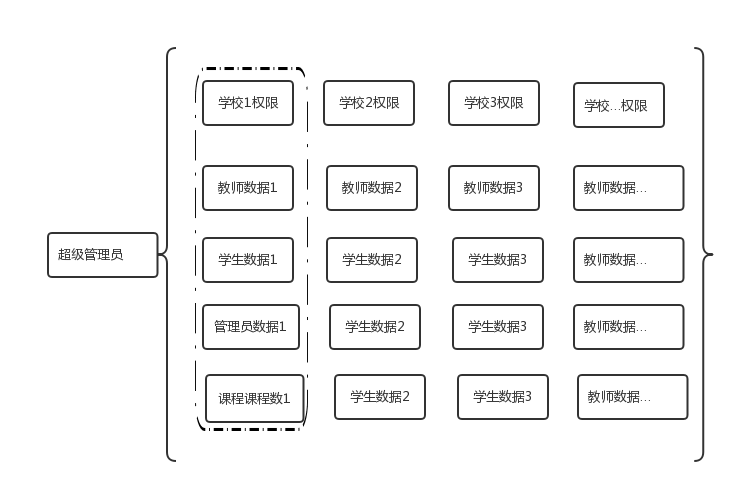
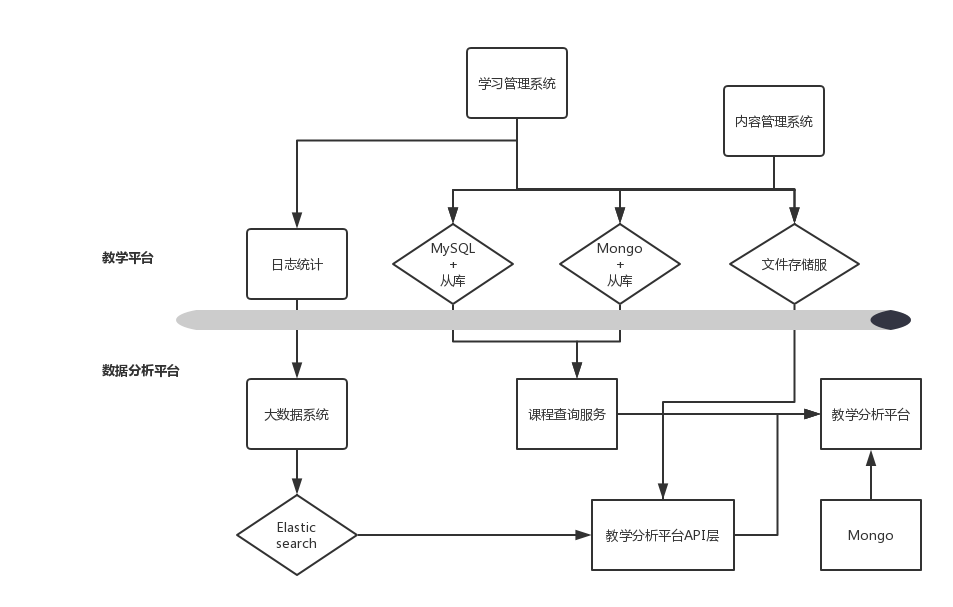
# 整体逻辑

**支持多个学校的学生、教师、课程、管理员分别管理，也可以由超级管理员统一管理数据逻辑如下：**



**主要分为大数据分析平台和教学平台两部分，部署结构如下：**



# 教学平台部分

**一、概述**

1，功能与架构概述

私有云（也称BOX）是将学堂在线主站的各主要模块，部署在一套独立的硬件环境中，提供给特定院校使用（比如可能因为访问公网受限，等等）。所有的模块，包括视频转码和存储模块，都只在这套硬件资源中运行，不与外部公网发生联系。

模块关系结构如下图1.1所示。

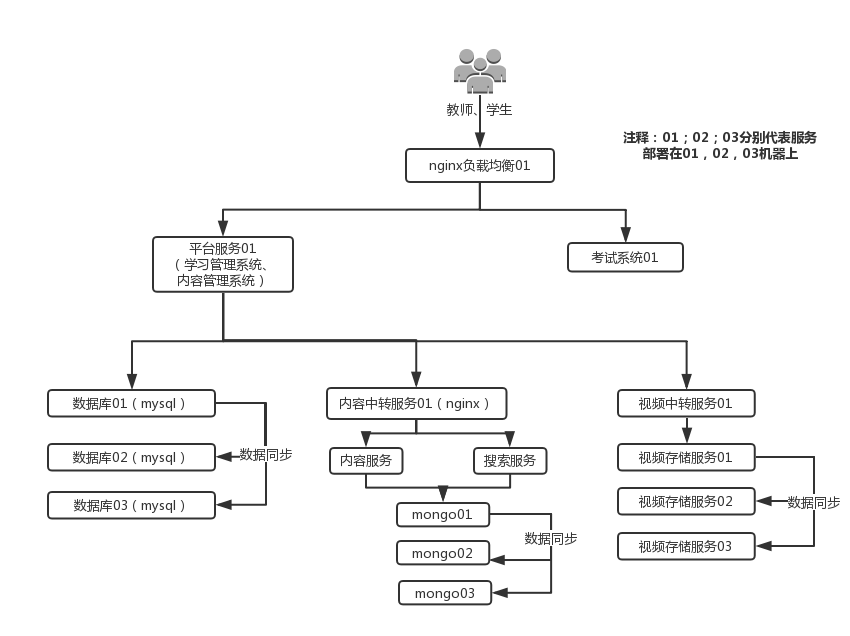


图1.1 学堂私有云模块关系图

2，参考模型各模块配置

**注意，这个架构图，以及下面的表格，都只是一个参考模型，可以根据实际情况，增加各个模块的实例个数（例如下表中platform就有两个）。**

参考模型下，学堂云BOX由一台二层交换机和三台物理物理器组成，物理服务器又分两台计算型和一台存储型。每台物理服务器中又分别运行若干Linux容器(LinuX Container,后面简称LXC)，各个容器运行着不同的服务。各模块的配置如下表所示：

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 物理机 | 虚拟机 | 校园局域网IP | 私网IP | 系统盘G | 数据盘G(最低|正常配置) | 内存(最低|正常配置) |
| SPOC01（计算型） |  | 192.168.9.181 | 172.16.0.11 |  |  |  |
| mongo02 |  | 172.16.0.116 | 15 | 80|130 | 4G|32G |
| mysql02 |  | 172.16.0.114 | 15 | 80|130 | 4G|16G |
| videostore02 |  | 172.16.0.112 | 15 | 150|230 | 4G|16G |
| platform01 |  | 172.16.0.102 | 100 |  | 8G|32G |
| public01 |  | 172.16.0.101 | 50 |  | 2G|16G |
| exam01 |  | 172.16.0.103 | 50 |  | 2G|16G |
|  |  |  |  |  |  |  |
| SPOC02（计算型） |  | 192.168.9.182 | 172.16.0.12 |  |  |  |
| mongo03 |  | 172.16.0.117 | 15 | 150|230 | 4G|32G |
| mysql03 |  | 172.16.0.115 | 15 | 80|130 | 4G|16G |
| videostore03 |  | 172.16.0.113 | 15 | 150|230 | 4G|16G |
| search01 |  | 172.16.0.106 | 50 |  | 2G|16G |
| forum01 |  | 172.16.0.105 | 50 |  | 4G|16G |
| inner01 |  | 172.16.0.104 | 50 |  | 2G|16G |
|  |  |  |  |  |  |  |
| SPOC03（存储型） |  | 192.168.9.183 | 172.16.0.13 |  |  |  |
| mongo01 |  | 172.16.0.109 | 15 | 80|130 | 8G|32G |
| mysql01 |  | 172.16.0.108 | 15 | 80|130 | 8G|16G |
| videocontrol |  | 172.16.0.107 | 15 | 80|130 | 4G|16G |
| videostore01 |  | 172.16.0.111 | 15 | 150|130 | 8G|16G |
|  |  |  |  |  |  |  |

**二，下面分别详细介绍各个组成部分的安装部署步骤。**

机器要求（总）：

1. 机器数量(虚拟机)：16台
2. 内存：配置如上图，共72G
3. 磁盘：1.5T
4. CPU：4核 \* 2线程

物理机准备好之后，将备份的各容器的镜像和配置文件及脚本文件，拷贝至每台物理机的/home/目录下。备份的各容器镜和配置文件，在192.168.9.15:/mnt/newspoc\_box目录中，分别为spoc01\_new，spoc02\_new，spoc03\_new。

**交换机与服务器的安装与配置**

交换机不需要特别地配置。用于各LXC间的联通，使用172.16网段。

**交换机与服务器的安装与配置说明:**

    物理机磁盘配置为RAID5，安装的操作系统为ubuntu 12.04。

    交换机不需要特别地配置。用于各LXC间的联通，使用172.16网段。

    磁盘配置成LVM，卷组名设置为xuetangxvg。物理机除了/boot和swap分区外，系统空间再配置200G的空间（比如/和/home各分配100G）。

    选择软件安装包时，需要选择上openssh和虚拟化组件。

    装好系统后，安装如下软件：

    apt-get install lxc lvm2 bridge-utils build-essential nginx(spoc01,spoc02,spoc03依次安装)

    /etc/hostname依次设置为spoc01、spoc02、spoc03（存储型服务器，存储空间最大的那台服务器）。密码设置为xuetangXspoc，包括后面各个虚拟机，root帐号密码也都                          配置为xuetangXspoc

    若安装时没有配置卷组，则需要建立卷组，以将/dev/sda4配置成卷组为例(spoc01,spoc02,spoc03依次安装)：

    pvcreate /dev/sda4

    vgcreate xuetangxvg /dev/sda4

**学堂私有云BOX服务快速安装部署:**

1，服务器网络配置

    物理机配置内网和外网网桥，如下，以spoc01为例（外网IP需要根据实际情况进行调整）：

     root@spoc01:~# cat /etc/network/interfaces

    (参考文件/home/spoc01\_new/etc\_config/interfaces)

    cat /etc/resolv.conf

    (参考文件/home/spoc01\_new/etc\_config/resolv.conf)

    cat /etc/hostname

    (参考文件/home/spoc01\_new/etc\_config/hostname)

3，nginx配置

platform服务、public(部分网站静态文件)服务、videostore服务（三台videostore分别位于不同的三台物理服务器中，由各自所在的物理服务器进行转发）、videocontrol（由SPOC03转发）服务，都由各模块所在的物理服务器通过nginx进行转发。

spoc01:cp -r /home/spoc01\_new/nginx/\* /etc/nginx/sites-enabled/

spoc02:cp -r /home/spoc02\_new/nginx/\* /etc/nginx/sites-enabled/

spoc03:cp -r /home/spoc03\_new/nginx/\* /etc/nginx/sites-enabled/

每次nginx文件修改后，要

# /etc/init.d/nginx reload

重新载入生效

5，容器配置文件

然后将每个物理服务器中的容器的配置文件，拷贝至相应目录：

spoc01: cp -r /home/spoc01\_new/lxc\_config/lxc /var/lib/

spoc02: cp -r /home/spoc02\_new/lxc\_config/lxc /var/lib/

spoc01: cp -r /home/spoc02\_new/lxc\_config/lxc /var/lib/

**三，各容器安装与配置**

spoc01为例:

cd /home/spoc01\_new/lxc\_create\_sh/

依次执行各个脚本

spoc02，spoc03同上

**四，各容器开关机步骤**

spoc01:

cd /home/spoc01\_new

sh [service\_start.sh](http://service_start.sh/)

spoc02，spoc03同上

**线上静态文件服务安装部署**

线上静态文件仅在spoc02中部署(nginx已经配置好，可查看/etc/nginx/sites-enabled/edxapp文件，使用端口号11111)

将s.tar.gz解压到/var/www/中

口令

mkdir /var/www/

cd /home/spoc02\_new

tar xzf s.tar.gz -C /var/www/

# 私有云升级步骤

使用升级后的platform镜像和mysql、forum镜像。

升级后的私有云使用的也是vpc\_spoc\_vpcsite中的一条记录，与公有云无异。

但在CMS端去掉了网站管理的功能，因此图片配置等需要手动配置，切要存在本地，不可使用线上storage的资源。

静态文件存放的路径 例如

microsites/basic/assets/images/home.jpg

## 不使用域名访问

如果不想使用域名访问只用IP访问的话，需要修改以下文件

common/djangoapps/microsite\_configuration/middleware.py

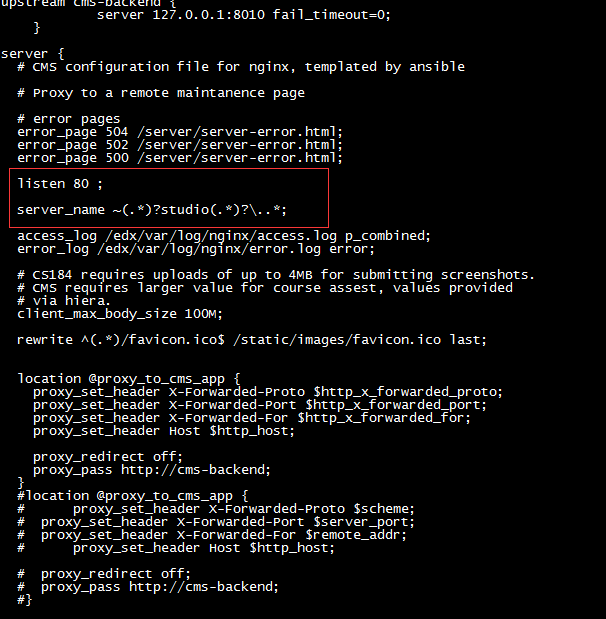
将里面的spocdemo.xuetangx.com改成具体使用的vpc\_spoc\_vpcsite的域名。

这个文件的这段代码的意思就是如果没有检测到输入的域名就使用一个默认的域名。

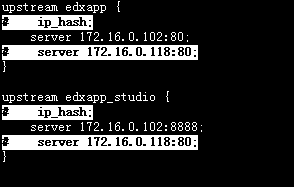
studio端则需要修改172网段platform的nginx的cms文件

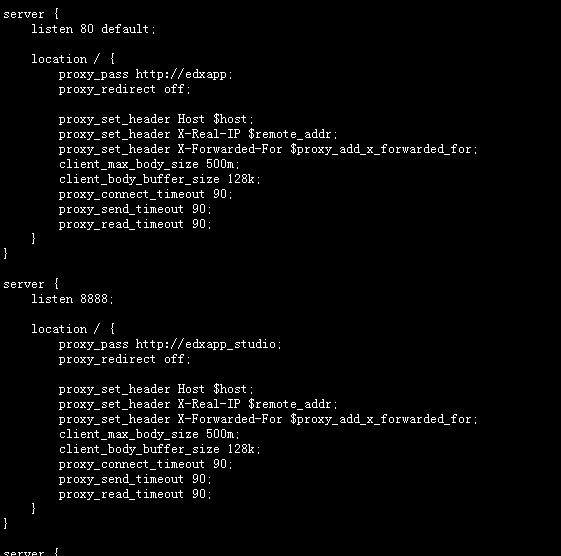
172.16.0.102

vim /etc/nginx/sites-enabled/cms



修改物理机（spoc01）的nginx





将监听的80端口和sever\_name注释换成需要的端口例如：8888

这样访问时则输入ip加端口号就可以访问studio端，例如：

192.168.9.181:8888

账户管理部分和考试系统目前暂不可用 考试系统因为我之前没有了解还没办法或者不知道能不能结合私有云使用

账户管理则是因为使用了work的异步任务 且需要storage的上传服务 等等

**三，学堂在线私有云服务安装完成后的配置操作**

检查mongo服务是否启动成功：

    测试ping 172.16.0.109(mongo01为例)

    连接ssh 172.16.0.109

    root@mongo01:~# mongo

    MongoDB shell version: 2.4.7

    connecting to: test

    Server has startup warnings:

    ... ...

    xuetang\_set:SECONDARY>  # 此时还只是SECONDARY mongo

    mongo01中还有memcache进程，还需要检查：

    # ps aux | grep memcache

    三个mongo都起来后，可以在mongo02、mongo03中，进到mongo中，可以看到SECONDARY，mongo01中进到mongo，可以看到PRIMARY(其实只要mongo02起来后，mongo01就是PRIMARY了)。

mysql设置主从同步：

    mysql01(172.16.0.13)，mysql02(172.16.0.11)，mysql03(172.16.0.12)

    mysql服务启动后，要在两个从机中，检查一下主从是否同步

    show slave status \G;

    若有错误，需要重新设置主从

    在主中，show master status;得到当前的File和Position

    然后在从中，都从该文件和位置处同步：

    slave stop;

    change master to master\_log\_file='mysql-bin.000004', master\_log\_pos=107;

    start slave;

videostore服务启动转码：

ssh到videostore服务

进入到~目录，然后执行

# sh [run\_storage.sh](http://run_storage.sh/)

并检查

# ps aux | grep python

有重启的新的python [server.py](http://server.py/) 8000和python [server.py](http://server.py/) 8001进程才可以。

这一步比较重要，不仅是首次安装时需要，以后每次重启videostore后，都要进行这样一步操作，以起动转码服务。

search服务启动：

进入search服务，启动search服务

# cd /opt/wwc/site-search/

# make parse

# make index

# nohup make search &

回车

现在索引是每天凌晨填加一遍，若想在每建完一门课后，很快生效，需要修改crontab的任务执行间隔，或者手动执行：

# make -C /opt/wwc/site-search parse

# make -C /opt/wwc/site-search index

IP、域名的配置

配置为客户名的物理机IP、域名，除了配置物理机的网络配置外，还有以下几个地方：

如果最终需要使用外网IP提供视频服务，需要回到SPOC03中

修改videocontrol中记录的对应的三个videostore的IP

ssh 172.16.0.107（进入videocontrol中）

# mysql

# use videoservice

然后修改videoserver\_storageserver表中的server\_ip字段，将三个videostore的server\_ip字段中的ip地址修改为当前对应物理机的ip地址。

# quit

# 大数据分析平台部分

## 一、系统

### 1、软件依赖

| **软件** | **版本** |
| --- | --- |
| python | 2.7 |
| java | 1.8.0 (8u45) |
| mysql | 5.5 |
| mongo | 2.4 |
| nginx | 1.4.6 |
| ubuntu | 14.04 |
| pip | 1.5.4 |

### 2、服务及硬件依赖

最优配置：

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **服务** | **域名（x为ip最后一位）** | **磁盘** | **内存 大小(G)** | **机器数量** | **系统** | **CPU** |
| cdh | cdh-{x} | 2000G | 64G | 6 | ubuntu 14.04 | 12核, 每核2线程 |
| cdhmysql | cdhmysql-{x} | 200G | 8G | 2 | ubuntu 14.04 | 4核, 每核2线程 |
| elasticsearch | elasticsearch-{x} | 300G | 32G | 3 | ubuntu 14.04 | 4核, 每核2线程 |
| tapapi + tap | tap-{x} | 50G | 8G | 2 | ubuntu 14.04 | 4核, 每核2线程 |
| mongo | mongo-{x} | 50G | 8G | 3 | ubuntu 14.04 | 4核, 每核2线程 |
| course\_service+ansible | course-service-{x} | 50G | 8G | 2 | ubuntu 14.04 | 4核, 每核2线程 |
|  |  |  |  | 共18台 |  |  |

最低配置（总）：

1. 机器数量：6台
2. 内存：16G \* 6台
3. 磁盘：650G \* 6台
4. CPU：4核 \* 2线程 \* 6台

## 二.部署方案

1.mysql主从复制搭建

2.monggo 集群搭建

3.cdh集群搭建

4.azkaban部署

5.elasticsearch集群搭建

6.教学分析平台部署（tap tapapi course-service）

下面分别介绍每种搭建步骤

## ****1.mysql 主从复制搭建****

1. 安装mysql
2. #本次测试使用两个虚拟机：
3. ip：192.168.6.235（主）
4. ip：192.168.6.236（从）

apt-get install mysql-server 安装过程中会提示输入用户名及密码

1. 修改mysql配置文件（主 和 从）
2. vim /etc/my.cnf
3. 在[mysqld]下添加
4. server-id=1
5. log-bin=mysql-bin
6. 保存，其中主从数据库的server-id必须唯一 主1 从2 或者各自ip末位

server mysql restart mysql 启动重启 /etc/init.d/mysqld restart

1. 查看主数据库状态（主）
2. #登陆mysql命令行，键入：
3. GRANT REPLICATION SLAVE ON \*.\* TO 'root'@'192.168.6.%' IDENTIFIED BY 'root123';

show master status; #查看File和Position的值

1. 从数据库连接主数据库（从）
2. #登陆mysql命令行，键入：
3. change master to
4. master\_host='192.168.6.235',
5. master\_user='root',
6. master\_password='root123',
7. master\_port=3306,
8. master\_log\_file='mysql\_binary\_log.000001',
9. master\_log\_pos=1163;

#注意其中master\_log\_file的值为步骤2中的File，master\_log\_pos值为步骤一中的Position

1. 启动从服务器复制功能（从）
2. #登陆mysql命令行，键入
3. show slave status\G;
4. #其中返回下列两行内容说明设置成功 Slave\_IO\_Running：Yes Slave\_SQL\_Running:Yes

#在主库上创建一个数据库试试吧！

1. 验证

用一个简单的例子：在主数据库创建名为mysql\_test的数据库，那么在从数据库中也能查询出名为mysql\_test数据库

## ****2.monggo 集群搭建****

1. Mongdb集群模式有三种安装模式：
   * Master-Slaver  
     仅提供为数据同步功能。主节点挂掉需要手动处理，已被官方宣称废弃。
   * **Replica Set**  
     可提供数据同步、主从切换的高可用功能。此处采取本方式。  
     官方：automatic failover and data redundancy。
   * Sharding  
     真正大数据解决方案，在Replica Set的基础上还需要配备路由、分片等，配备较为复杂，且吃资源。此处不必要采用。  
     官方：[Sharding](https://docs.mongodb.com/manual/sharding/#sharding-introduction) distributes data across a cluster of machines、supports creating [zones](https://docs.mongodb.com/manual/core/zone-sharding/#zone-sharding) of data based on the [shard key](https://docs.mongodb.com/manual/reference/glossary/#term-shard-key)。
2. 节点架构

Mongodb(M)表示主节点，Mongodb(S)表示备节点，Mongodb(A)表示仲裁节点。主备节点存储数据，仲裁节点不存储数据。客户端同时连接主节点与备节点，不连接仲裁节点。

默认设置下，主节点提供所有增删查改服务，备节点不提供任何服务。但是可以通过设置使备节点提供查询服务，这样就可以减少主节点的压力，当客户端进行数据查询时，请求自动转到备节点上。可以参考[Read Preference Modes](http://docs.mongodb.org/manual/applications/replication/#read-preference-modes)

仲裁节点是一种特殊的节点，它本身并不存储数据。但没有它就无法主备切换。

1. 服务安装
2. 1.在所有节点上安装Mongodb服务
3. apt-get install mongodb
4. 2.修改各节点配置文件
5. dbpath=/var/lib/mongodb #数据存放目录
6. logpath=/var/log/mongodb/mongodb.log #where to log
7. logappend=true #以追加的方式记录日志
8. replSet=replSets #replica set的名字
9. directoryperdb=true #为每一个数据库按照数据库名建立文件夹存放
10. bind\_ip = 192.168.6.23X #本服务绑定的ip地址
11. port = 27017 #进程所使用的端口号，默认为27017
12. journal=true # Enable journaling
13. fork=true #以后台方式运行进程
14. noprealloc=true #不预先分配存储
15. #注意：bind\_ip需要按照各节点设置。其他可相同。
16. 3.启动各服务
17. # mongd –f /etc/mongodb.conf
18. 4.初始化设置主、备、仲裁节点
19. 选择一个节点，连接到mongo
20. # mongo host:port
21. 例如：# mongo 192.168.6.237:27017
22. >use admin
23. >cfg={\_id:"replSets",members:[{\_id:0,host:'192.168.6.237:27017',priority:2}, {\_id:1,host:'192.168.6.236:27017',priority:1},
24. {\_id:2,host:'192.168.6.235:27017',arbiterOnly:true}] };
25. >rs.initiate(cfg)
26. >rs.status()
28. cfg只是一个名词，只要不是mongo关键字即可。
29. 注意id需要为replSet的名称，配备host:port以及priority优先级，最高者为master。仲裁节点需要指定arbiterOnly:true。
30. 5.测试

可测试主备切换、数据同步功能、DSL功能。

## ****3.cdh集群搭建****

## 一、引言

### 1、CDH

hadoop是一个开源项目，所以很多公司在这个基础进行商业化，Cloudera对hadoop做了相应的改变。Cloudera公司的发行版，我们将该版本称为CDH（Cloudera Distribution Hadoop）

### 2、术语约定

CDH：Cloudera Distribution Hadoop

## 二、下载和准备CDH的Parcel文件

在CDH-2-120上, 下载CDH的Parcel文件, 放在/srv/cdh-parcels/cdh5/parcels/5.7.0/

mkdir -p /srv/cdh-parcels/cdh5/parcels/5.7.0/

wget https://archive.cloudera.com/cdh5/parcels/5.7.0/CDH-5.7.0-1.cdh5.7.0.p0.45-precise.parcel \

-O /srv/cdh-parcels/cdh5/parcels/5.7.0/CDH-5.7.0-1.cdh5.7.0.p0.45-precise.parcel

wget https://archive.cloudera.com/cdh5/parcels/5.7.0/manifest.json \

-O /srv/cdh-parcels/cdh5/parcels/5.7.0/manifest.json

安装nginx, 把parcel文件做成http可下载的, 因cloudera-manager-agent需要使用apache2(占用80端口), 需要配置nginx不使用80端口

apt-get install nginx

配置nginx, 删除文件/etc/nginx/sites-enabled/default, 添加文件/etc/nginx/sites-enabled/cdh, 内容如下

server {

listen 10080;

root /srv/cdh-parcels/;

server\_name localhost;

}

重启nginx即可

service nginx restart

# 也可以用reload

在其他服务器上测试一下, 确认能返回 200 OK

curl -I http://10.0.2.120:10080/cdh5/parcels/5.7.0/CDH-5.7.0-1.cdh5.7.0.p0.45-precise.parcel

HTTP/1.1 200 OK

Server: nginx/1.1.19

Date: Wed, 18 May 2016 02:28:19 GMT

Content-Type: application/octet-stream

Content-Length: 1535640207

Last-Modified: Thu, 31 Mar 2016 19:55:14 GMT

Connection: keep-alive

Accept-Ranges: bytes

## 安装MySQL数据库

在CDH-2-118安装mysql, 并且创建3个数据库, 安装过程中会提示输入root的密码, 输入\*\*\*\*\*\*\*\*

apt-get install mysql-server

# 默认安装之后已经启动了, 停止mysql, 配置之后再启动

service mysql stop

编辑 /etc/mysql/my.cnf, 找到 "bind-address      = 127.0.0.1" 这行, 注释掉, 添加文件/etc/mysql/conf.d/cloudera.cnf, 内容如下

[mysqld]

binlog\_format = row

删掉mysql之前生成的一些文件(与innodb有冲突, 不删起不来), 再启动mysql

rm -f /var/lib/mysql/ib\*

service mysql start

登录到mysql

mysql -u root -p\*\*\*\*\*\*\*

创建数据库

> create database DBNAME default charset utf8;

> grant all on DBNAME.\* to DBNAME@'10.0.%' identified by 'DBPASSWORD';

> flush privileges;

总共需要创建3个数据库, cmf, hive, oozie. 记录root和这三个数据库,用户的密码

## 安装ansible

使用ansible是为了能同时操作多台服务器, 不需要在每台服务器上做同样的事情.

在CDH-2-120上创建ansible帐号, 安装ansible, 的过程略.

在CDH-2-120的/etc/hosts文件中添加如下几行

10.0.2.117 CDH-2-117

10.0.2.118 CDH-2-118

10.0.2.119 CDH-2-119

10.0.2.120 CDH-2-120

添加文件 /etc/ansible/hosts文件, 内容如下

[cdh]

CDH-2-[117:120]

### 配置ansible

注:各机器安装python 才可以使用ansible命令

以下都是用ansible帐号操作

创建ssh的key

ssh-keygen -t rsa

# 一路回车即可

然后建立信任关系

ssh-copy-id root@CDH-2-117

# 包括118, 119, 120

可以测试一下是否成功配置

# 使用ansible登录之后, 在运行任何ansible命令之前, 先运行如下三行命令

source ansible/hacking/env-setup

ssh-agent /bin/bash

ssh-add ~/.ssh/id\_rsa

# 测试, 返回4个SUCCESS表示安装成功

ansible cdh -u root -m ping

CDH-2-120 | SUCCESS => {

"changed": false,

"ping": "pong"

}

CDH-2-119 | SUCCESS => {

"changed": false,

"ping": "pong"

}

CDH-2-118 | SUCCESS => {

"changed": false,

"ping": "pong"

}

CDH-2-117 | SUCCESS => {

"changed": false,

"ping": "pong"

}

# 安装server和agent

使用ansible来安装

##### 设置hostname

vi ~/cdh-install/etc/hosts

127.0.0.1 localhost

# The following lines

::1 ip6-localhost

fe00::0 ip6-localnet

ff00::0 ip6-mcastpref

ff02::1 ip6-allnodes

ff02::2 ip6-allrouter

192.168.6.235 cdh-235

192.168.6.236 cdh-236

192.168.6.237 cdh-237

192.168.6.238 cdh-238

192.168.6.239 cdh-239

192.168.6.240 cdh-240

ansible cdh -u root -m copy -a "src=~/cdh-install/etc/hosts dest=/etc/hosts mode=0644"

ansible cdh -u root -m hostname -a 'name={{ ansible\_host }}'

##### 设置locale

ansible cdh -u root -m copy -a "src=~/cdh-install/etc/default/locale dest=/etc/default/locale mode=0644"

ansible cdh -u root -a "locale-gen en\_US.UTF-8"

ansible cdh -u root -a "locale-gen zh\_CN.UTF-8"

##### 准备apt的源

ansible cdh -u root -m apt\_key -a "url=http://archive.cloudera.com/cdh5/ubuntu/trusty/amd64/cdh/archive.key"

ansible cdh -u root -m copy -a "src=~/cdh-install/etc/apt/sources.list.d/cloudera.list dest=/etc/apt/sources.list.d/cloudera.list mode=0644"

ansible cdh -u root -m apt -a "update\_cache=yes"

##### 安装依赖项

ansible cdh -u root -m command -a "apt-get install -y apache2 fuse fuse-utils libfuse2"

ansible cdh -u root -m command -a "apt-get install -y libsasl2-modules libsasl2-modules-gssapi-mit libssl-dev libxslt1.1"

ansible cdh -u root -m command -a "apt-get install -y openssl perl psmisc python-mysqldb python-psycopg2"

ansible cdh -u root -m command -a "apt-get install -y rpcbind libmysql-java"

##### 安装JDK 1.8

ansible cdh -u root -m file -a "dest=/usr/java mode=755 state=directory"

# 需要先把jdk-8u45-linux-x64.tar.gz 拷贝到~/cdh-install/jdk/

ansible cdh -u root -m unarchive -a "src=~/cdh-install/jdk/jdk-8u45-linux-x64.tar.gz dest=/usr/java"

##### 安装Cloudera Manager Server

方法1:

因为网络比较慢, 需要下载500多M的文件, 所以不推荐这种安装方法, 虽然也能安装成功,但是要等N个小时. 推荐使用 方法2

ansible cdh -u root -m apt -a "name=cloudera-manager-daemons"

ansible CDH-2-118 -u root -m apt -a "name=cloudera-manager-server"

方法2:

登录CDH-2-118, 安装 cloudera-manager-daemons cloudera-manager-server. 这个过程比较慢, 如果速度很慢了,就停止掉, 再运行. apt会从断的地方继续开始下.

apt-get install cloudera-manager-daemons cloudera-manager-server

安装完之后, 把 /var/cache/apt/archives/cloudera-manager-daemons\_5.7.0-1.cm570.p0.76~precise-cm5\_all.deb 拷贝到每台机器的 /opt/下面

scp /var/cache/apt/archives/cloudera-manager-daemons\_5.7.0-1.cm570.p0.76~precise-cm5\_all.deb root@CDH-2-117:/opt

# 119, 120 也是一样

然后登录每台机器, 使用的dpkg进行安装

dpkg -i /opt/cloudera-manager-daemons\_5.7.0-1.cm570.p0.76~precise-cm5\_all.deb

##### 安装Cloudera Manager Agent

ansible cdh -u root -m apt -a "name=cloudera-manager-agent"

##### 启动Cloudera Manager Server

登录CDH-2-118, 修改 /etc/cloudera-scm-server/db.properties, 内容如下

com.cloudera.cmf.db.type=mysql

com.cloudera.cmf.db.host=CDH-2-118

com.cloudera.cmf.db.name=cmf

com.cloudera.cmf.db.user=cmf

com.cloudera.cmf.db.password=cmf\_password

启动

service cloudera-scm-server start

第一次启动时间比较长, 可以看启动的日志

tail -f /var/log/cloudera-scm-server/cloudera-scm-server.log

看到类似如下的文字, 表示启动完成

2016-05-17 08:36:30,900 INFO WebServerImpl:org.mortbay.log: jetty-6.1.26.cloudera.4

2016-05-17 08:36:30,902 INFO WebServerImpl:org.mortbay.log: Started SelectChannelConnector@0.0.0.0:7180

2016-05-17 08:36:30,902 INFO WebServerImpl:com.cloudera.server.cmf.WebServerImpl: Started Jetty server.

# 安装CDH

打开Cloudera Manager Server的页面进行CDH的安装. 用浏览器打开 [http://ip:7180](http://10.0.2.118:7180), 用户名和密码都是 admin

过程见下面的pdf文档



## ****4.azkaban部署****

## 一、引言

#### AZKABAN简介

            azkaban是一种类似于Oozie的工作流程引擎，可以用来解决多个Hadoop（或Spark等）离线任务计算任务之间的依赖关系问题。也可以替代crontab来对周期任务进行调度，并且更为直观，可靠，同时提供了美观的可视化管理界面。

## 二、系统部署

#### 下载源码azkaban3.0

* 1. 安装git wget
     1. sudo apt-get install git
     2. sudo apt-get install wget

sudo wget https://github.com/azkaban/azkaban/archive/3.0.0.tar.gz

* + 1. 解压及编译及报错
    2. #解压azkaban至任意目录，我选择/opt，解压之后会azkaban3.0.0目录
    3. tar zxvf 3.0.0.tar.gz /opt
    4. #进入azkaban3.0.0目录执行编译
    5. cd azkaban3.0.0
    6. ./gradlew distZip --info

#编译15分钟报错，发现是fatal: Not a git repository (or any parent up to mount parent )Stopping at filesystem boundary (GIT\_DISCOVERY\_ACROSS\_FILESYSTEM not set).:azkaban-execserver:createVersionFile FAILED，因为下载的zip不是git repo,所以用git命令创建版本文件错误了

* + 1. 正式编译
    2. #下载git repo
    3. git clone https://github.com/azkaban/azkaban.git
    4. #执行后会在/opt/azkaban-3.0.0 下出现一个azkaban目录，cd /opt/azkaban-3.0.0/azkaban
    5. #编译（gradlew文件）
    6. git checkout 3.0.0 （会出现HEAD is now at ff3d3eb... Azkaban 3.0.0字样）
    7. ./gradlew distTar --stacktrace --info
    8. #如果发现报错Could not find Linker 'g++' in system path. 需要先安装g++，sudo apt-get g++
    9. ./gradlew distZip --info

#发现在在azkaban目录下创建了两个目录 /opt/azkaban-3.0.0/azkaban/build/distributions build和distributions

#### ****环境准备 安装mysql****

* 1. 安装mysql-server
     1. sudo apt-get install mysql-server    (安装过程中，会提示输入root账户和密码，按提示输入即可，安装后默认开启server)
  2. 创建azkaban用户，并授权
  3. mysql -u root -proot
  4. mysql> create user 'azkaban' identified by 'azkaban'
  5. grant all privileges on \*.\* to 'azkaban'@ '%' with grant option;
  6. GRANT ALL PRIVILEGES ON \*.\* TO azkaban@"%" IDENTIFIED BY "azkaban" WITH GRANT OPTION;
  7. flush privileges
  8. #创建azkaban数据库

create database azkaban

* 1. 进入zakaban目录执行sql
  2. cd /opt/azkaban-3.0.0/azkaban/azkaban-sql/src/sql
  3. #执行下列命令
  4. for file in `ls \*sql`;do mysql -h 10.0.2.129 -u azkaban -pazkaban -D azkaban < $file ; done;
  5. #注：如果mysql主机和azkaban不在同一台主机，只需把 sql这个目录 scp到 mysql所在主机，续执行上条命令
  6. 远程拷贝命令：scp -r /opt/azkaban-3.0.0/azkaban/azkaban-sql/src root@10.0.2.129:/root/
  7. 在mysql中执行
  8. use azkaban;
  9. show tables;

#发现很多表证明这里没有问题

#### ****环境准备Azkaban-exec-server-3.0.0和Azkaban-web-server-3.0.0****

* 1. 解压web服务和exec服务
  2. #为了方便，我直接再/opt下创建一个新的azkaban3.0 目录 cd /opt/ mkdir azkaban-3.0
  3. #如果用root安装需要更改属主和属组 root@cdh-2-134:/opt# chown hdfs:hdfs azkaban-3.0
  4. #将编译好的/opt/azkaban-3.0.0/azkaban/build/distributions下 Azkaban Web Server和Executor Server包 解压到/opt/azkaban-3.0，和Web Server一样,记得在azkaban-web-server-3.0.0/extlib 加上mysql的jar

tar zxvf azkaban-exec-server-3.0.0.tar.gz /opt/azkaban3.0 tar zxvf azkaban-web-server-3.0.0.tar.gz /opt/azkaban3.0

* 1. 下载mysql JDBC连接包
  2. #因为azkaban web sever ,azkaban excutor server 都需要该mysql-connector-java-5.1.38-bin.jar包放在exitlib 下。解压mysql-connector-java-5.1.38.tar.gz
  3. tar zxvf mysql-connector-java-5.1.38.tar.gz
  4. #下载地址： http://dev.mysql.com/downloads/file/?id=460362
  5. #解压后 将mysql-connector-java-5.1.38-bin.jar 放到exitlib下#上述两个server运行时会报slf4j jar找不到的问题, 解决方法是下载 slf4j-api-X.Y.Z.jar和slf4j-simple-X.Y.Z.jar 到 extlib
  6. #到SLF4J官网下载：http://www.slf4j.org/download.html slf4j压缩包解压， 拷贝slf4j-api-1.7.7.jar和slf4j-simple-1.7.7.jar 到 两个 server的extlib下
  7. #首先把 slf4j-api-1.7.7.jar和slf4j-simple-1.7.7.jar 拷贝到/opt/azkaban-3.0/azkaban-web-server-3.0.0/extlib下

scp slf4j-api-1.7.21.jar slf4j-simple-1.7.21.jar /opt/azkaban-3.0/azkaban-exec-server-3.0.0/extlibcp slf4j-api-1.7.21.jar slf4j-simple-1.7.21.jar /opt/azkaban-3.0/azkaban-web-server-3.0.0/extlib

* 1. 配置Azkaban-web-server的配置文件
  2. hdfs@cdh-2-134:/opt/azkaban-3.0.0/azkaban-web-server-3.0.0/conf$ vi azkaban.properties
  3. # Azkaban Personalization Settings
  4. azkaban.name=Data Platform
  5. azkaban.label=Make Job manageable
  6. azkaban.color=#FF3601
  7. azkaban.default.servlet.path=/index
  8. web.resource.dir=web/
  9. default.timezone.id=Asia/Shanghai
  10. # Azkaban UserManager class
  11. user.manager.class=azkaban.user.XmlUserManager
  12. user.manager.xml.file=conf/azkaban-users.xml
  13. # Loader for projects
  14. executor.global.properties=conf/global.properties
  15. azkaban.project.dir=projects
  16. database.type=mysql
  17. mysql.port=3306
  18. mysql.host=10.0.2.129
  19. mysql.database=azkaban
  20. mysql.user=azkaban
  21. mysql.password=azkaban
  22. mysql.numconnections=100
  23. # Velocity dev mode
  24. velocity.dev.mode=false
  25. # Azkaban Jetty server properties.
  26. jetty.use.ssl=false
  27. jetty.maxThreads=25
  28. jetty.ssl.port=8185
  29. jetty.port=8085
  30. #jetty.keystore=/opt/azkaban-3.0/server-keystore
  31. #jetty.password=xuetangX@2014
  32. #jetty.keypassword=xuetangX@2014
  33. #jetty.truststore=/opt/azkaban-3.0/server-keystore
  34. #jetty.trustpassword=xuetangX@2014
  35. #jetty.excludeCipherSuites=
  36. ## Azkaban Executor settings
  37. executor.port=12321
  38. # mail settings
  39. mail.sender=noreply@xuetangx.com
  40. mail.host=smtp.qiye.163.com
  41. mail.user=noreply@xuetangx.com
  42. mail.password=mg4c5IUC9
  43. job.failure.email=
  44. job.success.email=
  45. lockdown.create.projects=false
  46. cache.directory=cache
  47. # JMX stats
  48. jetty.connector.stats=true

executor.connector.stats=true

* 1. 配置Azkaban-exec-server-3.0.0
  2. hdfs:/opt/azkaban-3.0/azkaban-exec-server-3.0.0/conf$ vi azkaban.properties
  3. # Azkaban
  4. default.timezone.id=Asia/Shanghai
  5. # Azkaban JobTypes Plugins
  6. azkaban.jobtype.plugin.dir=plugins/jobtypes
  7. # Loader for projects
  8. executor.global.properties=conf/global.properties
  9. azkaban.project.dir=projects
  10. database.type=mysql
  11. mysql.port=3306
  12. mysql.host=10.0.2.129
  13. mysql.database=azkaban
  14. mysql.user=azkaban
  15. mysql.password=azkaban
  16. mysql.numconnections=100
  17. # Azkaban Executor settings
  18. executor.maxThreads=50
  19. executor.port=12321
  20. executor.flow.threads=30
  21. # JMX stats
  22. jetty.connector.stats=true
  23. executor.connector.stats=true
  24. # uncomment to enable inmemory stats for azkaban
  25. #executor.metric.reports=true
  26. #executor.metric.milisecinterval.default=60000
  27. #hdfs browser
  28. azkabn.viewr.plugin.dir=plugins/viwer/hdfs
  29. # mail settings
  30. mail.sender
  31. mail.host=liulijun@xuetangx.com
  32. mail.user=liulijun@xuetangx.com
  33. mail.password=azkaban

#和web server对应 即web server就知道了这个端口

## 四、启动

### 启动azkaban-exec-server-3.0.0

1. cd /opt/azkaban-3.0/azkaban-web-server-3.0.0/

./bin/azkaban-executor-start.sh

### 启动azkaban-web-server

1. cd /opt/azkaban-3.0/azkaban-web-server-3.0.0

./bin/azkaban-web-server-start.sh

* 1. 访问：ip:8085 访问登陆界面    用户：azkaban  密码：azkaban

## 五、关闭

### 关闭azkaban-exec-server-3.0.0

1. cd /opt/azkaban-3.0/azkaban-exec-server-3.0.0

./bin/azkaban-executor-shutdown.sh

### 关闭azkaban-web-server-3.0.0

cd /opt/azkaban-3.0/azkaban-web-server-3.0.0/

./bin/azkaban-web-server-shutdown.sh

## ****5.elasticsearch集群搭建****

一.背景：

      当前搜索使用的是ES1.0，版本陈旧，与新版本5.X的特性有差异，现在的文档也多是新的版本，所以决定升级。

二.机器组织结构：

     新的ES集群由五台10.0.2.X网段的机器组成(配置还没有约定)，每台的环境都是一样的。

三.以其中一台的安装为例说明安装步骤，其他相同。

1.登入一台要安装的服务器

ssh   [root@xxx.xxx.xxx.xxx](mailto:root@xxx.xxx.xxx.xxx)

apt-get install curl

apt-get install htop     # 安装linux工具

2.创建搜索系统的账户（ES不运行在root上）

sudo adduser xtes      # 输入密码 xuetangX,然后一路回车(账户最好都约定成这个，不然后面的有些命令路径会不对，那就要自行调整路径名了)

sudo vim  /etc/sudoers   添加如下一行以使xtes获得sudo权限(退出保存输入wq!) (xtes    ALL=(ALL)NOPASSWD:ALL)

xtes    ALL=(ALL) ALL



现在还是在root用户下

sudo vim /etc/security/limits.conf   在末尾添加下面两行“打开内存限制”

xtes soft memlock unlimited  
xtes hard memlock unlimited

\* hard nofile 131074  
\* soft nofile 131074

vim /etc/profile

末尾添加一行

ulimit -l unlimited  
ulimit -n 131074

退出登录，重新登入

ulimit -n      看看返回值是不是你修改的结果确实修改是否生效

--------------------------------------------------------

su xtes      #切换到新创建的账户（下面的所有操作都在这个新账户中进行的）

cd ~

mkdir es   （在主目录下创建）

cd es

pwd   看下是否在这个目录   /home/xtes/es

scp  [root@192.168.9.63](mailto:root@192.168.9.63):/files/es5all.tar.bz2 ./      密码是 xuetangX，这是打包好的各种安装包，避免自己分别下载麻烦

tar -jxvf /files/es5all.tar.bz2  ./          解压到当前的文件夹

cd /home/xtes/es/es5all      进入解压文件目录

tar -zxvf jdk-8u45-linux-x64.tar.gz   解压JDK安装包到/usr/java

rm jdk-8u45-linux-x64.tar.gz    删除JDK安装包

-------------配置JDK------

cd ~

vim .bashrc

文件末尾追加下面两行

export JAVA\_HOME=/home/xtes/es/es5all/jdk1.8.0\_131

export PATH=$JAVA\_HOME/bin:$PATH

退出  .bashrc

source .bashrc

java -version     看返回结果正常与否

javac -version   看返回结果正常与否

-------------------配置es------------------

cd /

sudo mkdir es\_path

sudo chmod 777 es\_path/

cd /es\_path

mkdir data logs    (在/es\_path 下创建这两个文件夹)

----------------------------

该配置需要在物理机器修改（一个进程能拥有的最多的内存区域）

使用下面的方法临时使其生效

$ sudo sysctl -w vm.max\_map\_count=655360

或者修改 /etc/sysctl.conf 文件永久生效，添加

vm.max\_map\_count=655360

设置后，可以使用

sysctl -a | grep "vm.max\_map\_count"  查看是否更改生效

--------------------------

cd  /home/xtes/es/es5all/elasticsearch-5.1.1/config

vim elasticsearch.yml 对应修改如下配置：这里面的配置文件已经是配置好的，但是机器之间有少量差异，只需要修改如下字段(ES1中的这个文件有很多配置，但是都不再适用ES5，要么是配置位置变化，要么是取消了)

1. node.name: "ip-2-63" #2是对应的网段，63是本机器IP的最后一个字段（希望名字做一下约定）
2. network.host: 192.168.9.63         #改成本机器的IP
3. discovery.zen.ping.unicast.hosts: ["192.168.9.63","xxx.xxx.xxx.xx","xxx.xxx.xxx.xx"] #将5台机器的IP都加入这个数组中，不要求顺序
4. discovery.zen.minimum\_master\_nodes: 3    #（机器数/2）+1
5. gateway.recover\_after\_nodes: 3           #（机器数/2）+1

这个修改一定要注意，每个字段，如node.name: "ip-2-63"   冒号后面有一个空格，千万不忘紧跟：，如果你的vim打开了颜色设置，就会看到一旦紧跟:就会字段颜色变化。

-----------安装ES插件-------------

cd  /home/xtes/es/es5all/elasticsearch-5.1.1

 bin/elasticsearch-plugin install [file:///home/xtes/es/es5all/x-pack-5.1.1.zip](http://file///home/xtes/es/es5all/x-pack-5.1.1.zip)       (会让你选择，要输入'y'，否则就安装失败了)

vim bin/elasticsearch  中第46行添加如下一句话(就是在配置文件注释之后的第一行)

ES\_JAVA\_OPTS="-Xms32g -Xmx32g"    # 这是设置ES的内存数量，（我们约定设置成32G）

退出vim

setsid ./bin/elasticsearch  后台启动ES（你会看到报错，只要ES不退出就行，报错是因为集群只有出现3个节点后，才会可用，当把5台机器都启动完了就可以了，如果ES退出，请查看ES退出原因）

----------激活ES许可--------------------------------

cd  /home/xtes/es/es5all         #激活许可证（请替换IP为本机器的IP）

curl -XPUT -u elastic:changeme  'http://192.168.9.63:9200/\_xpack/license?acknowledge=true' -H "Content-Type: application/json" -d @license.json

curl -XGET -u elastic:changeme 'http://192.168.9.63:9200/\_xpack/license'      #查看许可证的有效日期（请替换IP为本机器的IP）

如果显示正常到2018年，那就完成了一台机器安装了。

---------------------------------------------------------

然后安装剩下的4台机器

---------------测试每台节点的返回情况------------------------

curl ' http://192.168.9.63:9200/?pretty'      输入这个命令看ES是否有返回（ 请替换IP为测试机器的IP）

curl "http://192.168.9.63:9200/\_analyze?analyzer=ik\_max\_word&pretty=true&text=java%E9%AB%98%E7%BA%A7%E5%B7%A5%E7%A8%8B%E5%B8%88"

测试ik分词器，看是否正常（请替换IP为测试机器的IP）

curl "http://192.168.9.63:9200/\_analyze?analyzer=pinyin&pretty=true&text=java%E9%AB%98%E7%BA%A7%E5%B7%A5%E7%A8%8B%E5%B8%88"

测试pinyin分词器，看是否正常（请替换IP为测试机器的IP）

如果有测试返回失败，就是这个节点有问题，请返回前面核对是不是哪里操作问题

--------------------最后的检查------------------------------------

curl -XGET 'http://10.0.0.163:9200/\_cluster/health'     把IP换成前面安装的任何一台机器的IP都行

返回信息的格式如下

{  
    "cluster\_name":"elasticsearch",  
    "status":"green",  
    "timed\_out":false,  
    "number\_of\_nodes":3,     # 这个数一定要是你的集群机器数，如果这个数少，那就是有几台不能正常工作，请回头排查。这个务必要是5（集群节点数）  
    "number\_of\_data\_nodes":3,  
    "active\_primary\_shards":85,  
    "active\_shards":170,  
    "relocating\_shards":0,  
    "initializing\_shards":0,  
    "unassigned\_shards":0  
}

如果检查节点数完整，那就结束啦0\_0

## ****6.tap tapapi course-service服务****

## 一.引言

### 1.系统介绍

教学分析平台（Teaching Assistant Platform ）通过统计学生在学习过程中产生的数据进行分析，帮助教师了解教学开展情况及效果，教师可以细粒度掌握学生学习进度和效果，了解每位学生在每个章节中的每个单元学习信息，教学分析平台主要包含8个分析面板：课程热度、了解学员、学习活跃、学习习惯、得分分布、学习进度、互动模式、课程健康度。

### 2.术语约定

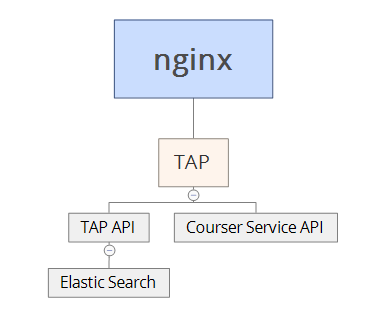
TAP：Teaching Assistant Platform 教学分析平台

TAP API：TAP平台依赖的API接口服务，提供学习情况的各类统计信息

Course Service API：课程信息API接口服务

## 二.系统部署

### 1.系统逻辑部署图



### 2.基本软件依赖

|  |  |
| --- | --- |
| **软件** | **版本** |
| Python | 2.7 |
| Nginx | 1.1.19 |
| JDK | 1.8.0(8u45) |
| Maven | 3.2.3 |

### 3.服务部署

#### 3.1 部署TAP API

##### a.检出代码

检出代码（master分支）到本地目录/opt/app/tapapi

##### b.安装依赖

pip install -r requirements.txt

##### c.修改配置

cp settings.yaml.simple settings.yaml

打开settings.yaml，修改其中的es\_cluster配置项为你的es集群。

##### d.Supervisor启动

修改你的supervisor配置，举例如下：

[group:tapapi]

programs=tapapi\_8000

[program:tapapi\_8000]

command=python /opt/app/tapapi/server.py --port=8000 --debug=false

stdout\_logfile=/var/log/supervisor/%(program\_name)-stdout.log

stderr\_logfile=/var/log/supervisor/%(program\_name)-stderr.log

autostart=true

autorestart=true

其中的端口号和log存放位置可由您指定。

#### 3.2 部署Course Service API

##### a.检出代码

检出代码（master分支）到本地目录/opt/app/course-service

##### b.项目打包

cd /opt/course-service

mvn -DskipTests package

mvn dependency:copy-dependencies

##### c.项目部署

将项目目录下target中的 course-service-{version}.jar和dependency目录拷贝到部署目录, 比如 /opt/app/course-service/run

在项目目录下创建脚本 course-service.sh, 内容如下

#!/bin/sh

D=/opt/app/course-service/run

CP=.

for jar in `find $D -name "\*.jar"`;

do

CP=$CP:$jar

done

OPTS="-cp $CP"

java $OPTS com.xuetangx.services.server.ServiceStartup $D/config.yml

如果部署目录不是 /opt/course-service, 修改脚本第二行为相应的目录

##### d.配置文件

在部署目录下建立配置文件 config.yml, 内容如下

port: 8080

mysql:

url: jdbc:mysql://192.168.9.104/edxapp

user: root

pass: ""

mongo:

host: 192.168.9.104

各项说明如下

port: 监听的端口

mysql: 主站使用的mysql, 配置为slave的地址即可

需要修改mysql下的url的ip地址, user和pass

mongo: 主站使用的mongo的地址

##### e.启动和重启

启动, 直接执行之前创建的course-service.sh

nohup sh course-service.sh &

重启, 先杀死进程, 然后再启动

# kill process

ps -ef|grep com.xuetangx.services.server.ServiceStartup | grep -v grep | awk '{print $2}' | xargs kill -9

# start

nohup sh course-service.sh &

#### 3.3 部署TAP

##### a.检出代码

检出代码到本地目录/opt/app/tap

##### b.安装依赖

pip install -r requirements.txt

##### c.修改配置

cp settings.yaml.simple settings.yaml

打开settings.yaml，需要修改的配置项有：

|  |  |  |
| --- | --- | --- |
| **配置** | **举例** | **修改说明** |
| course\_service\_api | 127.0.0.1:8001 | Course Service服务地址 |
| data\_service\_api | 127.0.0.1:8002 | TAP API服务地址 |
| statistics\_service | 127.0.0.1:8003 | memcache |
| kafka | 192.168.9.110:9092 | kafka地址 |

d.Supervisor启动

修改你的supervisor配置，举例如下：

[group:tap]

programs=tap\_8002

[program:tap\_8002]

command=python /opt/app/tap/app/server.py --port=8002 --debug=false

stdout\_logfile=/var/log/supervisor/%(program\_name)-stdout.log

stderr\_logfile=/var/log/supervisor/%(program\_name)-stderr.log

autostart=true

autorestart=true

其中的端口号和log存放位置可由您指定。

#### 3.4部署Nginx

在/etc/nginx/sites-enabled/中添加TAP和TAP API相应的配置文件：

**TAP**

upstream tap {

server 127.0.0.1:8001;

server 127.0.0.1:8002;

}

server {

listen 8000;

rewrite ^(.\*)/favicon.ico$ /static/images/favicon.ico last;

location ~ ^/static/(?P<file>.\*) {

root /opt/app/tap/app/static;

try\_files /$file =404;

}

location / {

auth\_basic off;

proxy\_pass http://tap$request\_uri;

proxy\_redirect off;

proxy\_set\_header Host $host;

proxy\_set\_header X-Real-IP $remote\_addr;

proxy\_set\_header X-Forwarded-For $proxy\_add\_x\_forwarded\_for;

client\_max\_body\_size 500m;

client\_body\_buffer\_size 128k;

proxy\_connect\_timeout 180;

proxy\_send\_timeout 180;

proxy\_read\_timeout 180;

}

}

**TAP API**

upstream tapapi {

server 127.0.0.1:9001;

server 127.0.0.1:9002;

}

server {

listen 9000;

location / {

proxy\_pass http://tapapi;

proxy\_redirect off;

proxy\_set\_header Host $host;

proxy\_set\_header X-Real-IP $remote\_addr;

proxy\_set\_header X-Forwarded-For $proxy\_add\_x\_forwarded\_for;

client\_max\_body\_size 500m;

client\_body\_buffer\_size 128k;

proxy\_connect\_timeout 180;

proxy\_send\_timeout 180;

proxy\_read\_timeout 180;

}

}

刷新Nginx配置

nginx -s reload