



GOPS2017
Shenzhen



全球运维大会

2017



深圳站

指导单位：



主办单位：



魅族容器云平台基于k8s的自动化运维实践

曾彬

目录



1

基本介绍

2

K8s集群

3

容器网络

4

外部访问4/7层

5

监控/告警/日志

6

业务发布/镜像/多机房

基本介绍

- 定位

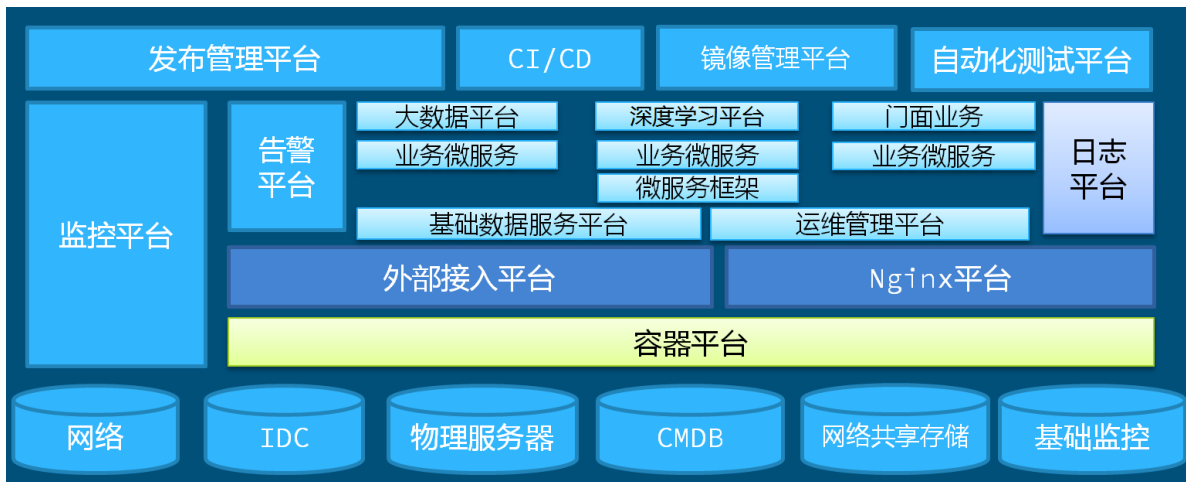
- 私有云平台
- 网站业务+高级业务
- 替换传统虚拟化

- 现状

- 网站业务，17年完成90%迁移
- 3个数据中心

- 运作方式

- 低成本试错，几个人小团队
- 紧跟k8s步伐
- 局部和关键创新
- 保证核心稳定，重视非功能性



目录

1 基本介绍

➔ 2 K8s集群

3 容器网络

4 外部访问4/7层

5 监控/告警/日志

6 业务发布/镜像/多机房

K8s集群-单一镜像

1. 快速部署/升级，docker run一键安装
2. Hyperkube Image
 - hyperkube二进制
 - 安装/升级脚本
 - kubelet service
 - static pod yml
 - 自签证书
3. Master安装/升级
 - apiserver/scheduler/manager
4. Minion安装/升级
 - kubelet/kube-proxy

meizu/hyperkube:
1.5.2

apiserver.yaml
scheduler.yaml
manager.yaml
kube-proxy.yaml

master.sh
minion.sh

kubelet.service

Cert files

hyperkube Binary

K8s集群-Master核心组件

1. Static Pod保证自动加载

- API Server
- Controller
- Scheduler

kube-apiserver-	1/1	Running	0	20d
kube-apiserver-	1/1	Running	3	20d
kube-controller-manager-	1/1	Running	0	20d
kube-controller-manager-	1/1	Running	0	20d
kube-scheduler-	1/1	Running	0	20d
kube-scheduler-	1/1	Running	0	20d

2. 自动修复

- Liveness probe
- 自动重启

3. 自动升级

- 指定新的hyperkube version

K8s集群-Master高可用设计

1. API Server

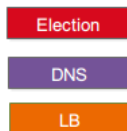
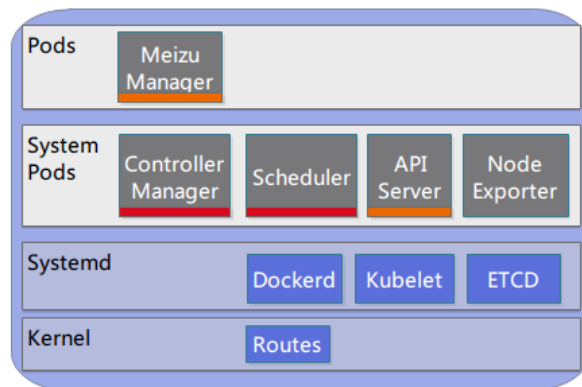
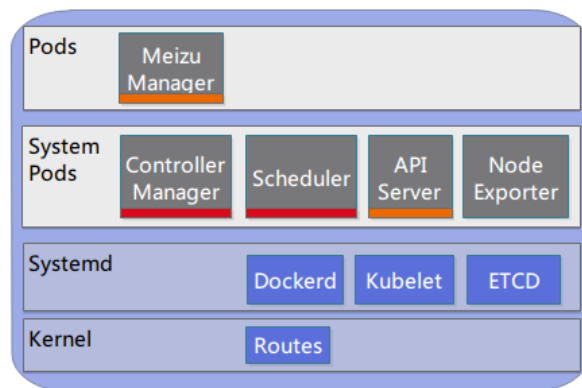
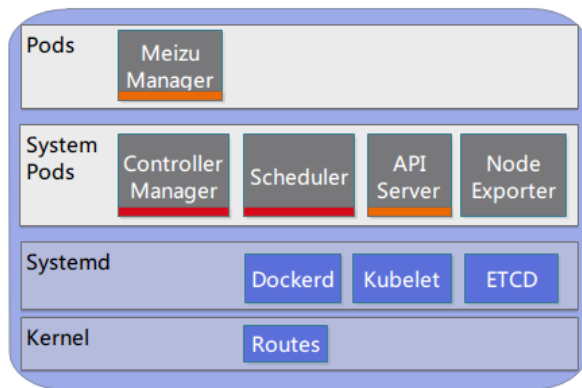
- LB or DNS
- LB多活，DNS单活

2. Scheduler

- Leader Election

3. Controller

- Leader Election



K8s集群-Master异常处理

1. k8s核心Pod运行状况

- 重启告警
- Leader Election导致Pod重启

2. 频繁的Leader Election

- 时钟同步?
- 与API Server通信不稳定?
- 重点关注Controller Manager的重启，及时介入检查有无异常状态

```
ALERT KubeControllerManagerRestart
IF changes(kube_pod_container_status_restarts{container="kube-controller-manager"}[2m]) != 0
LABELS {severity="warn", type="action"}
ANNOTATIONS {description="environment {{ $labels.env }}s {{ $labels.pod }} is restart for {{ $labels.pod }}
```

K8s集群-Minion常用配置

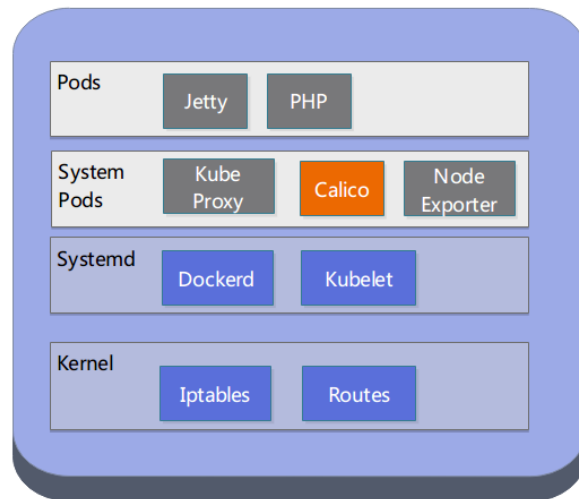
1. 硬件

- E5-2620 24core(HT)
- 内存 128GB
- Intel千兆

2. 参数调整

- 中断相关
- TCP backlog
- Swapness

```
Capacity:
  alpha.kubernetes.io/nvidia-gpu:      0
  cpu:                                  32
  memory:                               131987564Ki
  pods:                                 110
Allocatable:
  alpha.kubernetes.io/nvidia-gpu:      0
  cpu:                                  32
  memory:                               131987564Ki
  pods:                                 110
System Info:
  Machine ID:                           1dd00654bd744a65a6a94879
  System UUID:                           4C4C4544-0032-4810-804D-
  Boot ID:                               fb1bbc39-020f-49c6-a9a7-
  Kernel Version:                        3.16.41
  OS Image:                              CentOS Linux 7 (Core)
  Operating System:                      linux
  Architecture:                          amd64
  Container Runtime Version:              docker://1.12.0
  Kubelet Version:                        v1.5.2
  Kube-Proxy Version:                     v1.5.2
```



K8s集群-Minion-存储

1. Docker存储

- DeviceMapper
- Direct LVM
- 实际用量并不大

└─sda5	198.2G	0	part
└─┬─VG-docker--pool_tmeta	204M	0	lvm
└─└─VG-docker--pool	188.1G	0	lvm

2. 本地普通存储

- 日志/临时文件等存储于EmptyDir，EmptyDir映射到本地目录
- 大量读写
- 修改EmptyDir落地路径 --root-dir=/data/kubelet (默认 /var/lib/kubelet)
- 考虑使用普通分区
- 使用lvm参数设置不当导致meta full

sdb	8:16	0	1.1T	0	disk
└─sdb1	8:17	0	1.1T	0	part /data

K8s集群-Minion-存储踩坑

- Kernel issue
 - [Bug 1292481](https://bugzilla.redhat.com/show_bug.cgi?id=1292481) - device mapper hung tasks on an openshift/docker system
 - https://bugzilla.redhat.com/show_bug.cgi?id=1292481
- So,经常更新内核

Gerard Ryan 2016-02-18 06:34:38 EST

Comment 25

Hi, is there a known date when we can expect this to be pushed to stable repos?

Jeremy Eder 2016-02-18 06:57:19 EST

Comment 26

Hi Gerard -- we shipped this fix on Tuesday Feb 16.

Do a yum update and make sure you get kernel-3.10.0-327.10.1.el7 or later.

There was a different BZ tracking the RHEL7.2.z kernel release:

https://bugzilla.redhat.com/show_bug.cgi?id=1296566

This BZ is to track the fix into RHEL7.3.

Bruno Goncalves 2016-07-04 04:17:48 EDT

Comment 30

Tested kernel-3.10.0-445.el7 and it did pass on our Regression test and the tickets are closed.

Setting SanityOnly as we are not able to reproduce the problem in our environment.

K8s集群-Minion-内核

- 版本

- OS Centos 7.2 , 3.10

- **bugfix back porting**

- 4.1以后,hacking

tcp_v4_syn_recv_sock

会有问题

- 我们的选择是**3.16**

Longterm release kernels

Version	Maintainer	Released	Projected EOL
4.9	Greg Kroah-Hartman	2016-12-11	Jan, 2019
4.4	Greg Kroah-Hartman	2016-01-10	Feb, 2018
4.1	Sasha Levin	2015-06-21	Sep, 2017
3.16	Ben Hutchings	2014-08-03	Apr, 2020
3.12	Jiri Slaby	2013-11-03	May, 2017
3.10	Willy Tarreau	2013-06-30	Oct, 2017
3.4	Li Zefan	2012-05-20	Apr, 2017
3.2	Ben Hutchings	2012-01-04	May, 2018

K8s集群-Minion-Label管理

1. 标签

```
xx.xx.52.22 Ready 80d calico=v2,kubernetes.io/hostname=xx.xx.52.22,logging=N3,rack=0702
```

- 结合CMDB，为Minion打上标签
- rack/网络/机型/功能等

2. 调度

- Node Affinity
- Pod Anti-Affinity

K8s集群-总结

1. 部署便利度

- 集群部署
- 集群升级

2. 高可用

- 核心组件进程
- 核心组件健康检测
- 核心组件冗余设计

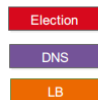
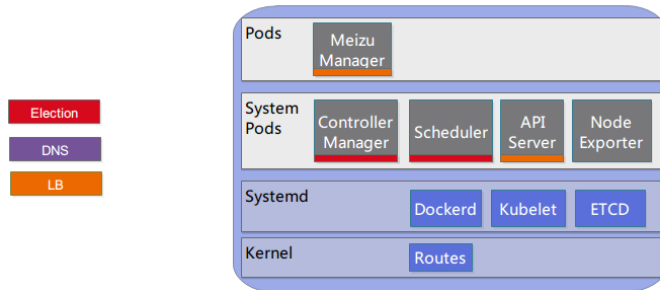
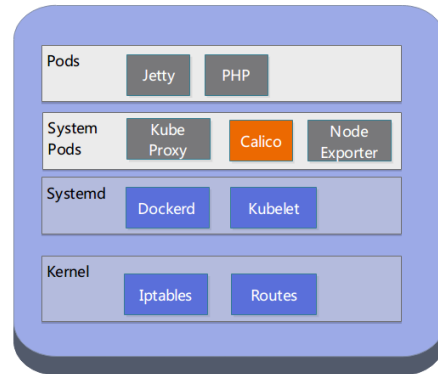
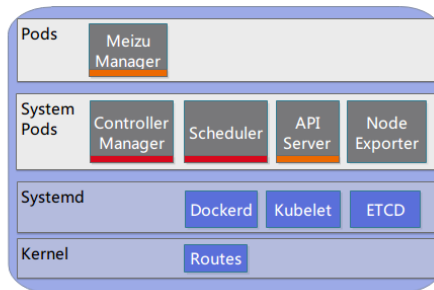
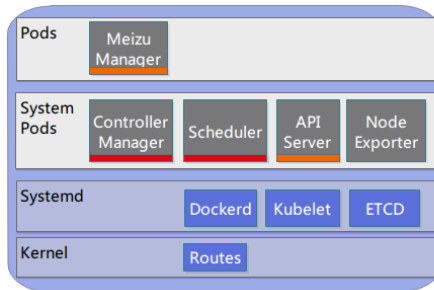
3. 存储

- 容器存储
- 非容器存储

4. 内核

- 版本
- 参数

5. 标签管理



目录

1 基本介绍

2 K8s集群

➔ 3 容器网络

4 外部访问4/7层

5 监控/告警/日志

6 业务发布/镜像/多机房

网络-整体

1. 方案

- Calico

2. 控制层面

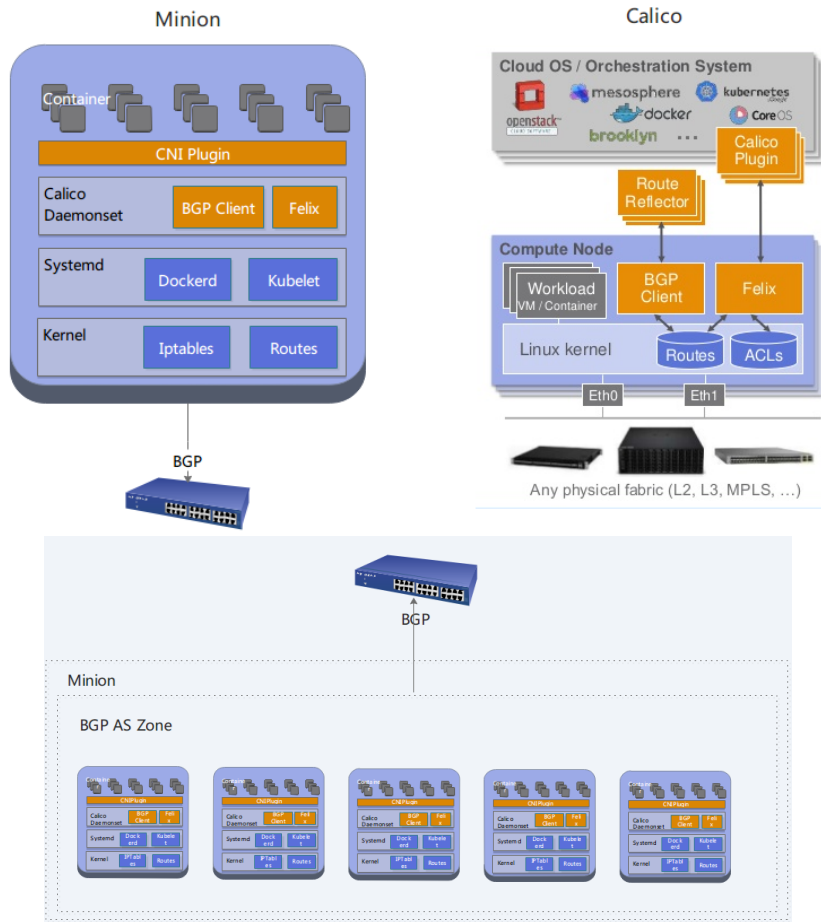
- BGP No Mesh
- Bird对接核心路由设备

3. 数据层面

- 3层路由
- Netfilter Conntrack
- Forward Chain

4. 部署

- Daemonset



网络-优化点

1. 配置

- 合理的Conntrack参数设置
- 使用headless service

```
--conntrack-max-per-core int32
--conntrack-min int32

--conntrack-tcp-timeout-close-wait duration
--conntrack-tcp-timeout-established duration
```

2. 监控告警

- 监控conntrack用量
- Calico Prometheus Met

```
# HELP node_nf_conntrack_entries Number of currently allocated flow entries for connection tracking.
# TYPE node_nf_conntrack_entries gauge
node_nf_conntrack_entries 406
# HELP node_nf_conntrack_entries_limit Maximum size of connection tracking table.
# TYPE node_nf_conntrack_entries_limit gauge
node_nf_conntrack_entries_limit 655350
```

3. Readness ping

- 容器Ping 核心交换机，确保网络联通性
- Readness检查fail，endpoint下线，并触发告警

网络-优化conntrack

1. Bypass Tracking

- Tracking多用于访问控制
- LVS过来的流量大，信任度高，Tracking意义不大
- Bypass LVS过来的流量

```
[root@ bin]iptables -t filter -L FORWARD
Chain FORWARD (policy ACCEPT)
target     prot opt source                destination
ACCEPT     all  --  10.0.0.0/24            anywhere
ACCEPT     all  --  anywhere              10.0.0.0/24
ACCEPT     all  --  10.0.0.0/24            anywhere
ACCEPT     all  --  anywhere              10.0.0.0/24
felix-FORWARD all  --  anywhere              anywhere
```

2. 方法

- felix insert改为append，保证自定义规则的最高优先级
- 在raw表PREROUTING和filter表FORWARD链添加规则

```
[root@10.0.0.1 bin]iptables -t raw -L PREROUTING
Chain PREROUTING (policy ACCEPT)
target     prot opt source                destination
CT         all  --  anywhere              10.0.0.0/24      NOTRACK
CT         all  --  10.0.0.0/24           anywhere         NOTRACK
CT         all  --  10.0.0.0/24           anywhere         NOTRACK
CT         all  --  anywhere              10.0.0.0/24      NOTRACK
```

网络-总结

1. 稳定性

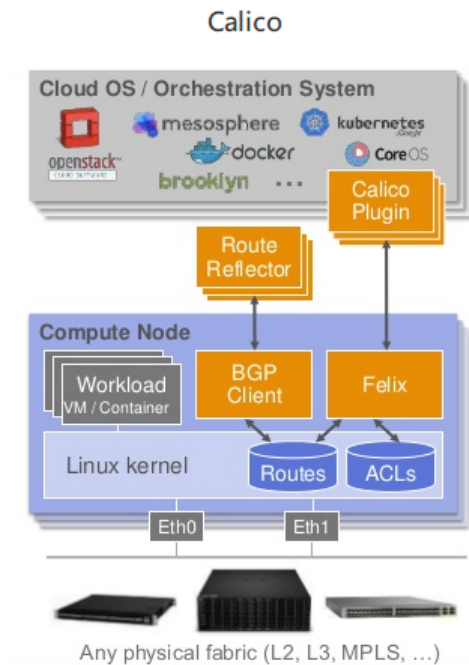
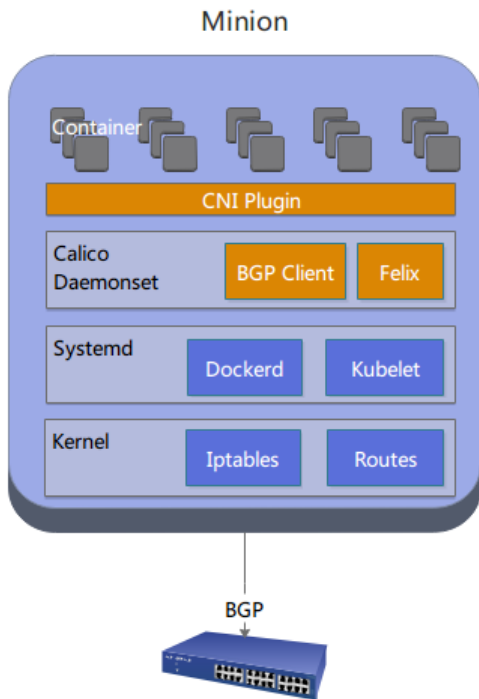
- mesh vs no mesh

2. 性能优化

- Netfilter Conntrack

3. 异常处理

- Pod主动检测网络
- Calico整体健康告警



目录

1 基本介绍

2 K8s集群

3 容器网络

➔ 4 外部访问4/7层

5 监控/告警/日志

6 业务发布/镜像/多机房

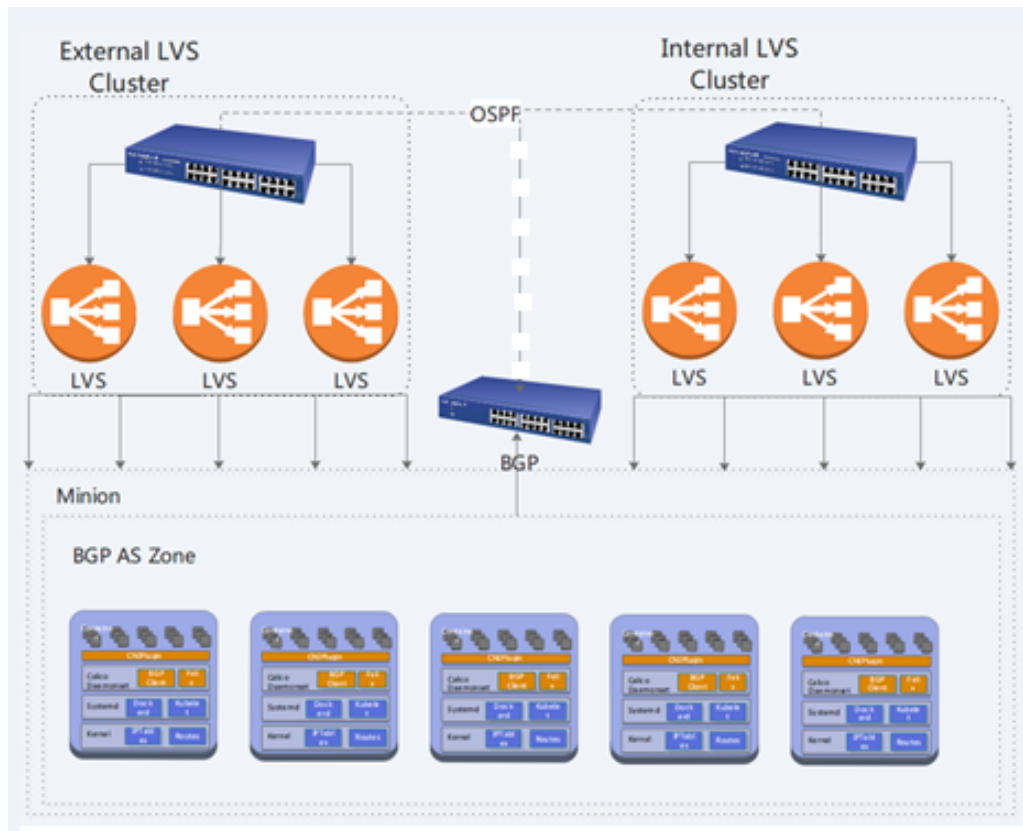
外部访问-4层

1. LVS集群

- Alibaba LVS
- FullNat
- 内/外集群

2. 流量进出

- Client <-> LVS <-> Minion <-> Pod
- Minion上不做Nat



外部访问-自动配置

- 自动化配置
 - VIP OSPF宣告（预先宣告，减少变更）
 - LVS配置
 - Virtual Service->K8s Service->Endpoints

```
apiVersion: v1
data:
  xx.xx.x3.225-80: yy.yy.y0.130:80, yy.yy.y51.20:80
  xx.xx.x3.225-443: yy.yy.y0.130:443, yy.yy.y51.20:443
  xx.xx.x3.226: gslb/nginx-ingress-controller-new-lb:s
  xx.xx.x3.227: openapi/openapi-nginx-lb:s
kind: ConfigMap
```

外部访问-4层-流量采集

- 流量采集

- 暴露Prometheus Metrics
- 业务信息(命名空间, 服务名)
- Virtual Service
- RealServer(Endpoint)

lvs
lvs_endpoint_stat_conns_total
lvs_endpoint_stat_inbytes_total
lvs_endpoint_stat_inpkts_total
lvs_endpoint_stat_outbytes_total
lvs_endpoint_stat_outpkts_total
lvs_service_stat_conns_total
lvs_service_stat_inbytes_total
lvs_service_stat_inpkts_total
lvs_service_stat_outbytes_total
lvs_service_stat_outpkts_total

```
lvs_endpoint_stat_conns_total{endpoint="10.0.0.1:8080",env="GGZ-YG",instance="10.0.0.1:8080",job="external-lvs",linkaddr="10.0.0.1:8080",namespace="gob",port="443",service="nginx-ingress-controller-new-lb",vport="443"}
```

5026006

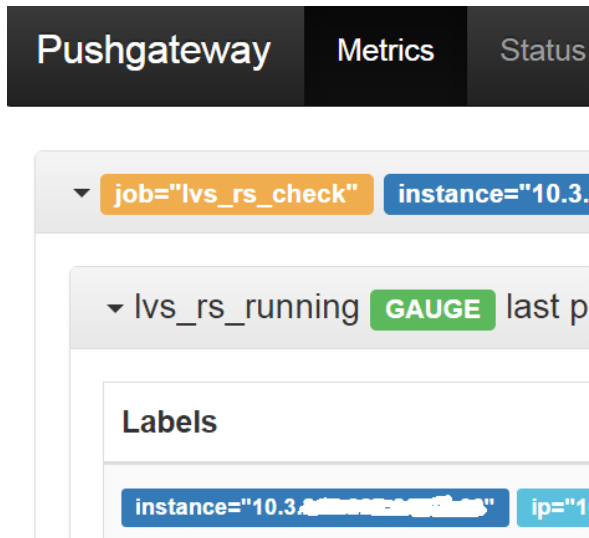
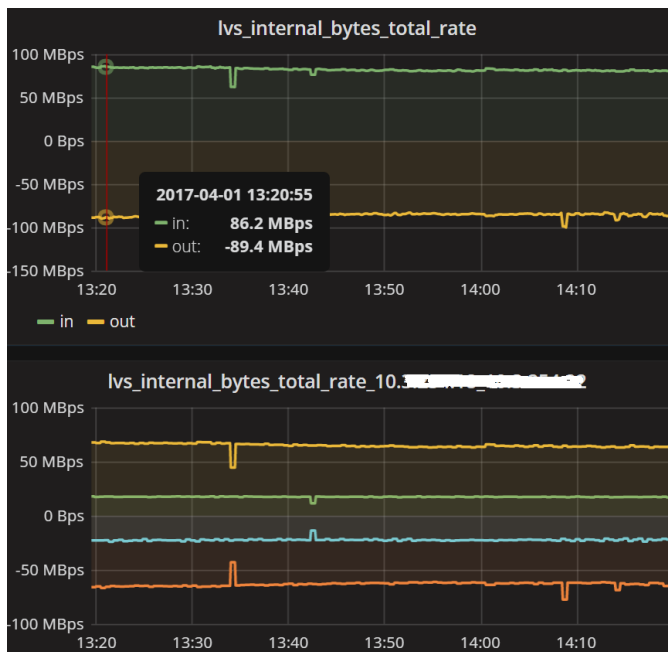
外部访问-4层-监控告警

1. 监控

- Grafana可视化
- 聚合集群数据

2. 告警

- 流量异常
- RealServer
MISC_CHECK
检测异常



外部访问-4层-总结

1. 性能/扩展性

- 方案选型

2. 与容器连通

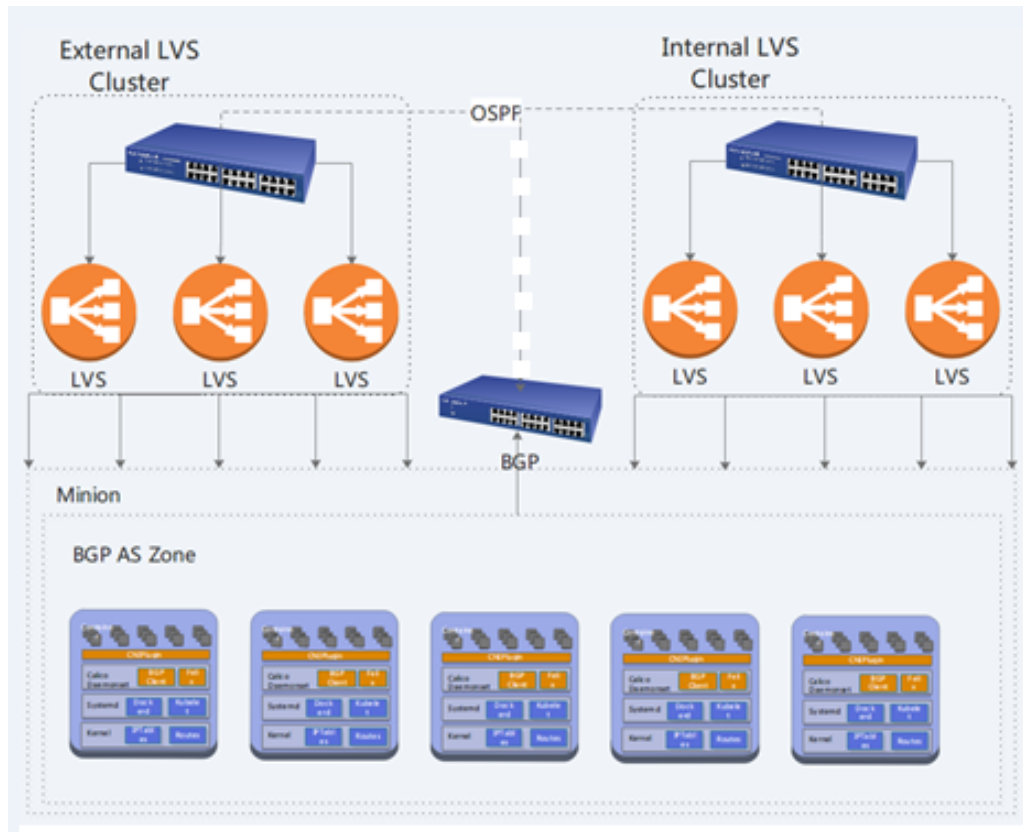
- 路由

3. 自动化运维

- 尽量减少手工配置

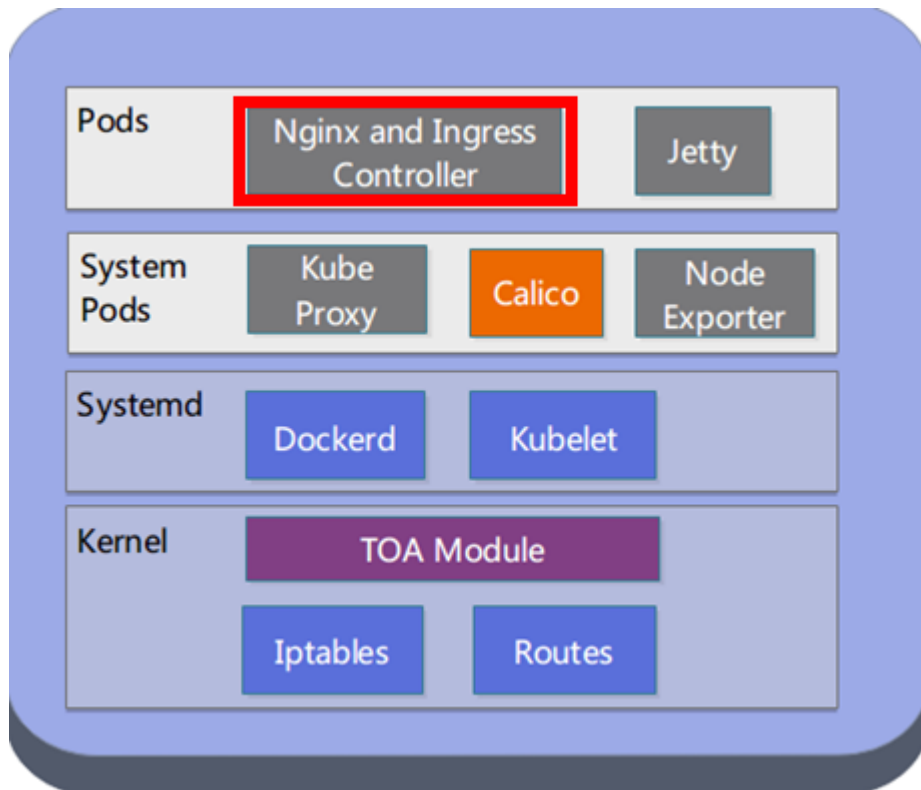
4. 监控告警

- 数据聚合
- 检测异常



外部访问-7层-Nginx

- Nginx
 - Nginx + Ingress Controller
 - 业务专属
 - 容器化
 - 自动伸缩



外部访问-Nginx-获取用户地址

- 获取client地址

- 默认获取到的是LVS的Local ip+Port , 要从TCP Option里”挖出来”

- TOA补丁只支持2.6.32

- 移植到3.16内核

```
[root@toy-docker-node-001 ~]# uname -r
3.16.41
[root@toy-docker-node-001 ~]# modinfo toa.ko
filename:       /root/toa-master/toa.ko
license:        GPL
srcversion:     BFA597D0C6615463A995C7D
depends:
vermagic:       3.16.41 SMP mod_unload modversions
```

```
Transmission Control Protocol, Src Port: 5015 (5015), Dst Port: 8080 (8080), Seq: 1, Ack: 1, Len: 0
Source Port: 5015
Destination Port: 8080
[Stream index: 0]
[TCP Segment Len: 0]
Sequence number: 1 (relative sequence number)
Acknowledgment number: 1 (relative ack number)
Header Length: 28 bytes
Flags: 0x010 (ACK)
Window size value: 29200
[Calculated window size: 29200]
[Window size scaling factor: -2 (no window scaling used)]
Checksum: 0x2fb2 [validation disabled]
Urgent pointer: 0
Options: (8 bytes), Experimental
  TCP Option - Experimental: fe08b5f3ac10108a
    Kind: RFC3692-style Experiment 2 (254)
    Length: 8
    Magic Number: 0xb5f3
  [SEQ/ACK analysis]
```

```
0000 00 00 00 01 00 06 00 50 56 92 53 0f 00 00 00 00 .....P V.S....
0010 45 00 00 30 ee 82 40 00 3c 06 c6 f7 ac 11 20 82 E..@.<.....
0020 c0 a8 fc 11 13 97 1f 90 c7 69 a0 5d fd e7 5b 4e .....i.]..[N
0030 70 10 72 10 2f b2 00 00 fe 08 b5 f3 ac 10 10 8a p.r./... ..
```

```
cat /proc/net/toa_stats
CPU16 CPU17 CPU18 CPU19 CPU20 CPU21 CPU22 CPU23 CPU24 CPU25 CPU26 CPU27 CPU28 CPU29 CPU30
syn_recv_sock_toa : 210722769 229438508 23346715 237758960 242640661 252758640 235125087 245413511 169267640 166447970 165505915
syn_recv_sock_no_toa : 171039454 173819700 171210801 174446579 173172526 193848363 189489481 158893341 162995508 159397445 168723939 168867308 173304645
112938 109978 : 146094 176650 174723 168299 177199 190807 163891 172783 130418 124577 121835
getname_toa_ok : 272245758 294641947 306976548 319348847 336113338 366164731 345077318 377157575 143122568 134549520 131691328
138929167 146699347 : 139310452 148320882 145908581 178219239 165723624 105366604 122985066 112452022 116746135 145473735 160868420
getname_toa_mismatch : 0 0 0 0 0 0 0 0 0 0 0 0 0
getname_toa_bypass : 0 0 0 0 0 0 0 0 0 0 0 0 0
getname_toa_empty : 241241563 262152225 273675863 286299214 301930384 331874612 313484582 345615230 134632947 127203429 124736255
148897891 155287161 : 147717479 156235587 154023483 185648664 174120399 115294143 132404895 121764088 126222961 154041546 169274242
```

外部访问-Nginx-性能

1. 高延迟问题

- 观察到upstream的响应时间长达数秒,实际后端服务一直正常

2. 解决方案

- worker_processes auto , worker数大过limit较多, worker可能被饿死
 - 合理设置, 例如worker=4, limit=5
- worker_cpu_affinity auto , 导致前几个核占用率高
 - 禁用worker_cpu_affinity

```
Tasks: 862 total, 12 running, 850 sleeping, 0 stopped, 0 zombie
%Cpu0  : 35.1 us, 3.3 sy, 0.0 ni, 58.2 id, 0.0 wa, 0.0 hi, 3.3 si, 0.0 st
%Cpu1  : 35.2 us, 3.1 sy, 0.0 ni, 56.3 id, 0.0 wa, 0.0 hi, 5.5 si, 0.0 st
%Cpu2  : 39.5 us, 4.4 sy, 0.0 ni, 50.3 id, 0.0 wa, 0.0 hi, 5.8 si, 0.0 st
%Cpu3  : 35.3 us, 4.1 sy, 0.0 ni, 55.6 id, 0.0 wa, 0.0 hi, 5.1 si, 0.0 st
%Cpu4  : 45.8 us, 3.7 sy, 0.0 ni, 46.1 id, 0.0 wa, 0.0 hi, 4.4 si, 0.0 st
%Cpu5  : 33.0 us, 4.1 sy, 0.0 ni, 58.2 id, 0.0 wa, 0.0 hi, 4.8 si, 0.0 st
%Cpu6  : 35.5 us, 3.7 sy, 0.0 ni, 58.1 id, 0.0 wa, 0.0 hi, 2.7 si, 0.0 st
%Cpu7  : 38.9 us, 4.7 sy, 0.0 ni, 53.4 id, 0.0 wa, 0.0 hi, 3.0 si, 0.0 st
%Cpu8  : 44.7 us, 4.3 sy, 0.0 ni, 47.0 id, 0.3 wa, 0.0 hi, 3.7 si, 0.0 st
%Cpu9  : 42.1 us, 5.4 sy, 0.0 ni, 48.8 id, 0.0 wa, 0.0 hi, 3.7 si, 0.0 st
%Cpu10 : 44.4 us, 3.4 sy, 0.0 ni, 49.5 id, 0.0 wa, 0.0 hi, 2.7 si, 0.0 st
%Cpu11 : 38.7 us, 3.0 sy, 0.0 ni, 55.9 id, 0.0 wa, 0.0 hi, 2.4 si, 0.0 st
%Cpu12 : 40.9 us, 4.4 sy, 0.0 ni, 51.3 id, 0.3 wa, 0.0 hi, 3.0 si, 0.0 st
%Cpu13 : 38.3 us, 3.4 sy, 0.0 ni, 55.6 id, 0.0 wa, 0.0 hi, 2.7 si, 0.0 st
%Cpu14 : 46.6 us, 4.4 sy, 0.0 ni, 45.6 id, 0.0 wa, 0.0 hi, 3.4 si, 0.0 st
%Cpu15 : 35.9 us, 4.4 sy, 0.0 ni, 56.4 id, 0.0 wa, 0.0 hi, 3.4 si, 0.0 st
%Cpu16 : 43.7 us, 4.1 sy, 0.0 ni, 47.1 id, 0.0 wa, 0.0 hi, 5.1 si, 0.0 st
%Cpu17 : 34.4 us, 3.7 sy, 0.0 ni, 60.2 id, 0.0 wa, 0.0 hi, 1.7 si, 0.0 st
%Cpu18 : 33.9 us, 4.4 sy, 0.0 ni, 59.4 id, 0.0 wa, 0.0 hi, 2.3 si, 0.0 st
%Cpu19 : 34.5 us, 5.7 sy, 0.0 ni, 57.4 id, 0.0 wa, 0.0 hi, 2.4 si, 0.0 st
%Cpu20 : 38.7 us, 4.0 sy, 0.0 ni, 54.9 id, 0.0 wa, 0.0 hi, 2.4 si, 0.0 st
%Cpu21 : 28.3 us, 3.4 sy, 0.0 ni, 66.3 id, 0.0 wa, 0.0 hi, 2.0 si, 0.0 st
%Cpu22 : 34.5 us, 4.1 sy, 0.0 ni, 57.0 id, 0.0 wa, 0.0 hi, 4.4 si, 0.0 st
%Cpu23 : 32.3 us, 5.2 sy, 0.0 ni, 59.1 id, 0.0 wa, 0.0 hi, 3.4 si, 0.0 st
%Cpu24 : 39.5 us, 3.4 sy, 0.0 ni, 54.7 id, 0.0 wa, 0.0 hi, 2.4 si, 0.0 st
%Cpu25 : 45.3 us, 5.0 sy, 0.0 ni, 46.6 id, 0.0 wa, 0.0 hi, 3.0 si, 0.0 st
%Cpu26 : 33.3 us, 3.7 sy, 0.0 ni, 60.3 id, 0.0 wa, 0.0 hi, 2.7 si, 0.0 st
%Cpu27 : 37.8 us, 3.0 sy, 0.0 ni, 57.1 id, 0.0 wa, 0.0 hi, 2.0 si, 0.0 st
%Cpu28 : 34.6 us, 4.4 sy, 0.0 ni, 59.1 id, 0.0 wa, 0.0 hi, 2.0 si, 0.0 st
%Cpu29 : 35.0 us, 3.0 sy, 0.0 ni, 59.3 id, 0.0 wa, 0.0 hi, 2.7 si, 0.0 st
%Cpu30 : 39.4 us, 3.7 sy, 0.0 ni, 53.9 id, 0.0 wa, 0.0 hi, 3.0 si, 0.0 st
%Cpu31 : 33.7 us, 5.0 sy, 0.3 ni, 58.3 id, 0.0 wa, 0.0 hi, 2.7 si, 0.0 st
KiB Mem: 13183342+total, 15422684 used, 11641073+free, 364 buffers
KiB Swap: 2129916 total, 1564524 used, 565392 free. 2034388 cached Mem
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
16418	65534	20	0	140532	112812	11688	S	81.5	0.1	339:28.08	nginx: worker process
16417	65534	20	0	142192	112840	11692	R	80.8	0.1	350:23.17	nginx: worker process
16409	65534	20	0	143256	109064	11688	R	80.5	0.1	338:48.61	nginx: worker process
16405	65534	20	0	141804	110300	11688	R	80.2	0.1	349:02.19	nginx: worker process
16410	65534	20	0	141792	111228	11688	R	80.2	0.1	350:27.95	nginx: worker process
16420	65534	20	0	143460	110392	11692	S	80.2	0.1	349:08.57	nginx: worker process
16419	65534	20	0	141660	110676	11692	R	79.9	0.1	347:19.77	nginx: worker process
16412	65534	20	0	143516	107856	11688	R	79.2	0.1	346:42.49	nginx: worker process
16406	65534	20	0	141512	110492	11688	R	78.5	0.1	330:01.27	nginx: worker process
16407	65534	20	0	139852	108940	11704	S	78.2	0.1	330:00.51	nginx: worker process

外部访问-nginx-动态伸缩

1. 缩容

- 从容退出
 - 稍等一会，确保前端LB已摘除endpoint
 - 确保LB不会送来新的请求，把未完成请求处理完，再优雅退出

2. 扩容

- 使用**readinessProbe initialDelaySeconds**留足初始化时间

3. 注意

- 设置足够大的timeout，若Liveness/Readiness Probe轻易失败，可能引起雪崩效应
- 根据应用特点和实际运行情况，优化cpu request和 hpa

```
lifecycle:
  preStop:
    exec:
      command:
        - /bin/sh
        - -c
        - sleep 8 && /ensure_nginx_quit.sh
```

外部访问-7层-总结

1. 降低成本

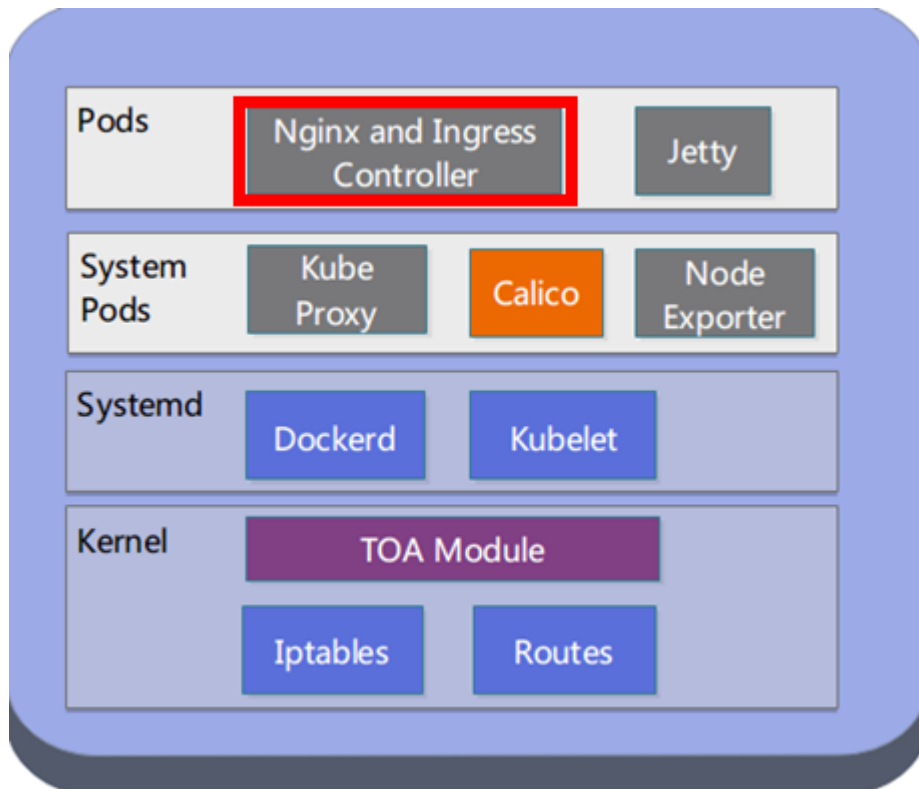
- 提升资源利用率
- 标准化/规范化

2. 动态化

- Nginx动态伸缩，注意柔性
- 后端upstream伸缩同样注意柔性

3. 满足业务需求

- 灰度发布
- 获取客户端IP
- 配置定制



目录

1 基本介绍

2 K8s集群

3 容器网络

4 外部访问4/7层

➔ 5 监控/告警/日志

6 业务发布/镜像/多机房

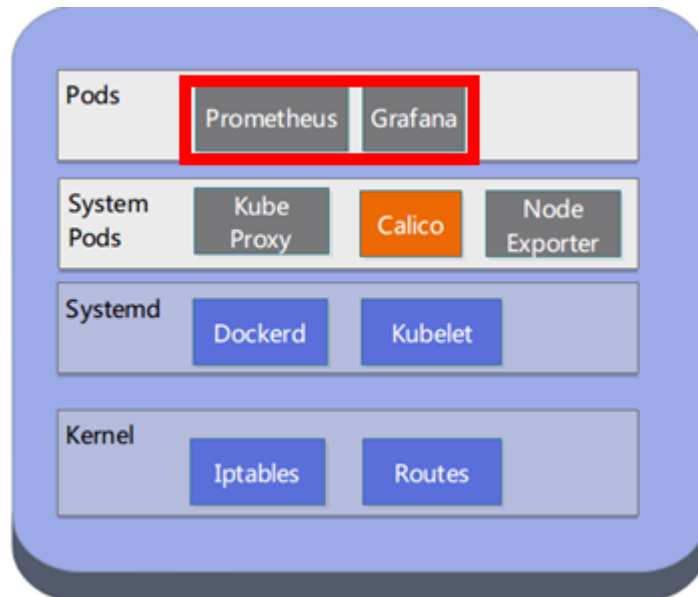
监控-Prometheus

- 部署

- Daemonset , 调度到特定机型
- 部署冗余副本

- 配置

- ConfigMap
- 按用途拆分



```
[devbasic@ip-10-0-1-100 ~]$ kubectl get configmap --namespace=monitor
NAME                                DATA  AGE
prometheus-alert-rules              1      28d
prometheus-alertmanager-config      1     134d
prometheus-config                   2     132d
prometheus-config-alone             1      28d
prometheus-recording-rules          1      29d
```

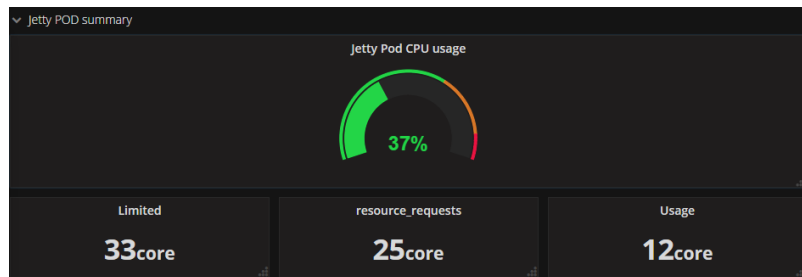
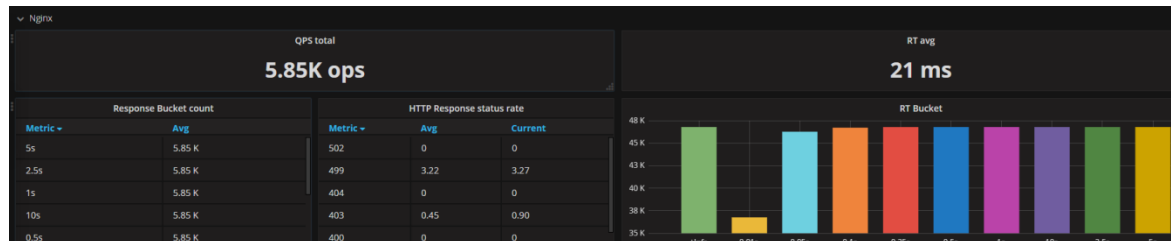
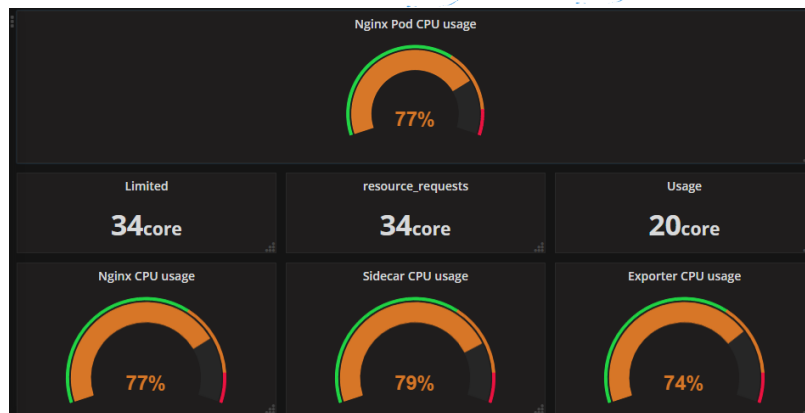
监控-业务“硬”指标

1. 从业务Nginx获取

- 当前QPS
- HTTP Code分布
- upstream平均rt
- upstream rt分布

2. 计算资源消耗状况

- Nginx
- 后端Jetty



监控-业务-“软”指标

1. “内部” 指标

- JVM
- Logging Counter
- 业务指标

Logging				
log4j				
Time ▾	debug	error	fatal	info
2017-03-08 20:16:54	0	0	0	0

2. 暴露方式

- 低侵入，可拔插
 - Java Agent运行http Prometheus Handler
 - -javaagent:lib/meizu-metric-agent-0.0.4.jar

Minor GC count (in past 1 hour)		Major GC count (from startup)		Major GC elapsed (from startup)		Minor GC elapsed (in past 1 hour)	
Metric ▾	Avg	Metric ▾	Avg	Metric ▾	Avg	Metric ▾	Avg
summary	4 K	10.1.14.20.100.1	1	10.1.14.20.100.1	570 ms	summary	32.4 s
10.1.14.20.100.1	904	10.1.14.20.100.1	1	10.1.14.20.100.1	149 ms	10.1.14.20.100.1	7.7 s

监控-Prometheus-总结

1. 暴露方式

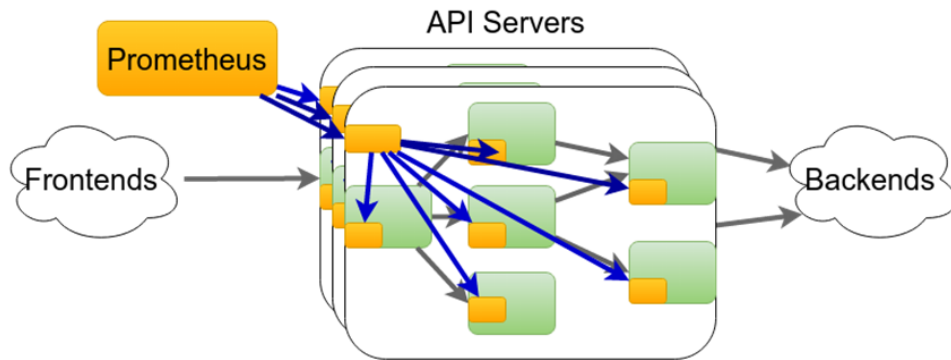
- pull
- push

2. 部署

- 易升级
- 调度到特定机型
- 冗余副本

3. 效率优化

- 查询速度
- 告警准确度
- 业务监控面板

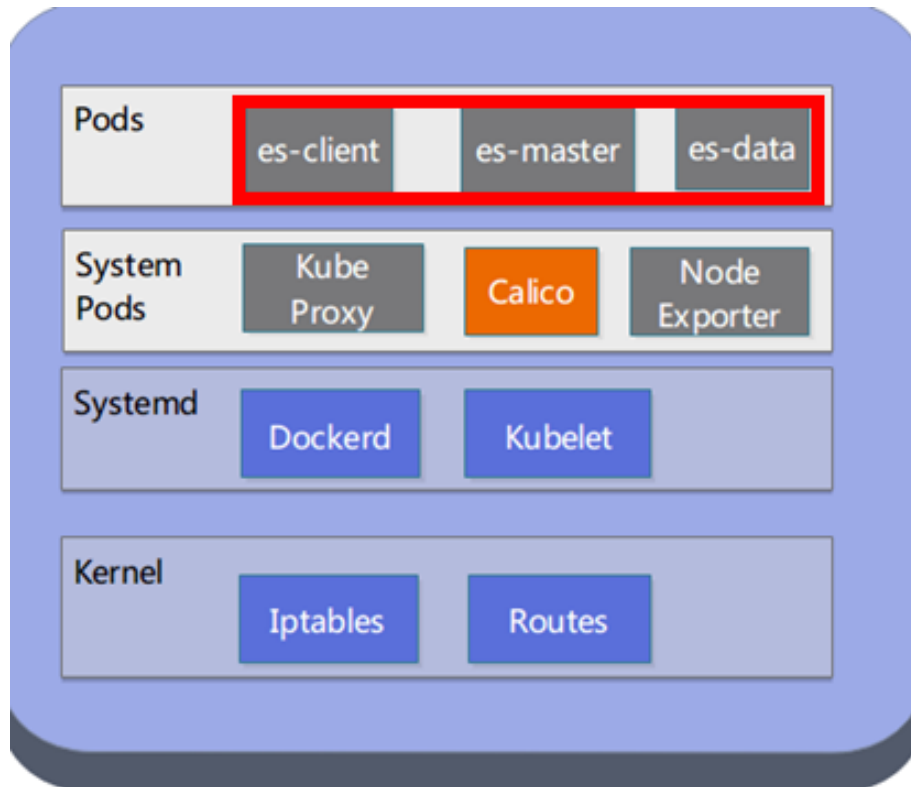


日志-集群部署

1. 部署

- 特定机型
- 快速扩容
- 容器化
- 线程数匹配CPU Limit

2. 收集效率



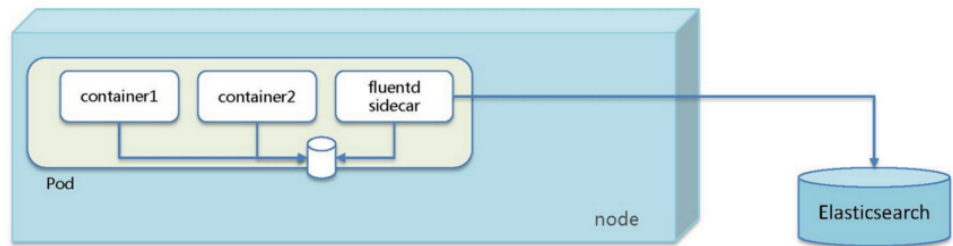
日志-收集方法

1. Logging Sidecar

- Fluentd agent
- 业务容器和agent使用EmptyDir非持久化存储
- 日志文件滚动，防止占用过多空间
- 直接发送到ElasticSearch

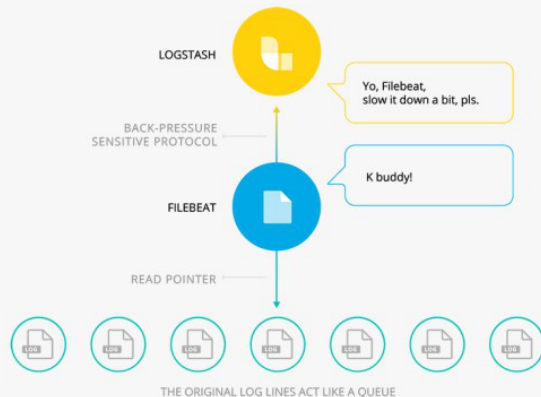
2. 问题

- 日志滞后，抓包发现是agent发送落后
- 优化bucket
- 记录传输进度的pos文件，放在EmptyDir
- 用filebeat取代



Filebeat Won't Let You Overload Your Pipeline

Filebeat uses a backpressure-sensitive protocol when sending data to Logstash or Elasticsearch to account for higher volumes of data. If Logstash is busy crunching data, it lets Filebeat know to slow down its read. Once the congestion is resolved, Filebeat will build back up to its original pace and keep on shippin'.



目录

1 基本介绍

2 K8s集群

3 容器网络

4 外部访问4/7层

5 监控/告警/日志

➔ 6 业务发布/镜像/多机房

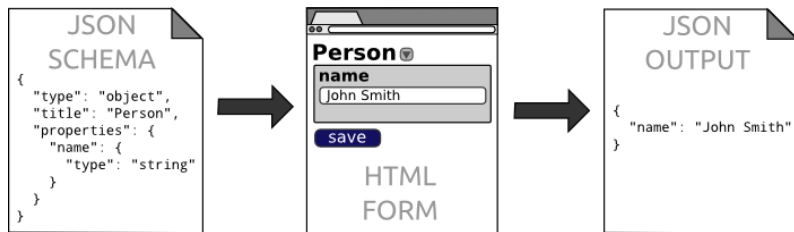
业务部署-实现

1. 生成K8s Resource Yaml

1. JSON SCHEMA生成界面
2. 界面输入生成JSON
3. JSON+Template= Resource Yaml

2. 执行动作

1. Ansible调用kubectl
2. 获取结果输出



The screenshot shows a web interface with two main sections: **jetty_hpa** and **Ingress**.

jetty_hpa section:

- mount_path**: /data/log/jetty
- inuse**: true
- maxReplicas**: 3
- minReplicas**: 1
- targetCPUUtilizationPercentage**: 70

Ingress section:

- inuse**: true
- http_host**: servcmeizu.com
- locations**:
 - path**: /
 - port**: 8080

业务部署-实现

• Schema

- 分组描述所有参数
- 类型
- 文字说明
- 默认值

schema.json 12.9 KB

```
1 {
2   "title": "Jetty-Dashboard",
3   "type": "object",
4   "properties": {
5     "global": {
6       "title": "Global",
7       "type": "object",
8       "properties": {
9         "create_ns": {
10          "type": "boolean",
11          "description": "创建namespace",
12          "enum": [
13            false,
14            true
15          ],
16          "default": "{{global.create_ns}}"
17        },
18        "org": {
19          "type": "string",
20          "description": "组织名",
21          "default": "{{global.org}}"
22        },
23        "registry": {
24          "type": "string",
25          "description": "镜像仓库地址",
26          "enum": [
27            "reg.local:5000",
28            "registry.meizu.com"
29          ],
30          "default": "{{global.registry}}"
31        }
32      }
33    },
34    "mfs": {
35      "title": "mfs",
36      "type": "object",
37      "properties": {
38        "inuse": {
39          "type": "boolean",
40          "description": "是否需要mfs",
41          "default": "{{mfs.inuse}}"
42        },
43        "mfs_name": {
44          "type": "string",
45          "description": "mfs名称",
46          "default": "{{mfs.mfs_name}}"
47        }
48      }
49    }
50  }
51 }
```

```
"jetty": {
  "title": "Jetty",
  "type": "object",
  "properties": {
    "version": {
      "type": "string",
      "description": "镜像版本",
      "minLength": 3,
      "default": "{{jetty.version}}"
    },
    "image": {
      "type": "string",
      "description": "镜像名字",
      "minLength": 1,
      "default": "{{jetty.image}}"
    },
    "jvm_xmx": {
      "type": "string",
      "description": "JVM最大分配内存",
      "minLength": 1,
      "default": "{{jetty.jvm_xmx}}"
    },
    "jvm_xms": {
      "type": "string",
      "description": "JVM最小分配内存",
      "minLength": 1,
      "default": "{{jetty.jvm_xms}}"
    },
    "limit_memory": {
      "type": "string",
      "description": "容器内存限制,单位Gi, Mi",
      "minLength": 2,
      "default": "{{jetty.limit_memory}}"
    },
    "limit_cpu": {
      "type": "string",
      "description": "容器cpu限制,单位(核)最小0.1",
      "minLength": 1,
      "default": "{{jetty.limit_cpu}}"
    }
  }
}
```

业务部署-Ansible

- 生成资源描述文件Yaml

- 模板库
- 角色库
- Resource template+json=yaml

- 创建K8s资源

- 直接调用kubectl，比API功能强大
- Ansible roles
- Namespace,Deployment,Service,Ingress,Configmap，HPA

- 创建其它资源

- Kibana Search，Dashboard
- Grafana面板
- 调用脚本

Name	master	ansible / jetty / roles /
basic	Name	Last Update
codis	..	
jetty	grafana	17 days ago
kiev	ingress	14 days ago
kiev_cpp	jetty-hpa	16 days ago
nginx_test	jetty	a day ago
nodeserver	kibana	3 months ago
php	nginx-hpa	16 days ago
task_launcher	nginx	about 4 hours a
	ns	4 months ago

master	ansible / jetty / roles / jetty / templates /
Name	Last Update
..	
jetty-deploy.yaml	a day ago
jetty-svc.yaml	2 months ago



业务部署-易用

- 方便使用

- 查看部署进度

- schema的模板，保存常用配置，输入更少

- 查看/修改历史发布

- 安全感

- 提供熟悉的console界面

执行结果

```
PLAY [master] *****
TASK [setup] *****
ok: [localhost]

TASK [ns : Generate Namespace Yaml] *****
skipping: [localhost]

TASK [ns : Create Namespace] *****
skipping: [localhost]

TASK [jetty : jetty::Prepare] *****
changed: [localhost]

TASK [jetty : Generate Jetty Deployment Yaml] *****
changed: [localhost]

TASK [jetty : Generate Jetty Service Yaml] *****
changed: [localhost]

TASK [jetty : Generate Jetty HPA Yaml] *****
changed: [localhost]

TASK [jetty : Create Jetty Deployment] *****
changed: [localhost]

TASK [jetty : Create Jetty Service] *****
changed: [localhost]

TASK [grafana : Create directory] *****
ok: [localhost]

a Dashboard [json] *****
```

确定

执行命令

```
nginx-ingress-0b  % 连接容器
2449 nginx-ingress-0b worker process
2450 nginx-ingress-0b worker process
2451 nobody 79:00 nginx: worker process
2452 nobody 90:00 nginx: worker process
2453 nobody 85:29 nginx: worker process
2454 nobody 71:20 nginx: worker process
4971 root 0:00 /bin/bash
4980 root 0:00 /bin/bash
5026 root 0:00 sleep 15
5027 root 0:00 /bin/bash
5033 root 0:00 ps
bash-4.3# ls -l
total 35780
drwxr-xr-x 2 root root 8192 Mar 5 2016 bin
drwxr-xr-x 3 root root 16 Dec 26 12:19 data
drwxr-xr-x 5 root root 380 Mar 9 14:23 dev
drwxr-xr-x 24 root root 4096 Mar 9 14:23 etc
drwxr-xr-x 2 root root 6 Oct 19 02:58 home
drwxr-xr-x 1 root root 369 Mar 5 2016 init
drwxr-xr-x 5 root root 4096 Dec 27 16:21 lib
drwxr-xr-x 2 root root 33 Oct 19 04:49 lib64
drwxr-xr-x 2 root root 27 Mar 5 2016 libexec
drwxr-xr-x 1 root root 12 Oct 19 02:58 linuxrc -> /bin/busybox
drwxr-xr-x 5 root root 41 Oct 19 02:58 media
drwxr-xr-x 2 root root 6 Oct 19 02:58 mnt
drwxr-xr-x 1 root root 36868112 Feb 6 10:31 nginx-ingress-controller
drwxr-xr-x 870 root root 0 Mar 9 14:23 proc
drwxr-xr-x 2 root root 6 Oct 19 02:58 root
drwxr-xr-x 2 root root 22 Mar 9 14:23 run
drwxr-xr-x 2 root root 4096 Dec 26 12:15 sbin
drwxr-xr-x 2 root root 6 Oct 19 02:58 srv
drwxr-xr-x 13 root root 0 Mar 9 14:23 sys
drwxr-xr-x 2 root root 6 Mar 9 15:01 tmp
drwxr-xr-x 8 root root 80 Dec 26 12:21 usr
drwxr-xr-x 12 root root 115 Dec 26 12:21 var
bash-4.3#
```

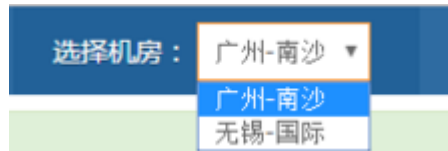
业务部署-多集群管理

1. 实现

- 每套集群自签证书不同
- 对每套Token和CA进行分组

2. 访问方式

- 管理界面，通过https访问
- Javascript web console,通过Nginx反向代理websocket访问
- 模板系统，ansible通过指定不同的k8s context访问



```
apiVersion: v1
clusters:
- cluster:
  certificate-authority: /etc/kubernetes/ssl/ca.pem
  server: https://kubernetes-ns.meizu.com:443
  name: ns-cluster
- cluster:
  insecure-skip-tls-verify: true
  server: https://kubernetes-bj.meizu.com:443
  name: bj-cluster
- cluster:
  insecure-skip-tls-verify: true
  server: https://kubernetes-wx.meizu.com:443
  name: wx-cluster
contexts:
- context:
  cluster: ns-cluster
  user: ns-cluster
  name: ns-cluster
- context:
  cluster: bj-cluster
  user: bj-cluster
  name: bj-cluster
- context:
  cluster: wx-cluster
  user: wx-cluster
```

镜像-alpine



1. 选择Alpine

- 小且够用
- 加入glibc支持，保证兼容性
- 搭建本地私有仓库

```
registry.meizu.com/codis/codis-server:rdb3.4 d3077932ef66 3 months ago 50.22 MB
registry.meizu.com/common/ingress-alpine:1.11.6.18 2697f0a9a16c 13 days ago 96.67 MB
```

2. 业务需求

- 可能需要多个进程配合
- 允许进程一次性执行，结束后检查返回值，并执行动作，如退出重试

PID	USER	TIME	COMMAND
1	root	0:00	s6-svscan -t0 /var/run/s6/services
23	root	0:00	s6-supervise s6-fdholderd
187	root	0:00	s6-supervise add-to-group
188	root	0:00	s6-supervise codis
190	root	0:00	s6-supervise create-group

3. 选择s6作为进程/服务管理器

- run，启动脚本
- finish，run退出后执行，根据返回值选择
- 根据add_to_group脚本返回值，1->不退出，0->退出服务

```
#run
/codis/codis-start add_to_group
```

```
#finish
if [ 0 -eq $1 ]; then
    s6-svc -d /var/run/s6/services/add-to-group
fi
exit 0
```

镜像-多机房缓存

1. 镜像缓存

- 镜像拉取加速
- 机房本地缓存
- 简单，小成本

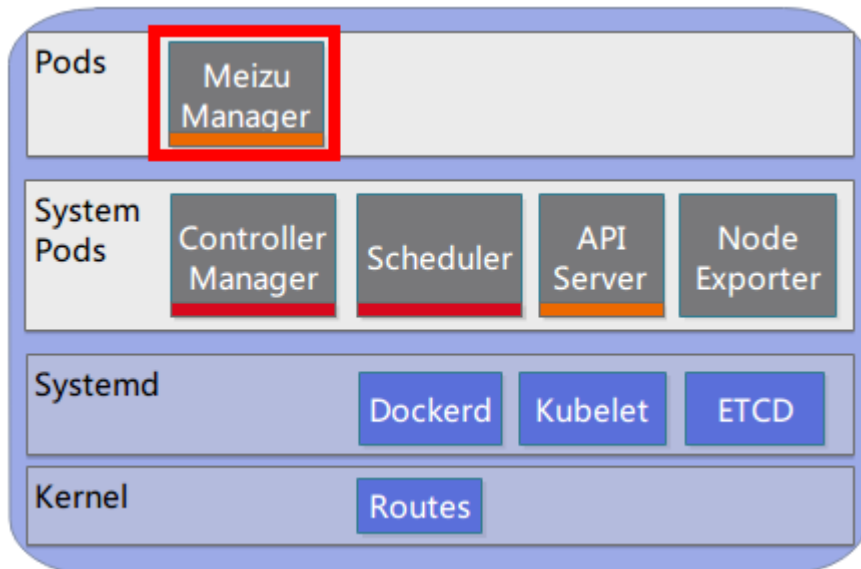
2. 方案

- 基于squid
- Pod化,自动运维



业务部署-总结

1. 用户接受度
 - 易用
2. 维护
 - 简单，易扩展
 - 问题易排查
3. 多集群管理
 - 简单，够用
 - 基于k8s context
4. 镜像
 - 小,快,够用





高效运维社区

GreatOPS Community

会议

- 3月18日 DevOpsDays 北京
- 8月18日 DevOpsDays 上海
- 全年 DevOps China 巡回沙龙
- 4月21日 GOPS深圳
- 11月17日 DevOps金融上海

培训

- EXIN DevOps Master 认证培训
- DevOps 企业内训
- DevOps 公开课
- 互联网运维培训

咨询

- 企业DevOps 实践咨询
- 企业运维咨询



商务经理：刘静女士
电话 / 微信：13021082989
邮箱：liujing@greatops.com



Thanks

高效运维社区
开放运维联盟

荣誉出品



想第一时间看到
高效运维社区公众号
的好文章吗？

请打开高效运维社区公众号，点击右上角小人，如右侧所示设置就好

