际的主机名。 | test-work01 |
| iptables-masquerade-bit | 如果使用纯iptables代理，则表示fwmark空间的位，用于标记需要SNAT的数据包。必须在[0,31]范围内 | 14 |
| iptables-min-sync-period | 当端点和服务发生变化时，iptables规则的刷新频率的最小间隔 | 0s |
| iptables-sync-period | 刷新iptables规则的最大间隔时间间隔（例如'5s'，'1m'，'2h22m'）。必须大于0. | 30s |
| ipvs-exclude-cidrs | ipvs proxier在清理IPVS规则时不触及的CIDR列表 | null |
| ipvs-min-sync-period | 在端点和服务发生变化时，ipvs规则的刷新频率的最小间隔 | 0s |
| ipvs-scheduler | 代理模式为ipvs时的ipvs调度程序类型 | rr |
| ipvs-sync-period | 刷新ipvs规则的最大间隔时间间隔。必须大于0 | 30s |
| kube-api-burst | 在与kubernetes apiserver通信时使用Burst | 10 |
| kube-api-content-type | 发送到apiserver的请求的内容类型。 | application/vnd.kubernetes.protobuf |
| kube-api-qps | 与 kubernetes apiserver 通信时使用的QPS | 5 |
| kubeconfig | 具有授权信息的kubeconfig文件的路径（master | /var/lib/kube-proxy/kubeconfig.conf |
| masquerade-all | 如果使用纯 iptables 代理，对所有通过集群 Service IP发送的流量进行 SNAT（通常不配置） | false |
| metrics-bind-address | metrics | 127.0.0.1:10249 |
| nodeport-addresses | 一个字符串值的片段，指定用于NodePorts的地址。 | [] |
| oom-score-adj | kube-proxy进程的oom-score-adj值。 | -999 |
| profiling | 通过Web界面启用性能分析 | false |
| proxy-mode | 使用哪种代理模式.'userspace'（较旧）或'iptables'（较快）或'ipvs'（实验）。如果选择了iptables代理，只要系统的内核或iptables版本不足，总是会回退到用户空间代理。 | ipvs |
| proxy-port-range | 主机端口的范围（beginPort-endPort，单端口或beginPort + offset），以便代理服务流量。如果（未指定，0或0-0）则随机选择端口。 |  |
| udp-timeout | 空闲UDP连接将打开多久（例如'250ms'，'2s'）。 | 250ms |

### dockerd参数

|  |  |  |
| --- | --- | --- |
| 项目 | 说明 | 示例 |
| registry-mirror | 首选的docker registry | <https://tu6zof29.mirror.aliyuncs.com> |
| data-root | 容器持久化根目录 | /var/lib/docker |
| log-opt | 设定日志驱动选项 | [max-size=50m, max-file=5] |
| iptables | 启用额外的iptables规则添加 | false |
| dns | 设置自定义DNS服务器 | 114.114.114.114 |
| dns-opt | 设置DNS选项 | [ndots:2, timeout:2 ,attempts:2] |
| dns-search | 设置自定义的DNS搜索域 | [default.svc.k8s.sh, svc.k8s.sh] |

## 集群访问

### 集群外创建资源流程

kube-apiserver做为kubernetes集群对外的唯一入口，云管理平台通过API来访问集群，对集群资源对象进行增删改查，并可以通过List/Watch接口来探测集群资源的变化。下面描述kubernetes集群里的Pod，ReplicaSet，Service的创建流程。

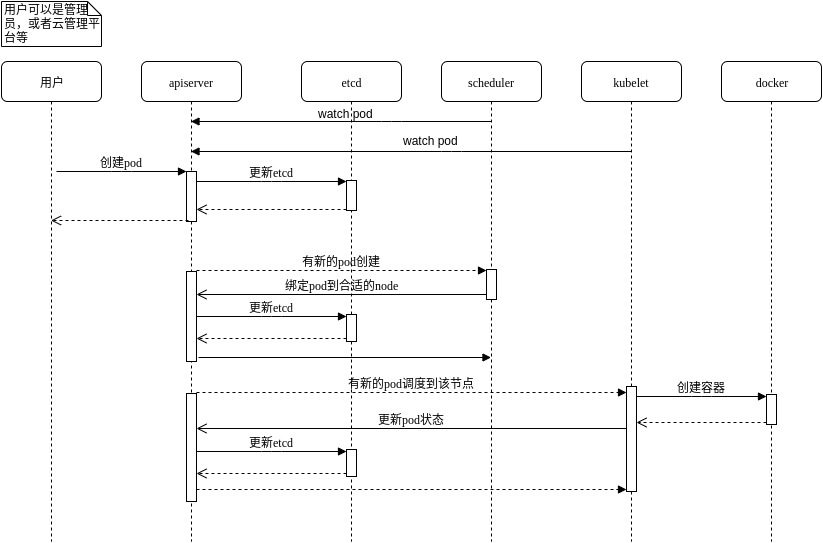


图1. Pod创建流程

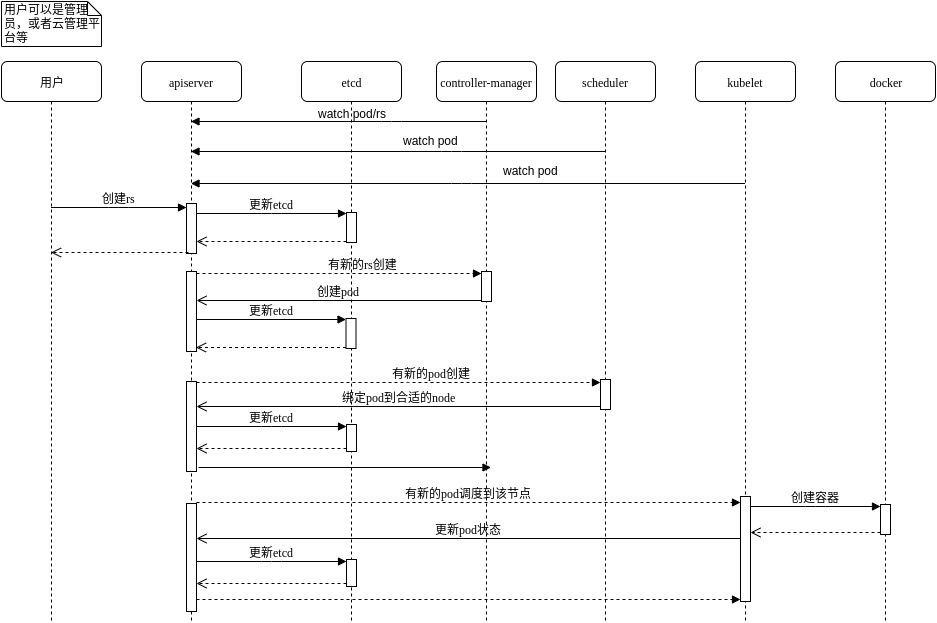


图2. ReplicaSet创建流程

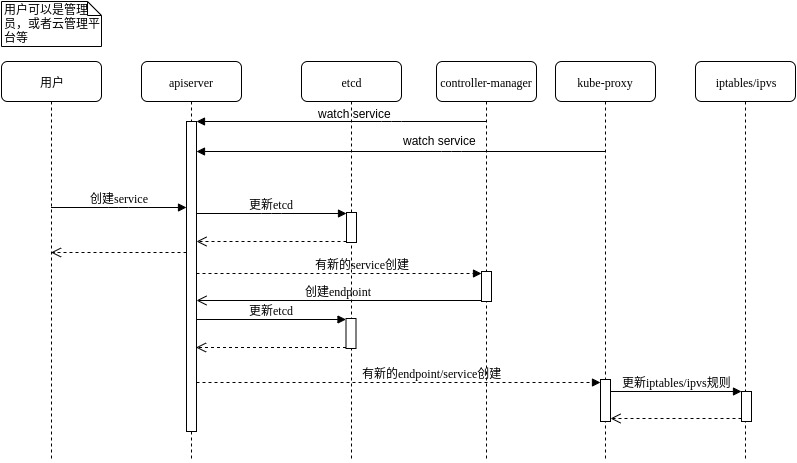
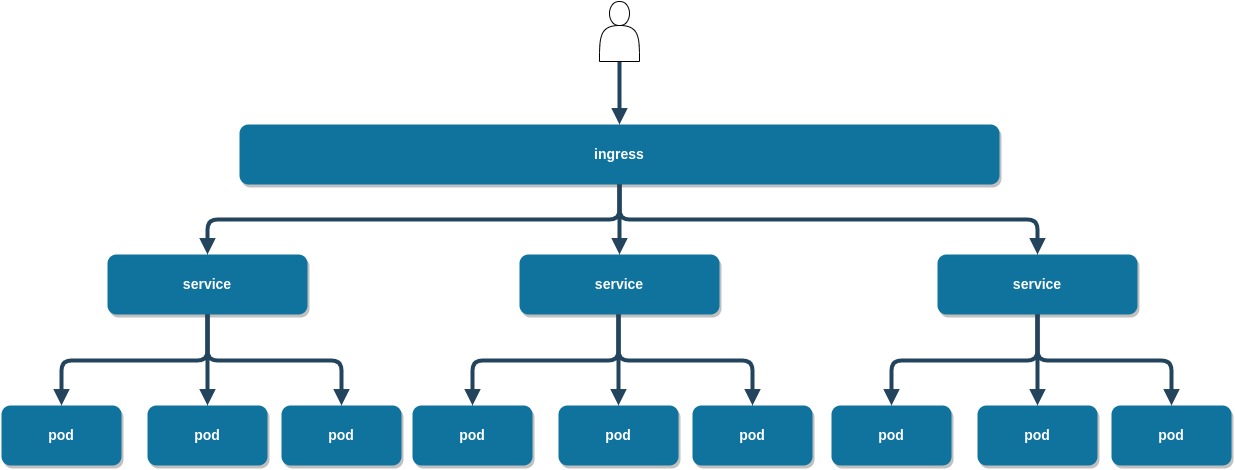


图3. Service创建流程

### 集群外访问部署在集群内业务的流程

在 Kubernetes 中通过 Ingress 来暴露服务到集群外部，Ingress资源对象，用于将不同URL的访问请求转发到后端不同的Service，Ingress 对象与一个或多个 [Service](https://cloud.google.com/kubernetes-engine/docs/concepts/service?hl=zh-cn) 对象相关联，每个 Service 对象与一组 Pod 相关联。

图4. 流量路径

### 创建资源对象

用户创建服务一般是创建一个Deployment, Deployment管理一组ReplicaSet，用来实现滚动升级和回滚等功能。ReplicaSet来维持Pod的副本数。可以根据业务的需求指定Pod所需的资源，包括内存，CPU，GPU等。按照资源的指定方式，Pod的QoS分为3个等级：Guaranteed，Burstable，BestEffort。

Guaranteed即指定资源的limits等于requests， 此类Pod的优先级最高。Burstable是指定limits小于requests，这种情况下相当于超售。而BestEffort是不制定limits和requests的至，该类Pod的的优先级最低，在Node资源紧张的时候，会被最先杀掉。

上层云管理平台，可以根据应用的优先级，在创建资源的时候通过指定合适的limits和reqests的值来达到应用分级的目的。

下图简要介绍了kubernetes核心资源的逻辑关系。

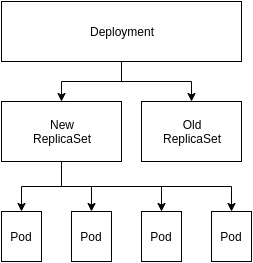


图5. 资源关系

## 集群部署

kubernetes集群采用kubespray来部署，需要在部署节点安装ansible. 安装步骤：

### **下载**kubespray代码

# cd ~

# git clone <https://gitlab.oneitfarm.com/idg-public/kubespray.git>

### 修改配置文件

根据对应的环境修改hosts.ini，<env>实际的环境来选择。

# cd kubespray

# vi inventory/<env>/hosts.ini

在hosts.ini里添加master和node的信息。格式如下表中的例子所示：

[all]

test-master01 ansible\_host=192.168.3.142 ip=192.168.3.142

test-master02 ansible\_host=192.168.3.143 ip=192.168.3.143

test-master03 ansible\_host=192.168.3.144 ip=192.168.3.144

test-work02 ansible\_host=192.168.3.152 ip=192.168.3.152

[kube-master]

test-master01

test-master02

test-master03

[kube-node]

test-work02

[etcd:children]

kube-master

[k8s-cluster:children]

kube-node

kube-master

[calico-rr]

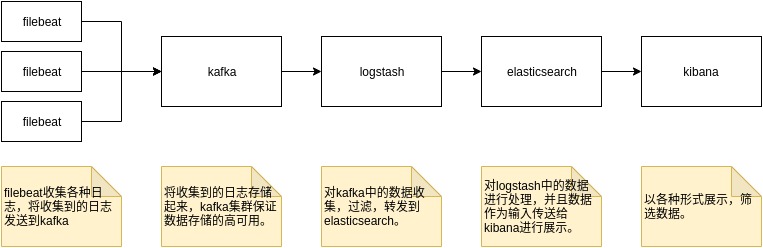
### 部署集群

# ansible-playbook cluster.yml -i inventory/<env>l/hosts.ini -v

## 集群管理

### 日志

采用ELK+kafka+filebeat的方案来做日志管理。



### 监控告警

通过prometheus配合各种exporter来采集集群的各种指标。

| **级别** | **监控什么** | **Exporter** |
| --- | --- | --- |
| 网络 | 网络协议：http、dns、tcp、icmp；网络硬件：路由器，交换机等 | BlockBox Exporter;SNMP Exporter |
| 主机 | 资源用量 | node exporter |
| 容器 | 资源用量 | cAdvisor |
| 应用 | 延迟，错误，QPS，内部状态等 | 代码中集成Prmometheus Client |
| 中间件状态 | 资源用量，以及服务状态 | 代码中集成Prmometheus Client |
| 编排工具 | 集群资源用量，调度等 | Kubernetes Components |

可以按照以下四个维度来设置告警规则。具体的阈值可以先设置一个值，根据经验来调整。

* 延迟：集群管理面主要监控apiserver， coredns等的请求延迟，比如apiserver请求延迟大于500ms产生一条告警信息。
* 通信量：监控apiserver 每秒请求数。
* 错误：监控当前系统所有发生的错误请求，衡量当前系统错误发生的速率，比如15秒内产生HTTP 4XX错误大于千分之1的时候告警。
* 饱和度：包括宿主机的磁盘，内存，CPU，网络的使用率，根据指标类型不同可以设置不同的阈值，比如磁盘使用率大于90%，网络使用带宽率达到总带宽80%产生告警。

告警通知的方式可以选择邮件，企业微信。也可以根据具体需求实现相应的webhook来做告警通知。