ES PAAS 云平台

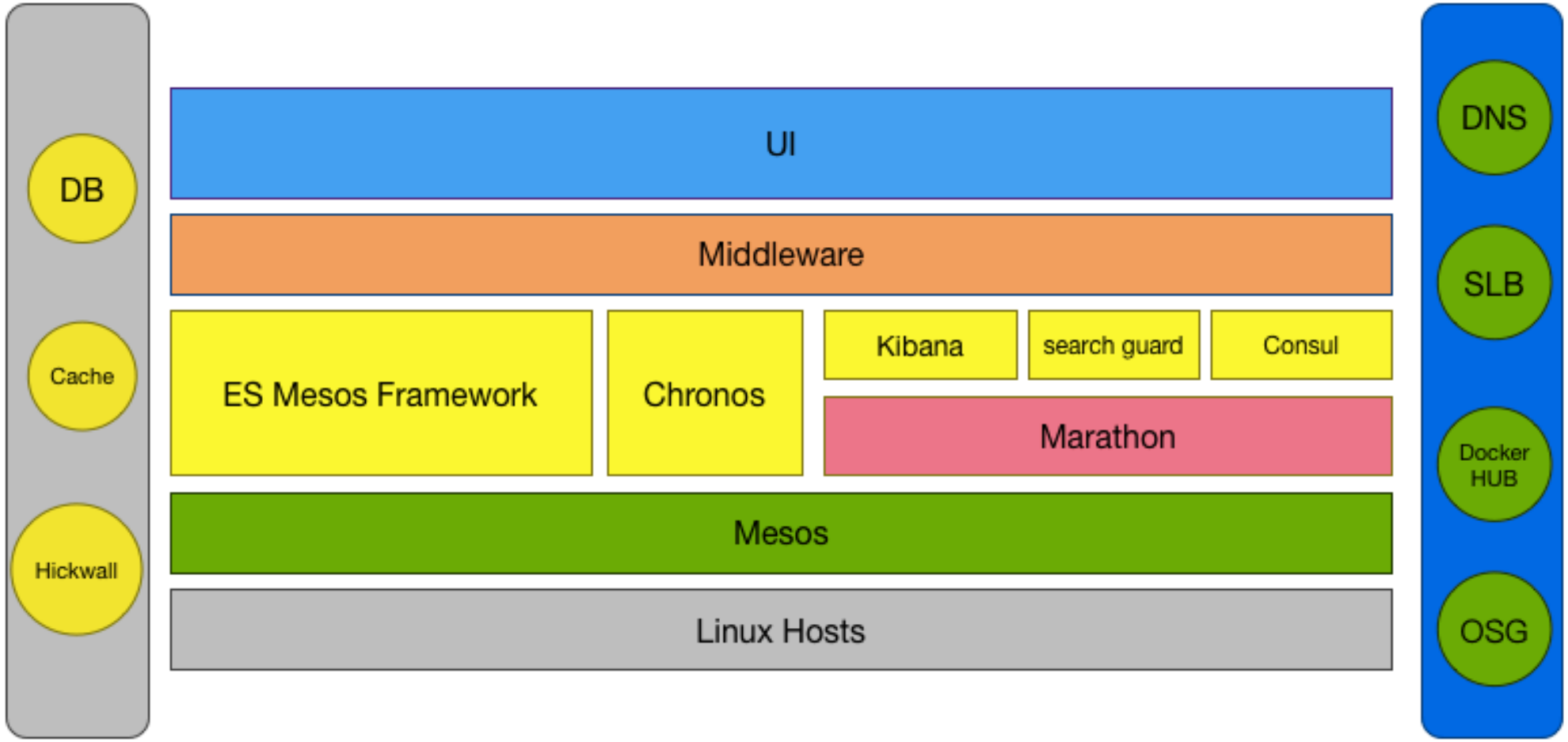
柯诗豪@携程

目录

- 为何有这个项目
- 总体架构
- 集群创建、扩容和缩容
- 服务发现
- 配置插件
- 集群监控
- 集群安全

为何有这个项目

ES PAAS 架构图


DB
Cache
Hickwall
UI
Middleware
ES Mesos Framework
Chronos
Kibana
search guard
Consul
Marathon
Mesos
Linux Hosts
DNS
SLB
Docker HUB
OSG

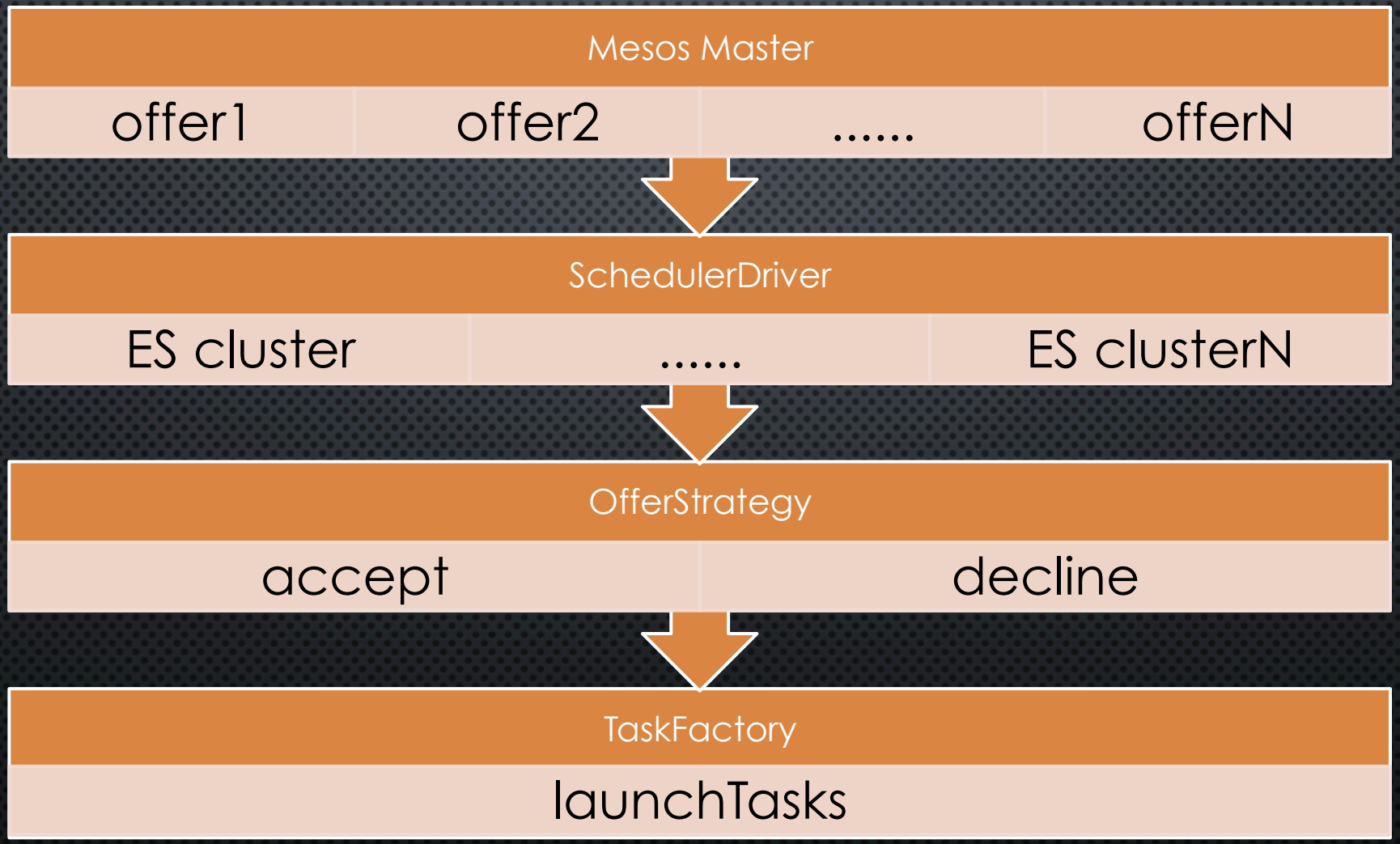
为什么不直接使用Marathon

资源调度器的模式不能完全适合 ES PAAS 的场景

由于ES 数据写在宿主机的磁盘上，所以期望的是调度是:
重启ES实例（类似 Marathon中的kill）时
kill掉实例之后，新启动的实例，尽可能的启动在先前被kill的实例所在的宿主机上
还需保证启动刚kill的实例获取资源的优先级高于其他集群扩容的优先级
否则丢失数据的概率会变高，而且不能支持 full restart

Marathon 对于这样的场景并不能满足，所以需要我们自己的Framework来实现

ES PaaS Framework



- Mesos master 将slave上可用的资源发送offer给 framework
- ES framework 收到resource offer后，有两类结点需要资源来启动
  1. 节点终止且所在集群结点数缺少，可能是结点挂了或者重启 需要尽快恢复
  2. 集群扩容后运行中结点数小于集群设置的结点数
- 如果存在类型1的结点，获取结点相关信息和资源需求，对得到的offer列表依次check 是否符合 OfferStrategy，符合则使用该offer启动新的结点实例（task）
- 直到类型1的结点不存在后，再对类型2的结点进行offer匹配
- 类型1的结点一定时间内无法匹配到offer则放弃该结点启动新结点

OfferStrategy 包含规则：

- offer的host resolveable
- offer的host上没有该ES集群的其他结点运行
- 有足够的cpu
- 有足够的memory
- 至少有2个空闲的port
- 有足够的磁盘空间
- 满足salve tag限制

Build Docker Task

- 配置Discovery从Resource中选择2个port作为http和transport使用
- Docker 的 net 使用bridge 模式，主要为了做容器网络流量监控
- 配置ES 启动参数，主要的几个：
  - discovery.zen.ping.unicast.hosts = 集群中已有的结点
  - discovery.zen.minimum_master_nodes = 集群总结点数 / 2 + 1
  - network.publish_host
  - transport.publish_port
  - http.publish_port
  - bootstrap.mlockall = true
  - bootstrap.memory_lock = true
  - discovery.zen.ping.multicast.enabled = false

Build Docker Task

- 配置docker参数
  - cpu-period 和 cpu-quota（默认的 cpu-shard不能真正限制CPU使用）
  - 画limit= memlock=-1:-1
  - memory-swappiness=0
  - ENV
  - Lable
  - PortMappingsImage
  - Network
  - Volume
    - 挂载log目录用于日志采集
    - 挂载data目录，目前是使用直接挂载，后续会采用convoy

Task status update

Framework 监听 来自 Mesos master 的task的status变化事件
更新DB中存储的结点运行状态，若task status变为：

- TASK_ERROR
- *TASK_FAILED*
- TASK_FINISHED
- TASK_LOST
- TASK_KILLED

此时无论是何种原因结点已不在运行，将结点标志为已经运行结束
如果此时集群中运行中的结点数小于集群设置的结点数，首先尝试重启该结点，超时后启动新的结点

ES PaaS Framework REST API

ES PaaS 管理平台通过 REST API 请求ES PaaS Framework来

- 创建集群实例
- 修改集群配置
- 集群扩容缩容
- 获取结点状态
- 停止 / 开始结点

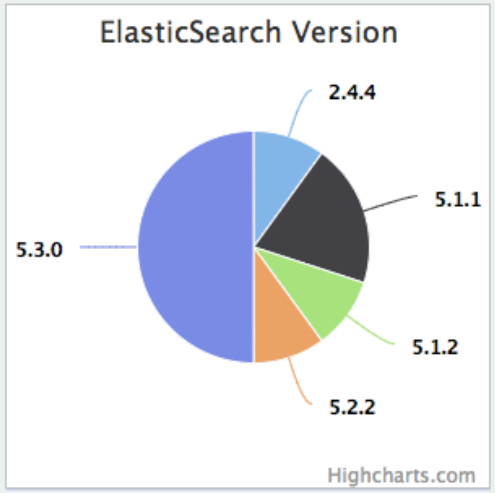
ES PaaS 平台

Elastic | Home | Cluster | Utils | Reports | Infrastructure | Admin | ksh柯诗豪

Resource Statistic CPU: 44/95 Mem: 92144/174771MB Disk: 1200/2450GB

10 Clusters	22 Nodes
NUMBER OF CLUSTERS	NUMBER OF NODES

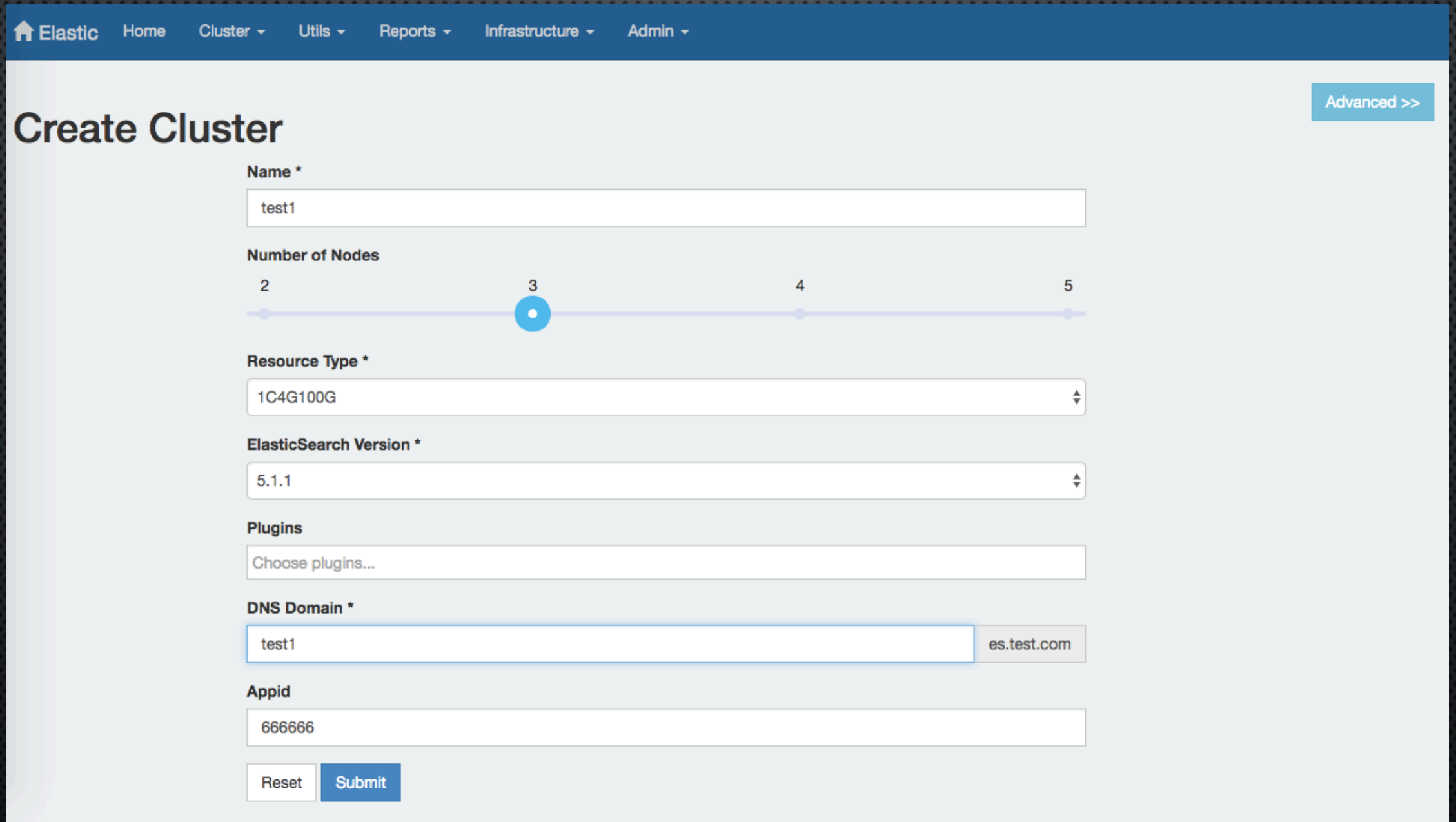
44 Cores	92,144 MB	1,200 GB
NUMBER OF CPU CORES	TOTAL OF MEMORY	GIGABYTE OF DISK SPACE



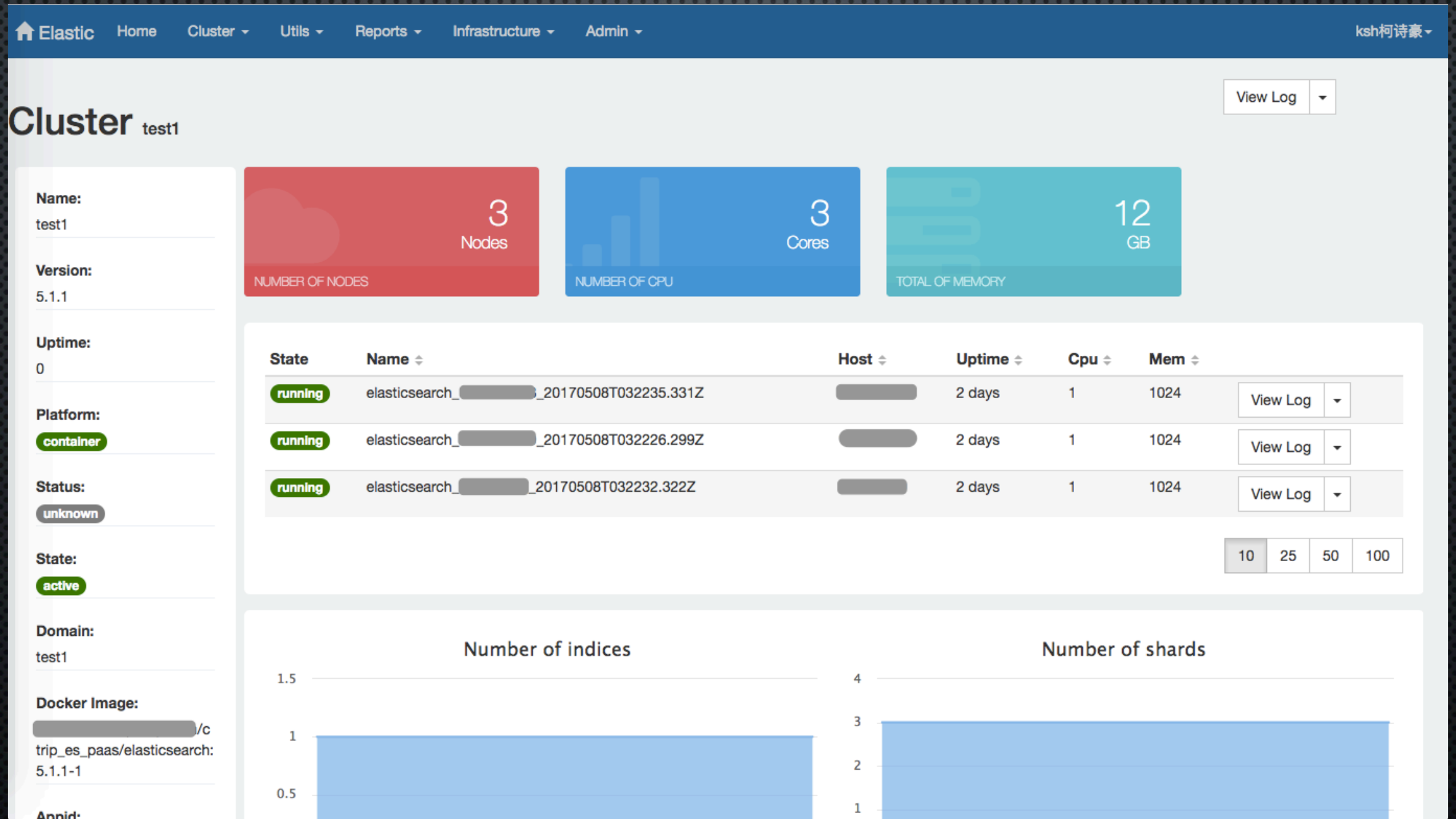
Online | Offline | Pending

Platform	State	Name	Domain	Version	Node	Uptime	Cpu	Mem	
container	unknown	test1	test1	5.1.1	3	0	3	12	Performance
container	green	test3	test3	5.1.1	3	0	3	12	Performance
container	green	test4	test2.ops.ctrip.com	5.3.0	1	0	0	0	Performance
container	green	test6	test6.ops.com	5.3.0	1	0	0	0	Performance
container	green	test7	test7.test.com	5.3.0	2	0	0	0	Performance
container	green	test8	test8.test.com	5.3.0	1	0	0	0	Performance

创建集群


Elastic
Home
Cluster
Utils
Reports
Infrastructure
Admin
Advanced >>
Create Cluster
Name *
test1
Number of Nodes
2
3
4
5
Resource Type *
1C4G100G
ElasticSearch Version *
5.1.1
Plugins
Choose plugins...
DNS Domain *
test1
es.test.com
Appid
666666
Reset
Submit

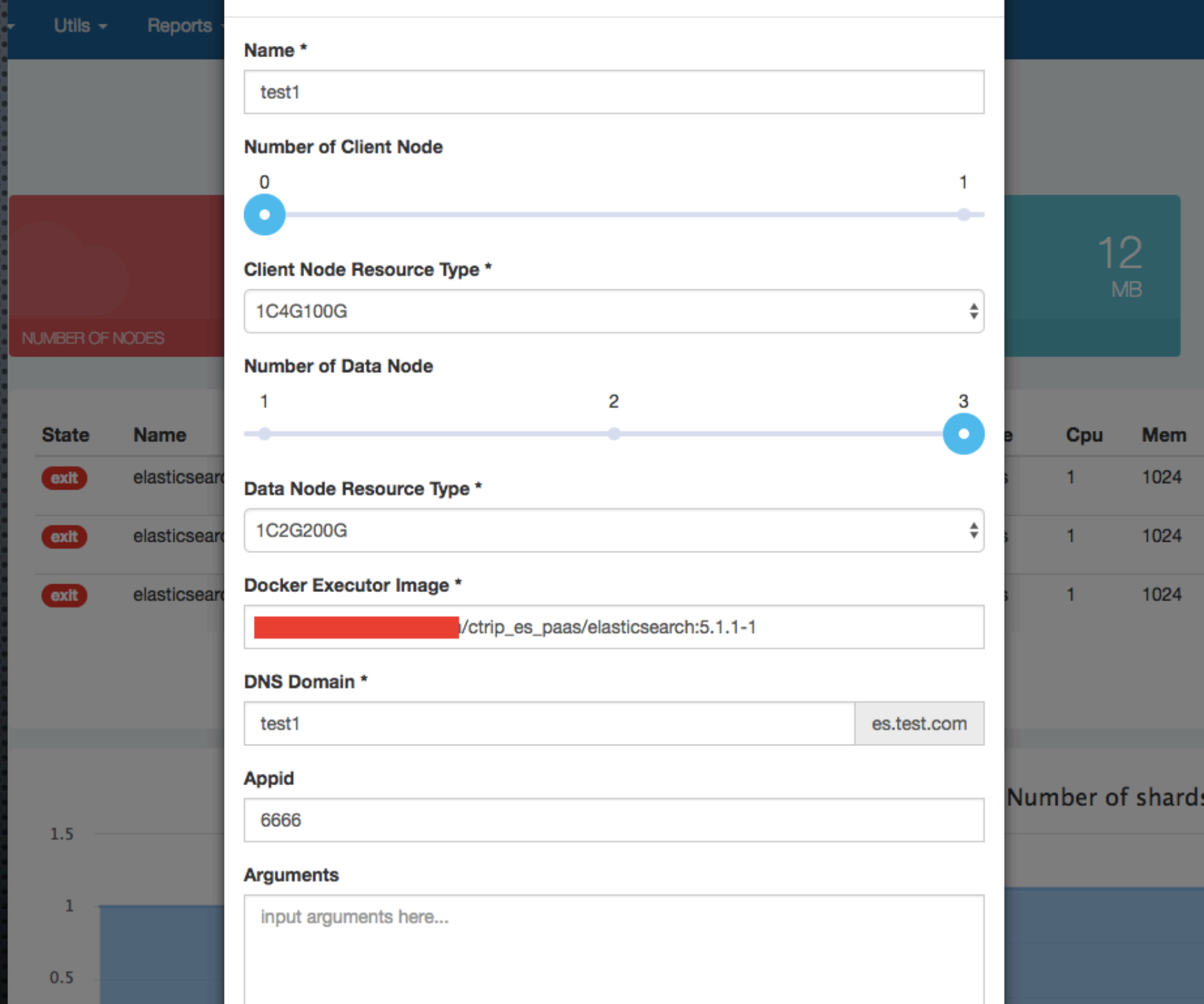
集群创建完成


Elastic
Home
Cluster
Utils
Reports
Infrastructure
Admin
ksh柯诗豪
View Log
Cluster test1
Name:
test1
Version:
5.1.1
Uptime:
0
Platform:
container
Status:
unknown
State:
active
Domain:
test1
Docker Image:
/c
trip_es_paas/elasticsearch:
5.1.1-1
Appid:
3
Nodes
NUMBER OF NODES
3
Cores
NUMBER OF CPU
12
GB
TOTAL OF MEMORY
State
Name
Host
Uptime
Cpu
Mem
running
elasticsearch_ _20170508T032235.331Z
2 days
1
1024
View Log
running
elasticsearch_ _20170508T032226.299Z
2 days
1
1024
View Log
running
elasticsearch_ _20170508T032232.322Z
2 days
1
1024
View Log
10
25
50
100
Number of indices
1.5
1
0.5
Number of shards
4
3
2
1

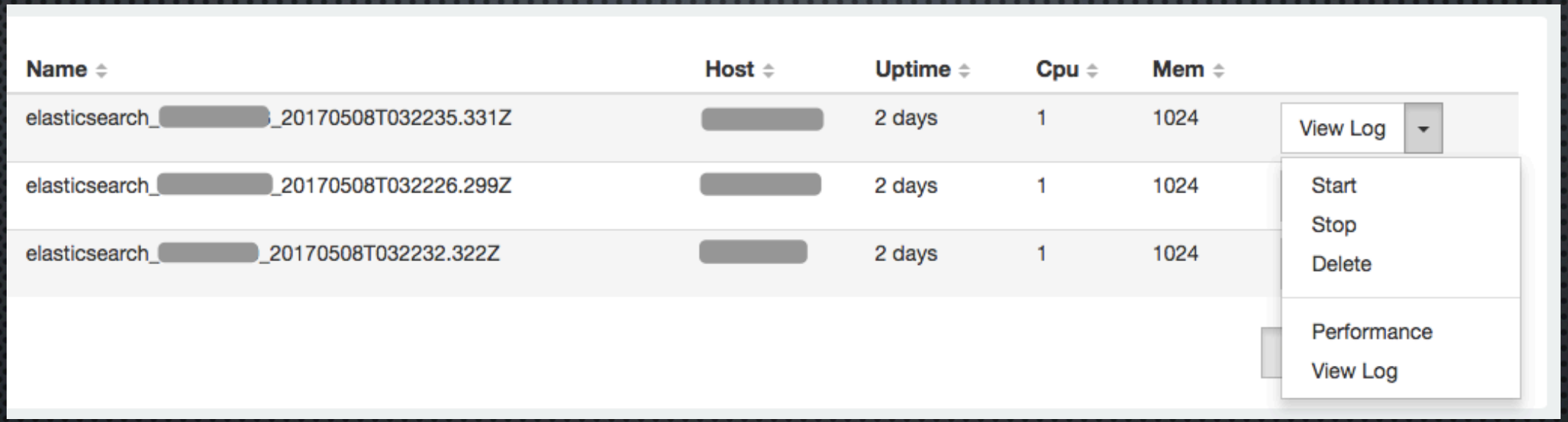
扩容和缩容

- 水平
- 垂直

垂直扩容需要结合水平扩容才能生效



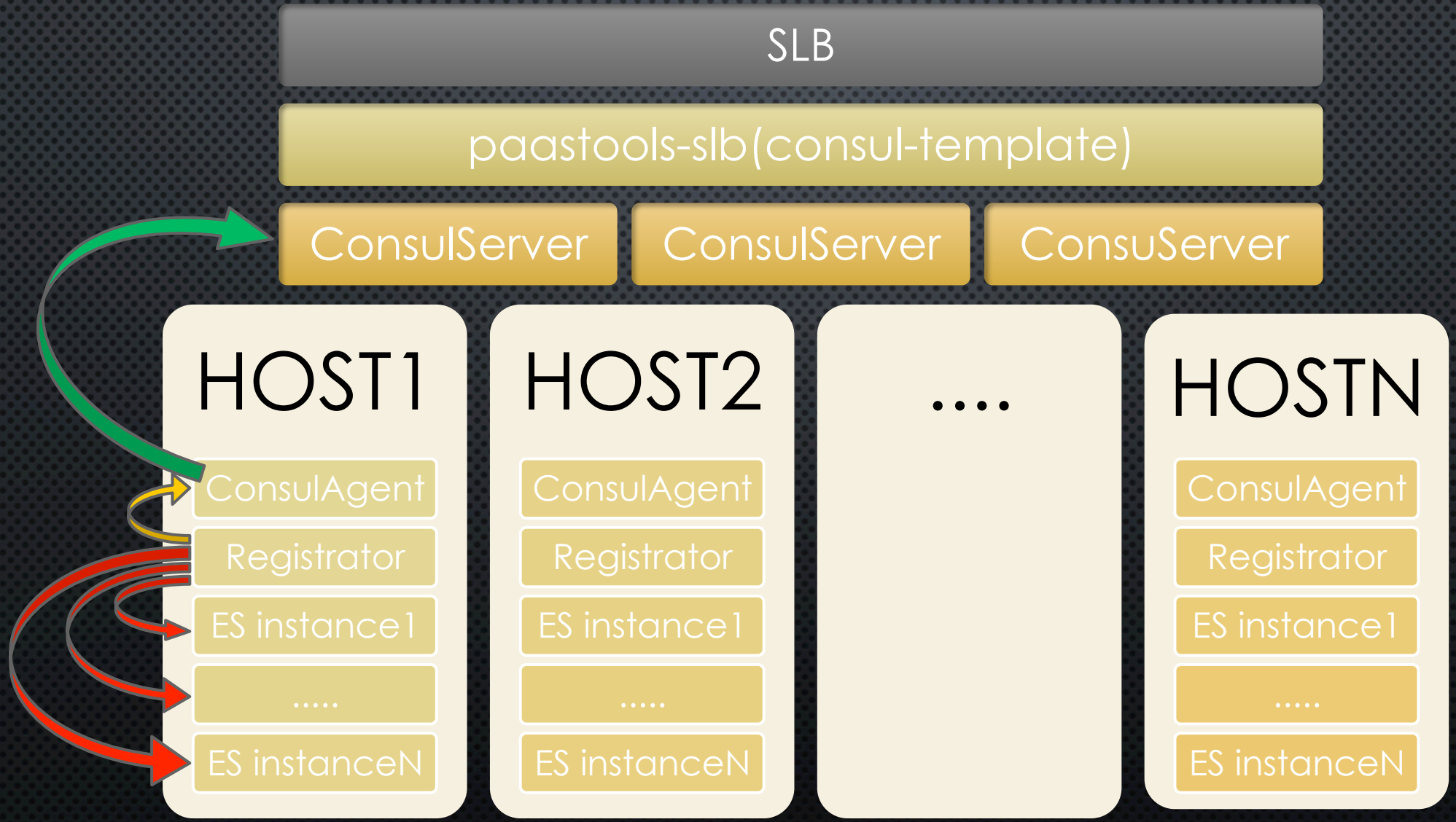
对单结点启停和删除



停止一个结点，ES会close之后结束进程，结点信息记录保留

启动一个结点一分钟内无法成功启动（可能因为无资源或者发生异常），会尝试在新的HOST上启动一个新结点

服务发现


SLB
paastools-slb(consul-template)
ConsulServer
ConsulServer
ConsuServer
HOST1
ConsulAgent
Registrator
ES instance1
.....
ES instanceN
HOST2
ConsulAgent
Registrator
ES instance1
.....
ES instanceN
....
HOSTN
ConsulAgent
Registrator
ES instance1
.....
ES instanceN

服务发现

- Consul
- SLB
- 范域名

Docker 运行时加上两个用于Registrator 检测服务的环境变量
-e SERVICE_9200_NAME=es-paas-test1
-e SERVICE_9200_TAGS=domain=test1.es.test.com,appid=66666666666

Registrator 注册服务到consul

使用Consul-template + SLB更新脚本，更新SLB上的配置

范域名 *.es.test.com 指向 SLB 的 VIP

服务发现

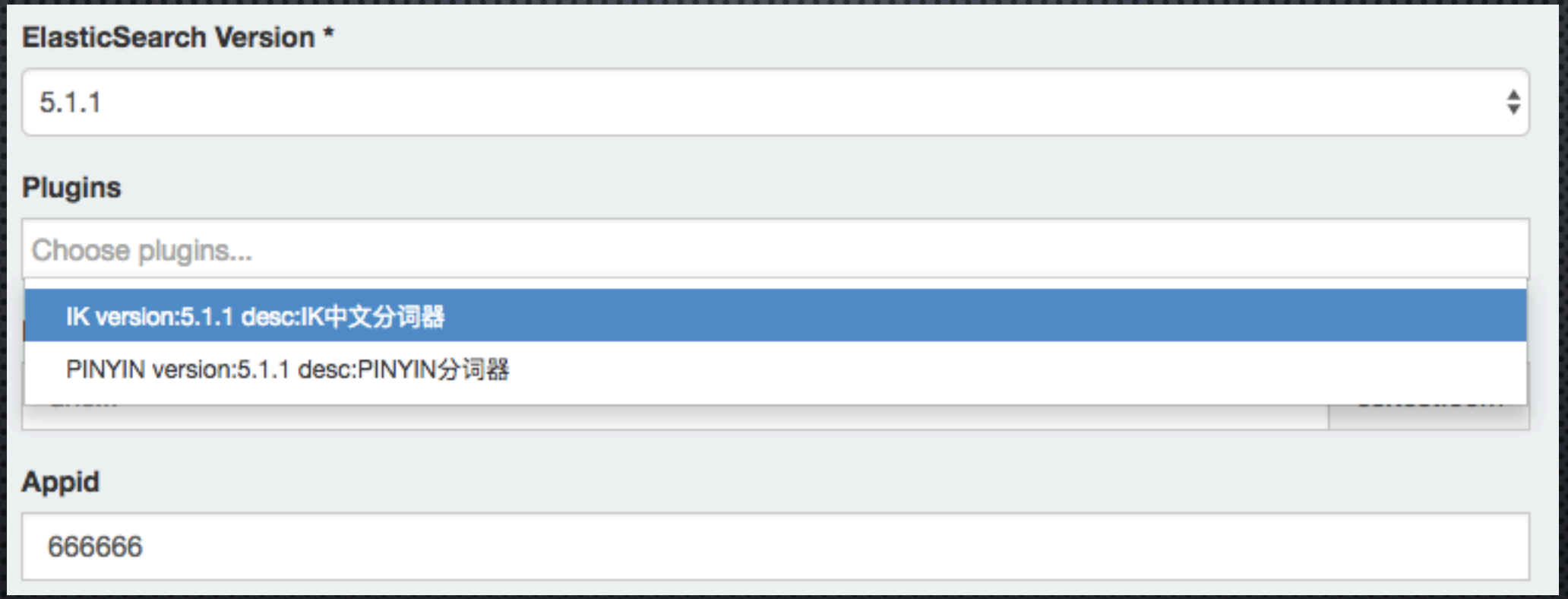
REST 方式访问：

通过域名，域名访问SLB会自动转发到正确的IP和PORT

Transport 方式访问：

先使用REST API /_cat/nodes?h=ip,port 得到ip以及transport的port，再进行连接

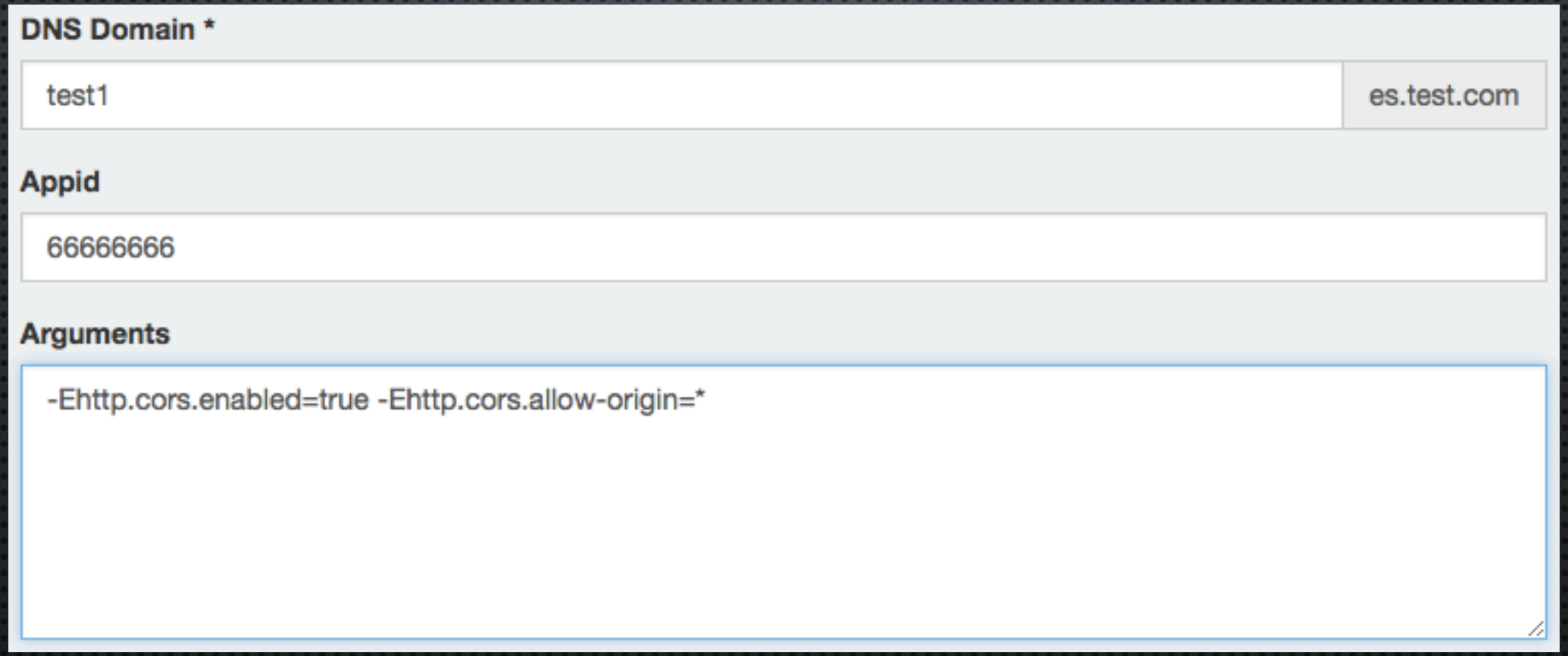
插件配置



插件转换为docker image, 位编码

无插件： ctrip_es_paas/elasticsearch:5.1.1
IK： ctrip_es_paas/elasticsearch:5.1.1-1
PINYIN： ctrip_es_paas/elasticsearch:5.1.1-2
IK+PINYIN： ctrip_es_paas/elasticsearch:5.1.1-3

集群参数设置



暂不支持修改 elasticsearch.yml 配置

2.X 版本 开启script功能

DNS Domain *

test2 | es.test.com

Appid

66666666

Arguments

```
--script.inline=true --script.indexed=true --script.engine.groovy.inline.update=on --script.groovy.sandbox.enabled=true
```

IK字典配置

用户自行上传和更新字典
提供字典FileServer 服务， http 请求返回头中包含 Last-Modifie

集群监控

采用我们团队的hickwall监控系统（数据存储也采用ES）

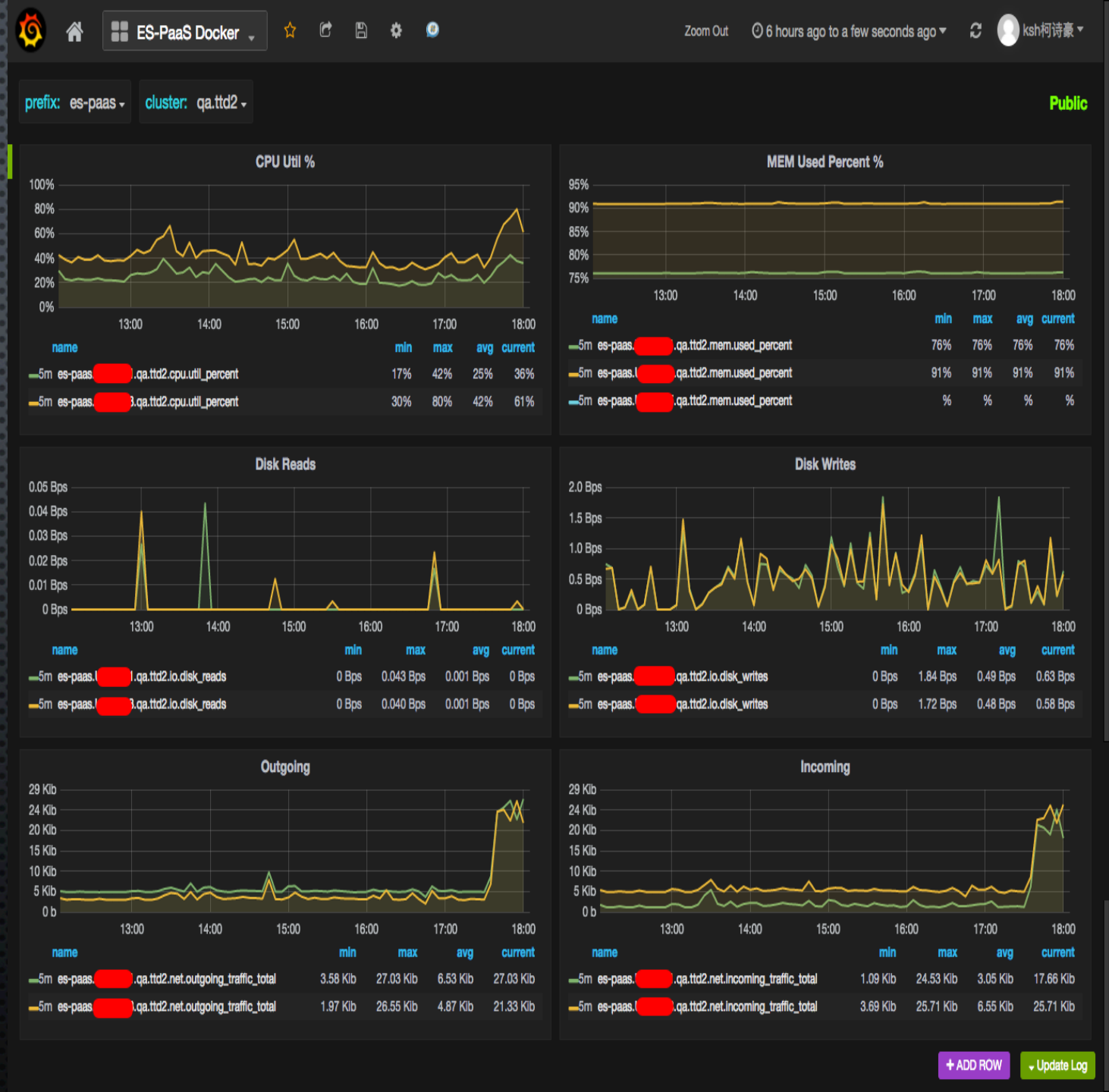
- ES 监控指标



集群监控

采用我们团队的hickwall监控系统（数据存储也采用ES）

- ES 监控指标
- 容器监控指标



集群监控

采用我们团队的hickwall监控系统（数据存储也采用ES）

- ES 监控指标
- 容器监控指标
- 宿主机监控指标



集群安全

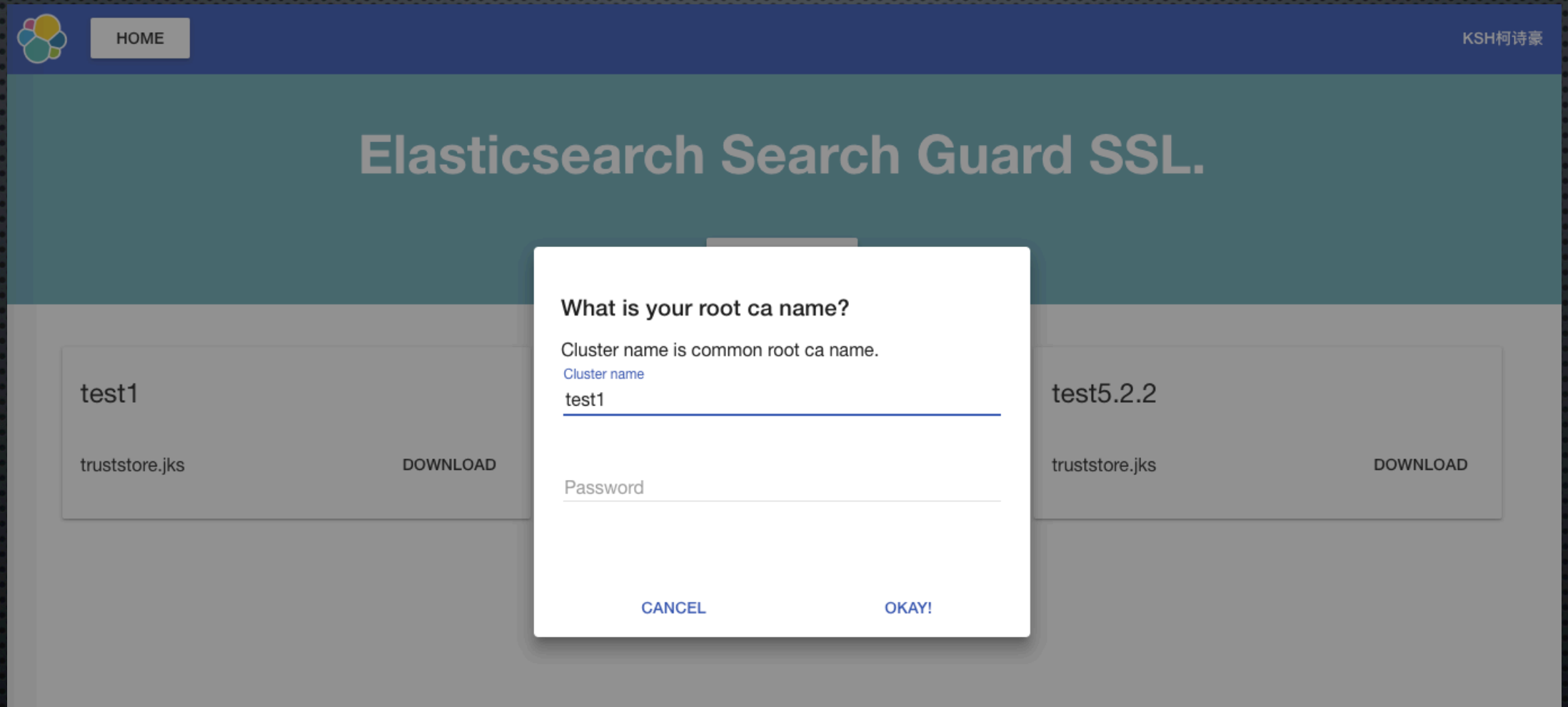
整合Search guard 插件

- 集群中节点之间使用SSL/TSL认证加密通信
- REST API HTTPS
- HTTP Basic Auth
- 针对不同HTTP Auth用户，设置不同的权限

签证书服务

提供证书用于Search guard认证

首先创建一个 ROOT CA，证书名与集群名相同



ES 启动前通过REST API 获取 证书

获取ROOT CA（必须先在证书系统中创建 ）

curl -k –XGET "${CERT_SERVICE}/api/v1/rootca/${CLUSTER_NAME}/"
> /usr/share/elasticsearch/config/truststore.jks

获取NODE 证书（如果证书不存在则会根据参数创建并返回）

curl -k –XGET “${CERT_SERVICE}/api/v1/nodecert/${CLUSTER_NAME}/${NODE_NAME}/?password=${CERT_PASSWORD}&ip=${NODE_IP}” > /usr/share/elasticsearch/config/node.jks

在ES启动前直接把Search guard index 复制到 data目录中，省去使用sgadmin初始化

ES 启动参数设置

-Esearchguard.ssl.transport.keystore_filepath=node.jks
-Esearchguard.ssl.transport.keystore_password=123456
-Esearchguard.ssl.transport.truststore_filepath=truststore.jks
-Esearchguard.ssl.transport.truststore_password=123456
-Esearchguard.ssl.http.clientauth_mode=OPTIONAL
-Esearchguard.ssl.http.enabled=true
-Esearchguard.ssl.http.keystore_filepath=node.jks
-Esearchguard.ssl.http.keystore_password=123456
-Esearchguard.ssl.http.truststore_filepath=truststore.jks
-Esearchguard.ssl.http.truststore_password=123456
-Esearchguard.authcz.admin_dn.0=CN=admin,OU=client,O=Ctrip,L=SH,C=CN

REST 访问需要 HTTP Basic Auth

```
[root@[redacted] ~]# curl --tlsv1.1 -k -XGET -I https://[redacted]:10002
HTTP/1.1 401 Unauthorized
WWW-Authenticate: Basic realm="Search Guard"
content-type: text/plain; charset=UTF-8
content-length: 12

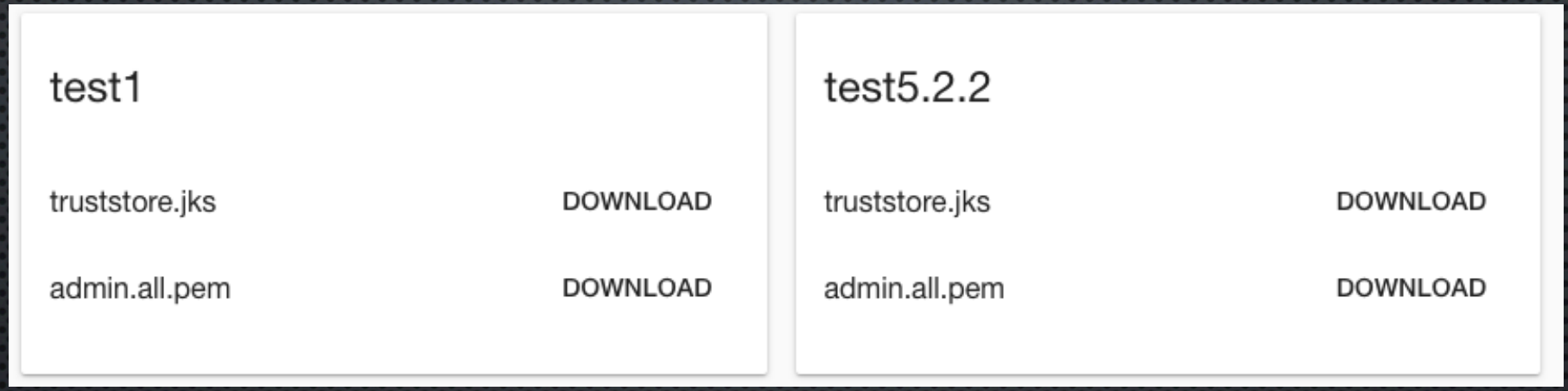
[root@[redacted] ~]# curl --tlsv1.1 -k -XGET -I https://test:test@[redacted]:10002
HTTP/1.1 200 OK
content-type: application/json; charset=UTF-8
content-length: 327
```

对于REST API 具体的权限控制

```
[root@[redacted] ~]# curl --tlsv1.1 -k -XGET https://test:test@[redacted]:10002/_stats?pretty
{
  "error" : {
    "root_cause" : [
      {
        "type" : "security_exception",
        "reason" : "no permissions for indices:monitor/stats"
      }
    ],
    "type" : "security_exception",
    "reason" : "no permissions for indices:monitor/stats"
  },
  "status" : 403
}
```

权限管理

从证书系统中获取Admin证书（DN与 searchguard.authcz.admin_dn 中设置的相同的为admin证书）



使用Admin证书调用REST API进行权限管理

```
[root@[redacted] test5.2.2]# curl --tlsv1.1 -k --cert ./admin.all.pem --key ./admin.all.pem
 https://[redacted]:10002/_searchguard/api/user/test
{
  "test" : {
    "hash" : "$2a$12$1HqHxm3QTfzwkse7vwzhF0V4gDv787cZ8BwmCwNEyJhn0CZoo8VVu"
  }
}[root@[redacted] test5.2.2]#
```

Thanks !