

Table of Contents

声明	1.1
平台概述	1.2
平台简介	1.2.1
基本架构	1.2.2
最低需求	1.2.3
建议硬件配置	1.2.4
集群环境准备	1.3
规划集群主机	1.3.1
创建Yum本地源	1.3.2
创建DNS域名解析服务	1.3.3
创建NTP时钟同步服务	1.3.4
初始化集群部署环境	1.3.5
创建Postgresql数据库	1.3.6
创建Yum缓存代理服务	1.3.7
管理中心安装	1.4
启动Server	1.4.1
创建PG实例	1.4.2
配置Server	1.4.3
启动Server	1.4.4
集群安装和配置	1.5
Web登录	1.5.1
安装向导	1.5.2
集群命名	1.5.3
选择版本	1.5.4
选项配置	1.5.5
主机检查	1.5.6
选择服务	1.5.7

调整服务主组件	1.5.8
调整服务从主键和客户端	1.5.9
调整服务参数	1.5.10
保存配置报告	1.5.11
安装过程	1.5.12
安装完成	1.5.13

版权声明

本产品包装内之物件所有权归北京达沃时代科技股份有限公司所有。

达沃商标归北京达沃时代科技股份有限公司所有。

本手册中提及之商标，均属其合法注册公司所有。

本产品使用手册保留变更内容而不另行通知之权利，未经北京达沃时代科技股份有限公司许可，不得自行转载、复制或散布。

责任声明

本手册所提及之內容仅代表北京达沃时代科技股份有限公司，对于非正确使用本产品所导致的任何损坏或损失，北京达沃时代科技股份有限公司不承担任何责任。对于使用本产品可能导致的第三方的损失或索赔，北京达沃时代科技股份有限公司不承担任何责任。

平台概述

本文档主要介绍达沃大数据平台的整个安装流程，包括平台介绍说明、安装前准备工作、详细的安装过程和注意事项等内容。

- 平台简介
- 基本架构
- 最低需求
- 建议硬件配置

平台简介

达沃大数据平台是基于Web，专注于Hadoop分布式集群的配置管理工具，它支持Hadoop集群的部署、管理和监控，目前已支持绝大多数Hadoop组件，包括HDFS、MapReduce、Hive、Pig、Hbase、Zookeper、Sqoop和Hcatalog等。

在用户体验上，达沃大数据平台提供了简单的操作模式、多维的状态展现以及全面的性能指标。与传统的同类产品相比较，它最突出的是以下几个特点：

- 通过一步一步的安装向导简化了集群部署过程。
- 预先配置好关键的运维指标 Metrics，可以直接查看 Hadoop Core 及相关服务（如HBase、Hive和HCatalog）是否健康。
- 支持作业与任务执行的可视化与分析，能够更好地查看依赖和性能。
- 通过一个完整的 RESTful API 把监控信息暴露出来，集成了现有的运维工具。
- 用户界面非常直观，用户可以轻松有效地查看信息并控制集群。

基本架构

组成部分

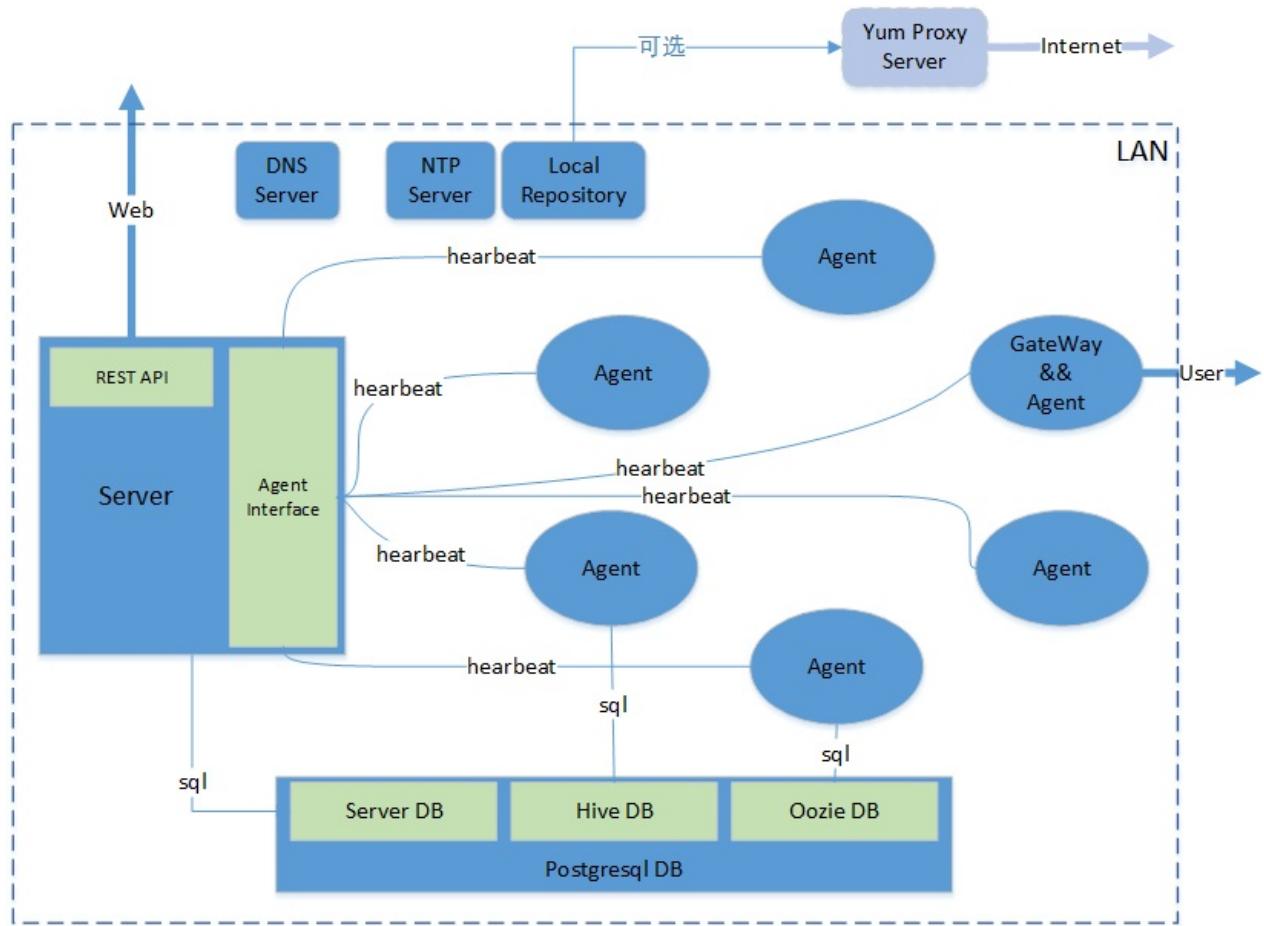
达沃大数据平台基本架构采用的是Server/Client的模式，其核心由两部分组成：若干个Agent和一个Server。

- **Server:** 接收 Agent Interface 的状态上报请求，完成集中式管理和监控逻辑。
- **Agent:** 负责所在节点主机的状态采集、维护并利用 Agent Interface 来完成上报。

另外还需要一些必要的系统及第三方服务来保证平台的正常安装和运行。

- **DNS Server:** 为集群内的各个主机提供域名解析服务，实现主机域名和IP地址的相互映射。
- **NTP Server:** 为集群内的各个主机提供系统时间同步服务，主要是Server及Agent之间的时间同步。
- **Local Repository:** 包含了集群安装过程中需必须的所有rpm包，方便离线安装并提高安装速度。
- **Postgresql DB:** 为Server和Hadoop组件分别创建数据库实例，集中提供数据库服务，方便后期维护和管理。

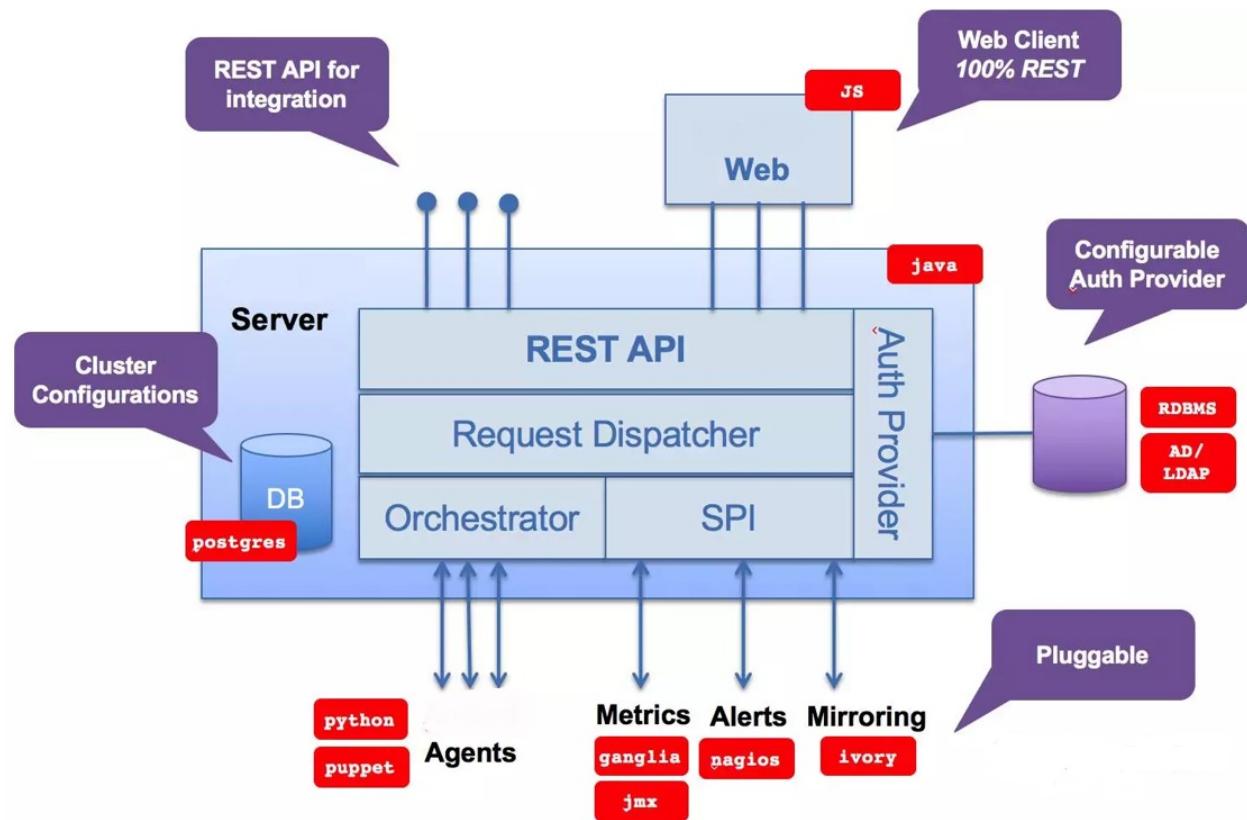
注意：Postgresql可替换为Oracle、 MySql等其他数据库。



技术实现

在技术实现上，达沃大数据平台广泛地依赖其它已经成熟的开源工具，达到集群安装的步骤化、集群状态的可视化、集群资源管理的一键化、配置变更历史的版本化。

- 使用 **Ganglia** 收集度量指标。
- 用 **Nagios** 支持系统报警，当需要引起管理员的关注时（比如，节点停机或磁盘剩余空间不足等问题），系统将向其发送邮件。
- 支持安装基于 **Kerberos** 的安全认证组件，以此实现了对**Hadoop**安全的支持。
- 提供了基于角色的用户认证、授权和审计功能，并为用户管理集成了**LDAP**和**Active Directory**。



正是这些自动化手段的运用，极大的降低了集群运维的难度，减少了集群运维的工作量。

最低需求

操作系统版本

操作系统	发行版本
CentOS (64-bit)	CentOS v7.x
	CentOS v6.x
Debian	Debian v7.x
Red Hat (64-bit)	RHEL 7.0, 7.1, 7.2
	RHEL 6.1, 6.2, 6.3, 6.4, 6.5, 6.6, 6.7, 6.8
SUSE (64-bit)	(SLES) Enterprise Linux 12, SP2 for Teradata
	(SLES) Enterprise Linux 12, SP1
	(SLES) Enterprise Linux 11, SP4 (HDP 2.2 and later)
	(SLES) Enterprise Linux 11, SP3 for Teradata (HDP 2.2 and later)
	(SLES) Enterprise Linux 11, SP1 (HDP 2.2)
Ubuntu (64-bit)	Ubuntu 16.04 (Xenial)
	Ubuntu 14.04 (Trusty)

Web浏览器版本

操作系统	浏览器类型及版本
Linux	Chrome 56.0.2924.87, 57.0.2987
	Firefox 51, 52
Mac OS X	Chrome 56.0.2924.87, 57.0.2987
	Firefox 51, 52
	Safari 10.0.1, 10.0.3
Windows	Chrome 56.0.2924.87, 57.0.2987
	Edge 38
	Firefox 51.0.1, 52.0
	Internet Explorer 10, 11

数据库版本

组件	数据库	描述说明
Ambari	PostgreSQL 9.1.13+, 9.3, 9.4*	
	MariaDB 10*	
	MySQL 5.6**	
	Oracle 11gr2	
	Oracle 12c**	
Druid	PostgreSQL 9.1.13+, 9.3, 9.4*	
	MariaDB 10*	
	MySQL 5.6**	
Hive	PostgreSQL 9.1.13+, 9.3, 9.4*	
	MariaDB 10*	
	MySQL 5.6**	
	Oracle 11gr2	
	Oracle 12c**	
Oozie	PostgreSQL 9.1.13+, 9.3, 9.4*	
	MariaDB 10*	
	MySQL 5.6**	
	Oracle 11gr2	
	Oracle 12c**	
Ranger	PostgreSQL 9.1.13+, 9.3, 9.4*	
	MariaDB 10*	
	MySQL 5.6**	
	Oracle 11gr2	
	Oracle 12c**	

集群节点数量与存储容量

一般情况下，随着集群中节点主机数量的增加，平台中定义的管理主机所需要的内存和磁盘空间应该不断增大。可参考下面列出的集群规模与存储之间的关系来确定管理主机的恰当配置。

最低需求

主机数量	内存	磁盘空间
1	1024 MB	10 GB
10	1024 MB	20 GB
50	2048 MB	50 GB
100	4096 MB	100 GB
300	4096 MB	100 GB
500	8096 MB	200 GB
1000	12288 MB	200 GB
2000	16384 MB	500 GB

建议硬件配置

主机类型	数量	CPU	内存	网卡	硬盘
管理服务主机	1	CPU e5-2620及以上	16G	千兆网口/万兆网口	独立操作系统硬盘
数据库服务主机	1	CPU e5-2620及以上	16G	千兆网口/万兆网口	独立操作系统硬盘，独立数据硬盘
DNS服务主机	1	CPU e5-2620及以上	8G	千兆网口/万兆网口	独立操作系统硬盘
本地仓库服务主机	1	CPU e5-2620及以上	8G	千兆网口/万兆网口	独立操作系统硬盘，独立数据硬盘
组件节点主机	1	CPU e5-2620及以上	8G	千兆网口/万兆网口	独立操作系统硬盘，独立数据硬盘

集群环境准备

- 规划集群主机
- 创建Yum本地源
- 创建DNS域名解析服务
- 创建NTP时钟同步服务
- 初始化集群部署环境
- 创建Postgresql数据库
- 创建Yum缓存代理服务

规划集群主机

在开始部署大数据平台之前，我们需要对集群主机的角色和功能进行总体的规划，比如哪些是NameNode节点，哪些是DataNode节点，哪些节点需要安装Client，用户通过怎样的方式访问这些客户端等等。下面，让我们看看在集群规划过程中的几个要点。

- 定义一个GateWay节点，使其成为用户操作的唯一接入口。

在GateWay上安装相应的Client，要求所有的用户都通过连接到GateWay的方式来操作集群，不允许直接访问集群内的其他主机。

- 组件使用统一且独立的数据库服务器，便于集中管理和维护。

有数关系数据库存储需求的组件都应该尽量使用同一种类的数据库，并在集群中定义专门的数据库服务器为这些组件来提供数据库实例服务，这样将大大的减少DBA的后期运维难度和运维成本。

- 遵循 FQDN 命名规则，让主机名变得有意义。

定义主机名时，应该包含尽可能多的机器信息，比如操作系统、应用类型/角色、位置、环境、实例等。我们需要根据实际情况（那种信息是我们组织中最重要的部分）来约定一个标准，然后保证集群中所有的主机都遵循标准来命名。

创建YUM本地源

最常用的创建和加载自定义yum本地源的方法有以下三种：

- 下载已整理好的自定义源ISO镜像文件，直接挂载到本地文件系统，在本机上通过 `file:` 的方式访问。
- 下载已整理好的自定义源TAR压缩文件，解压后创建Ftp文件服务器，在局域网通过 `ftp:` 的方式访问。
- 下载已整理好的自定义源TAR压缩文件，解压后创建Web文件服务器，在局域网通过 `http:` 的方式访问。

我们选择安装和配置都比较简单 `Apache httpd` 来实现基于http的Web服务器。

获取自定义源压缩包

我们对大数据平台所包含的Server、Agent、Hadoop服务组件，以及所依赖的第三方软件rpm包和其他资源都进行了汇总和整理，制作成了本地的自定义源后重新打包并进行版本更新和维护。

- 下载

从公司内网的Ftp服务器上下载 `bigdata-2.5.0.3-centos7.tar.gz`，其地址为 <ftp://192.168.16.100/bigdata-2.5.0.3-centos7.tar.gz>。

- 解压

将自定义源解压至 `/home/repo` 目录，通过`tree`命令查看其文件系统结构如下：

```
[root@repo repo]# tree -L 2  
.  
|-- ambari  #Server相关的rpm包  
|   '-- centos7  
|-- common  #依赖的第三方软件rpm包  
|   '-- centos7  
|-- component  #核心组件rpm包  
|   |-- HDF  
|   |-- HDP    #Hadoop组件  
|   '-- HDP-UTILS-1.1.0.21  #Hadoop组件工具包  
`-- resource  #Jdk、jdbc等资源  
    |-- ambari.repo  
    |-- CentOS7-Base-aliyun.repo  
    |-- jce_policy-8.zip  
    |-- jdk-8u112-linux-x64.tar.gz  
    |-- jdk-8u77-linux-x64.tar.gz  
    |-- postgresql-jdbc.jar  
    '-- repodata  
  
10 directories, 6 files
```

配置本机的**YUM**源

- 有永久或临时外网访问权限

为了取得更快的安装速度，将默认的国外源更新为国内的阿里源。首先备份备份默认源。

```
mv /etc/yum.repos.d /etc/yum.repos.d.bak
```

然后下载对应版本的**repo**文件，放入`/etc/yum.repos.d/`。

```
mkdir /etc/yum.repos.d  
wget -O /etc/yum.repos.d/CentOS7-Base-aliyun.repo http://mirrors  
.aliyun.com/repo/Centos-7.repo
```

- 没有外网访问权限

直接使用之前解压的/home/repo/common目录作为源，通过file://的方式来配置repo文件。同样首先备份下默认源。

```
mv /etc/yum.repos.d /etc/yum.repos.d.bak
```

然后编辑生成使用本地文件系统的repo文件，放入/etc/yum.repos.d/。

```
cat << eof > /etc/yum.repos.d/localfile.repo
[localfile]
name=local file system repository
baseurl=file:///home/repo/common
gpgcheck=0
enabled=1
priority=1
eof
```

- 更新yum缓存的数据。

```
yum clean all
yum makecache
```

安装YUM工具和插件

为了方便后期扩充自己所需的软件包，以及在涉及到多个源之后所需要的优先级管理，接下来安装yum周边的工具和插件。

```
#createrepo命令创建版本库所需要的基础rpm包
yum install -y yum-utils createrepo

#yum插件，安装后可根据priority值来设置优先级
yum install -y yum-plugin-priorities
```

配置Apache httpd服务

通过配置httpd虚拟主机的方式实现一个简单的Web文件服务器，以此为基础来架设集群内的本地源。

- 首先安装Apache httpd服务。

```
yum install -y httpd
```

安装完成后进入httpd生成的主配置目录/etc/httpd，它的文件结构及意义如下。

```
[root@repo httpd]# tree
#主配置目录，通过修改其中的httpd.conf文件来完成配置
|-- conf
|   |-- httpd.conf
|   `-- magic
#扩展配置目录，在主配置文件httpd.conf中，
#通过Include conf.d/*的方式来载入该目录中的配置
|-- conf.d
|   |-- autoindex.conf
|   |-- README
|   |-- userdir.conf
|   `-- welcome.conf
#httpd核心模块目录
|-- conf.modules.d
|   |-- 00-base.conf
|   |-- 00-dav.conf
|   |-- 00-lua.conf
|   |-- 00-mpm.conf
|   |-- 00-proxy.conf
|   |-- 00-systemd.conf
|   `-- 01-cgi.conf
|-- logs -> ../../var/log/httpd
|-- modules -> ../../usr/lib64/httpd/modules
`-- run -> /run/httpd

6 directories, 13 files
```

- 在80端口配置虚拟主机用于通过http方式访问自定义源 /home/repo 目录，将新建虚拟主机配置文件放在httpd扩展目录 /etc/httpd/conf.d 中。

```
cat << eof > /etc/httpd/conf.d/local_repo.conf
<VirtualHost *:80>                      #虚拟主机工作在80端口
    DocumentRoot "/home/repo"            #我们定义的根目录
    <Directory "/home/repo">
        #访问控制组合指令
        Options Indexes FollowSymLinks
        AllowOverride None
        Require all granted
    </Directory>
</VirtualHost>
eof
```

Options指令含义如下：

Indexes：在无默认主页面又无欢迎页时，将所有资源以列表形式呈现给用户

FollowSymLinks：允许跟踪符号链接文件

AllowOverride指令含义：

表示支持在每个页面目录下创建 .htaccess 文件，来定义对此目录中资源的访问控制，设置为None时表示忽略.htaccess 文件

Require指令含义：

all：包含所有主机
ip：包含指定的地址
not ip：排除指定的地址
granted：表示允许访问
denied：表示拒绝访问

- 将新创建虚拟主机 conf.d/local_repo.conf 添加至主配置 httpd.conf 末尾。

```
[root@repo httpd]# vi conf/httpd.conf

#
# ServerName gives the name and port that the server uses to identify itself.
# This can often be determined automatically, but we recommend you specify
# it explicitly to prevent problems during startup.
#
# If your host doesn't have a registered DNS name, enter its IP
# address here.
#
ServerName repo.bigdata.wh.com          #设置为当前主机的FQDN

#EnableMMAP off
EnableSendfile on

# Supplemental configuration
#
# Load config files in the "/etc/httpd/conf.d" directory, if any
.

#IncludeOptional conf.d/*.conf           #注释掉默认的apache欢迎页面
IncludeOptional conf.d/local_repo.conf  #在文件末尾包含新增的虚拟主机
扩展配置文件
IncludeOptional conf.d/autoindex.conf   #autoindex.conf中定义了显示文件列表时的系统图标资源
```

- 通过httpd自带的命令来检测当前配置是否有错误。

```
[root@repo httpd]# httpd -S
VirtualHost configuration:
*:80                  repo.bigdata.wh.com (/etc/httpd/conf.d/local_repo.conf:1)
ServerRoot: "/etc/httpd"
Main DocumentRoot: "/var/www/html"
Main ErrorLog: "/etc/httpd/logs/error_log"
Mutex proxy-balancer-shm: using_defaults
Mutex rewrite-map: using_defaults
Mutex authdigest-client: using_defaults
Mutex proxy: using_defaults
Mutex authn-socache: using_defaults
Mutex default: dir="/run/httpd/" mechanism=default
Mutex mpm-accept: using_defaults
Mutex authdigest-opaque: using_defaults
PidFile: "/run/httpd/httpd.pid"
Define: DUMP_VHOSTS
Define: DUMP_RUN_CFG
User: name="apache" id=48
Group: name="apache" id=48
```

- 启动 `httpd.service` 服务，并设置跟随系统自启动。

```
systemctl restart httpd.service
systemctl enable httpd.service
```

- 关闭防火墙 `firewalld.service`。

```
systemctl stop firewalld.service
systemctl disable firewalld.service
```

- 关闭 Selinux

```
#临时关闭Selinux  
setenforce 0  
  
#开机不启动Selinux  
sed -i 's/SELINUX=.*/SELINUX=disabled/' /etc/selinux/config
```

- 通过Web浏览器访问自定义源80端口的文件目录。



The screenshot shows a web browser window with the URL `192.168.36.247` in the address bar. The page displays the contents of a directory, specifically listing four subfolders: `ambari/`, `common/`, `component/`, and `resource/`. Each entry includes a small folder icon, the folder name, the last modified date, and a size indicator of '-'.

Name	Last modified	Size	Description
ambari/	2017-05-09 10:17	-	
common/	2017-05-22 11:23	-	
component/	2017-05-22 11:30	-	
resource/	2017-05-17 11:22	-	

- 查看 `/home/repo/resource` 目录下的本地源repo文件，检查其默认的 `baseurl` 是否与新建的本地源路径相同。

```
[root@repo repo]# cat resource/ambari-2.5.0.3.repo
[ambari-2.5.0.3]
name=ambari local repository
baseurl=http://repo.bigdata.wh.com/ambari/centos7/2.x/updates/2.
5.0.3/
gpgcheck=0
enabled=1
priority=1
proxy=_none_

[common]
name=common local repository
baseurl=http://repo.bigdata.wh.com/common/centos7/
gpgcheck=0
enabled=1
priority=1
proxy=_none_
```

以上步骤全部完成之后，集群所需要的自定义本地源就创建成功了。

创建DNS域名解析服务

大数据平台严重依赖于DNS，在正常的操作期间会执行许多次DNS查找，我们必须为集群内的所有主机配置正向和反向DNS，完成主机名（Fully Qualified Domain Name）到IP地址的映射。要达到这个目的，可以通过编辑集群中所有主机上的 `/etc/hosts` 文件或者创建一个DNS服务器来解决，这里我们来详细描述利用bind9来创建DNS服务器，配置集群内的 `bigdata.wh.com` 子域，以及结果测试的过程。

- 系统安装后配置hostname

```
hostnamectl set-hostname dns.bigdata.wh.com #设置主机名
```

- 编辑 `/etc/hosts` 文件

首先编辑hosts文件添加本地源的IP地址和主机名之间的映射关系。

```
[root@dns ~]# cat /etc/hosts
127.0.0.1 localhost localhost.localdomain localhost4 localhost
4.localdomain4
::1 localhost localhost.localdomain localhost6 localhost
6.localdomain6

192.168.36.247 repo.bigdata.wh.com repo
```

- 下载本地源repo文件，更新yum缓存

```
wget -O /etc/yum.repos.d/ambari.repo http://repo.bigdata.wh.com/
resource/ambari-2.5.0.3.repo
yum clean all
yum repolist
```

- 安装bind及相关工具

```
yum install -y bind          #提供DNS服务, 程序名named  
yum install -y bind-utils    #测试域名的周边工具, 如dig、host、nslookup
```

bind安装完成后，其包含的配置文件和区域文件如下。

```
[root@dns etc]# rpm -ql bind
/etc/logrotate.d/named
/etc/named
/etc/named.conf                                #bind主配置文件
/etc/named.iscdlv.key
/etc/named.rfc1912.zones                      #区域(zone)配置文件
/etc/named.root.key
/etc/rndc.conf                                  #rndc配置文件
/etc/rndc.key
/etc/rwtab.d/named
/etc/sysconfig/named
/run/named
/usr/lib/systemd/system/named-setup-rndc.service #服务脚本文件
/usr/lib/systemd/system/named.service
/usr/sbin/ddns-confgen                         #生成rndc密钥
/usr/sbin/named
/usr/sbin/named-checkconf                      #检测/etc/named.conf文件语法
/usr/sbin/named-checkzone                      #检测zone和对应zone文件的语法
/usr/sbin/named-compilezone
/usr/sbin/named-journalprint
/usr/sbin/nsec3hash
/usr/sbin/rndc                                 #远程dns管理工具
/usr/sbin/rndc-confgen                         #生成rndc密钥
/var/log/named.log
/var/named
/var/named/data
/var/named/dynamic
/var/named/named.ca                            #根区域文件
/var/named/named.empty
/var/named/named.localhost                     #默认的本地主机解析库
/var/named/named.loopback                     #添加区域后新建一个该区域的解析库文件
/var/named/slaves                             #从文件夹
```

- 修改主配置文件 /etc/named.conf

```
[root@dns named]# cat /etc/named.conf
options {
    listen-on port 53 { 127.0.0.1;192.168.36.149; };      #添加本机内
网IP
    listen-on-v6 port 53 { ::1; };
    directory      "/var/named";
    dump-file      "/var/named/data/cache_dump.db";
    statistics-file "/var/named/data/named_stats.txt";
    memstatistics-file "/var/named/data/named_mem_stats.txt";
    allow-query     { localhost; any; };      #添加允许查询的主机地址，
以分号分隔
                                                #可以是某个主机:192.168
.36.x
                                                #或某个网段:192.168.36.
0/255
                                                #或所有主机:any
    allow-transfer  {};  #允许区域传送主机，白名单
    allow-recursion {}; #允许递归的主机
    allow-update   {};   #允许更新区域数据库中的内容
};

logging {
    channel default_debug {
        file "data/named.run";
        severity dynamic;
    };
};

zone "." IN {
    type hint;
    file "named.ca";
};

include "/etc/named.rfc1912.zones";
include "/etc/named.root.key";
```

- 追加区域 `bigdata.wh.com` 的正向解析定义。

```
cat << eof >> /etc/named.rfc1912.zones
zone "bigdata.wh.com" IN {
    #设置类型为master
    type master;

    #解析库文件名称为bigdata.wh.com.zone，默认的存储目录为/var/named/
    file "bigdata.wh.com.zone";
    allow-update { none; };

};

eof
```

- 创建区域的正向解析库文件。

```
cat << 'eof' > /var/named/bigdata.wh.com.zone
$TTL 600 #定义全局默认超时时间
$ORIGIN bigdata.wh.com. #定义域名后缀
@    IN  SOA     dns.bigdata.wh.com. admin.bigdata.wh.com. (
                20170510 #序列号
                1H        #刷新时
                5M        #重试时间
                1W        #超时时间
                10M )     #否定答案缓存TTL值
                IN      NS      dns          #定义区域内的一台nameserver
dns    IN      A       192.168.36.149 #dns这台nameserver所对应的I
P
admin  IN      CNAME   dns          #admin是dns的别名，admin.bi
gdata.wh.com.将解析到dns.bigdata.wh.com.
*      IN      A       192.168.30.1   #泛域名解析，以上都不是的解析
到192.168.30.1
eof
```

- 检测配置及启动DNS服务。

```
#检查主配置文件语法  
named-checkconf  
  
#检查区域所对应的解析库文件  
named-checkzone "bigdata.wh.com" /var/named/bigdata.wh.com.zone  
systemctl enable named.service  
systemctl start named.service
```

- 测试域名解析成IP是否正常，如果遇到错误，可以观察日志文件 `/var/log/message` 的内容来分析问题。

```
[root@dns named]# host -t NS bigdata.wh.com 192.168.36.149  
Using domain server:  
Name: 192.168.36.149  
Address: 192.168.36.149#53  
Aliases:  
  
bigdata.wh.com name server dns.bigdata.wh.com.  
[root@dns named]# host -t A dns.bigdata.wh.com 192.168.36.149  
Using domain server:  
Name: 192.168.36.149  
Address: 192.168.36.149#53  
Aliases:  
  
dns.bigdata.wh.com has address 192.168.36.149
```

- 追加区域bigdata.wh.com的反向解析定义。

```
cat << eof >> /etc/named.rfc1912.zones  
zone "168.192.in-addr.arpa" IN {  
    type master;  
    file "168.192.in-addr.arpa.zone";  
    allow-update { none; };  
};  
eof
```

- 创建区域的反向解析库文件。

```
cat << 'eof' > /var/named/168.192.in-addr.arpa.zone
$TTL 600
$ORIGIN 168.192.in-addr.arpa.
@ IN SOA dns.bigdata.wh.com. admin.bigdata.wh.com. (
20170510
1H
5M
1W
10M )
IN NS dns.bigdata.wh.com.
149.36 IN PTR dns.bigdata.wh.com. #ip到hostname的反向记录
eof
```

- 修改新增区域文件的所有者和权限。

```
[root@dns named]# ps aux|grep named
named 2668 0.0 1.4 165652 15008 ? Ssl 12:13 0:00 /usr/sbin/named
-u named
root 2680 0.0 0.0 112640 960 pts/0 S+ 12:14 0:00 grep --color=au
to named

chown :named bigdata.wh.com.zone
chmod 640 bigdata.wh.com.zone

chown :named 68.192.in-addr.arpa.zone
chmod 640 168.192.in-addr.arpa.zone
```

- 检测新增配置并重启DNS服务。

```
named-checkconf
named-checkzone "168.192.in-addr.arpa" /var/named/168.192.in-add
r.arpa.zone

systemctl restart named.service
```

- 测试IP反向解析成域名是否正常

```
[root@dns named]# host -t PTR 192.168.36.149 192.168.36.149
Using domain server:
Name: 192.168.36.149
Address: 192.168.36.149#53
Aliases:

149.36.168.192.in-addr.arpa domain name pointer dns.bigdata.wh.c
om.
```

- 在 `bigdata.wh.com` 区域内为集群其他主机添加A记录和PTR记录

```
[root@dns named]# cat bigdata.wh.com.zone
$TTL 600
$ORIGIN bigdata.wh.com.
@ IN SOA dns.bigdata.wh.com. admin.bigdata.wh.com. (
                20170510
                1H
                5M
                1W
                10M )
        IN      NS      dns
dns    IN      A       192.168.36.149
repo   IN      A       192.168.36.247 #区域内其他主机的A记录
proxy   IN      A       192.168.36.132
db      IN      A       192.168.36.101
admin   IN      CNAME   dns
*       IN      A       192.168.30.1
```

```
[root@dns named]# cat 168.192.in-addr.arpa.zone
$TTL 600
$ORIGIN 168.192.in-addr.arpa.
@ IN SOA dns.bigdata.wh.com. admin.bigdata.wh.com. (
20170510
1H
5M
1W
10M )
        IN NS      dns.bigdata.wh.com.
149.36  IN PTR     dns.bigdata.wh.com.
247.36  IN PTR     repo.bigdata.wh.com. #区域内其他主机的PTR记录
132.36  IN PTR     proxy.bigdata.wh.com.
101.36  IN PTR     db.bigdata.wh.com.
```

创建NTP时钟同步服务

为了保障大数据平台中Server收集到的状态信息、配置信息、任务调度信息以及保存的历史记录信息的有序和一致，集群内所有的节点主机和通过浏览器访问Web管理页面的客户机之间必须彼此时钟同步。因此我们通过在所有主机上安装Ntp服务来保证时间同步，即以某台主机作为Ntp Server，其他主机作为Ntp Client从Server端同步时间，这里我们选择repo主机作为server。

- 安装并启动ntp服务。

```
yum install -y ntp

systemctl start ntpd.service
systemctl enable ntpd.service
```

- 设置语言区域、时区。

```
localectl set-locale LANG=en_US.utf8      #设备系统语言及区域
timedatectl set-timezone Asia/Shanghai   #设置时区
```

- 同步网络时间或手动设置当前时间。

```
ntpdate time.ntp.org                      #有外网条件下，直接更新互联网标准时间
timedatectl set-ntp yes                   #启用NTP同步

timedatectl set-time 2017-05-11          #无外网条件下，手动设置当前日期
timedatectl set-time 16:47:00            #设置当前时间
```

- 修改Ntp服务的配置文件，创建Ntp Server端，确保如下配置项。

```
[root@repo yum.repos.d]# cat /etc/ntp.conf
# Permit all access over the loopback interface. This could
# be tightened as well, but to do so would effect some of
# the administrative functions.
restrict 127.0.0.1    #确保localhost本身有足够的权限
restrict ::1

# Hosts on local network are less restricted.
#restrict 192.168.1.0 mask 255.255.255.0 nomodify notrap
restrict 192.168.0.0 mask 255.255.0.0 nomodify notrap  #配置客户端
访问NTP Server的权限，                                         #授权192.1
68.0.0~192.168.255.255网段                                         #客户机被允
许从这台机器上查询和同步时间

#broadcast 192.168.1.255 autokey      # broadcast server
#broadcastclient                  # broadcast client
#broadcast 224.0.1.1 autokey        # multicast server
#multicastclient 224.0.1.1         # multicast client
#multicastserver 239.255.254.254     # multicast server
#multicastclient 239.255.254.254 autokey # multicast client

server 127.127.1.0    #将localhost的本地时钟作为时间供给源，这样，即便
该机器失去网络连接，也可以继续为网络提供服务
fudge  127.127.1.0 stratum 10
```

- 重启Ntp服务，并查询同步状态。

```
[root@repo yum.repos.d]# systemctl restart ntpd.service

[root@repo yum.repos.d]# ntpstat          #查询ntp服务同步状态
synchronised to NTP server (115.28.122.198) at stratum 3
    time correct to within 1804 ms
    polling server every 64 s

[root@repo yum.repos.d]# ntpq -p      #查看连接上的ntp远程服务器
      remote           refid      st t when poll reach   delay
offset jitter
=====
=====
  79.98.105.18    .INIT.        16 u      -    64    0   0.000
  0.000  0.000
+52.187.51.163  202.83.101.70   2 u      43    64    1  60.413  -
1586.1    3.586
*time6.aliyun.co 10.137.38.86   2 u      45    64    1  47.904  -
1582.3    2.975
  61.216.153.107 211.22.103.158   3 u      48    64    1  50.326  -
1583.0    0.106
 LOCAL(0)         .LOCL.        10 l      67    64    2   0.000
  0.000  0.000
```

- 值得注意的是，如果Ntp服务设置为开机启动，但是系统重启之后Ntp并没有启动，一般引起这个问题的最为常见的原因是系统安装了一个与Ntp相冲突的工具**chrony**，并且设置了自启动。解决这个问题的方法就是关闭 `chronyd.server` 的启动项，或是直接卸载掉工具**chrony**。

```
[root@gw ~]# systemctl status ntpd.service
● ntpd.service - Network Time Service
  Loaded: loaded (/usr/lib/systemd/system/ntp.service; enabled
; vendor preset: disabled)
  Active: inactive (dead)

[root@gw ~]# systemctl status chronyd.service
● chronyd.service - NTP client/server
  Loaded: loaded (/usr/lib/systemd/system/chronyd.service; enabled
; vendor preset: enabled)
  Active: active (running) since Mon 2017-05-22 02:44:19 UTC; 1
8min ago
    Process: 628 ExecStartPost=/usr/libexec/chrony-helper update-d
aemon (code=exited, status=0/SUCCESS)
    Process: 609 ExecStart=/usr/sbin/chronyd $OPTIONS (code=exited
, status=0/SUCCESS)
   Main PID: 616 (chronyd)
      CGroup: /system.slice/chronyd.service
              └─616 /usr/sbin/chronyd

[root@gw ~]# systemctl disable chronyd.service
Removed symlink /etc/systemd/system/multi-user.target.wants/chro
nyd.service.
```

初始化集群部署环境

前面几章我们分别完成了YUM本地源服务，DNS服务以及NTP服务的创建和配置，接下来我们要为集群中的所有主机配置这些基础服务，进一步还要进行安装部署前的环境初始化和检查。

- 设置主机名。

```
hostnamectl set-hostname server.daowoowh.com
```

- 设置静态的IP地址。

```
[root@server .ssh]# cat /etc/sysconfig/network-scripts/ifcfg-eth1
#VAGRANT-BEGIN
# The contents below are automatically generated by Vagrant. Do
not modify.
NM_CONTROLLED=no
BOOTPROTO=none
ONBOOT=yes
IPADDR=192.168.70.100
NETMASK=255.255.255.0
DEVICE=eth1
PEERDNS=no
#VAGRANT-END
```

- 设置系统语言区域和时区。

```
localectl set-locale LANG=en_US.utf8      #设备系统语言及区域
timedatectl set-timezone Asia/Shanghai    #设置时区
timedatectl set-ntp yes
```

- 禁用防火墙。

```
systemctl stop firewalld.service  
systemctl disable firewalld.service  
  
setenforce 0 #临时关闭Selinux  
sed -i 's/SELINUX=.*/SELINUX=disabled/' /etc/selinux/config #开机不启动Selinux
```

- 检查UMASK值。

UMASK值设置了用户在系统中创建新文件或文件夹时被授予的默认权限或基本权限，大数据平台支持的umask值为022/0022和027/0027。集群中的所有主机均要利用umask命令检查或修改umask值。

```
[root@gw ~]# umask #查询umask值  
0022  
  
[root@gw ~]# umask 022 #设置umask值为022
```

- SSH免密登录。

在大数据平台部署过程中，为了让Server自动地在所有集群主机上安装Agent，必须在Server主机和其他所有主机之间设置无密码的ssh连接。这样Server主机就能够使用ssh公钥通过身份验证来访问和安装Agent。

在Server主机上生成ssh公钥和私钥。

```
ssh-keygen  
  
[root@server ~]# ls -l ~/.ssh  
total 12  
-rw----- 1 root root 409 May  3 02:56 authorized_keys  
-rw----- 1 root root 1679 May 11 08:54 id_rsa          #私钥  
-rw-r--r-- 1 root root 408 May 11 08:54 id_rsa.pub     #公钥
```

将生成的ssh公钥复制到其他主机的 /root/.ssh 目录。

```
scp /root/.ssh/id_rsa.pub root@192.168.70.101:/root/.ssh/
```

在目标主机上将ssh公钥添加到 `authorized_keys` 文件中。

```
cat id_rsa.pub >> authorized_keys
```

然后验证从Server中利用ssh登录目标主机，确保无需输入密码而能够正常登录。

```
ssh root@192.168.70.101  
Are you sure you want to continue connecting (yes/no)? #第一登录会  
询问是否继续
```

- 系统DNS设置。

Centos7新增了NetworkManager来提供默认的网络服务，它是一个动态的网络控制和配置守护进程，在 `/etc/resolv.conf` 中修改nameserver后，它会定期的恢复成初始值。所以，我们需要禁止NetworkManager定期恢复dns。

```
cat << eof > /etc/NetworkManager/NetworkManager.conf  
[main]  
plugins=ifcfg-rh  
dns=none      #禁止更新dns  
  
[logging]  
#level=DEBUG  
#domains=ALL  
eof  
  
#重启 NetworkManager服务  
[root@server ~]# systemctl restart NetworkManager.service
```

```
#配置dns  
cat << eof > /etc/resolv.conf  
# Generated by NetworkManager  
search bigdata.wh.com  
nameserver 192.168.36.149  
eof
```

- 在DNS服务器中为集群其他主机添加A记录和PTR记录。

```
[root@dns named]# cat bigdata.wh.com.zone
$TTL 600
$ORIGIN bigdata.wh.com.
@ IN SOA dns.bigdata.wh.com. admin.bigdata.wh.com. (
                20170510
                1H
                5M
                1W
                10M )
          IN      NS      dns
dns      IN      A       192.168.36.149
repo     IN      A       192.168.36.247 #区域内其他主机的A记录
proxy    IN      A       192.168.36.132
db       IN      A       192.168.36.101

server   IN      A       192.168.70.100
gw       IN      A       192.168.70.101
nn       IN      A       192.168.70.102
snn      IN      A       192.168.70.103
hive     IN      A       192.168.70.104
dn001    IN      A       192.168.70.105
dn002    IN      A       192.168.70.106
dn003    IN      A       192.168.70.107

admin    IN      CNAME   dns
*        IN      A       192.168.30.1
```

```
[root@dns named]# cat 168.192.in-addr.arpa.zone
$TTL 600
$ORIGIN 168.192.in-addr.arpa.
@ IN SOA dns.bigdata.wh.com. admin.bigdata.wh.com. (
20170510
1H
5M
1W
10M )
      IN NS  dns.bigdata.wh.com.
149.36 IN PTR dns.bigdata.wh.com.
247.36 IN PTR repo.bigdata.wh.com. #区域内其他主机的PTR记录
```

```
132.36 IN PTR proxy.bigdata.wh.com.  
101.36 IN PTR db.bigdata.wh.com.  
  
100.70 IN PTR server.bigdata.wh.com.  
101.70 IN PTR gw.bigdata.wh.com.  
102.70 IN PTR nn.bigdata.wh.com.  
103.70 IN PTR snn.bigdata.wh.com.  
104.70 IN PTR hive.bigdata.wh.com.  
105.70 IN PTR dn001.bigdata.wh.com.  
106.70 IN PTR dn002.bigdata.wh.com.  
107.70 IN PTR dn003.bigdata.wh.com.
```

- 为集群其他主机设置Yum本地源。

备份系统默认源。

```
mv /etc/yum.repos.d /etc/yum.repos.d.bak  
mkdir /etc/yum.repos.d
```

下载之前创建的本地源repo文件，更新yum缓存

```
wget -O /etc/yum.repos.d/ambari.repo http://repo.bigdata.wh.com/  
resource/ambari-2.5.0.3.repo  
yum clean all  
yum repolist
```

- NTP时间同步。

安装Ntp，在Ntp Client客户端中配置本地Ntp Server的地址。

```
yum install -y ntp #安装ntpd服务  
  
#设置ntp server地址，修改为内网ntp服务器  
sed -i 's/server [0-3].centos.*/server repo.bigdata.wh.com/' /et  
c/ntp.conf
```

配置ntpd.service服务自启动。

```
[root@dns named]# systemctl enable ntpd.service
ln -s '/usr/lib/systemd/system/ntp.service' '/etc/systemd/system/multi-user.target.wants/ntp.service'
[root@server yum.repos.d]# systemctl restart ntpd.service
```

与本地的Ntp Server保持时钟同步。

```
[root@server yum.repos.d]# ntpdate -u repo.bigdata.wh.com
12 May 05:50:47 ntpdate[4136]: step time server 192.168.36.247 offset -3.068888 sec
[root@server yum.repos.d]# ntpq -p
      remote          refid      st t when poll reach   delay
offset jitter
=====
=====
repo.bigdata.wh 115.28.122.198    3 u    34    64      3    5.780  -
3068.4  29.757
```

- 优化swap分区设置。

```
[root@centos7 ~]# sysctl -w vm.swappiness=0          #优先使用内存，减少与磁盘交互
vm.swappiness = 0
[root@centos7 ~]# echo "vm.swappiness = 0" >> /etc/sysctl.conf
```

- 删除版本冲突的包。

Centos7镜像中默认安装了高版本的snappy，hdp-util中使用的低版本的snappy，这会导致安装datanode client的时候出现版本冲突，我们先把高版本的snappy卸载掉，后续安装过程中再从本地源中下载并安装低版本。

```
yum erase -y snappy.x86_64
```

创建Postgresql数据库

- 安装Postgresql二进制包。

```
yum install -y postgresql-server      #安装服务主程序  
yum install -y postgresql-contrib     #安装第三方包和分布式包（可选）
```

- Postgresql安装后的目录说明。

```
[root@db libexec]# rpm -ql postgresql-server.x86_64  
/usr/bin/*                                     #initdb、pos  
gres、psql等可执行程序存放在此目录  
/usr/lib/systemd/system/postgresql.service    #系统服务注册  
脚本，包含有服务自启动时PGDATA变量的定义  
/usr/lib/tmpfiles.d/postgresql.conf           #系统服务自启  
动时的主配置文件  
/usr/lib64/pgsql/*                            #动态库目录，  
程序运行所需要的动态库均在此目录下  
/usr/libexec/initscripts/legacy-actions/postgresql #initdb执行初  
始化时的脚本  
/usr/share/pgsql                                #配置文件模板  
文件以及一些扩展包的sql脚本文件  
/var/lib/pgsql                                    #postgres用户  
的根目录，.bash_profile文件中定义了PGDATA环境变量
```

- 初始化数据库，我们直接使用postgres默认的数据目
录 /var/lib/pgsql/data

```
[root@db ~]# su postgres          #切换至安装PG时创建的系统账户postgres
bash-4.2$ ls
backups  data
bash-4.2$ initdb -D data/ -U postgres -W      #初始化数据库
The files belonging to this database system will be owned by user "postgres".
This user must also own the server process.

The database cluster will be initialized with locale "en_US.utf8".
.

The default database encoding has accordingly been set to "UTF8"
.

The default text search configuration will be set to "english".

fixing permissions on existing directory data ... ok
creating subdirectories ... ok
selecting default max_connections ... 100
selecting default shared_buffers ... 32MB
creating configuration files ... ok
creating template1 database in data/base/1 ... ok
initializing pg_authid ... ok
Enter new superuser password:      #设置PG数据库超级管理员用户postgres的密码
Enter it again:
setting password ... ok
initializing dependencies ... ok
creating system views ... ok
loading system objects' descriptions ... ok
creating collations ... ok
creating conversions ... ok
creating dictionaries ... ok
setting privileges on built-in objects ... ok
creating information schema ... ok
loading PL/pgSQL server-side language ... ok
vacuuming database template1 ... ok
copying template1 to template0 ... ok
copying template1 to postgres ... ok
```

- 数据库初始化后，在数据库目录data中会生成一系列的文件夹和文件，它们的具体含义如下。

```
bash-4.2$ tree -L 1 data/
data/
|-- base  #默认表空间的目录，与每个数据库对应的子目录存储在该目录中
|-- global #集群范围的表存储在该目录中，比如'pg\_\_database'
|-- pg\_\_clog #包含事务提交状态数据的子目录
|-- pg\_\_hba.conf #访问认证配置文件，包括允许哪些IP的主机访问，采用的认证方法是什么等信息
|-- pg\_\_ident.conf #'ident'认证方式的用户映射文件
|-- pg\_\_multixact #包含多重事务状态数据的子目录\(使用共享的行锁\
|-- pg\_\_notify #包含LISTEN/NOTIFY状态数据的子目录
|-- pg\_\_serial #包含已提交可串行化事务信息的子目录
|-- pg\_\_snapshots #包含输出快照的子目录
|-- pg\_\_stat\_\_tmp #用于统计子系统的临时文件存储在该目录中
|-- pg\_\_subtrans #包含子事务状态数据的子目录
|-- pg\_\_tblspc #包含指向表空间的符号链接的子目录
|-- pg\_\_twophase #包含用于预备事务的状态文件的子目录
|-- PG\_\_VERSION #一个包含PostgreSQL主版本号的文件
|-- pg\_\_xlog #包含WAL\(预写日志\)文件的子目录
`-- postgresql.conf #数据库实例的主配置文件，基本上所有的配置参数均在此文件中

12 directories, 4 files
```

- 配置PG监听的IP和端口，对于拥有多个IP的主机，可根据实际需要选择监听哪几个IP。

```
bash-4.2$ cd data/
bash-4.2$ vi postgresql.conf
# - Connection Settings -
listen_addresses = '*'          #监听所有ip的连接，默认是本机
port = 5432                      #这个不写也行，默认就是5432端口
```

- 配置访问权限，为了提高安全性，可以配置成只允许特定的几台主机访问。

```
bash-4.2$ vim pg_hba.conf
# TYPE  DATABASE        USER        ADDRESS         METHOD
# "local" is for Unix domain socket connections only
local  all            all
trust
# IPv4 local connections:
host   all            all            127.0.0.1/32
trust
host   all            all            0.0.0.0/0
md5    #添加行，所有IP和用户，密码对都可以连接
# IPv6 local connections:
host   all            all            ::1/128
trust
```

- 利用PG默认提供的管理工具**pg_ctrl**来启动数据库，然后再查看5432端口是否正常开放。

```
bash-4.2$ pg_ctl start -D data/  #利用pg_ctl启动指定数据库实例
server starting
bash-4.2$ ss -tponl |grep 5432  #查看数据库服务监听的端口是否正常
LISTEN      0          128                  *:5432
*:          users:(( "postgres", 10297, 3 ))
LISTEN      0          128                  :::5432
:::*        users:(( "postgres", 10297, 4 ))
```

- 利用PG默认提供的客户端工具**psql**来连接并操作数据库，**psql**的操作命令本文不作详细描述，请参考**psql --help**。

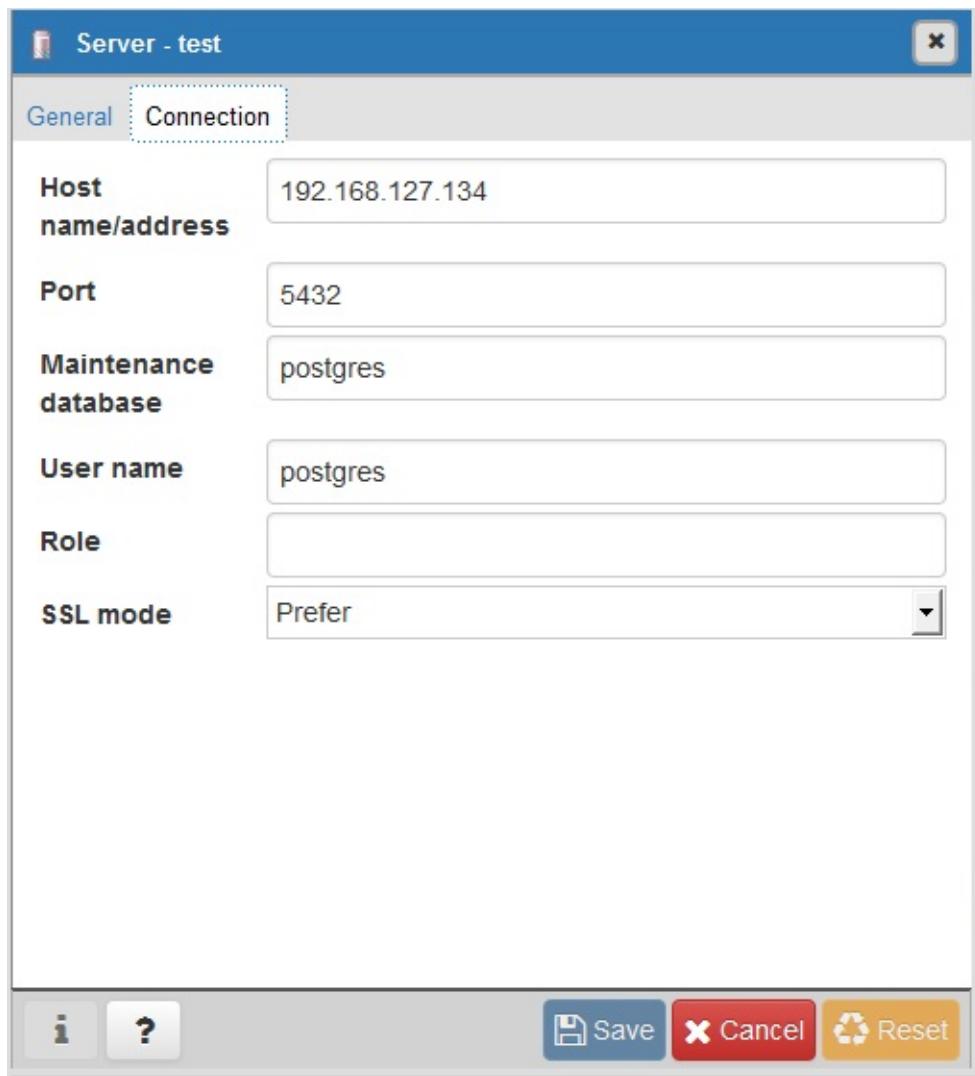
```
bash-4.2$ psql -U postgres    #psql客户端采用postgres用户登陆数据库（  
当前登陆的是默认的全局数据库postgres）  
psql (9.2.18)  
Type "help" for help.  
  
postgres=# \l          #执行psql客户端环境下提供的命令，该命令是列出该  
数据库实例中的所有数据库及模板  
                                         List of databases  
   Name    |  Owner   | Encoding | Collate  | Ctype    |  A  
access privileges  
-----+-----+-----+-----+-----+-----  
-----  
postgres | postgres | UTF8      | en_US.utf8 | en_US.utf8 |  
template0 | postgres | UTF8      | en_US.utf8 | en_US.utf8 | =c/  
postgres           +  
                   |          |          |          |          | pos  
tgres=CTc/postgres  
template1 | postgres | UTF8      | en_US.utf8 | en_US.utf8 | =c/  
postgres           +  
                   |          |          |          |          | pos  
tgres=CTc/postgres  
(3 rows)  
  
postgres=# \t  
Showing only tuples.  
postgres=# ALTER USER postgres WITH PASSWORD '1';      #利用SQL语  
句修改管理员密码  
ALTER ROLE  
postgres=# \q          #退出psql客户端  
bash-4.2$ exit          #退出postgres用户  
exit
```

- 利用GUI工具连接并操作数据库

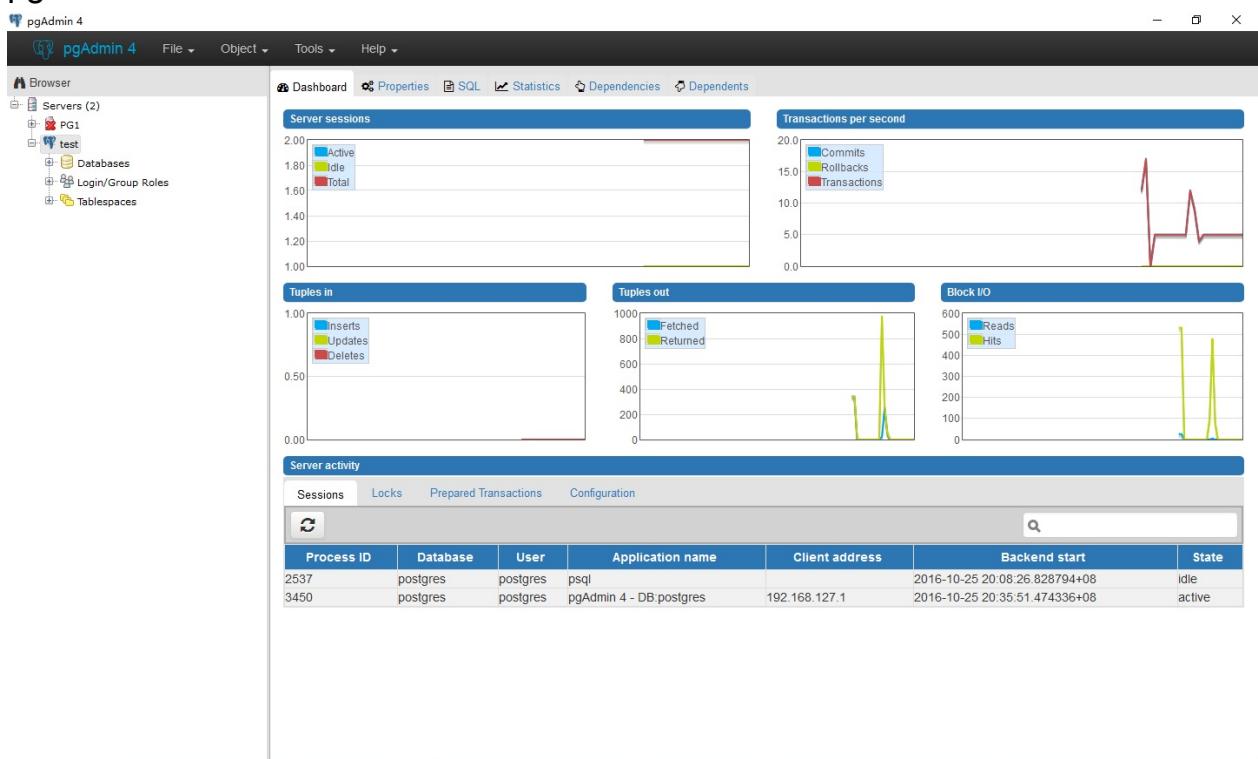
postgresql数据库的GUI工具有很多，与PG配合比较好用的主要有Navicat Premium和pgAdmin 3/4了。其中pgAdmin 4采用QT开发，支持跨平台，GUI非常赞，还添加了性能实时监视图表，不过目前多语言和流畅性都还有点问题。

pgAdmin4登录界面：

创建Postgresql数据库



pgAdmin4操作主界面：



- 配置PG跟随系统自启动

二进制安装时会自动把postgresql.service配置文件拷贝

到 /usr/lib/systemd/system/ 目录下，如果需要对自启动服务进行自定义配置，建议不要直接修改 /usr/lib/systemd/system/postgresql.service 文件，而是将该文件拷贝至 /etc/systemd/system/ 目录，然后再修改目标文件，并在目标文件内通过**.include**指令包含源文件。

```
systemctl enable postgresql.service  
systemctl start postgresql.service
```

```
[root@localhost system]# vi /etc/systemd/system/postgresql-9.5.service  
  
# It's not recommended to modify this file in-place, because it  
will be  
# overwritten during package upgrades. If you want to customize  
, the  
# best way is to create a file "/etc/systemd/system/postgresql-9  
.5.service",  
# containing  
#       .include /lib/systemd/system/postgresql-9.5.service  
#       ...make your changes here...  
# For more info about custom unit files, see  
# http://fedoraproject.org/wiki/Systemd#How_do_I_customize_a_uni  
t_file.2F_add_a_custom_unit_file.3F  
# Note: changing PGDATA will typically require adjusting SELinux  
# configuration as well.  
# Note: do not use a PGDATA pathname containing spaces, or you w  
ill  
# break postgresql-setup.  
.include /lib/systemd/system/postgresql-9.5.service      #通过'.  
include'指令包含源文件中的配置项
```

```
[Service]  
Environment=PGDATA=/home/pgsql/data/                      #然对指  
定配置项目进行修改
```


创建Yum缓存代理服务

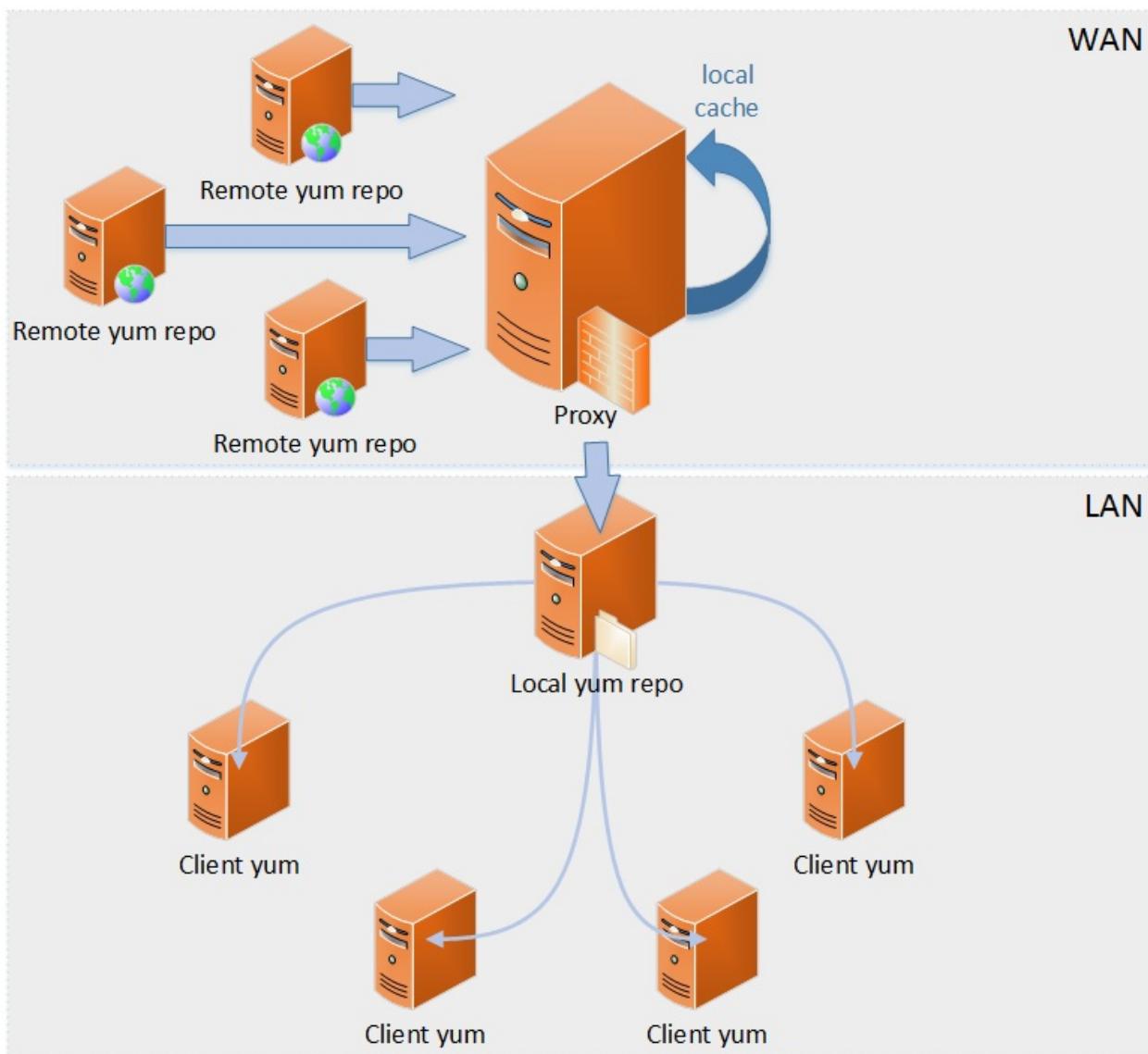
通常情况下，我们提供的自定义yum本地源压缩包中包含了大数据平台部署、运行和维护过程中的所有软件rpm包。除非是有特殊需要，比如开发人员试验某些新软件包的功能，或是运维人员安装自己熟悉的某些工具软件等，才需要从外部导入rpm包。因此，绝大多数场景下这个步骤都是可选的，运维人员要根据自己的实际需求来决定是否执行该步骤。

在需要导入外部rpm包的情况下，由于生产环境通常都是与外网进行隔离的，就算是可以通过其他手段获取到外网权限，但是通过修改集群中某些主机的网络配置和repo文件很显然是非常低效和十分危险的，极易造成集群主机之间的配置不一致，甚至数据泄露或丢失。

理想的解决方案应该是：

- yum安装软件时优先从集群内的本地源中查找，若存在则直接下载并安装。
- 若没有找到合适的rpm包，则将http请求转发到一台具有外网访问权限的proxy主机。
- proxy解析并执行该http请求进而从外部的标准源中获取rpm包，并缓存在本地。
- 将rpm包转发给原始请求的那台主机，yum完成软件安装。

其结构图如下：



Centos7环境下目前没有开源的工具能够满足以上的解决方案，不过Ubuntu14环境下有一个**apt-cacher-ng**的工具能满足要求，并且其最新版本增加了对yum所使用的rpm及repodata的支持。接下来描述的就是其实际安装和配置过程。

proxy主机初始化

利用Usb的系统安装盘在proxy主机上安装Ubuntu 14.04.5 LTS操作系统，安装完成后首先配置主机名。

```
hostnamectl set-hostname proxy.bigdata.wh.com #设置主机名
```

然后利用ifconfig查询Ubuntu系统识别的网卡信息，执行结果如下。

```

root@proxy:/etc/network# ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:27:54:e0:3b
#以太网口eth1
          inet  addr:10.0.2.15  Bcast:10.0.2.255  Mask:255.255.25
5.0
          inet6 addr: fe80::a00:27ff:fe54:e03b/64 Scope:Link
UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
RX packets:22809 errors:0 dropped:0 overruns:0 frame:0
TX packets:4259 errors:0 dropped:0 overruns:0 carrier:
0
          collisions:0 txqueuelen:1000
          RX bytes:31386840 (31.3 MB)  TX bytes:344498 (344.4 KB)
)

eth1      Link encap:Ethernet  HWaddr 08:00:27:a0:59:b6
#以太网口eth1
          inet  addr:192.168.36.111  Bcast:192.168.37.255  Mask:2
55.255.254.0
          inet6 addr: fe80::a00:27ff:fea0:59b6/64 Scope:Link
UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
RX packets:76679 errors:0 dropped:0 overruns:0 frame:0
TX packets:3088 errors:0 dropped:0 overruns:0 carrier:
0
          collisions:0 txqueuelen:1000
          RX bytes:8766387 (8.7 MB)  TX bytes:401980 (401.9 KB)

lo       Link encap:Local Loopback
#单机环回网卡
          inet  addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
UP LOOPBACK RUNNING  MTU:65536  Metric:1
RX packets:34 errors:0 dropped:0 overruns:0 frame:0
TX packets:34 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:4547 (4.5 KB)  TX bytes:4547 (4.5 KB)

```

Ubuntu 14系统的网络主配置文件为 `/etc/network/interfaces`，通常情况下主配置文件只配置loopback本地环回，其他的以每个网口对应于一个 `eth*.cfg` 配置文件的方式存放在 `/etc/network/interfaces.d` 目录下，并且通过主配置文

件来全部加载。

```
root@proxy:/etc/network# cat interfaces
# This file describes the network interfaces available on your system
# and how to activate them. For more information, see interfaces(5).

# The loopback network interface
auto lo
iface lo inet loopback

# Source interfaces
# Please check /etc/network/interfaces.d before changing this file
# as interfaces may have been defined in /etc/network/interfaces.d
# NOTE: the primary ethernet device is defined in
# /etc/network/interfaces.d/eth0
# See LP: #1262951
#其他的网口配置以eth0.cfg、eth1.cfg的方式存放在该目录，并在此处加载
source /etc/network/interfaces.d/*.cfg

#VAGRANT-BEGIN
# The contents below are automatically generated by Vagrant. Do not modify.
#Vagrant工具自动配置的网口，直接在主配置文件中附加了定义
auto eth1
iface eth1 inet dhcp
    post-up route del default dev $IFACE || true
#VAGRANT-END
```

然后，根据proxy主机的物理网口连接情况和机房内所属网段的路由设置，选择使用哪个网口以及何种网络配置方式。

- 若路由器支持DHCP方式连接，则按照如下方式配置所选的以太网口。

```
root@proxy:/etc/network# cat interfaces.d/eth1.cfg
# The primary network interface
auto eth1
iface eth1 inet dhcp
```

- 若路由器只允许以指定的静态IP方式连接，则按照如下方式来配置所选的以太网口。

```
root@proxy:/etc/network# cat interfaces.d/eth1.cfg
# The primary network interface
auto eth1
iface eth1 inet static
address 192.168.36.100
gateway 192.168.37.254
netmask 255.255.254.0
```

完成配置后，需要利用 `ifdown/ifup` 命令来重启该以太网口。

```
vagrant@proxy:~$ sudo ifdown eth1
Internet Systems Consortium DHCP Client 4.2.4
Listening on LPF/eth1/08:00:27:a0:59:b6
Sending on  LPF/eth1/08:00:27:a0:59:b6
Sending on  Socket/fallback
DHCPRELEASE on eth1 to 192.168.30.254 port 67 (xid=0x157b26fb)

vagrant@proxy:~$ sudo ifup eth1
Internet Systems Consortium DHCP Client 4.2.4
Listening on LPF/eth1/08:00:27:a0:59:b6
Sending on  LPF/eth1/08:00:27:a0:59:b6
Sending on  Socket/fallback
DHCPDISCOVER on eth1 to 255.255.255.255 port 67 interval 3 (xid=
0x89fef906)
DHCPREQUEST of 192.168.36.111 on eth1 to 255.255.255.255 port 67
(xid=0x6f9fe89)
DHCPOFFER of 192.168.36.111 from 192.168.37.254
DHCPACK of 192.168.36.111 from 192.168.37.254
bound to 192.168.36.111 -- renewal in 17083 seconds.
SIOCDELRT: No such process
```

为了让集群中的其他主机能够通过主机名来访问proxy缓存，需要在本地DNS服务器的 `bigdata.wh.com` 区域正向和反向解析库文件中分列添加A记录和PTR记录。

```
proxy    IN      A        192.168.36.111 #bigdata.wh.com.zone

111 IN PTR proxy.bigdata.wh.com.          #36.168.192.in-addr.arpa
.zone
```

最后，手动修改 `/etc/resolv.conf` 文件添加ISP提供的DNS服务器地址，以便能够访问外网。

```
root@proxy:~# cat /etc/resolv.conf
# Dynamic resolv.conf(5) file for glibc resolver(3) generated by
resolvconf(8)
#      DO NOT EDIT THIS FILE BY HAND -- YOUR CHANGES WILL BE OVER
WRITTEN
nameserver 10.0.2.3
nameserver 192.168.30.1
```

apt-cacher-ng安装及配置

Ubuntu 14操作系统默认源修改成国内源，以加快访问和下载速度。

```
cat << eof > /etc/apt/sources.list
deb http://mirrors.aliyun.com/ubuntu/ trusty main restricted uni
verse multiverse
deb http://mirrors.aliyun.com/ubuntu/ trusty-security main restr
icted universe multiverse
deb http://mirrors.aliyun.com/ubuntu/ trusty-updates main restri
cted universe multiverse
deb http://mirrors.aliyun.com/ubuntu/ trusty-proposed main restr
icted universe multiverse
deb http://mirrors.aliyun.com/ubuntu/ trusty-backports main rest
ricted universe multiverse
deb-src http://mirrors.aliyun.com/ubuntu/ trusty main restricted
universe multiverse
deb-src http://mirrors.aliyun.com/ubuntu/ trusty-security main r
estricted universe multiverse
deb-src http://mirrors.aliyun.com/ubuntu/ trusty-updates main re
stricted universe multiverse
deb-src http://mirrors.aliyun.com/ubuntu/ trusty-proposed main r
estricted universe multiverse
deb-src http://mirrors.aliyun.com/ubuntu/ trusty-backports main
restricted universe multiverse
eof
```

利用apt-get命令安装apt-cacher-ng和服务管理程序sysv-rc-conf。

```
apt-get install -y apt-cacher-ng sysv-rc-conf
```

启动apt-cacher-ng服务并配置跟随系统自启动。

```
service apt-cacher-ng start  
sysv-rc-conf apt-cacher-ng on
```

关闭防火墙ufw。

```
ufw disable
```

通过Web浏览器访问apt-cacher-ng主页。



apt-cacher-ng的主配置文件路径为/etc/apt-cacher-ng/acng.conf，编辑acng.conf确保以下配置项有效。

```

CacheDir: /var/cache/apt-cacher-ng      #存储已下载完毕的rpm包的缓存
目录
LogDir: /var/log/apt-cacher-ng        #默认的日志文件存放路径
SupportDir: /usr/lib/apt-cacher-ng    #存放辅助文件及脚本的默认端口
Port:3142                            #默认的http访问端口
ReportPage: acng-report.html         #在默认的web生成统计报告
VerboseLog: 1                         #记录更详细的Log信息

```

另外，在acng.conf中有多列以'Remap-'作为前缀的配置项，这每一行都表示一个资源重定向规则，其语法表示为： Remap-RepositoryName: MergingURLs ; TargetURLs ; OptionalFlags 。不过需要注意的是 MergingURLs 默认的根路径是 SupportDir ，而 TargetURLs 默认的根路径是/etc/apt-cacher-ng 。

```

# Repository remapping. See manual for details.
# In this example, some backends files might be generated during
package
# installation using information collected on the system.
# Examples:
Remap-debrep: file:deb_mirror*.gz /debian ; file:backends_debian
# Debian Archives
Remap-uburep: file:ubuntu_mirrors /ubuntu ; file:backends_ubuntu
# Ubuntu Archives
Remap-debvol: file:debvol_mirror*.gz /debian-volatile ; file:bac
kends_debvol # Debian Volatile Archives
Remap-cygwin: file:cygwin_mirrors /cygwin # ; file:backends_cygw
in # incomplete
Remap-sfnet: file:sfnet_mirrors # ; file:backends_sfnet # incom
plete
Remap-alxrep: file:archlx_mirrors /archlinux # ; file:backend_ar
chlx # Arch Linux
Remap-fedora: file:fedora_mirrors # Fedora Linux
Remap-epel: file:epel_mirrors # Fedora EPEL
Remap-slrep: file:sl_mirrors # Scientific Linux

```

于是我们来添加一个基于Centos7的rpm资源包的重定向规则。

```
Remap-centos: file:centos_mirrors /centos ; file:backends_centos  
# Centos Rpm
```

然后再创建文件centos_mirrors和backends_centos，表示所有来自于centos_mirrors定义地址的http请求全部被重定向到backends_centos定义的某个地址。

- centos_mirrors文件

该文件所包含的是Centos官方所定义的若干外部源地址，我们可以通过一个简单的脚本来生成它。

```
cat << 'eof' > /usr/lib/apt-cacher-ng/fetch-centos.sh  
#!/bin/bash  
  
URL="http://www.centos.org/download/full-mirrorlist.csv"  
INFILE=$(mktemp -t mirror-list-centos.XXXXXX)  
OUTFILE="centos_mirrors"  
  
wget --no-check-certificate -q -O "${INFILE}" "${URL}"  
  
tail -n+2 "${INFILE}" | awk -F '"' '{print $5}' > ${OUTFILE}  
tail -n+2 "${INFILE}" | awk -F '"' '{print $6}' >> ${OUTFILE}  
  
sed -i '' '/^$/d' ${OUTFILE}  
  
rm -f ${INFILE}  
eof
```

为fetch-centos.sh添加执行权限并执行脚本生成centos_mirrors文件。

```
chmod +x /usr/lib/apt-cacher-ng/fetch-centos.sh  
source /usr/lib/apt-cacher-ng/fetch-centos.sh  
mv centos_mirrors /usr/lib/apt-cacher-ng/
```

- backends_centos文件

```
cat << eof > /etc/apt-cacher-ng/backends_centos
http://mirrors.aliyun.com/centos/
http://mirrors.cn99.com/centos/
eof
```

但是外部源开启GPG-KEY验证时，安装部分软件会出现“403 Forbidden”错误。这是由于在apt-cacher-ng的默认配置中未包含RPM-GPG-KEY类型的文件，yum通过其代理下载相应KEY文件时被自动屏蔽了，解决该文件需要配置VfilePatternEx参数来添加RPM-GPG-KEY类型的文件。

```
VfilePatternEx: ^(/\\?release=[0-9]+&arch=.*)|.*/RPM-GPG-KEY-.*/m
etalink\\?repo=epel\\-[0-9]+&arch=.*)$
```

重新启动apt-cacher-ng服务。

```
root@proxy:/etc/apt-cacher-ng# service apt-cacher-ng restart
* Restarting apt-cacher-ng apt-cacher-ng
```

集群主机代理配置

为了让集群主机使用我们已经创建好的apt-cacher-ng缓存代理服务，有在yum.conf中添加proxy配置项和在repo文件中添加proxy配置项两种，前者是针对所有源的全局性修改，后者是针对特定源的局部性修改。

根据我们的实际需求，在创建的本地源中，那些大数据平台包含的Server、Agent以及各种Hadoop组件应该通过直接访问的方式来更新和下载，而它们所依赖的一些系统基础包或第三方辅助包应该在离线环境下，通过代理服务来完成更新和下载。

- 首先下载阿里云的repo文件。

```
wget -O /etc/yum.repos.d/CentOS7-Base-aliyun.repo http://repo.b
gdata.wh.com/resource/CentOS7-Base-aliyun.repo
```

- 然后修改阿里云的repo文件/etc/yum.repos.d/CentOS7-Base-aliyun.repo，使

其http请求通过 `proxy.bigdata.wh.com` 来进行代理和重定向。

```
[root@server yum.repos.d]# cat CentOS7-Base-aliyun.repo

[base]
name=CentOS-$releasever - Base - mirrors.aliyun.com
failovermethod=priority
baseurl=http://mirrors.aliyun.com/centos/$releasever/os/$basearc
h/
    http://mirrors.aliyuncs.com/centos/$releasever/os/$basea
rch/
#mirrorlist=http://mirrorlist.centos.org/?release=$releasever&r
ch=$basearch&repo=os
gpgcheck=1
gpgkey=http://mirrors.aliyun.com/centos/RPM-GPG-KEY-CentOS-7
proxy=http://proxy.bigdata.wh.com:3142/          #为base添加代理

#released updates
[updates]
name=CentOS-$releasever - Updates - mirrors.aliyun.com
failovermethod=priority
baseurl=http://mirrors.aliyun.com/centos/$releasever/updates/$ba
search/
    http://mirrors.aliyuncs.com/centos/$releasever/updates/$
basearch/
#mirrorlist=http://mirrorlist.centos.org/?release=$releasever&r
ch=$basearch&repo=updates
gpgcheck=1
gpgkey=http://mirrors.aliyun.com/centos/RPM-GPG-KEY-CentOS-7
proxy=http://proxy.bigdata.wh.com:3142/          #为updates添加代理

#additional packages that may be useful
[extras]
name=CentOS-$releasever - Extras - mirrors.aliyun.com
failovermethod=priority
baseurl=http://mirrors.aliyun.com/centos/$releasever/extras/$bas
earch/
    http://mirrors.aliyuncs.com/centos/$releasever/extras/$b
asearch/
#mirrorlist=http://mirrorlist.centos.org/?release=$releasever&r
```

```
ch=$basearch&repo=extras
gpgcheck=1
gpgkey=http://mirrors.aliyun.com/centos/RPM-GPG-KEY-CentOS-7
proxy=http://proxy.bigdata.wh.com:3142/          #为extras添加代理

#additional packages that extend functionality of existing packages
[centosplus]
name=CentOS-$releasever - Plus - mirrors.aliyun.com
failovermethod=priority
baseurl=http://mirrors.aliyun.com/centos/$releasever/centosplus/
$basearch/
    http://mirrors.aliyuncs.com/centos/$releasever/centosplus/$basearch/
#mirrorlist=http://mirrorlist.centos.org/?release=$releasever&arch=$basearch&repo=centosplus
gpgcheck=1
enabled=0
gpgkey=http://mirrors.aliyun.com/centos/RPM-GPG-KEY-CentOS-7
proxy=http://proxy.bigdata.wh.com:3142/          #为centosplus添加代理

#contrib - packages by Centos Users
[contrib]
name=CentOS-$releasever - Contrib - mirrors.aliyun.com
failovermethod=priority
baseurl=http://mirrors.aliyun.com/centos/$releasever/contrib/$basearch/
    http://mirrors.aliyuncs.com/centos/$releasever/contrib/$basearch/
#mirrorlist=http://mirrorlist.centos.org/?release=$releasever&arch=$basearch&repo=contrib
gpgcheck=1
enabled=0
gpgkey=http://mirrors.aliyun.com/centos/RPM-GPG-KEY-CentOS-7
proxy=http://proxy.bigdata.wh.com:3142/          #为contrib添加代理
```

- 更新yum缓存，检测配置是否成功。

```
yum clean all
yum makecache
```

查看proxy缓存

完成yum缓存更新后，可以在proxy主机的代理缓存目录 `/var/cache/apt-cacher-ng` 内找到 `repodata` 信息，如果proxy下载了rpm包，它们也会被存储在这个目录下。

```
root@proxy:/var/cache/apt-cacher-ng# tree -L 5 centos/
centos/
└── 7
    ├── extras
    │   └── x86_64
    │       └── repodata
    │           ├── b9ef6cf87b7680ec9aeba7da9fbddd3c0f672850f420
    │           │   42e90a5571f3d22dd2a0-filelists.sqlite.bz2
    │           │       ├── b9ef6cf87b7680ec9aeba7da9fbddd3c0f672850f420
    │           │       42e90a5571f3d22dd2a0-filelists.sqlite.bz2.head
    │           │           ├── ccd64088c86ba61f69b059b50c00c34a5e756b5cae3a
    │           │           2c671bd47e372a711a85-primary.sqlite.bz2
    │           │               ├── ccd64088c86ba61f69b059b50c00c34a5e756b5cae3a
    │           │               2c671bd47e372a711a85-primary.sqlite.bz2.head
    │           │                   ├── dbcca46a3dcc5a733cd9f02ab4aed05943047f4995e9
    │           │                   3fdb219213c7e009f62e-other.sqlite.bz2
    │           │                       ├── dbcca46a3dcc5a733cd9f02ab4aed05943047f4995e9
    │           │                       3fdb219213c7e009f62e-other.sqlite.bz2.head
    │           │                           ├── e4659f8012fd7e99c8adccbf4b4eecf0af3d38bc64c5
    │           │                           1d8bc061c40ffb2d508b-prestodelta.xml.gz
    │           │                               ├── e4659f8012fd7e99c8adccbf4b4eecf0af3d38bc64c5
    │           │                               1d8bc061c40ffb2d508b-prestodelta.xml.gz.head
    │           │                                   ├── repomd.xml
    │           │                                   └── repomd.xml.head
    │           └── os
    │               └── x86_64
    │                   └── repodata
    │                       ├── 3a1b41925bb25892c1003b22979ea0705aa815fed57f
    │                       992cf0229b76539a9ac4-filelists.sqlite.bz2
```

```

|           └── 3a1b41925bb25892c1003b22979ea0705aa815fed57f
992cf0229b76539a9ac4-filelists.sqlite.bz2.head
|           └── bd50ff3d861cc21d254a390a963e9f0fd7b7b96ed9d3
1ece2f2b1997aa3a056f-primary.sqlite.bz2
|           └── bd50ff3d861cc21d254a390a963e9f0fd7b7b96ed9d3
1ece2f2b1997aa3a056f-primary.sqlite.bz2.head
|           └── c55e5b7bbe933fa8dac2cffca4596c265812b74ed12e
f3968d487dd6eb22ad93-c7-x86_64-comps.xml.gz
|           └── c55e5b7bbe933fa8dac2cffca4596c265812b74ed12e
f3968d487dd6eb22ad93-c7-x86_64-comps.xml.gz.head
|           └── f7ed48f490360933445293386b7c300d55b31bc639a9
bc98d6c5365c702719a9-other.sqlite.bz2
|           └── f7ed48f490360933445293386b7c300d55b31bc639a9
bc98d6c5365c702719a9-other.sqlite.bz2.head
|           └── repomd.xml
|               └── repomd.xml.head
└── updates
    └── x86_64
        └── repodata
            ├── 3ea8c61f1c12741135aba7c0422e4b74fb072be9a437
75385c12fed4acb4059-prestodelta.xml.gz
            ├── 3ea8c61f1c12741135aba7c0422e4b74fb072be9a437
75385c12fed4acb4059-prestodelta.xml.gz.head
            ├── 5515a534747cd7cd585f9c77641a0d7a18125e291017
6e132fe8d88d07fdb0bc-other.sqlite.bz2
            ├── 5515a534747cd7cd585f9c77641a0d7a18125e291017
6e132fe8d88d07fdb0bc-other.sqlite.bz2.head
            ├── 68826be45b4c9e22f7b586290945b0d1cec66297b13a
73a81215d1fdf2c7297f-filelists.sqlite.bz2
            ├── 68826be45b4c9e22f7b586290945b0d1cec66297b13a
73a81215d1fdf2c7297f-filelists.sqlite.bz2.head
            ├── 84362bc7c7e9732f7781d7fdc02ace9c9a3157bc94a2
ebe40448a5759c714f22-primary.sqlite.bz2
            ├── 84362bc7c7e9732f7781d7fdc02ace9c9a3157bc94a2
ebe40448a5759c714f22-primary.sqlite.bz2.head
            ├── repomd.xml
            └── repomd.xml.head

```

10 directories, 30 files

管理中心安装

- 安装Server
- 创建PG实例
- 配置Server
- 启动Server

安装Server

- 利用root用户登录server主机

```
Connecting to 192.168.70.100:22...
Connection established.
To escape to local shell, press 'Ctrl+Alt+]'.

WARNING! The remote SSH server rejected X11 forwarding request.
Last login: Sun May 14 18:34:36 2017 from 192.168.70.1
Welcome to your Vagrant-built virtual machine.

[root@server ~]#
```

- 确认本地源及配置repo列表

```
yum repolist
```

- 安装server端

```
[root@server ~]# yum install ambari-server
Total download size: 670 M
Installed size: 755 M
Is this ok [y/d/N]: y
Downloading packages:
(1/4): postgresql-9.2.18-1.el7.x86_64.rpm
| 3.0 MB 00:00:03
(2/4): postgresql-server-9.2.18-1.el7.x86_64.rpm
| 3.8 MB 00:00:05
(3/4): postgresql-libs-9.2.18-1.el7.x86_64.rpm
| 232 KB 00:00:00
(4/4): ambari-server-2.5.0.3-7.x86_64.rpm
| 663 MB 00:01:00
-----
-----4/4
Installed:
ambari-server.x86_64 0:2.5.0.3-7

Dependency Installed:
postgresql.x86_64 0:9.2.18-1.el7  postgresql-libs.x86_64 0:9.2
.18-1.el7  postgresql-server.x86_64 0:9.2.18-1.el7

Complete!
```

创建PG实例

Server端默认情况下，在其启动过程中会在本机安装并创建postgresql数据库实例，并调用 `/var/lib/ambari-server/resources/` 目录的SQL脚本来初始化数据库实例。为了便于集中管理，我们使用了独立的主机来创建postgresql数据库实例，并且hadoop集群的其他组件所需的数据库均在该主机上创建。

- 利用root用户登录登录db主机。

```
WARNING! The remote SSH server rejected X11 forwarding request.  
Last login: Sun May 14 18:35:36 2017 from 192.168.1.54  
Welcome to your Vagrant-built virtual machine.  
[root@db ~]#
```

- 利用psql登录默认数据库。

```
[root@db ~]# sudo -u postgres psql
#以postgres用户登录
could not change directory to "/root"
psql (9.2.18)
Type "help" for help.

postgres=# CREATE DATABASE ambari;
#创建数据库ambari
CREATE DATABASE
postgres=# CREATE USER ambari WITH PASSWORD '123';
#创建用户ambari并设置密码
CREATE ROLE
postgres=# GRANT ALL PRIVILEGES ON DATABASE ambari TO ambari;
#给用户ambari添加操作数据库ambari的授权
GRANT
postgres=# \c ambari
#切换登录到ambari数据库
You are now connected to database "ambari" as user "postgres".
ambari=# CREATE SCHEMA ambari AUTHORIZATION ambari;
#创建从属于用户ambari的模式ambari
CREATE SCHEMA
ambari=# ALTER SCHEMA ambari OWNER TO ambari;
#修改模式ambari的拥有者为用户ambari
ALTER SCHEMA
ambari=# ALTER ROLE ambari SET search_path to 'ambari', 'public'
;
ALTER ROLE
ambari=# \q
```

- 从server主机的 /var/lib/ambari-server/resources/ 目录拷贝DDL初始化脚本。

```
[root@db ~]# scp vagrant@server.bigdata.wh.com:/var/lib/ambari-server/resources/Ambari-DDL-Postgres-CREATE.sql ./
The authenticity of host 'server.bigdata.wh.com (192.168.70.100)
' can't be established.
ECDSA key fingerprint is 75:06:22:b7:e2:3b:46:88:51:97:c4:4f:27:
20:21:26.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'server.bigdata.wh.com,192.168.70.100
' (ECDSA) to the list of known hosts.
vagrant@server.bigdata.wh.com's password:
Ambari-DDL-Postgres-CREATE.sql
100%    77KB   77.3KB/s   00:00

[root@db ~]# ls
Ambari-DDL-Postgres-CREATE.sql  anaconda-ks.cfg  ec2-keypair
```

- 利用DDL脚本初始化ambari数据库

```
[root@db ~]# psql -U ambari -d ambari
psql (9.2.18)
Type "help" for help.

ambari=> \i /root/Ambari-DDL-Postgres-CREATE.sql
```

- 列出ambari数据库中的所有数据表

```
ambari=> \dt
      ambari | adminpermission          | table | ambari
      ambari | adminprincipal          | table | ambari
      ambari | adminprincipaltype     | table | ambari
      ambari | adminprivilege          | table | ambari
      ambari | adminresource           | table | ambari
      ambari | adminresourcetype      | table | ambari
      ambari | alert_current           | table | ambari
      ambari | alert_definition       | table | ambari
      ambari | alert_group             | table | ambari
      ambari | alert_group_target      | table | ambari
      ambari | alert_grouping          | table | ambari
      ambari | alert_history           | table | ambari
--More--
```

至此自定义的数据库就成功创建了，当在server主机上启动Server时，我们就使用ambari用户来登录 db.bigdata.wh.com 主机上的ambari数据库。

配置 Server

在启动 Server 之前，必须先完成 Server 的数据库配置、JDK 安装、自定义账户。

- 执行 setup 命令

```
[root@server resources]# ambari-server setup
Using python /usr/bin/python
Setup ambari-server
Checking SELinux...
SELinux status is 'disabled'
```

- 默认情况下，使用 root 用户运行 Server 进程。

```
Customize user account for ambari-server daemon [y/n] (n)? n
Adjusting ambari-server permissions and ownership...
Checking firewall status...
```

- 选择 JDK 版本。

```
Checking JDK...
[1] Oracle JDK 1.8 + Java Cryptography Extension (JCE) Policy Files 8
[2] Oracle JDK 1.7 + Java Cryptography Extension (JCE) Policy Files 7
[3] Custom JDK
=====
=====
Enter choice (1):
```

- 拷贝 repo 主机 `/home/repo/resource` 目录中的 jdk-8u112 安装包至 Server 的资源目录。

```
scp vagrant@repo.bigdata.wh.com:/home/repo/resource/jdk-8u112-li  
nux-x64.tar.gz /var/lib/ambari-server/resources/  
scp vagrant@repo.bigdata.wh.com:/home/repo/resource/jce_policy-8  
.zip /var/lib/ambari-server/resources/
```

- 选择安装【1】。

```
Enter choice (1): 1  
JDK already exists, using /var/lib/ambari-server/resources/jdk-8  
u112-linux-x64.tar.gz  
Installing JDK to /usr/jdk64/  
Successfully installed JDK to /usr/jdk64/  
JCE Policy archive already exists, using /var/lib/ambari-server/  
resources/jce_policy-8.zip  
Installing JCE policy...  
Completing setup...
```

- 配置postgresql数据库。

```
Enter advanced database configuration [y/n] (n)? y      #启用高级配置
Configuring database...
=====
=====
Choose one of the following options:
[1] - PostgreSQL (Embedded)
[2] - Oracle
[3] - MySQL / MariaDB
[4] - PostgreSQL
[5] - Microsoft SQL Server (Tech Preview)
[6] - SQL Anywhere
[7] - BDB
=====
=====
Enter choice (1): 4                                #选择自定义的
PG数据库
Hostname (localhost): db.bigdata.wh.com            #配置PG主机的
hostname
Port (5432): 5432                                #配置PG主机的
数据库端口
Database name (ambari): ambari                   #配置数据库名
称
Postgres schema (ambari): ambari                 #配置数据库内
的模式名
Username (ambari): ambari                         #配置数据库用
户名
Enter Database Password (bigdata):                #输入该用户的
密码
Re-enter password:
Configuring ambari database...
Configuring remote database connection properties...
WARNING: Before starting Ambari Server, you must run the following DDL against the database to create the schema:
/var/lib/ambari-server/resources/Ambari-DDL-Postgres-CREATE.sql
```

- 确认数据库连接信息，完成Server安装过程。

```
Proceed with configuring remote database connection properties [y/n] (y)? y #确认连接信息
Extracting system views...
ambari-admin-2.5.0.3.7.jar
.....
Adjusting ambari-server permissions and ownership...
Ambari Server 'setup' completed successfully.
```

启动Server

- 通过start命令启动Server。

```
[root@server resources]# ambari-server start
Using python /usr/bin/python
Starting ambari-server
Ambari Server running with administrator privileges.
Organizing resource files at /var/lib/ambari-server/resources...
Ambari database consistency check started...
Server PID at: /var/run/ambari-server/ambari-server.pid
Server out at: /var/log/ambari-server/ambari-server.out
Server log at: /var/log/ambari-server/ambari-server.log
Waiting for server start.....
```

- 通过status命令查看Server进制状态。

```
[root@server resources]# ambari-server status
Using python /usr/bin/python
Ambari-server status
Ambari Server running
Found Ambari Server PID: 17791 at: /var/run/ambari-server/ambari
-server.pid
```

- 通过stop命令可以关闭Server。

```
[root@server yum.repos.d]# ambari-server stop
Using python /usr/bin/python
Stopping ambari-server
Waiting for server stop...
Ambari Server stopped
```

- 数据库一致性检查。

Server启动时，会运行数据库一致性检查来查找问题，如果发现任何问题，启动将中止并显示以下消息：DB配置一致性检查失败。并将有关数据库一致性检查结果的更多详细信息写入 /var/log/ambari-server/ambari-server-check-database.log 文件。

```
[root@server resources]# cat /var/log/ambari-server/ambari-server-check-database.log
2017-05-14 21:30:18,384 INFO - Checking DB store version
2017-05-14 21:30:20,214 INFO - DB store version is compatible
2017-05-14 21:30:37,208 INFO - ****
Check database started ****
2017-05-14 21:30:37,208 INFO - Ensuring that the schema set for Postgres is correct
2017-05-14 21:30:37,244 INFO - Checking for configs that are not mapped to any service
2017-05-14 21:30:37,259 INFO - Checking for configs not mapped to any cluster
2017-05-14 21:30:37,269 INFO - Checking for configs selected more than once
2017-05-14 21:30:37,282 INFO - Checking for hosts without state
2017-05-14 21:30:37,293 INFO - Checking host component states count equals host component desired states count
2017-05-14 21:30:37,326 INFO - Checking services and their configs
2017-05-14 21:30:37,326 INFO - Getting ambari metainfo instance
2017-05-14 21:30:37,327 INFO - Executing query 'GET_SERVICES_WITHOUT_CONFIGS'
2017-05-14 21:30:37,337 INFO - Executing query 'GET_SERVICE_CONFIG_WITHOUT_MAPPING'
2017-05-14 21:30:37,348 INFO - Getting stack info from database
2017-05-14 21:30:37,359 INFO - Executing query 'GET_SERVICES_WITH_CONFIGS'
2017-05-14 21:30:37,371 INFO - Comparing service configs from stack with configs that we got from db
2017-05-14 21:30:37,371 INFO - Getting services which has mapped configs which are not selected in clusterconfigmapping
2017-05-14 21:30:37,384 INFO - Checking Topology tables
2017-05-14 21:30:37,406 INFO - ****
Check database completed ****
```

启动Server

- 通过浏览器访问Server在默认端口8080开启的Web主页。

```
http://server.bigdata.wh.com:8080/
```

集群安装和配置

- [Web登录](#)
- [安装向导](#)
- [集群命名](#)
- [选择版本](#)
- [选项配置](#)
- [主机检查](#)
- [选择服务](#)
- [调整服务主组件](#)
- [调整服务从组件和客户端](#)
- [调整服务参数](#)
- [保存配置报告](#)
- [安装过程](#)
- [安装完成](#)

Web登录

在浏览器中，使用<http://server.bigdata.wh.com:8080> 以及默认账户admin/admin，登录Server启动的Web管理页面。

The screenshot shows the Apache Ambari web interface. At the top, there is a browser header with the URL 'server.bigdata.wh.com:8080/views/ADMIN_VIEW/2.5.0.3/INSTANCE/#/'. Below the header, the Ambari logo and the word 'Ambari' are visible. On the right side of the header, there is a user dropdown menu showing 'admin'. The main content area has a dark header bar with the text 'Welcome to Apache Ambari' and a sub-instruction 'Provision a cluster, manage who can access the cluster, and customize views for Ambari users.' Below this, there is a large central panel titled 'Create a Cluster' with the sub-instruction 'Use the Install Wizard to select services and configure your cluster'. It features a cloud icon and a blue button labeled 'Launch Install Wizard'. To the left of this central panel, there is a sidebar with three sections: 'Clusters' (with 'No Clusters' and 'Remote Clusters' options), 'Views' (with 'Views' and 'View URLs' options), and 'User + Group Management' (with 'Users' and 'Groups' options). To the right of the central panel, there are two smaller panels: 'Manage Users + Groups' (with a user icon and 'Users' and 'Groups' buttons) and 'Deploy Views' (with a grid icon and a 'Views' button).

安装向导

对于新集群，第一次登陆将显示欢迎页面，此时选择启动安装向导。



Manage Users + Groups

Manage the users and groups that can access Ambari



Users **Groups**

Deploy Views

Create view instances and grant permissions

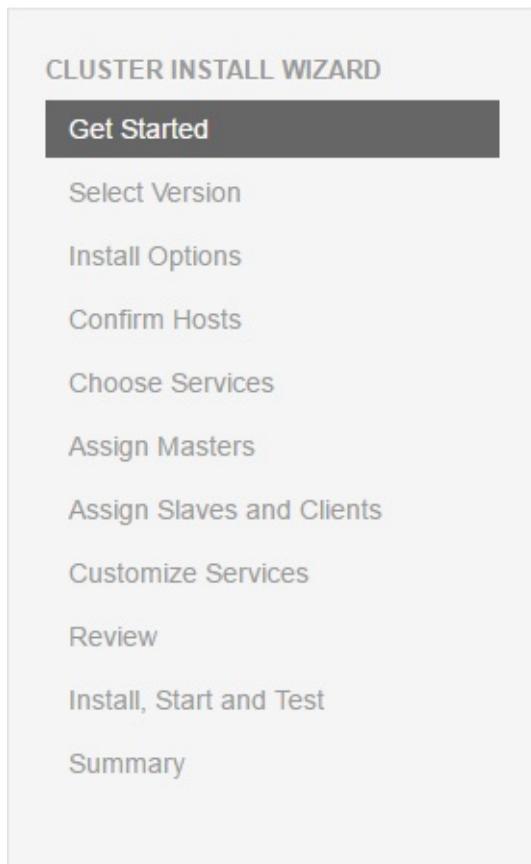


Views

集群命名

启动集群安装向导后，页面主页包含集群安装步骤列表和当前步骤配置两部分。

- 集群安装步骤，展示当前处于哪一个步骤，同时还可以返回到之前的步骤以便进行修改。



- 新集群命名，注意集群名字不能使用特殊符号。

The screenshot shows the 'Get Started' step of the wizard. The title 'Get Started' is at the top in a large, bold, dark font. Below it, a light blue box contains the text: 'This wizard will walk you through the cluster installation process. First, start by naming your new cluster.' Underneath this, there is a form field with the placeholder 'Name your cluster [Learn more](#)'. A red oval highlights the input field where the name 'bigdata' has been typed. To the right of the input field is a green 'Next →' button.

命名完成后进入下一步。

选择版本

该页面通过Tab式的按钮来选择准备安装的平台组件版本，并在其右边以列表的方式，详细展示了该HDP版本所对于的各个组件版本关系。

- 在本文中的示例中，我们选择安装最新的HDP-2.6版本。

Select Version

Select the software version and method of delivery for your cluster. Using a Public Repository requires Internet connectivity. Using a Local Repository requires you have configured the software in a repository available in your network.

HDP-2.6	HDP-2.6 ▾														
HDP-2.5															
HDP-2.4															
HDP-2.3															
	<table border="1"> <tr><td>Accumulo</td><td>1.7.0.2.6</td></tr> <tr><td>Ambari Infra</td><td>0.1.0</td></tr> <tr><td>Ambari Metrics</td><td>0.1.0</td></tr> <tr><td>Atlas</td><td>0.8.0.2.6</td></tr> <tr><td>Druid</td><td>0.9.2.2.6</td></tr> <tr><td>Falcon</td><td>0.10.0.2.6</td></tr> <tr><td>Flume</td><td>1.5.2.2.6</td></tr> </table>	Accumulo	1.7.0.2.6	Ambari Infra	0.1.0	Ambari Metrics	0.1.0	Atlas	0.8.0.2.6	Druid	0.9.2.2.6	Falcon	0.10.0.2.6	Flume	1.5.2.2.6
Accumulo	1.7.0.2.6														
Ambari Infra	0.1.0														
Ambari Metrics	0.1.0														
Atlas	0.8.0.2.6														
Druid	0.9.2.2.6														
Falcon	0.10.0.2.6														
Flume	1.5.2.2.6														

- 根据实际使用的系统类型及版本来设置HDP组件及工具集的本地源路径，以便Server生成对应的repo文件。

Use Public Repository

Use Local Repository

Repositories

Provide Base URLs for the Operating Systems you are configuring.

OS	Name	Base URL	Actions
redhat7	HDP-2.6	http://repo.bigdata.wh.com/hadoop/HDP/centos7/	
	HDP-UTILS-1.1.0.21	http://repo.bigdata.wh.com/hadoop/HDP-UTILS-1.1.0.21/	Remove

Skip Repository Base URL validation (Advanced)

Use RedHat Satellite/Spacewalk

[← Back](#) [Next →](#)

点击下一步后，系统会首先检测输入的URL地址能够正常访问，然后才真正的进入下一步。

选择版本

选项配置

在该页面需要录入Server用于自动化的访问集群中其他目标主机的相关配置信息。

- 输入目标主机名列表，注意每一行代表一个目标主机。

Target Hosts

Enter a list of hosts using the Fully Qualified Domain Name (FQDN), one per line. Or use [Pattern Expressions](#)

```
gw.bigdata.wh.com  
nn.bigdata.wh.com  
sn.bigdata.wh.com  
dn001.bigdata.wh.com  
dn002.bigdata.wh.com
```

- 之前我们配置了server主机针对其他目标主机的SSH免密登陆，这里我们先获取其SSH私钥。

```
[root@server .ssh]# cat /root/.ssh/id_rsa
-----BEGIN RSA PRIVATE KEY-----
MIIEpgIBAAKCAQEazXYScP6GHLivxzsytl4eeFss6oi2nsHHLsjsi2cHlnm+R6Vu
f0Mk0Xw0rtxowp7I7op7fv+gmv+Jv3sY+ja0RdhbbYspDS1jHxfMERRwYhVpexPa
WQ3n1cy0YDAONqKGzQ86+9zCQ1l0Mvdzjd53NI48j/u5bDCbhwcMPVGw3z9880t
Qo0ZbEVhJ5F1Sbz/Ap3UGIbILLqfVBgVnvEYhEwsmY9joUXX0jSLGiNejr/wH616
KvOLmhSDqh6w/hxq+uFcga7HbEmL+82Gwp8Z8z+hCh19tq8arpswhb2T7TykSwi
m7vZBgP3t1sdGePw9BwPT8uVqhQY43J0tFBmhwIDAQABoIBAQCA0NM1Fs78u0o0
LjBYWGAgM4HQtdBRbsqz0XNE317JgCDFiLniAQMYK4BYVYXzsvPlYtuUvy5xn188
ty/syFl0JPcFkic1xMwN1XzzBG6FgEk2yjaue0JGcczy3A49QN71N/RSLpF5akd1
+uDvBJiWUcs0tq0FY0BR5fySUWWBb/+OK2Vkv8ijnhEZcNvZBqPpQHiJKw9fLaw1
25gUDqqcHu+tuyijfHGJH26dNx6619SYVK/9P29fRw06AKflPqIEfL1z/m8RCUQo
Fd1802738Al1pPnz+nTz0vZDvVqk/2k80a+3fDqPC30G1/IAJ7IyfdQk70onxLpj
0Gg1Ng4BAoGBA0xZ5etKtaItgDFcB09ElQK+NDLyAT8T4YB/BvN+87TvAsvmJrmt
c9gNd2ltcghYBK6zIlVXVEn0Gr9Xc+izfbYxAJdPme2LSo+sWWU/5uJmD82jGeYj
eRGQmRn0+RtoE2+Z0Rw6ghTJx8uTCodIrS4F4Q05CUADH5u1Km0o3G0BAoGBAN6K
w5yMy1glZg4JPj+8QPjPcNBmAcUhdjZcTf1WC4diQ2u38e5L4ofAdqwFwXrI0pc
FMPB/Muj+rLD76W7cUwuBtA4BQtTk6wfir/2CcV6K1phGUOL103rfI7h1uqiMby7
99Pmcz3TXXLMarKdCVCbs/EZT2Izvl2G73oJA+uHAoGBAJkuTohnnD6m9L2I4R3d
uiHT+mrG15WtIeqw6WVo8zRh79MMsC6JD1qIp8rphw2HVkmPigh7noJrteYrHNFF
e4VYTtTCL4Y4T708RRgNCWvUMAvb2I5CkVXj/IZJMiYkFuyuqUt9VA97E4WKIDm7
zZnVb5eFFkypeZPmH7oFmA8BAoGBANCB+11GnKR4xjD1Cd8yHueh11DHuIWJuQ7A
TNA9U+2BRtRHMOyUmFIzsy0PJ8Mn1qM+u0XF9hNP6sW4gbKZREXXtLgaf1+yTHQ
K9RH1kfseppLt7wN2+c/aGkHOLKGXUUUY1Nr7DXVQA07cg0ADaY0/Je90x+rk4VV
1DLnF4EpAoGBAI5uqN894BYxhzXrmK7 cwdGzg3IBg1GR48dFK0ilApacxyew4UWF
FPxZTxXTpfJaNNnW0QWaughAGt7W6NBtbqMte9TTG+trf/7S2MUvbI75WxcMUhLw
A6Lg2p05bvTCDIUTaL4sEG9S1k9G03zxrw/VmlffxREqW1+CL8NfzTAq
-----END RSA PRIVATE KEY-----
```

- 拷贝server主机的SSH私钥至输入框。

Host Registration Information

- Provide your [SSH Private Key](#) to automatically register hosts

选择文件 未选择任何文件

-----BEGIN RSA PRIVATE KEY-----
MIIEpgIBAAKCAQEAzXYScP6GHLivxzsytl4eeFss6oi2nsHHLsjsi2cHlnm+R6Vu
fOMkOXw0rtxoWp7I7op7fv+gmv+Jv3sY+ja0RdhbbYspDS1jHxfMERRwYhVpexPa
WQ3n1cy0YDAONqKGzQ86+9zCQ1l0Mvdzjd53NI48j/u5bDCbhwicMPVGw3z9880t
QoOZbEVhJ5FlSbz/Ap3UGIbI1LqfVBgVnvEYhEwsmY9joUXXOjSLGiNejr/wH616
KvOLmhSDqhs6w/hxq+uFcga7HbEmL+82Gwp8Z8z+hCh19tq8arpwhb2T7TyrkSwi
m7u7BqB3t1edGeBu9BvPTsuvVchQY43J0tFBmhvIDAQABAcIBACQANM1E=78uOo

- 默认使用root用户和22端口登陆SSH。

SSH User Account

root

SSH Port Number

22

- Perform [manual registration](#) on hosts and do not use SSH

[← Back](#)

Register and Confirm →

主机检查

检查集群中准备安装组件的主机是否能够正常访问并登录，以及创建Agent相关的目录并拷贝软件包和脚本资源，最后启动Agent进程来与Server通信和交互，确保下一步的安装过程顺利完成。

- 逐个项目地检查主机。

Confirm Hosts

Registering your hosts.
Please confirm the host list and remove any hosts that you do not want to include in the cluster.

	Host	Progress	Status	Action
<input type="checkbox"/>	gw.bigdata.wh.com		Installing	
<input type="checkbox"/>	nn.bigdata.wh.com		Installing	
<input type="checkbox"/>	sn.bigdata.wh.com		Installing	
<input type="checkbox"/>	dn001.bigdata.wh.com		Installing	
<input type="checkbox"/>	dn002.bigdata.wh.com		Installing	

Show: All (5) | [Installing \(5\)](#) | [Registering \(0\)](#) | [Success \(0\)](#) | [Fail \(0\)](#)

Show: 25 ▾ 1 - 5 of 5 ⌂ ⌃ ⌁ ⌂ ⌁

[← Back](#)[Next →](#)

- 可以通过Status栏中的链接地址来弹出信息窗口查看检查进度，如果检查失败还可以查看到具体的失败原因。

Progress	Status
	Success
	Failed
	Failed
	Failed
	Registering
	Success
	Success

Registration log for gw.bigdata.wh.com

```
=====
Creating target directory...
=====

Command start time 2017-05-16 01:55:51
chmod: cannot access '/var/lib/ambari-agent/data': No such file or directory

Warning: Permanently added 'gw.bigdata.wh.com' (ECDSA) to the list of known hosts.
Connection to gw.bigdata.wh.com closed.
SSH command execution finished
host=gw.bigdata.wh.com, exitcode=0
Command end time 2017-05-16 01:55:52
```

click to highlight

- 排除错误后通过Retry按钮再重新检查失败的目标主机。

<input type="checkbox"/> Host	Progress	Status	Action
<input type="checkbox"/> server.bigdata.wh.com		Success	<input type="button" value="Remove"/>
<input type="checkbox"/> gw.bigdata.wh.com		Failed	<input type="button" value="Remove"/>

<input type="checkbox"/> Host	Progress	Status	Action
<input type="checkbox"/> server.bigdata.wh.com		Success	<input type="button" value="Remove"/>
<input type="checkbox"/> gw.bigdata.wh.com		Installing	<input type="button" value="Remove"/>
<input type="checkbox"/> nn.bigdata.wh.com		Installing	<input type="button" value="Remove"/>
<input type="checkbox"/> snn.bigdata.wh.com		Installing	<input type="button" value="Remove"/>

- 主机环境检测全部完成后进入下一步。

<input type="checkbox"/> Host	Progress	Status	Action
<input type="checkbox"/> gw.bigdata.wh.com		Success	<input type="button" value="Remove"/>
<input type="checkbox"/> nn.bigdata.wh.com		Success	<input type="button" value="Remove"/>
<input type="checkbox"/> snn.bigdata.wh.com		Success	<input type="button" value="Remove"/>
<input type="checkbox"/> hive.bigdata.wh.com		Success	<input type="button" value="Remove"/>
<input type="checkbox"/> dn001.bigdata.wh.com		Success	<input type="button" value="Remove"/>
<input type="checkbox"/> dn002.bigdata.wh.com		Success	<input type="button" value="Remove"/>
<input type="checkbox"/> dn003.bigdata.wh.com		Success	<input type="button" value="Remove"/>

Show: 25 ▾ 1 - 7 of 7 ⏪ ⏴ ⏵ ⏩ ⏹

All host checks passed on 7 registered hosts. [Click here to see the check results.](#)

← Back ↑ 这里可以查看主机环境检测的详细过程和执行的相关命令 Next →

选择服务

根据之前版本选择步骤中所选中的大数据平台版本，该页面以列表的方式展示出其中所包含的服务组件、版本号、描述等信息。我们可以通过勾选的方式来决定现在将要安装哪些可用服务，或者第一次只安装核心服务，之后再逐步添加其他可用的服务。

- 根据需求来选择若干服务组件，然后下一步。

<input checked="" type="checkbox"/> HDFS	2.7.3.2.6	Apache Hadoop Distributed File System
<input checked="" type="checkbox"/> YARN + MapReduce2	2.7.3	Apache Hadoop NextGen MapReduce (YARN)
<input type="checkbox"/> Tez	0.7.0	Tez is the next generation Hadoop Query Processing framework written on top of YARN.
<input checked="" type="checkbox"/> Hive	1.2.1.2.6	Data warehouse system for ad-hoc queries & analysis of large datasets and table & storage management service
<input checked="" type="checkbox"/> HBase	1.1.2.2.6	A Non-relational distributed database, plus Phoenix, a high performance SQL layer for low latency applications.
<input checked="" type="checkbox"/> Oozie	4.2.0.2.6	System for workflow coordination and execution of Apache Hadoop jobs. This also includes the installation of the optional Oozie Web Console which relies on and will install the ExtJS Library.
<input checked="" type="checkbox"/> ZooKeeper	3.4.6	Centralized service which provides highly reliable distributed coordination
<input type="checkbox"/> Falcon	0.10.0.2.6	Data management and processing platform
<input type="checkbox"/> Storm	1.0.1	Apache Hadoop Stream processing framework

- Server会自动检查所选组件的关联和依赖关系，并将检查结果以弹出窗体的方式通知给你。

<input type="checkbox"/> Slider	0.91.0	A framework for deploying, managing and monitoring existing distributed applications on YARN.
---------------------------------	--------	---

← Back

Next →



Tez Needed

X

You did not select Tez, but it is needed by other services you selected. We will automatically add Tez. Is this OK?

Cancel

OK

调整服务主组件

该页面为选中的服务在集群中分配适当主机来作为Master组件。页面左侧列显示服务和所选择的当前主机，右列显示这些主机当前的组件分配情况，并指示主机CPU内核的数量和RAM的数量。

初始情况下，系统会给出默认的建议分配方案，我们可以直接采用该默认方案或者自己动手进行调整。

- 默认的分配方案，如果集群中节点较多，建议平均分配这些组件，避免单个节点的负荷过重，进而影响集群性能。

Assign Masters

Assign master components to hosts you want to run them on.
* HiveServer2 and WebHCat Server will be hosted on the same host.

SNameNode:	dn002.bigdata.wh.com (992.7 MB, 1 cores)
NameNode:	dn001.bigdata.wh.com (992.7 MB, 1 cores)
ResourceManager:	dn002.bigdata.wh.com (992.7 MB, 1 cores)
App Timeline Server:	dn002.bigdata.wh.com (992.7 MB, 1 cores)
History Server:	dn002.bigdata.wh.com (992.7 MB, 1 cores)
HiveServer2:	dn003.bigdata.wh.com (992.7 MB, 1 cores)
WebHCat Server:	dn003.bigdata.wh.com (992.7 MB, 1 cores)
Hive Metastore:	dn003.bigdata.wh.com (992.7 MB, 1 cores)
HBase Master:	dn003.bigdata.wh.com (992.7 MB, 1 cores)
Oozie Server:	dn003.bigdata.wh.com (992.7 MB, 1 cores)
ZooKeeper Server:	dn001.bigdata.wh.com (992.7 MB, 1 cores)

- 根据之前规划的主机名来调整Master组件位置和个数。

1. NameNode和ResourceManager组件调整到nn主机；
2. SNameNode和HST等其他服务组件调整到snn主机；
3. Hive和HBase相关的组件调整到hive主机；
4. Grafana和Activity等余下的组件则全部调整到gw主机；
5. ZooKeeper数量为三个，分别分布在nn、snn和hive三台主机上。

NameNode:	nn.bigdata.wh.com (992.7 MB, ▾)	gw.bigdata.wh.com (992.7 MB, 1 cores)
SNameNode:	snn.bigdata.wh.com (992.7 MB ▾)	Metrics Collector Grafana Activity Analyzer Activity Explorer
ResourceManager:	nn.bigdata.wh.com (992.7 MB, ▾)	hive.bigdata.wh.com (992.7 MB, 1 cores)
App Timeline Server:	nn.bigdata.wh.com (992.7 MB, ▾)	Hive Metastore HiveServer2 WebHCat Server HBase Master ZooKeeper Server
History Server:	snn.bigdata.wh.com (992.7 MB ▾)	nn.bigdata.wh.com (992.7 MB, 1 cores)
Hive Metastore:	hive.bigdata.wh.com (992.7 ME ▾)	NameNode ResourceManager App Timeline Server ZooKeeper Server
HiveServer2:	hive.bigdata.wh.com (992.7 ME ▾)	snn.bigdata.wh.com (992.7 MB, 1 cores)
WebHCat Server:	hive.bigdata.wh.com *	SNameNode History Server ZooKeeper Server HST Server
HBase Master:	hive.bigdata.wh.com (992.7 ME ▾)	
ZooKeeper Server:	nn.bigdata.wh.com (992.7 MB, ▾)	

- 可以通过主机旁边绿色的‘+/-’图标来增加/减少Master组件的实例个数。

ZooKeeper Server:	snn.bigdata.wh.com (992.7 MB ▾)		
ZooKeeper Server:	hive.bigdata.wh.com (992.7 ME ▾)		
Metrics Collector:	gw.bigdata.wh.com (992.7 MB, ▾)		
Grafana:	gw.bigdata.wh.com (992.7 MB, ▾)		

调整服务从组件和客户端

该页面为服务在集群中分配适当主机来作为 Slave 组件，例如 DataNodes、NodeManagers 和 RegionServers 等。同时还可以选择特定的主机以安装所有服务的客户端。

- 默认的分配方案，同样我可以通过集群的规划来进行调整。

Assign Slaves and Clients

Assign slave and client components to hosts you want to run them on.

Hosts that are assigned master components are shown with *.

"Client" will install HDFS Client, YARN Client, MapReduce2 Client, Tez Client, HCat Client, Hive Client, HBase Client, Pig Client, ZooKeeper Client and Slider Client.

Host	all none	all none	all none	all none	all none	all none	all					
gw.bigdata.wh.com *	<input type="checkbox"/>	DataNode	<input type="checkbox"/>	NFSGateway	<input type="checkbox"/>	NodeManager	<input type="checkbox"/>	RegionServer	<input type="checkbox"/>	Phoenix Query Server	<input checked="" type="checkbox"/>	Cli
nn.bigdata.wh.com *	<input type="checkbox"/>	DataNode	<input type="checkbox"/>	NFSGateway	<input type="checkbox"/>	NodeManager	<input type="checkbox"/>	RegionServer	<input type="checkbox"/>	Phoenix Query Server	<input type="checkbox"/>	Cli
snn.bigdata.wh.com *	<input type="checkbox"/>	DataNode	<input type="checkbox"/>	NFSGateway	<input type="checkbox"/>	NodeManager	<input type="checkbox"/>	RegionServer	<input type="checkbox"/>	Phoenix Query Server	<input type="checkbox"/>	Cli
hive.bigdata.wh.com *	<input checked="" type="checkbox"/>	DataNode	<input type="checkbox"/>	NFSGateway	<input checked="" type="checkbox"/>	NodeManager	<input checked="" type="checkbox"/>	RegionServer	<input type="checkbox"/>	Phoenix Query Server	<input type="checkbox"/>	Cli
server.bigdata.wh.com	<input checked="" type="checkbox"/>	DataNode	<input type="checkbox"/>	NFSGateway	<input checked="" type="checkbox"/>	NodeManager	<input checked="" type="checkbox"/>	RegionServer	<input type="checkbox"/>	Phoenix Query Server	<input type="checkbox"/>	Cli
dn001.bigdata.wh.com	<input checked="" type="checkbox"/>	DataNode	<input type="checkbox"/>	NFSGateway	<input checked="" type="checkbox"/>	NodeManager	<input checked="" type="checkbox"/>	RegionServer	<input type="checkbox"/>	Phoenix Query Server	<input checked="" type="checkbox"/>	Cli
dn002.bigdata.wh.com	<input checked="" type="checkbox"/>	DataNode	<input type="checkbox"/>	NFSGateway	<input checked="" type="checkbox"/>	NodeManager	<input checked="" type="checkbox"/>	RegionServer	<input type="checkbox"/>	Phoenix Query Server	<input type="checkbox"/>	Cli
dn003.bigdata.wh.com	<input checked="" type="checkbox"/>	DataNode	<input type="checkbox"/>	NFSGateway	<input checked="" type="checkbox"/>	NodeManager	<input checked="" type="checkbox"/>	RegionServer	<input type="checkbox"/>	Phoenix Query Server	<input checked="" type="checkbox"/>	Cli

- 我们在 dn 系列主机上安装 DataNode、NodeManager、RegionServers 和 Phoenix Query Server，在 gw 主机上安装 NFSGateway 以及服务的客户端，后期在集群中添加限制条件，只能通过 gw 主机才能访问集群资源和运行任务。

Host	all	none	all	none	all	none	all	none	all	r
gw.bigdata.wh.com*	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
nn.bigdata.wh.com*	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
snn.bigdata.wh.com*	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
hive.bigdata.wh.com*	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
server.bigdata.wh.com	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
dn001.bigdata.wh.com	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>				
dn002.bigdata.wh.com	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>				
dn003.bigdata.wh.com	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>				

master components hosted on

Show: 1 - 8 of 8 ◀ ▶

[← Back](#)[Next →](#)

调整服务参数

在初始的安装过程中，建议优先采用默认的参数来完成组件的安装，待平台中所有的组件都正常启动后再根据自己的需求调整各项配置参数来提高性能。

Customize Services

We have come up with recommended configurations for the services you selected. Customize them as you see fit.

HDFS YARN MapReduce2 Tez **Hive 1** HBase Pig ZooKeeper Ambari Metrics 1 SmartSense 1
Slider Misc

Hive组件所必须的数据库连接设置。

这里我们选择连接在db主机上创建的postgresql数据库，以便实现所有组件数据库的集中管理和维护。

- 首先在db主机上创建hive数据库。

```
[root@db ~]# sudo -u postgres psql
could not change directory to "/root"
psql (9.2.18)
Type "help" for help.

postgres=# CREATE DATABASE hive;
CREATE DATABASE
postgres=# CREATE USER hive WITH PASSWORD '123';
CREATE ROLE
postgres=# GRANT ALL PRIVILEGES ON DATABASE hive TO hive;
GRANT
postgres=# \c hive
You are now connected to database "hive" as user "postgres".
hive=# CREATE SCHEMA hive AUTHORIZATION hive;
CREATE SCHEMA
hive=# ALTER SCHEMA hive OWNER TO hive;
ALTER SCHEMA
hive=# ALTER ROLE hive SET search_path to 'hive', 'public';
ALTER ROLE
```

```

hive=# \q
[root@db ~]# psql -U hive -d hive
psql (9.2.18)
Type "help" for help.

hive=> \l
                                         List of databases
   Name    |  Owner   | Encoding | Collate | Ctype | Access privileges
-----+-----+-----+-----+-----+-----+
ambari | postgres | UTF8     | en_US.utf8 | en_US.utf8 | =Tc
/postgres          +                               |
                     |           |           |           |           | pos
tgres=CTc/postgres+
                     |           |           |           |           | amb
ari=CTc/postgres
hive    | postgres | UTF8     | en_US.utf8 | en_US.utf8 | =Tc
/postgres          +                               |
                     |           |           |           |           | pos
tgres=CTc/postgres+
                     |           |           |           |           | hiv
e=CTc/postgres
postgres | postgres | UTF8     | en_US.utf8 | en_US.utf8 | 
template0 | postgres | UTF8     | en_US.utf8 | en_US.utf8 | =c/
postgres          +                               |
                     |           |           |           |           | pos
tgres=CTc/postgres
template1 | postgres | UTF8     | en_US.utf8 | en_US.utf8 | =c/
postgres          +                               |
                     |           |           |           |           | pos
tgres=CTc/postgres
(5 rows)

```

- 完善数据库连接配置，测试数据库连接是否正常。

Hive Metastore host: hive.bigdata.wh.com

Hive Database:

- New MySQL Database
- Existing MySQL / MariaDB Database
- Existing PostgreSQL Database
- Existing Oracle Database
- Existing SQL Anywhere Database

Database Name: hive

Database Username: hive

Database Password: Type password [RETYPE] Retype Password [RETYPE] *This is required*

JDBC Driver Class: org.postgresql.Driver

Database URL: jdbc:postgresql://hive.bigdata.wh.com:5432/hive

Test Connection

Hive Database Type: postgres

Connection Failed

- 观察连接失败日志可知错误原因是Server中还未设置postgres JDBC驱动程序和程序路径。

首先从本地源中下载postgresql-jdbc.jar数据库驱动包。

```
[root@server ~]# wget http://repo.bigdata.wh.com/resource/postgresql-jdbc.jar
--2017-05-16 09:00:31--  http://repo.bigdata.wh.com/resource/postgresql-jdbc.jar
Resolving repo.bigdata.wh.com (repo.bigdata.wh.com)... 192.168.3
6.247
Connecting to repo.bigdata.wh.com (repo.bigdata.wh.com)|192.168.
36.247|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 446067 (436K) [application/x-java-archive]
Saving to: 'postgresql-jdbc.jar'

100%[=====>] 446,067      --.- K/s   in 0.1s

2017-05-16 09:00:32 (3.82 MB/s) - 'postgresql-jdbc.jar' saved [4
46067/446067]
```

然后在Server中通过ambari-server setup命令来配置postgres驱动包。

```
[root@server ~]# ambari-server setup --jdbc-db=postgres --jdbc-driver=/root/postgresql-jdbc.jar
Using python /usr/bin/python
Setup ambari-server
Copying /root/postgresql-jdbc.jar to /var/lib/ambari-server/resources
If you are updating existing jdbc driver jar for postgres with postgresql-jdbc.jar.
Please remove the old driver jar, from all hosts. Restarting services that need the driver,
will automatically copy the new jar to the hosts.
JDBC driver was successfully initialized.
Ambari Server 'setup' completed successfully.
```

- 重新测试postgres数据库连接。

Database URL: jdbc:postgresql://db.bigdata.wh.com:5432/hive

Connection OK 

Ambari Metrics组件所必须的管理员密码设置。

General 1

Metrics Service operation mode	embedded	embedded
Metrics Collector log dir	/var/log/ambari-metrics-collector	/var/log/ambari-metrics-collector
Metrics Collector pid dir	/var/run/ambari-metrics-collector	/var/run/ambari-metrics-collector
Metrics Monitor log dir	/var/log/ambari-metrics-monitor	/var/log/ambari-metrics-monitor
Metrics Monitor pid dir	/var/run/ambari-metrics-monitor	/var/run/ambari-metrics-monitor
Grafana Admin Username	admin	admin
Grafana Admin Password	Type password	Retype Password  ...

SmartSense组件所必须得管理员密码设置

调整服务参数

Basic Data Capture Operations Gateway Activity Analysis 1 Advanced

Activity Explorer

UI port: 9060

Anonymous access allowed: No

⚠ Password for user 'admin':
Type pass: Retype Pass:

>Password for user 'admin':
Type pass: Retype Pass:

保存配置报告

Server在开始安装服务组件之前，会根据之前的配置，生成包含主机与组件对应关系，服务实例个数等信息的配置报告。我们可以选择直接打印纸质文档或是生成PDF电子文档。

- 配置报告预览，可直接打印或保存。

Review

Please review the configuration before installation

```

Admin Name: admin
Cluster Name: bigdata
Total Hosts: 8 (8 new)
Repositories:
  redhat7 (HDP-2.6):
    http://repo.bigdata.wh.com/hadoop/HDP/centos7/
  redhat7 (HDP-UTILS-1.1.0.21):
    http://repo.bigdata.wh.com/hadoop/HDP-UTILS-1.1.0.21/
Services:
  HDFS
    DataNode : 3 hosts
    NameNode : nn.bigdata.wh.com
    NFSGateway : 1 host
    SNameNode : snn.bigdata.wh.com
  YARN + MapReduce2
    App Timeline Server : nn.bigdata.wh.com
    NodeManager : 3 hosts
    ResourceManager : nn.bigdata.wh.com
  Tez
    Clients : 4 hosts
  
```

[← Back](#)
[Print](#)
[Deploy →](#)

- 开始部署后，执行部署准备命令来在Server中生成若干任务。

[← Back](#)
[Print](#)
Deploy →



Preparing to Deploy: 3 of 47 tasks completed.

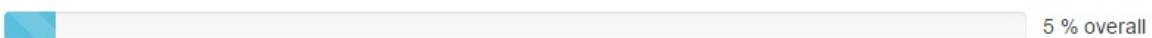
安装过程

该页面展示了在集群各主机上安装、启动以及测试每个组件的过程。

- 整个安装过程的总体状态显示在页面顶部的进度条中。

Install, Start and Test

Please wait while the selected services are installed and started.



- 各个主机各自的安装过程显示在页面下部的主机状态列表中。

Show: All (8) In Progress (8) Warning (0) Success (0) Fail (0)			
Host	Status	Message	
server.bigdata.wh.com	<div style="width: 23%;">23%</div>	Installing Metrics Monitor	
gw.bigdata.wh.com	<div style="width: 4%;">4%</div>	Installing Activity Analyzer	
nn.bigdata.wh.com	<div style="width: 5%;">5%</div>	Installing App Timeline Server	
snn.bigdata.wh.com	<div style="width: 4%;">4%</div>	Installing HDFS Client	
hive.bigdata.wh.com	<div style="width: 4%;">4%</div>	Installing HBase Master	
dn001.bigdata.wh.com	<div style="width: 5%;">5%</div>	Installing DataNode	
dn002.bigdata.wh.com	<div style="width: 5%;">5%</div>	Installing DataNode	
dn003.bigdata.wh.com	<div style="width: 5%;">5%</div>	Installing DataNode	

8 of 8 hosts showing - [Show All](#) Show: 25 ▾ 1 - 8 of 8 ⏪ ⏴ ⏵ ⏩ ⏹

- 单击相应主机Message列中的链接，通过弹出的任务窗口，可以查看该主机上任务的具体信息。

Message

- Installing Metrics Monitor
- Installing Activity Analyzer
- Installing App Timeline Server
- Installing HDFS Client**
- Installing HBase Master
- Installing DataNode
- Installing DataNode
- Installing DataNode

Tasks

- HDFS Client Install
- History Server Install
- HST Agent Install
- HST Server Install
- Metrics Monitor Install
- SNameNode Install
- Slider Client Install
- Tez Client Install

- 在任务弹出窗口中，单击单个任务以查看记录命令执行过程的日志文件。若任务执行失败，可逐条的查看日志来定位错误和分析原因。

server.bigdata.wh.com

Tasks

- HST Agent Install**
- Metrics Monitor Install

server.bigdata.wh.com

Tasks

HST Agent Install

stderr: /var/lib/ambari-agent/data/errors-308.txt

None

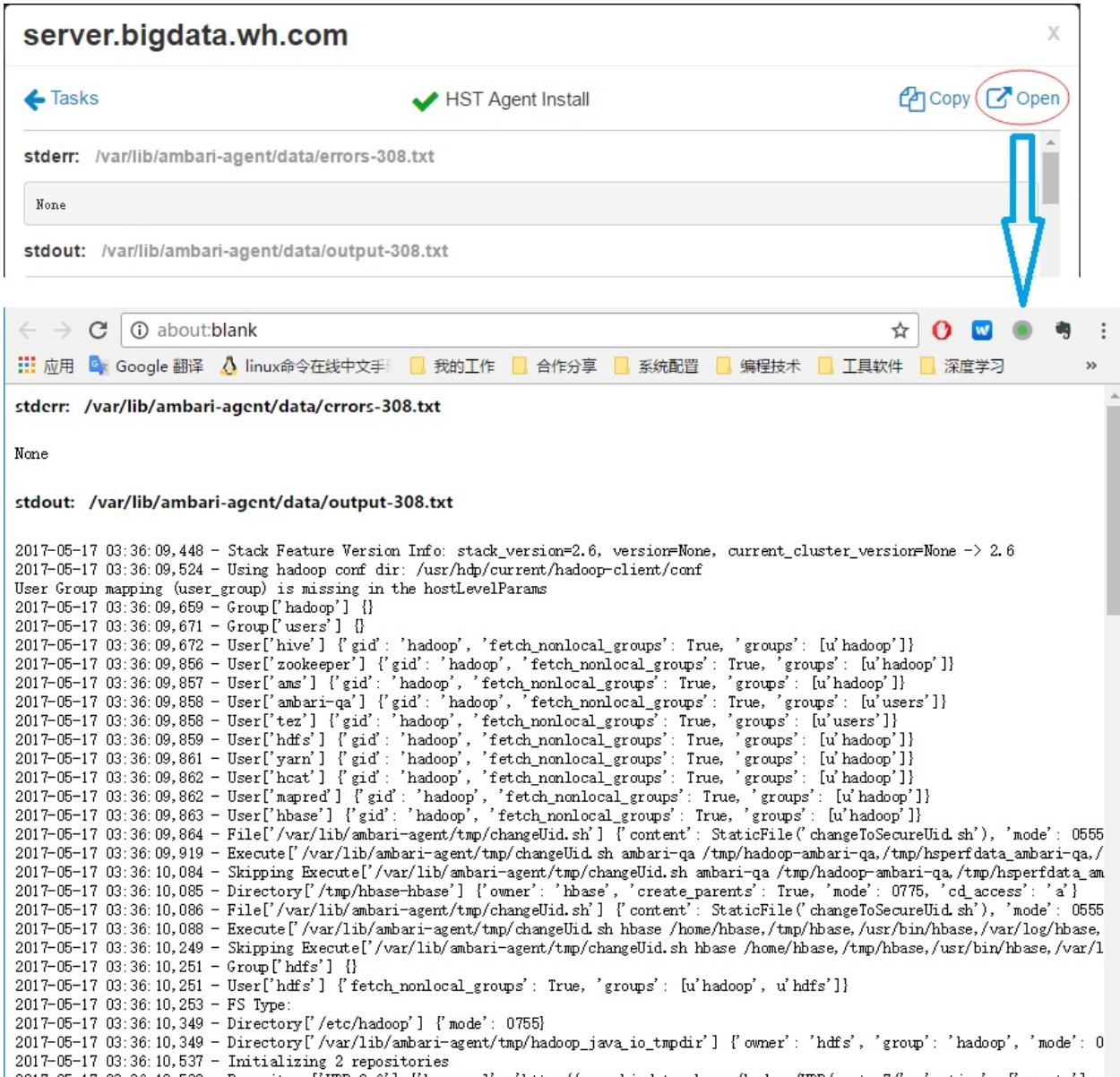
stdout: /var/lib/ambari-agent/data/output-308.txt

```

2017-05-17 03:36:09,448 - Stack Feature Version Info: stack_version=2.6, version=None, c
2017-05-17 03:36:09,524 - Using hadoop conf dir: /usr/hdp/current/hadoop-client/conf
User Group mapping (user_group) is missing in the hostLevelParams
2017-05-17 03:36:09,659 - Group['hadoop'] {}
2017-05-17 03:36:09,671 - Group['users'] {}
2017-05-17 03:36:09,672 - User['hive'] {'gid': 'hadoop', 'fetch_nonlocal_groups': True, 'c
2017-05-17 03:36:09,856 - User['zookeeper'] {'gid': 'hadoop', 'fetch_nonlocal_groups': T
2017-05-17 03:36:09,857 - User['ams'] {'gid': 'hadoop', 'fetch_nonlocal_groups': True, 'c
2017-05-17 03:36:09,858 - User['ambari-qe'] {'gid': 'hadoop', 'fetch_nonlocal_groups': T
2017-05-17 03:36:09,858 - User['tez'] {'gid': 'hadoop', 'fetch_nonlocal_groups': True, 'c
2017-05-17 03:36:09,859 - User['hdfs'] {'gid': 'hadoop', 'fetch_nonlocal_groups': True, 'c
2017-05-17 03:36:09,861 - User['yarn'] {'gid': 'hadoop', 'fetch_nonlocal_groups': True, 'c
2017-05-17 03:36:09,862 - User['heat'] {'gid': 'hadoop', 'fetch_nonlocal_groups': True, 'c
2017-05-17 03:36:09,862 - User['mapred'] {'gid': 'hadoop', 'fetch_nonlocal_groups': True,
2017-05-17 03:36:09,863 - User['hbase'] {'gid': 'hadoop', 'fetch_nonlocal_groups': True,

```

- 要查看更大版本的日志内容，请单击打开新的浏览器窗口来最大化查看。



安装完成

- 该页面展示了集群中已完成任务的摘要列表。

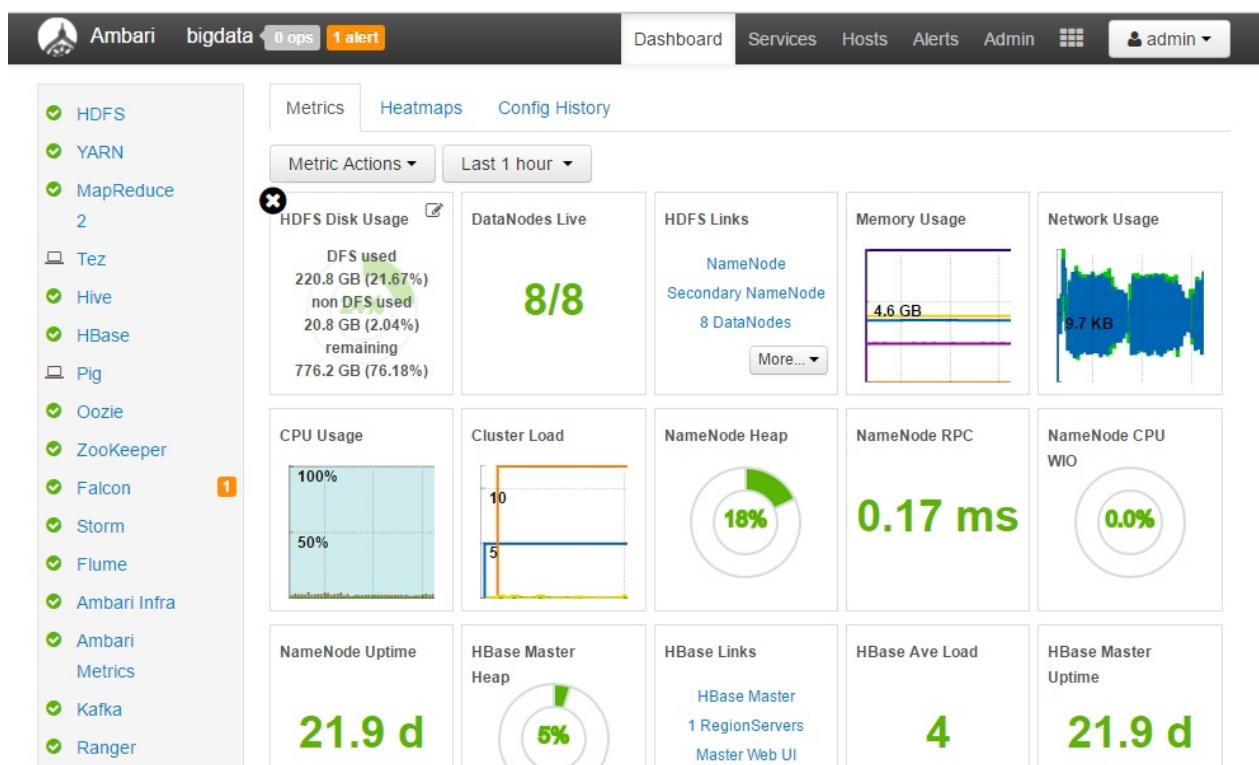
Summary

Here is the summary of the install process.

The cluster consists of 6 hosts
 Installed and started services successfully on 1 new host
 6 warnings
 Master services installed
 SNameNode installed on snn.bigdata.wh.com
 NameNode installed on nn.bigdata.wh.com
 ResourceManager installed on nn.bigdata.wh.com
 History Server installed on snn.bigdata.wh.com
 HiveServer2 installed on hive.bigdata.wh.com
 HBase Master installed on hive.bigdata.wh.com
 Oozie Server installed on gw.bigdata.wh.com
 Starting services failed

[Complete →](#)

- 单击完成按钮后页面会转跳到集群bigdata的管理主页。



安装完成
