RabbitMQ应用实践

易车二手车 杨伟

目录

- Rabbitmq介绍
- 应用实践
- 运维实践

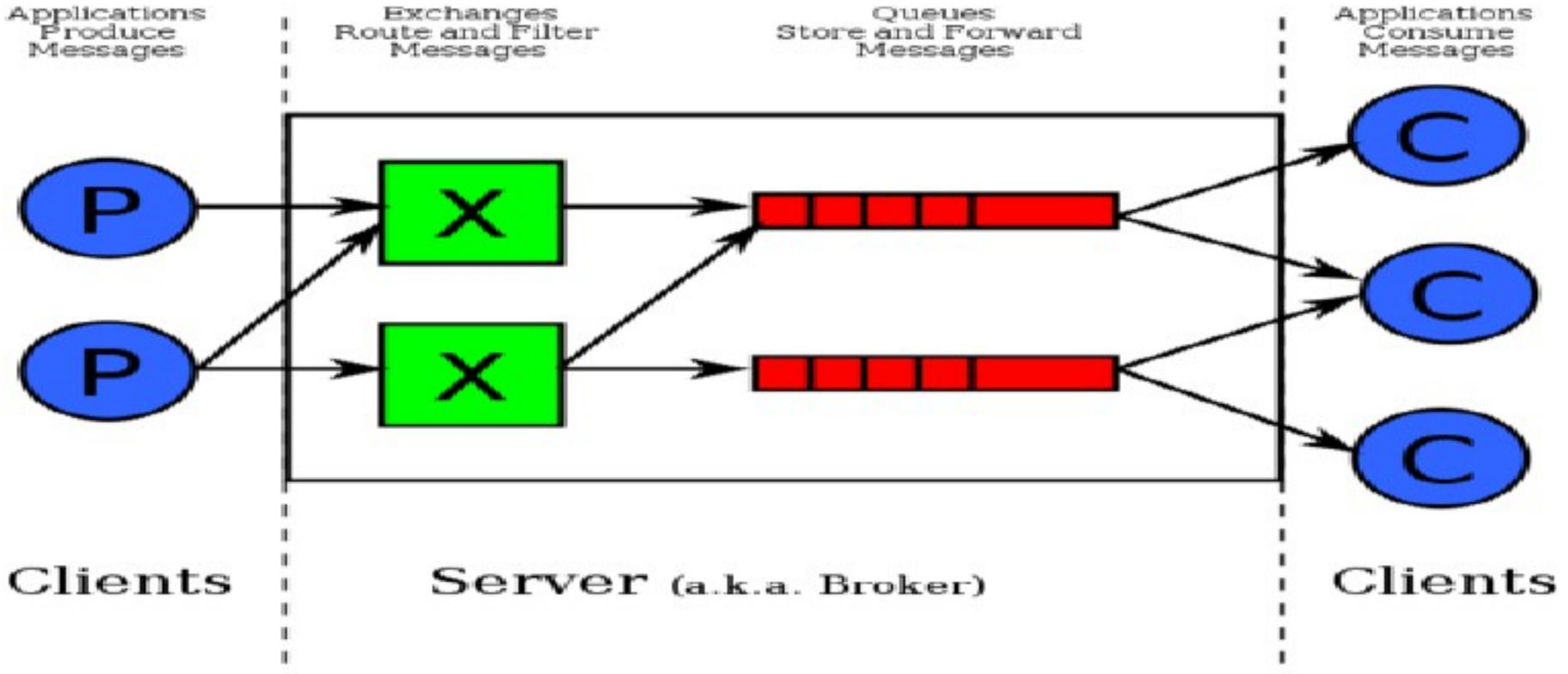
第一部分：Rabbitmq介绍

简要介绍

- 开源AMQP实现，Erlang语言编写，支持多种客户端
- 分布式、高可用、持久化、可靠、安全
- 支持多协议：AMQP、STOMP、MQTT、HTTP
- Rabbitmq主要概念对象：生产者、消费者、交换机、队列
- 业务解耦：解决多系统、异构系统间的数据交换，解耦生产者和消费者
- 适用场景：批量数据异步处理、并行任务串行化、高负载任务负载均衡

AMQP工作原理

- AMQP，即Advanced Message Queuing Protocol，高级消息队列协议

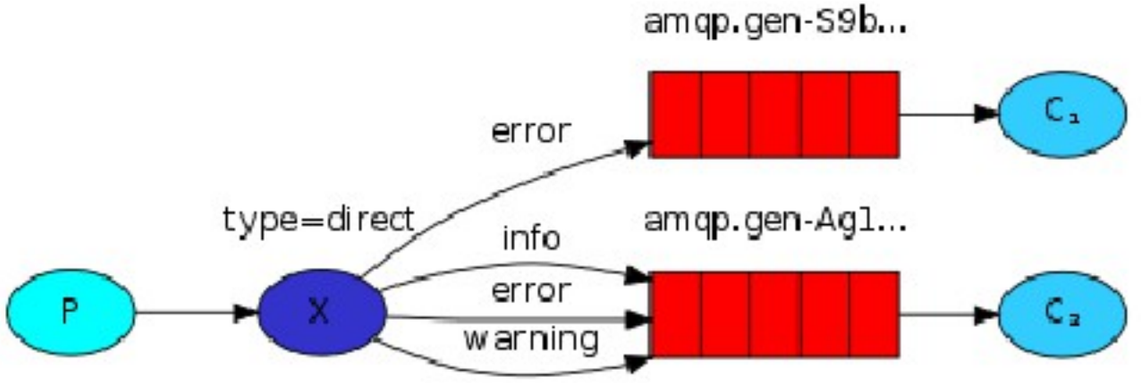


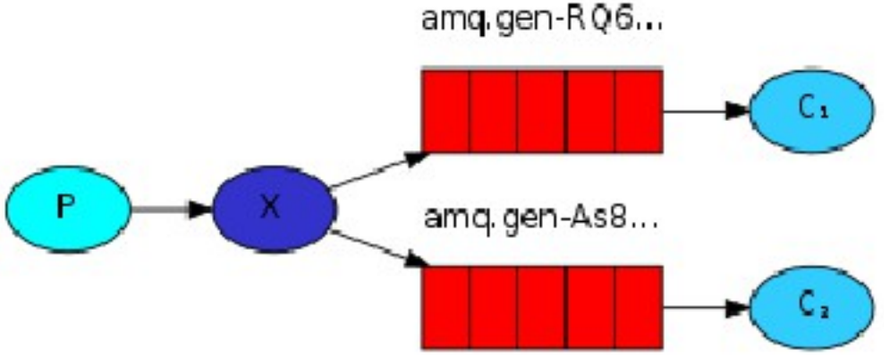
几个核心概念

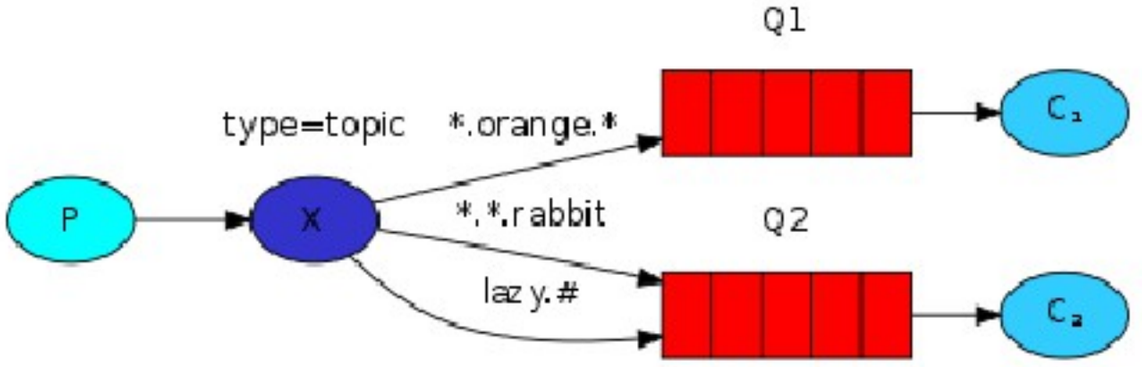
- 颗粒度：
  - Broker：消息队列服务器实体
  - vhost：虚拟主机，一个broker里可以开设多个vhost
  - Exchange：消息交换机
  - Queue：消息队列载体
- 消息流转：
  - Binding：绑定，根据路由规则绑定exchange和queue
  - Routing Key：路由关键字
  - Connection：连接
  - channel：消息通道，每个连接可建立多个channel
- 关联对象
  - producer：消息生产者
  - consumer：消息消费者

交换机类型

- Direct Exchange – 完全匹配的路由
- Topic Exchange – 模式匹配路由
- Fanout Exchange – 广播模式
- Headers exchange –键值对匹配路由







可靠性机制

- Message acknowledgment：消息回执
  - 应答机制下：收到回执才删除消息；未收到回执而连接断开，消息会转给其他消费者
  - 应答机制下：忘记回执会导致消息堆积，业务重复处理
  - 采用非应答机制可以提升队列处理效率
- Message durability：消息持久化
  - 可以避免绝大部分的消息丢失，如服务重启
  - 采用非持久化机制可以提升队列处理效率
- Prefetch count：每次发送给消费者消息数量，默认1，实践采用2

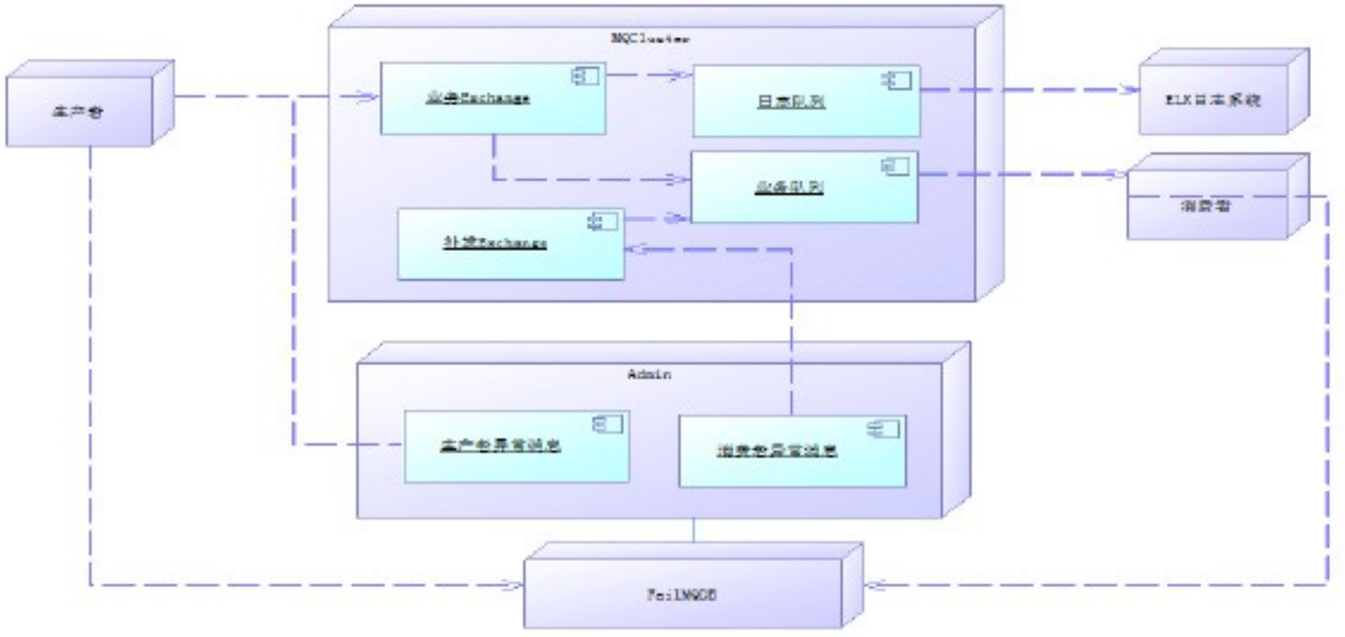
Overview					Messages			Message rates		
Virtual host	**Name**	**Node**	Features	**State**	**Ready**	**Unacked**	**Total**	**incoming**	**deliver / get**	**ack**
vhost-carindex	**qu_car_change**	rabbitmq199	D	idle	0	0	0	0.00/s	0.00/s	0.00/s
vhost-carindex	**qu_car_change_cdncache**	rabbitmq199	D	idle	99	0	99	0.00/s	0.00/s	0.00/s
vhost-carindex	**qu_car_change_refresh**	rabbitmq199	D	idle	0	0	0			
vhost-carindex	**qu_car_change_top**	rabbitmq199	D	idle	0	0	0	0.00/s	0.00/s	0.00/s

第二部分：部分应用实践

代码实现需注意细节

- 复用connection、复用channel
- 如果需要可靠业务，需要支持持久化和ack机制
  - 两台虚拟机集群，QPS >5000次/秒
- 如果希望高吞吐，可以采取非持久化、noack、自动删除机制
  - 两台虚拟机集群测试，QPS >20k次/秒

- 稳定性保障：
  - 生产者异常保障
  - 消费者异常保障



规划

- 生产者面对exchange，消费者面对queue；
- 一个queue只有一个消费者（可多个副本）进行处理；
- 命名规划：
  - exchange：ex_{bizobj}_{usecase}， 其中：ex为前缀，{bizobj}为业务对象（如Car、Dealer），{usecase}为某个业务场景，示例：ex_car_promotion；
  - routekey：对应于event（事件，某个业务动作），如：置顶（settop)、刷新(refresh)
  - queue：为方便使用，建议对应于event（采用exchange + event），如：qu_{bizobj}_{usecase}_{event}，其中：ex为前缀，{bizobj}为业务对象（如Car、Dealer），{usecase}为某个业务场景，示例：qu_car_promotion_settop；
- 串行或并行的业务方案：
  - 并行方案：一个事件发生后，多个消费者相互间没有依赖关系，可由exchange分发消息到多个队列，由各队列的消费者并行进行处理；
  - 串行方案：一个事件发生后，多个消费间有先后依赖关系，可以有先执行的消费者处理事件，处理完成后再次发送消息（此时为另外一个event）到exchange，由后续的队列进行处理。

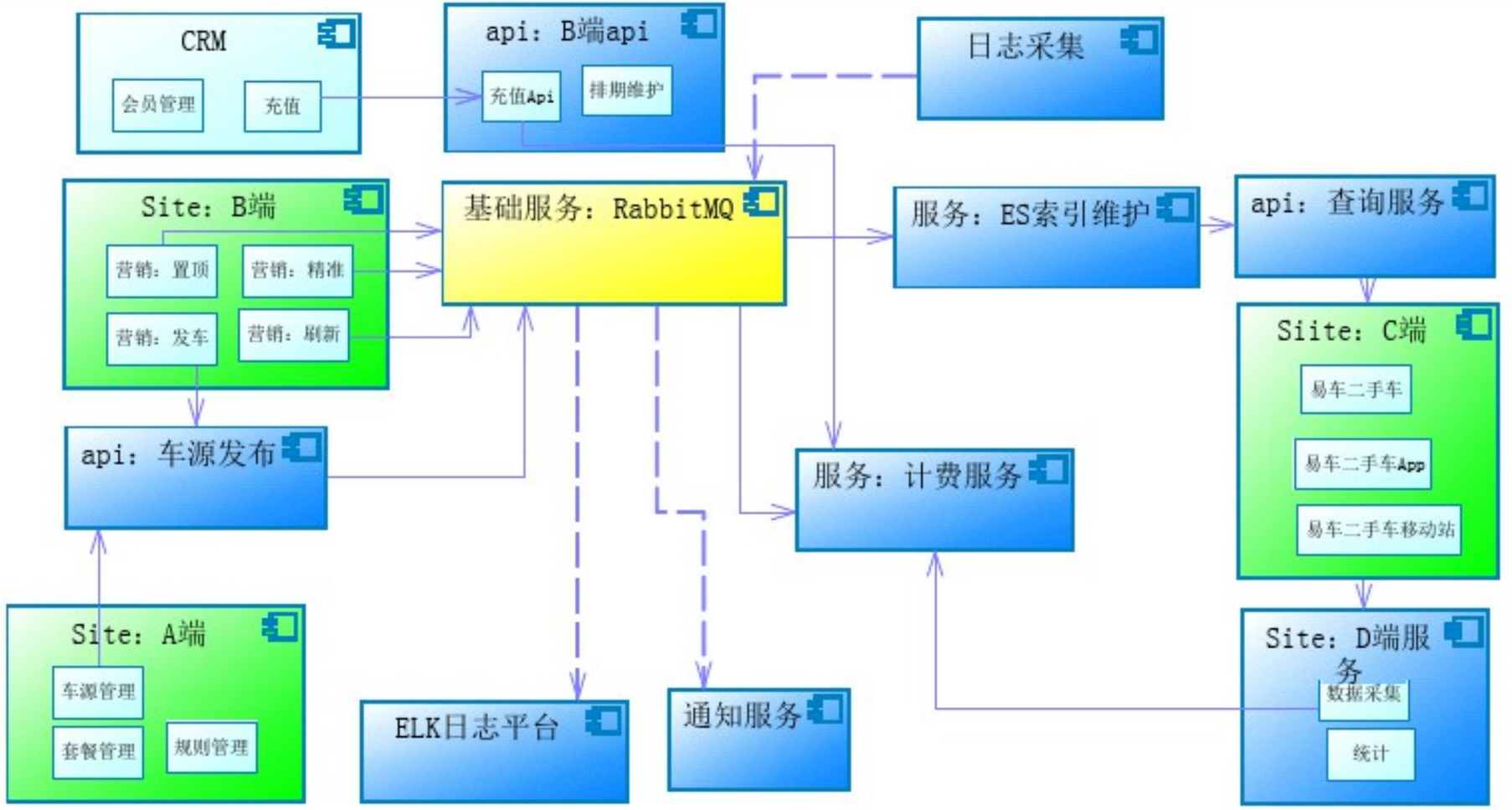
规划

- 根据不同的业务颗粒度规划

三、队列清单

序号	virtual host	exchange（交换机）	routekey/event（路由）	queue（队列名称）	说明
1	vh_ucar	ex_car_promotion	settop	qu_car_promotion_settop	车源置顶营销扣费队列（赵白勇读）
			refresh	qu_car_promotion_refresh	车源刷新营销扣费队列（赵白勇读）
			cpclimitchg	qu_car_promotion_cpc	广告限额修改队列（刘小云写，赵白勇读）
			cpcpricechange	qu_car_promotion_cpcpricechange	CPC车源价格变更，删除等【王剑锋用，刘小云写】
			cpcclick	qu_car_promotion_cpcclick qu_car_uncore_promotion_cpcclick	车源cpc点击营销【李军洁增量索引用，王剑锋写】（uncore后缀为非核心ES索引使用）
			publishcar	qu_car_promotion_publishcar	发布车源推送扣费消息队列【赵白勇扣费读取，卫勇写】

案例：新车源营销


CRM
会员管理
充值
api：B端api
充值Api
排期维护
日志采集
Site：B端
营销：置顶
营销：精准
营销：发车
营销：刷新
基础服务：RabbitMQ
服务：ES索引维护
api：查询服务
Siite：C端
易车二手车
易车二手车App
易车二手车移动站
api：车源发布
服务：计费服务
Site：A端
车源管理
套餐管理
规则管理
ELK日志平台
通知服务
Site：D端服务
数据采集
统计

案例：新车源营销

- 车源营销
  - 置顶：置顶→冻结费用、更新索引→曝光判断→计费
  - 精准：精准→更新索引→点击判断→计费
  - 刷新：刷新→更新索引、计费
  - 发车：发车→计费、自动审核→更新索引

Exchange: ex_car_promotion in virtual host vh_ucar

Overview

Message rates (chart: last minute) (?)

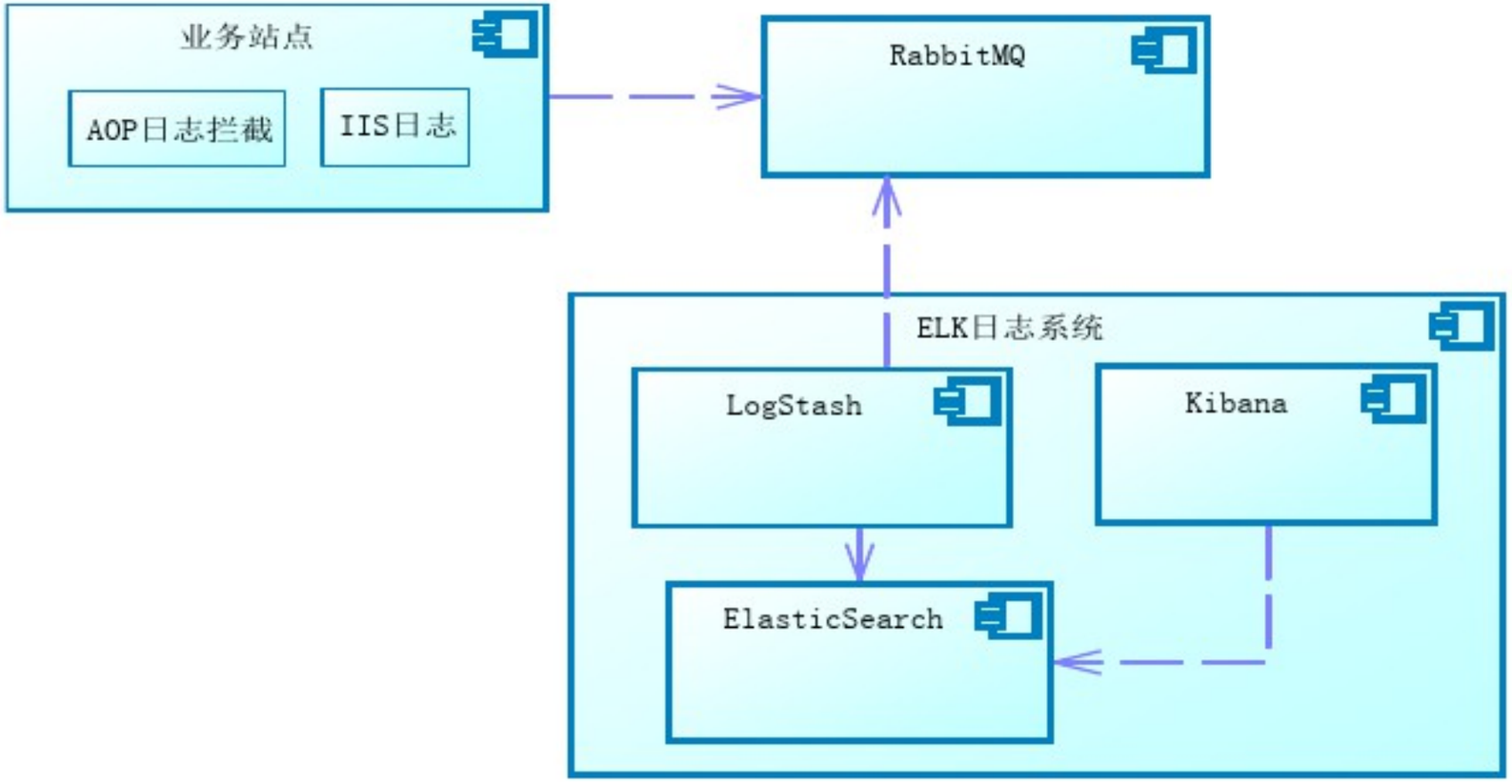
Currently idle

Details

Type	topic
Features	durable: true
Policy	ha-all

To	Routing key
qu_car_promotion_cpc	cpclimitchg
qu_car_promotion_cpcclick	cpcclick
qu_car_promotion_cpcpricechange	cpcpricechange
qu_car_promotion_publishcar_1	publishcar
qu_car_promotion_refresh	refresh
qu_car_promotion_settop	settop

案例：ELK日志平台

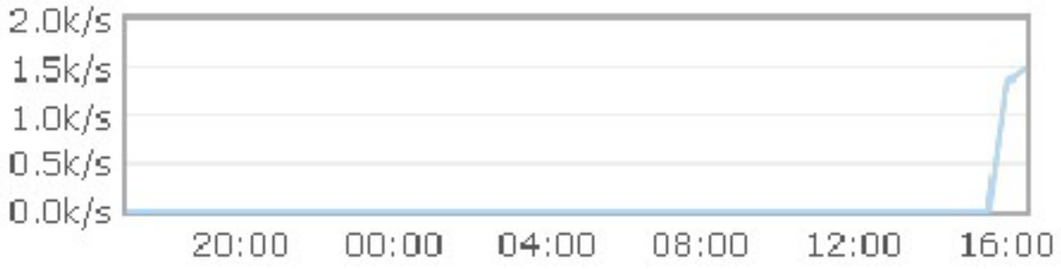

业务站点
AOP日志拦截
IIS日志
RabbitMQ
ELK日志系统
LogStash
Kibana
ElasticSearch

案例：ELK日志平台

Exchange: logstash-temp in virtual host vh_log

Overview

Message rates (chart: last day) (?)



Publish (In) 668/s

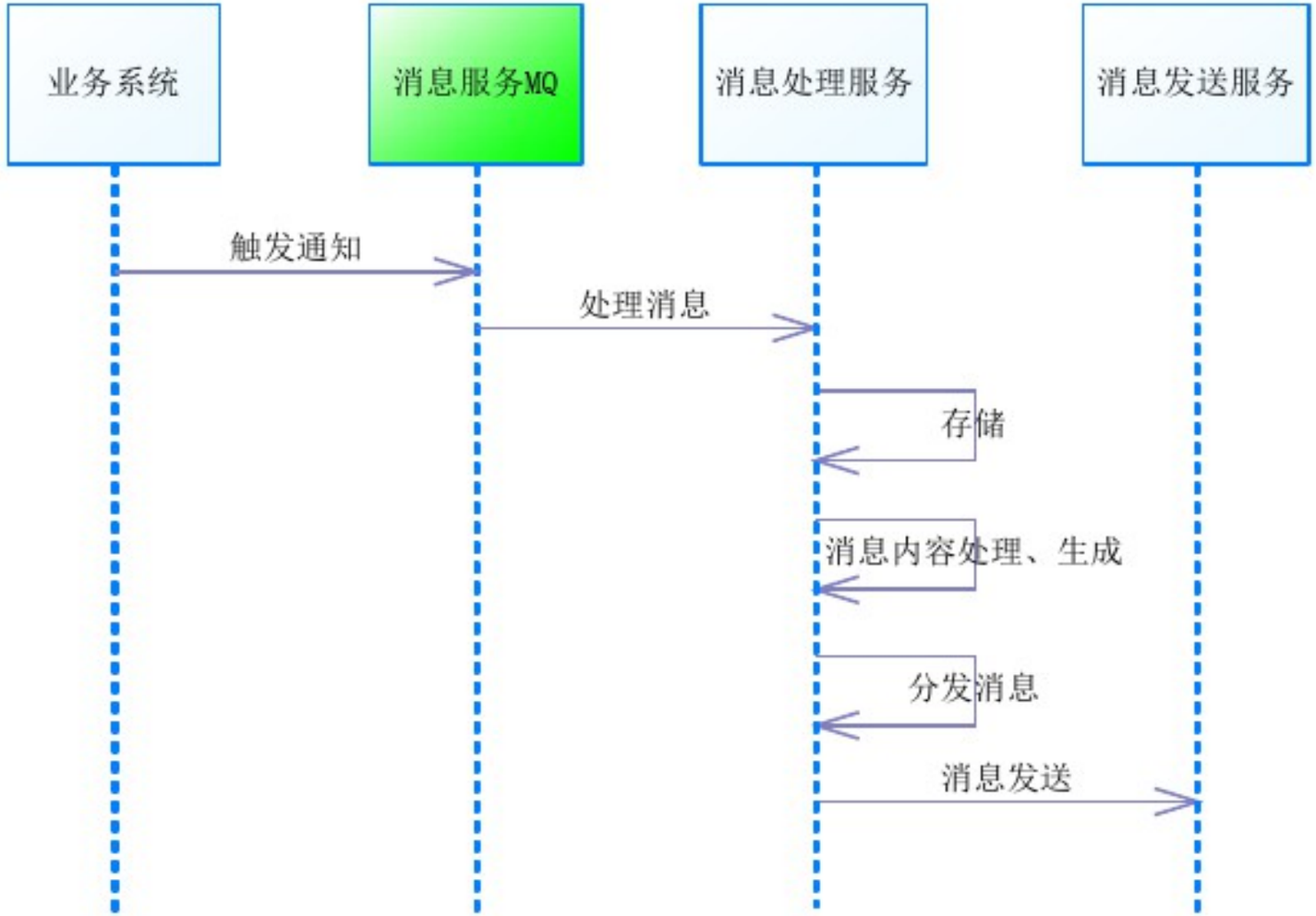
Publish (Out) 668/s

Details

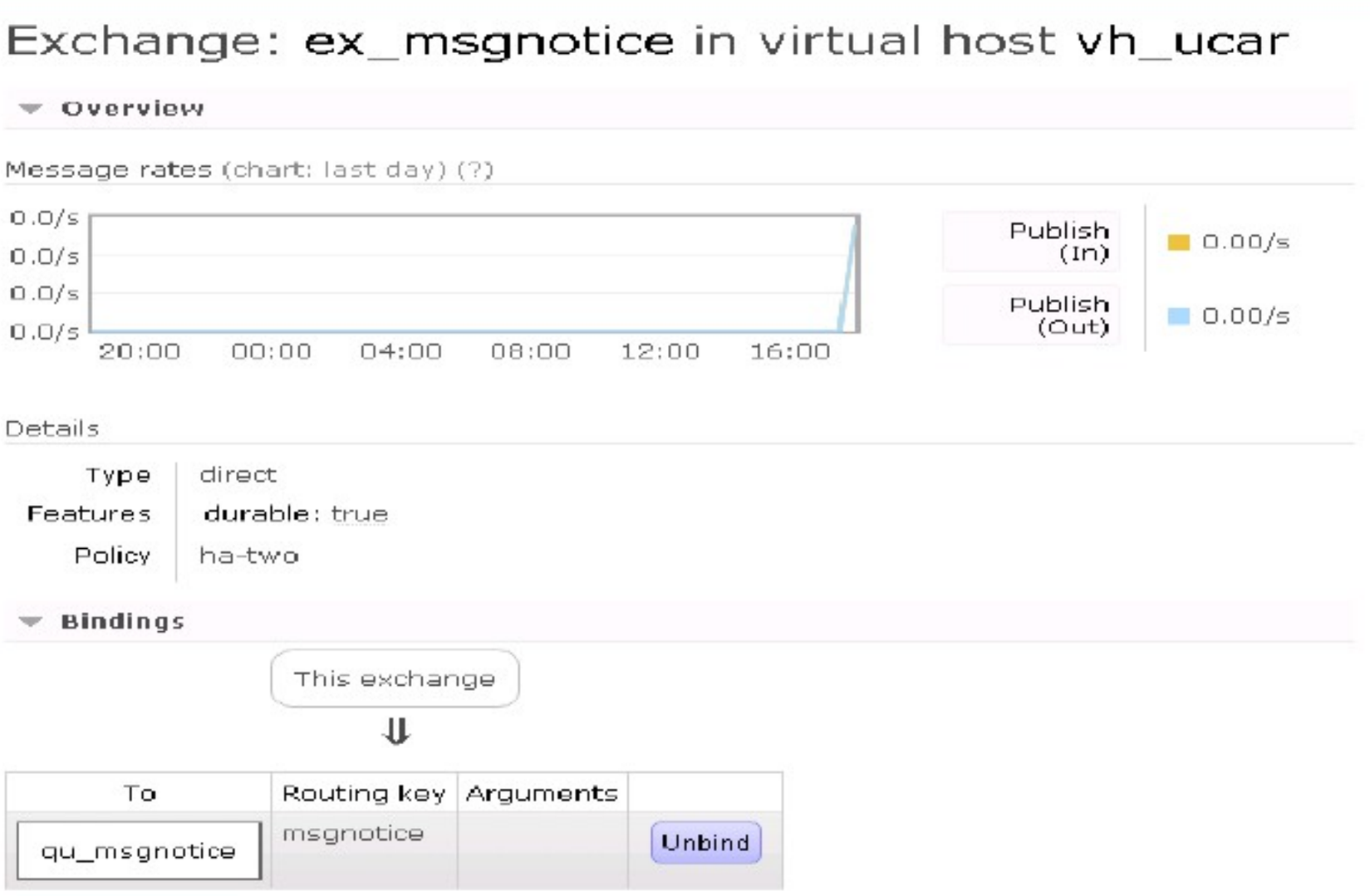
Type	direct
Features	auto-delete: true
Policy	ha-two

To	Routing key
bus-month	bus-month
bus-temp	bus-temp
bus-year	bus-year
line-admin	line-admin
line-dealer	line-dealer
line-essearch	line-essearch
line-mcityaction	line-mcityaction
line-mtaocheaction	line-mtaocheaction
line-pcctyaction	line-pcctyaction
line-pctaocheaction	line-pctaocheaction
line-suggestaction	line-suggestaction
line-test	line-test
line-topicaction	line-topicaction
line-yicheapi	line-yicheapi

案例：通知服务



案例：通知服务


Exchange: ex_msgnotice in virtual host vh_ucar
Overview
Message rates (chart: last day) (?)
0.0/s
0.0/s
0.0/s
0.0/s
20:00
00:00
04:00
08:00
12:00
16:00
Publish (In)
0.00/s
Publish (Out)
0.00/s
Details
Type direct
Features durable: true
Policy ha-two
Bindings
This exchange
To
Routing key
Arguments
qu_msgnotice
msgnotice
Unbind

第三部分：部分运维实践

运维实践

易车 yiche.com

- 安装
- 配置：
  - 服务器配置
  - 集群配置
  - 高可用集群配置
- 监控

安装

- 安装Erlang环境、安装RabbitMQ
- 启动rabbitmq，并验证启动情况：

  rabbitmq-server --detached &ps aux |grep rabbitmq
- 如果启用了防火墙的话，开启相关端口

  4369 (epmd),
  25672 (Erlang distribution)
  5672, 5671 (AMQP 0-9-1 without and with TLS)
  15672 (if management plugin is enabled)
- 启用web界面的监控插件：

  rabbitmq-plugins enable rabbitmq_management
- 登录账号密码默认都是 guest
- 添加用户

  rabbitmqctl add_user rabbitmq 123456

配置：服务配置

- 两个配置文件：
  - 环境变量的配置文件 rabbitmq-env.conf
    - 设置rabbitmq的数据存储位置

    RABBITMQ_MNESIA_BASE=/data/rabbitmq/data
    - 设置rabbitmq的日志存储位置

    RABBITMQ_LOG_BASE=/data/rabbitmq/log
  - 配置信息文件 rabbitmq.config
    - 内存阀值，超过时启动GC

    vm_memory_high_watermark, 0.6
    - 内存阀值，超过阀值时内存数据写到磁盘

    vm_memory_high_watermark_paging_ratio, 0.5
    - 脑裂问题的修复方式： ignore， autoheal， pause_minority

    cluster_partition_handling, autoheal
    - 自动加载broker信息

    {rabbitmq_management, [{load_definitions, "/etc/rabbitmq/rabbitmq_broker.json"}]}

配置：服务配置

- 日志切分：
  - 编写shell文件：/opt/scripts/rabbitmq_split_log.sh

```
#!/bin/bash

Date=`date +%Y%m%d`
logname=/data/rabbitmq/log/rabbit@rabbitmq248.log
cp $logname $logname.$Date
if [ $? -eq 0 ];then
    >$logname
fi
agologfile=$logname.`date --date "-10 day" +%Y%m%d`
if [ -e "$agologfile" ]; then
    echo 'rm'  $agologfile
    rm -f $agologfile
fi
```

  - 定时执行

    执行 crontab –e，加入定时任务

```
59 23 * * * /opt/scripts/rabbitmq_split_log.sh
```

配置：集群配置

- 修改hostname：
  修改 /etc/sysconfig/network
- 设置.erlang.cookie，基于Erlang的集群来实现
  cd /var/lib/rabbitmq/
  echo -n PXXXXEWPXODAMMALGXXXX > .erlang.cookie
- 打开端口： 25672，4369
- 加入集群：
  sudo rabbitmqctl join_cluster **--ram** rabbit@rabbitmq199
- 检查集群状态：
  rabbitmqctl cluster_status
- 设置镜像队列策略
- 从集群中分离
  - 在任一节点执行：rabbitmqctl forget_cluster_node rabbit@rabbit1
  - 在分离节点执行：rabbitmqctl reset

Rabbitmq集群监控

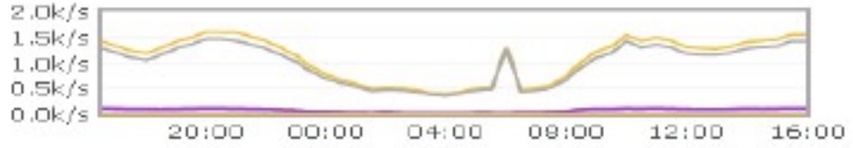
易车 yiche.com

Totals

Queued messages (chart: last day) (?)

Ready	199
Unacked	12
Total	211

Message rates (chart: last day) (?)



Publish	717/s	Redelivered	0.00/s	Deliver (noack)	6
Confirm	0.00/s	Acknowledge	3.8/s		
Deliver	3.8/s	Get	0.00/s	Get (noack)	0.

Global counts (?)

Connections: 325 | Channels: 430 | Exchanges: 37 | Queues: 46 | Consumers: 85

Nodes

Name	File descriptors (?)	Socket descriptors (?)	Erlang processes	Memory	Disk space	Info
rabbit@rabbitmq118	132 1024 available	85 829 available	1222 1048576 available	405MB 9.3GB high watermark	117GB 48MB low watermark	Disc 1
rabbit@rabbitmq119	106 1024 available	82 829 available	1219 1048576 available	393MB 9.3GB high watermark	117GB 48MB low watermark	RAM 1
rabbit@rabbitmq247	133 1024 available	88 829 available	1389 1048576 available	427MB 9.3GB high watermark	117GB 48MB low watermark	Disc 1
rabbit@rabbitmq248	115 1024 available	73 829 available	1201 1048576 available	454MB 9.3GB high watermark	117GB 48MB low watermark	RAM 1 Stats

+/-

Rabbitmq集群监控

Queues

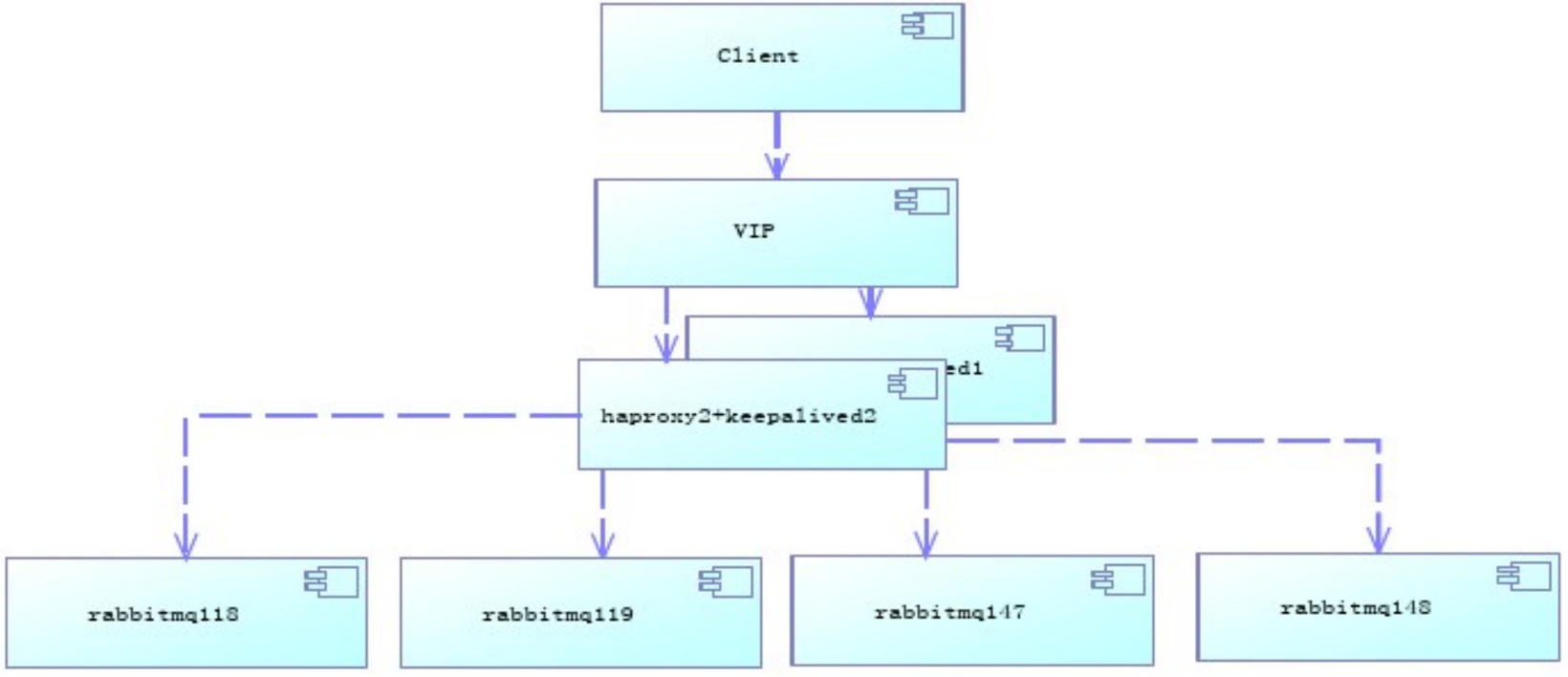
All queues

Filter: Regex (?)

Overview					Messages			Message rates			+/-
Virtual host	Name	Node	Features	State	Ready	Unacked	Total	incoming	deliver / get	ack	
vh_log	bus-month	rabbitmq248 +1	ha-two	running	0	0	0	0.00/s	0.00/s		
vh_log	bus-temp	rabbitmq248 +1	ha-two	running	0	0	0	0.00/s	0.00/s		
vh_log	bus-year	rabbitmq247 +1	ha-two	running	0	0	0	0.00/s	0.00/s		
vh_log	line-admin	rabbitmq247 +1	ha-two	idle	0	0	0	0.00/s	0.00/s		
vh_log	line-dealer	rabbitmq248 +1	ha-two	running	0	0	0	0.00/s	0.00/s		
vh_log	line-essearch	rabbitmq248 +1	ha-two	running	0	0	0	0.00/s	0.00/s		
vh_log	line-mcityaction	rabbitmq248 +1	ha-two	running	0	0	0	0.00/s	0.00/s		
vh_log	line-mtaocheaction	rabbitmq119 +1	ha-two	running	0	0	0	0.00/s	0.00/s		
vh_log	line-pccityaction	rabbitmq248 +1	ha-two	running	0	0	0	0.00/s	0.00/s		
vh_log	line-pctaocheaction	rabbitmq118 +1	ha-two	running	0	0	0	0.00/s	0.00/s		
vh_log	line-suggestaction	rabbitmq247 +1	ha-two	running	0	0	0	0.00/s	0.00/s		
vh_log	line-test	rabbitmq248 +1	ha-two	idle	0	0	0	0.00/s	0.00/s		
vh_log	line-topicaction	rabbitmq247 +1	ha-two	running	0	0	0	0.00/s	0.00/s		
vh_log	line-yicheapi	rabbitmq118 +1	ha-two	running	0	0	0	0.00/s	0.00/s		
vh_ucar	qu_car_change	rabbitmq247 +1	D ha-two	running	2	2	4	0.00/s	0.80/s	0.80/s	

配置：高可用集群配置

基于Haproxy+keepalived+rabbitmq实现集群



配置：高可用集群配置

- Haproxy配置
  - Haproxy日志配置
    - **安装rsyslog**
    - **添加haproxy的log配置：** vim /etc/rsyslog.d/haproxy.conf
    - **编辑/etc/sysconfig/rsyslog**
    - **创建日志文件并授权** touch haproxy.log
  - 配置防火墙端口：
    - **9188 用于haproxy的监控界面**
    - **5670 用于rabbimq的负载均衡端口**
  - 配置haproxy： /etc/haproxy/haproxy.cfg

    注意：因为要使用tcp的负载，屏蔽掉与http相关的默认配置
  - 启动haproxy

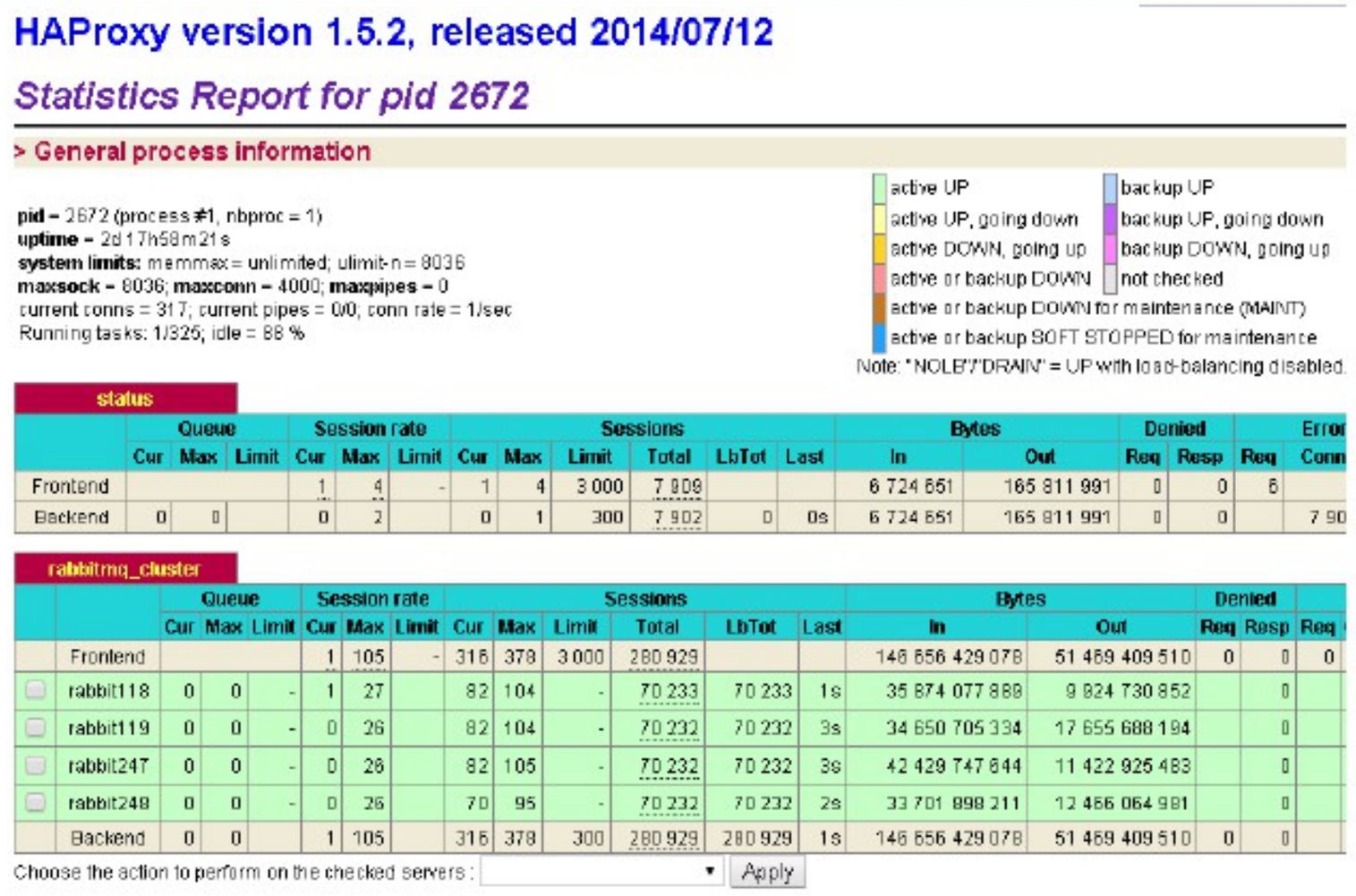
    haproxy -f /etc/haproxy/haproxy.cfg
  - 停止haproxy

    killall haproxy

配置：高可用集群配置

- Keepalived配置
  - VRRP：虚拟路由冗余协议（Virtual Router Redundancy Protocol）
  - **配置文件位置：**/etc/keepalived/keepalived.conf
  - 配置keepalived
    # 因为当前环境中VRRP组播有问题，改为使用**单播**发送VRRP报文
    unicast_src_ip 192.168.200.200
    unicast_peer {
    192.168.200.199
    }
  - **启动服务：**顺序启动
    启动haproxy： haproxy -f /etc/haproxy/haproxy.cfg
    启动keeepalived：先启动master节点，后启动BACKUP节点
    /etc/init.d/keepalived start
  - **停止keeepalived服务**
    /etc/init.d/keepalived stop
  - **检查keeepalived的运行日志，默认keepalived的日志位于/var/log/message**
    tail -n100 /var/log/message

配置：高可用集群配置



HAProxy version 1.5.2, released 2014/07/12

Statistics Report for pid 2672

> General process information

pid = 2672 (process #1, nbproc = 1)
uptime = 2d 17h58m21s
system limits: memmax = unlimited; ulimit-n = 8036
maxsock = 8036; **maxconn =** 4000; **maxpipes =** 0
current conns = 317; current pipes = 0/0; conn rate = 1/sec
Running tasks: 1/325; idle = 88 %

active UP
active UP, going down
active DOWN, going up
active or backup DOWN
active or backup DOWN for maintenance (MAINT)
active or backup SOFT STOPPED for maintenance
backup UP
backup UP, going down
backup DOWN, going up
not checked

Note: "NOLB"/"DRAIN" = UP with load-balancing disabled.

status

	Queue			Session rate			Sessions						Bytes		Denied		Error	
	Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Total	LbTot	Last	In	Out	Req	Resp	Req	Conn
Frontend				1	4	-	1	4	3 000	7 909			6 724 651	165 811 991	0	0	6	
Backend	0	0		0	2		0	1	300	7 902	0	0s	6 724 651	165 811 991	0	0		7 90

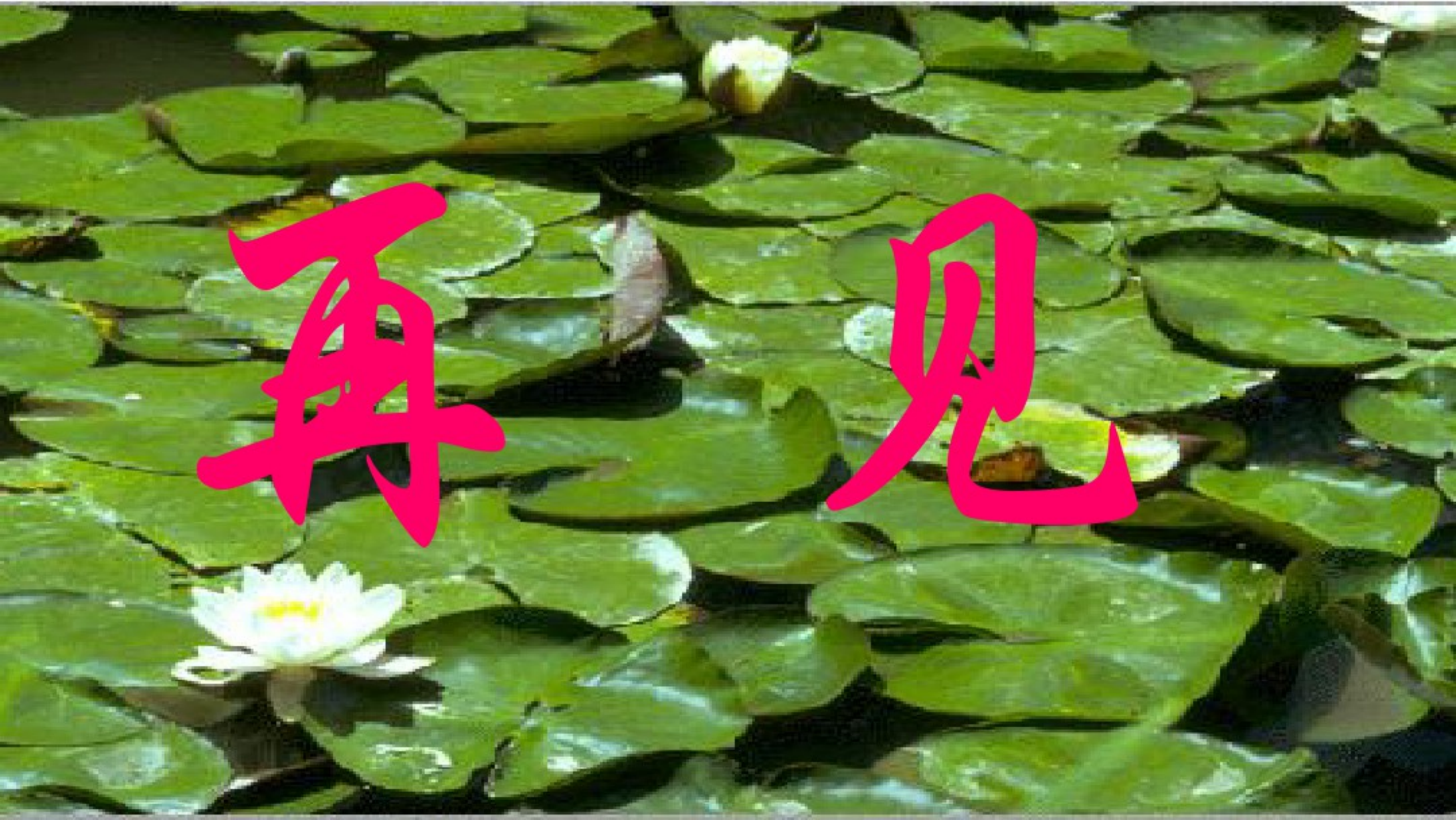
rabbitmq_cluster

		Queue			Session rate			Sessions						Bytes		Denied		
		Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Total	LbTot	Last	In	Out	Req	Resp	Req
	Frontend				1	105	-	316	378	3 000	280 929			146 656 429 078	51 469 409 510	0	0	0
	rabbit118	0	0	-	1	27		82	104	-	70 233	70 233	1s	35 874 077 888	9 824 730 852		0	
	rabbit119	0	0	-	0	26		82	104	-	70 232	70 232	3s	34 650 705 334	17 655 688 194		0	
	rabbit247	0	0	-	0	28		82	105	-	70 232	70 232	3s	42 429 747 844	11 422 925 483		0	
	rabbit248	0	0	-	0	26		70	95	-	70 232	70 232	2s	33 701 898 211	12 466 064 981		0	
	Backend	0	0		1	105		316	378	300	280 929	280 929	1s	146 656 429 078	51 469 409 510	0	0	

Choose the action to perform on the checked servers : Apply

易车
yiche.com

谢谢


再见