# Ansible工具使用案例

## 9.1 实战案例——Ansible部署集群商城网站实战

### 9.1.1 案例目标

（1）掌握Ansible目录结构创建与使用

（2）掌握Ansible Playbook的编写与使用

（3）掌握使用Ansible工具部署商城集群网站

### 9.1.2 案例分析

#### 1.规划节点

Ansible服务的节点规划，见表1-1-1。

表1-1-1规划节点

|  |  |  |
| --- | --- | --- |
| **IP** | **主机名** | **节点** |
| 172.30.11.12 | ansible | Ansible节点 |
| 172.30.11.13 | node1 | Node1节点 |
| 172.30.11.14 | node2 | Node2节点 |
| 172.30.11.15 | node3 | Node3节点 |

#### 2.基础准备

使用OpenStack平台创建两台云主机进行实验，云主机镜像使用提供的CentOS\_7.5\_x86\_64\_XD.qcow2镜像，flavor使用1核/2G内存/20G硬盘，自行配置网络并使用远程连接工具连接云主机。节点规划表中的IP地址为作者的IP地址，在进行实操案例的时候，按照自己的环境规划网络与IP地址。Ansible节点安装好Ansible服务。然后做好Ansible节点对所有node节点的无秘钥访问操作。

### 9.1.3 案例实施

#### 1.环境准备

该实战案例为使用Ansible工具部署一个三节点的集群商城系统，和一个zabbix监控系统。

##### （1）配置IP并连接

配置4台虚拟机IP，配置四台虚拟机的主机名分别为ansible、node1、node2、node3并使用远程工具连接。

##### （2）Ansible节点安装Ansible服务

在Ansible节点使用提供的ansible.tar.gz软件包，配置本地镜像源，安装ansible服务，操作方法在上面的实战案例中有描述，此处不再赘述。

##### （3）配置无秘钥登录

配置Ansible节点无秘钥登录node节点。配置完无秘钥登录后，使用Ansible节点ssh连接测试。

#### 2.目录结构

##### （1）项目目录

首先在/opt目录下创建一个项目目录gpmall\_ansible，这个就是本次企业应用实战的项目目录，命令如下：

[root@ansible ~]# mkdir /opt/gpmall\_ansible

##### （2）创建角色

在创建roles角色目录之前，考虑将集群商城+监控案例拆分为多个roles执行，这样的话，Playbook易于编写和读懂。

安装集群商城+监控，使用init（基础环境）、mariadb（数据库）、redis（数据库）、zookeeper（协调服务）、kafka（消息队列）、jar（运行jar包）、nginx（前端服务）这些roles来完成。下面创建这些roles和相应的项目目录，具体命令如下：

[root@ansible ~]# mkdir -p /opt/gpmall\_ansible/roles/{init,mariadb,redis,zookeeper,kafka,jar,nginx,zabbix}/{tasks,files,templates,meta,handlers,vars}

[root@ansible ~]# tree /opt/gpmall\_ansible/

/opt/gpmall\_ansible/

└── roles

├── init

│   ├── files

│   ├── handlers

│   ├── meta

│   ├── tasks

│   ├── templates

│   └── vars

├── jar

│   ├── files

│   ├── handlers

│   ├── meta

│   ├── tasks

│   ├── templates

│   └── vars

├── kafka

│   ├── files

│   ├── handlers

│   ├── meta

│   ├── tasks

│   ├── templates

│   └── vars

├── mariadb

│   ├── files

│   ├── handlers

│   ├── meta

│   ├── tasks

│   ├── templates

│   └── vars

├── nginx

│   ├── files

│   ├── handlers

│   ├── meta

│   ├── tasks

│   ├── templates

│   └── vars

├── redis

│   ├── files

│   ├── handlers

│   ├── meta

│   ├── tasks

│   ├── templates

│   └── vars

├── zabbix

│   ├── files

│   ├── handlers

│   ├── meta

│   ├── tasks

│   ├── templates

│   └── vars

└── zookeeper

├── files

├── handlers

├── meta

├── tasks

├── templates

└── vars

在每个角色目录下面都有一样的目录，这些目录中的task目录一般是一定会用到的，其他的目录视情况而定来使用。

##### （3）创建group\_vars目录

在项目目录/opt/gpmall\_ansible下创建group\_vars目录，并在该目录下创建all文件，该目录用来存放变量声明文件all。命令如下：

[root@ansible ~]# cd /opt/gpmall\_ansible/

[root@ansible gpmall\_ansible]# mkdir group\_vars

[root@ansible gpmall\_ansible]# cd group\_vars/

[root@ansible group\_vars]# touch all

##### （4）创建安装入口文件

进入/opt/gpmall\_ansible目录，创建“install\_gpmall\_cluster.yaml”文件，该文件是安装动作的入口文件。命令如下：

[root@ansible gpmall\_ansible]# touch install\_gpmall\_cluster.yaml

[root@ansible gpmall\_ansible]# ll

total 0

drwxr-xr-x. 2 root root 17 Aug 26 21:31 group\_vars

-rw-r--r--. 1 root root 0 Aug 26 21:33 install\_gpmall\_cluster.yaml

drwxr-xr-x. 10 root root 114 Aug 26 21:18 roles

到目前为止，ansible的目录结构与文件创建完毕。接下来对每一个role角色进行剧本的编写。

#### 3.编写Playbook剧本

##### （1）init角色

该角色执行的任务是用来部署三个node节点的基础环境，包括关闭防火墙，SELinux，配置yum源，安装Java JDK环境。在roles/init/tasks目录下，创建main.yaml文件，文件的内容如下：

- name: Selinux Config Setenforce

shell: getenforce

register: info

- name: when\_Selinux

shell: setenforce 0

when: info['stdout'] == 'Enforcing'

- name: stop firewalld

shell: systemctl stop firewalld

ignore\_errors: yes

- name: copy packages

copy: src=gpmall-repo.tar.gz dest=/opt

- name: tar gpmall

shell: tar -zxvf /opt/gpmall-repo.tar.gz -C /opt/

- name: move repos

shell: mv /etc/yum.repos.d/\* /media

- name: create local.repo

copy: src=local.repo dest=/etc/yum.repos.d/

- name: install jdk

yum: name={{ item }} state=present

with\_items:

- java-1.8.0-openjdk

- java-1.8.0-openjdk-devel

- name: copy hosts

template: src=hosts.j2 dest=/etc/hosts

该剧本用到了copy和template模块，copy模块使用的文件及镜像包，放入tasks同级目录的files目录下；template模块使用的Jinja2文件，放入tasks同级目录的templates目录下。

在该init角色剧本中，即把gpmall-repo.tar.gz镜像和local.repo文件拷贝至init/files目录下，把hosts.j2文件拷贝至init/templates目录下。下面贴出local.repo和host.j2的文件内容：

# cat local.repo

[gpmall]

name=gpmall

baseurl=file:///opt/gpmall-repo

gpgcheck=0

enabled=1

# cat hosts.j2

127.0.0.1 localhost localhost.localdomain localhost4 localhost4.localdomain4

::1 localhost localhost.localdomain localhost6 localhost6.localdomain6

{{hostip1}} {{hostname1}}

{{hostip2}} {{hostname2}}

{{hostip3}} {{hostname3}}

因为设置的变量，所以需要在/opt/gpmall\_ansible/group\_vars/all中声明变量，all文件内容如下：

hostip1: 172.30.11.13

hostip2: 172.30.11.14

hostip3: 172.30.11.15

hostname1: node1

hostname2: node2

hostname3: node3

至此，init角色剧本编写完成。

##### （2）mariadb角色

该角色的作用是在三个节点上安装数据库服务并配置成数据库MariaDB Galera Cluster集群，在roles/mariadb/tasks目录下，创建main.yaml文件，文件的内容如下：

- name: install mariadb

yum: name=mariadb-server state=present

- name: start mariadb

shell: systemctl start mariadb

- name: enable mariadb

shell: systemctl enable mariadb

- name: install expect

yum: name=expect state=present

- name: mysql\_secure\_installation

template: src=mysql\_secure\_installation.sh.j2 dest=/opt/mysql\_secure\_installation.sh

- name: sh shell

shell: sh /opt/mysql\_secure\_installation.sh

- name: config mariadb

template: src=server.cnf.j2 dest=/etc/my.cnf.d/server.cnf

- name: grant privileges

shell: mysql -uroot -p{{ DB\_PASS }} -e "grant all privileges on \*.\* to 'root'@'%' identified by '{{ DB\_PASS }}';"

- name: stop db

shell: systemctl stop mariadb

- name: new cluster

shell: galera\_new\_cluster

when: ansible\_fqdn=="node1"

- name: restart db

shell: systemctl restart mariadb

when: ansible\_fqdn=="node2"

- name: restart db

shell: systemctl restart mariadb

when: ansible\_fqdn=="node3"

在该mariadb角色剧本中，用到了template模块，需要把server.cnf.j2和mysql\_secure\_installation.sh.j2文件放入tasks同级别的templates目录下，下面贴出两个j2文件内容：

# cat server.cnf.j2

#

# These groups are read by MariaDB server.

# Use it for options that only the server (but not clients) should see

#

# See the examples of server my.cnf files in /usr/share/mysql/

#

# this is read by the standalone daemon and embedded servers

[server]

# this is only for the mysqld standalone daemon

[mysqld]

#

# \* Galera-related settings

#

[galera]

wsrep\_on=ON

wsrep\_provider=/usr/lib64/galera/libgalera\_smm.so

wsrep\_cluster\_address="gcomm://{{hostip1}},{{hostip2}},{{hostip3}}"

{% if ansible\_fqdn == "node1" %}

wsrep\_node\_name= {{hostname1}}

wsrep\_node\_address={{hostip1}}

{% elif ansible\_fqdn == "node2" %}

wsrep\_node\_name= {{hostname2}}

wsrep\_node\_address={{hostip2}}

{% else %}

wsrep\_node\_name= {{hostname3}}

wsrep\_node\_address={{hostip3}}

{% endif %}

binlog\_format=row

default\_storage\_engine=InnoDB

innodb\_autoinc\_lock\_mode=2

wsrep\_slave\_threads=1

innodb\_flush\_log\_at\_trx\_commit=0

innodb\_buffer\_pool\_size=120M

wsrep\_sst\_method=rsync

wsrep\_causal\_reads=ON

#

# Allow server to accept connections on all interfaces.

#

{% if ansible\_fqdn == "node1" %}

bind-address={{hostip1}}

{% elif ansible\_fqdn == "node2" %}

bind-address={{hostip2}}

{% else %}

bind-address={{hostip3}}

{% endif %}

#

# Optional setting

#wsrep\_slave\_threads=1

#innodb\_flush\_log\_at\_trx\_commit=0

# this is only for embedded server

[embedded]

# This group is only read by MariaDB servers, not by MySQL.

# If you use the same .cnf file for MySQL and MariaDB,

# you can put MariaDB-only options here

[mariadb]

# This group is only read by MariaDB-10.3 servers.

# If you use the same .cnf file for MariaDB of different versions,

# use this group for options that older servers don't understand

[mariadb-10.3]

server.cnf.j2文件为数据库的配置文件，因需要确保各个节点的配置，需要使用变量。

# mysql\_secure\_installation.sh.j2

#!/bin/bash

#检查是否是root用户执行

if [ $(id -u) != "0" ]; then

echo "Error: You must be root to run this script, please use root to install"

exit 1

fi

expect -c "

spawn /usr/bin/mysql\_secure\_installation

expect \"Enter current password for root (enter for none):\"

send \"\r\"

expect \"Set root password?\"

send \"y\r\"

expect \"New password:\"

send \"{{DB\_PASS}}\r\"

expect \"Re-enter new password:\"

send \"{{DB\_PASS}}\r\"

expect \"Remove anonymous users?\"

send \"y\r\"

expect \"Disallow root login remotely?\"

send \"n\r\"

expect \"Remove test database and access to it?\"

send \"y\r\"

expect \"Reload privilege tables now?\"

send \"y\r\"

expect eof 与spawn结束交互

"

mysql\_secure\_installation.sh.j2文件为一个shell脚本，用来初始化数据库，脚本中使用了DB\_PASS变量，所以需要在all文件中声明，编辑/opt/gpmall\_ansible/group\_vars/all文件，添加一条变量声明，内容如下：

# cat all

hostip1: 172.30.11.13

hostip2: 172.30.11.14

hostip3: 172.30.11.15

hostname1: node1

hostname2: node2

hostname3: node3

DB\_PASS: 123456

至此，mariadb角色剧本编写完毕。

##### （3）redis角色

该角色的作用是安装redis服务并配置启动，在roles/redis/tasks目录下，创建main.yaml文件，文件的内容如下：

- name: install redis

yum: name=redis state=present

when: ansible\_fqdn=="node1"

- name: config redis

copy: src=redis.conf dest=/etc/redis.conf

when: ansible\_fqdn=="node1"

- name: start redis

shell: systemctl restart redis

when: ansible\_fqdn=="node1"

该剧本使用了copy模块，故需要将redis.conf文件拷贝至tasks目录同级别的files目录下，因redis.conf文件内容过长，此处不再贴出。（可通过提供的文件查看）

至此，redis角色剧本编写完毕。

##### （4）zookeeper角色

该角色的作用是在三个节点上安装zookeeper服务并配置为zookeeper集群，在roles/zookeeper/tasks目录下，创建main.yaml文件，文件的内容如下：

- name: copy zookeeper.tar.gz

copy: src=zookeeper-3.4.14.tar.gz dest=/opt

- name: tar zookeeper

shell: tar -zxvf /opt/zookeeper-3.4.14.tar.gz -C /opt

- name: delete zoo\_sample.cfg

shell: rm -rf /opt/zookeeper-3.4.14/conf/zoo\_sample.cfg

- name: config zoo.cfg

template: src=zoo.cfg.j2 dest=/opt/zookeeper-3.4.14/conf/zoo.cfg

- name: mkdir /tmp/zookeeper

shell: mkdir -p /tmp/zookeeper

- name: config myid

template: src=myid.j2 dest=/tmp/zookeeper/myid

- name: start zookeeper

shell: sh /opt/zookeeper-3.4.14/bin/zkServer.sh start

该剧本使用了copy和template模块，故需要将zookeeper-3.4.14.tar.gz镜像包拷贝至tasks目录同级别的files目录下；将zoo.cfg.j2文件和myid.j2文件拷贝至templates下，下面贴出两个j2文件的内容：

# cat myid.j2

{% if ansible\_fqdn == "node1" %}

1

{% elif ansible\_fqdn == "node2" %}

2

{% else %}

3

{% endif %}

# cat zoo.cfg.j2

# The number of milliseconds of each tick

tickTime=2000

# The number of ticks that the initial

# synchronization phase can take

initLimit=10

# The number of ticks that can pass between

# sending a request and getting an acknowledgement

syncLimit=5

# the directory where the snapshot is stored.

# do not use /tmp for storage, /tmp here is just

# example sakes.

dataDir=/tmp/zookeeper

# the port at which the clients will connect

clientPort=2181

# the maximum number of client connections.

# increase this if you need to handle more clients

#maxClientCnxns=60

#

# Be sure to read the maintenance section of the

# administrator guide before turning on autopurge.

#

# http://zookeeper.apache.org/doc/current/zookeeperAdmin.html#sc\_maintenance

#

# The number of snapshots to retain in dataDir

#autopurge.snapRetainCount=3

# Purge task interval in hours

# Set to "0" to disable auto purge feature

#autopurge.purgeInterval=1

server.1={{hostip1}}:2888:3888

server.2={{hostip2}}:2888:3888

server.3={{hostip3}}:2888:3888

至此，zookeeper角色剧本编写完毕。

##### （5）kafka角色

该角色的作用是在三个节点上安装kafka服务并配置为kafka集群，在roles/kafka/tasks目录下，创建main.yaml文件，文件的内容如下：

- name: copy kafka.tgz

copy: src=kafka\_2.11-1.1.1.tgz dest=/opt

- name: tar kafka

shell: tar -zxvf /opt/kafka\_2.11-1.1.1.tgz -C /opt

- name: config

template: src=server.properties.j2 dest=/opt/kafka\_2.11-1.1.1/config/server.properties

- name: start kafka

shell: sh /opt/kafka\_2.11-1.1.1/bin/kafka-server-start.sh -daemon /opt/kafka\_2.11-1.1.1/config/server.properties

该剧本使用了copy和template模块，故需要将kafka\_2.11-1.1.1.tgz镜像包拷贝至tasks目录同级别的files目录下；将server.properties.j2文件拷贝至templates下，因server.properties.j2文件内容过长，此处不再贴出（可通过提供的文件查看）。

至此，kafka角色剧本编写完毕。

##### （6）jar角色

该角色的作用是在node2和node3节点上启动应用商城的jar包，在roles/jar/tasks目录下，创建main.yaml文件，文件的内容如下：

- name: copy sql

copy: src=gpmall.sql dest=/opt/gpmall.sql

when: ansible\_fqdn=="node1"

- name: config db

shell: mysql -uroot -p{{ DB\_PASS }} -e "create database gpmall;"

when: ansible\_fqdn=="node1"

- name: source gpmall.sql

shell: mysql -uroot -p{{ DB\_PASS }} gpmall < /opt/gpmall.sql

when: ansible\_fqdn=="node1"

- name: copy jar

copy: src=user-provider-0.0.1-SNAPSHOT.jar dest=/opt

when: ansible\_fqdn=="node2"

- name: copy jar

copy: src=user-provider-0.0.1-SNAPSHOT.jar dest=/opt

when: ansible\_fqdn=="node3"

- name: copy jar

copy: src=shopping-provider-0.0.1-SNAPSHOT.jar dest=/opt

when: ansible\_fqdn=="node2"

- name: copy jar

copy: src=shopping-provider-0.0.1-SNAPSHOT.jar dest=/opt

when: ansible\_fqdn=="node3"

- name: copy jar

copy: src=gpmall-shopping-0.0.1-SNAPSHOT.jar dest=/opt

when: ansible\_fqdn=="node2"

- name: copy jar

copy: src=gpmall-shopping-0.0.1-SNAPSHOT.jar dest=/opt

when: ansible\_fqdn=="node3"

- name: copy jar

copy: src=gpmall-user-0.0.1-SNAPSHOT.jar dest=/opt

when: ansible\_fqdn=="node2"

- name: copy jar

copy: src=gpmall-user-0.0.1-SNAPSHOT.jar dest=/opt

when: ansible\_fqdn=="node3"

- name: copy hosts

template: src=hosts.j2 dest=/etc/hosts

when: ansible\_fqdn=="node2"

- name: copy hosts

template: src=hosts.j2 dest=/etc/hosts

when: ansible\_fqdn=="node3"

- name: start jar

shell: nohup java -jar /opt/user-provider-0.0.1-SNAPSHOT.jar &

when: ansible\_fqdn=="node2"

- name: start jar

shell: nohup java -jar /opt/user-provider-0.0.1-SNAPSHOT.jar &

when: ansible\_fqdn=="node3"

- name: start jar

shell: nohup java -jar /opt/shopping-provider-0.0.1-SNAPSHOT.jar &

when: ansible\_fqdn=="node2"

- name: start jar

shell: nohup java -jar /opt/shopping-provider-0.0.1-SNAPSHOT.jar &

when: ansible\_fqdn=="node3"

- name: start jar

shell: nohup java -jar /opt/gpmall-shopping-0.0.1-SNAPSHOT.jar &

when: ansible\_fqdn=="node2"

- name: start jar

shell: nohup java -jar /opt/gpmall-shopping-0.0.1-SNAPSHOT.jar &

when: ansible\_fqdn=="node3"

- name: start jar

shell: nohup java -jar /opt/gpmall-user-0.0.1-SNAPSHOT.jar &

when: ansible\_fqdn=="node2"

- name: start jar

shell: nohup java -jar /opt/gpmall-user-0.0.1-SNAPSHOT.jar &

when: ansible\_fqdn=="node3"

同样的，该剧本使用了copy和template模块，需要将gpmall.sql和四个gpmall商城的jar包拷贝至tasks目录同级别的files目录下；将hosts.j2文件拷贝至templates目录下。下面贴出host.j2文件内容：

# cat hosts.j2

127.0.0.1 localhost localhost.localdomain localhost4 localhost4.localdomain4

::1 localhost localhost.localdomain localhost6 localhost6.localdomain6

{{hostip1}} redis.mall

{{hostip1}} mysql.mall

{{hostip1}} kafka1.mall

{{hostip2}} kafka2.mall

{{hostip3}} kafka3.mall

{{hostip1}} zk1.mall

{{hostip2}} zk2.mall

{{hostip3}} zk3.mall

至此，jar角色剧本编写完毕。

##### （7）nginx角色

该角色的作用是在node1节点上安装nginx服务，部署前端，并使用ip轮询的方式配置负载均衡，在roles/nginx/tasks目录下，创建main.yaml文件，文件的内容如下：

- name: install nginx

yum: name=nginx state=present

when: ansible\_fqdn=="node1"

- name: delete html

shell: rm -rf /usr/share/nginx/html/\*

when: ansible\_fqdn=="node1"

- name: copy file

copy: src=dist dest=/opt

when: ansible\_fqdn=="node1"

- name: cp dist

shell: cp -rvf /opt/dist/\* /usr/share/nginx/html/

when: ansible\_fqdn=="node1"

- name: config default.config

template: src=default.conf.j2 dest=/etc/nginx/conf.d/default.conf

when: ansible\_fqdn=="node1"

- name: start nginx

service: name=nginx state=started enabled=true

when: ansible\_fqdn=="node1"

该剧本使用了copy和template模块，故需要dist目录拷贝至tasks目录同级别的files目录下；将default.conf.j2文件拷贝至templates下。下面贴出default.conf.j2文件内容：

upstream myuser {

server {{hostip2}}:8082;

server {{hostip3}}:8082;

ip\_hash;

}

upstream myshopping {

server {{hostip2}}:8081;

server {{hostip3}}:8081;

ip\_hash;

}

upstream mycashier {

server {{hostip2}}:8083;

server {{hostip3}}:8083;

ip\_hash;

}

server {

listen 80;

server\_name localhost;

#charset koi8-r;

#access\_log /var/log/nginx/host.access.log main;

location / {

root /usr/share/nginx/html;

index index.html index.htm;

}

location /user {

proxy\_pass http://myuser;

}

location /shopping {

proxy\_pass http://myshopping;

}

location /cashier {

proxy\_pass http://mycashier;

}

#error\_page 404 /404.html;

# redirect server error pages to the static page /50x.html

#

error\_page 500 502 503 504 /50x.html;

location = /50x.html {

root /usr/share/nginx/html;

}

# proxy the PHP scripts to Apache listening on 127.0.0.1:80

#

#location ~ \.php$ {

# proxy\_pass http://127.0.0.1;

#}

# pass the PHP scripts to FastCGI server listening on 127.0.0.1:9000

#

#location ~ \.php$ {

# root html;

# fastcgi\_pass 127.0.0.1:9000;

# fastcgi\_index index.php;

# fastcgi\_param SCRIPT\_FILENAME /scripts$fastcgi\_script\_name;

# include fastcgi\_params;

#}

# deny access to .htaccess files, if Apache's document root

# concurs with nginx's one

#

#location ~ /\.ht {

# deny all;

#}

}

至此，nginx角色剧本编写完毕。

#### 4.执行Playbook

通过上面八个剧本的编写，完成了gpmall集群商城的一键部署。不过在执行剧本之前，还有一些工作需要完成，具体任务如下：

##### （1）修改hosts文件

此处需要修改的hosts文件并不只是/etc/hosts，还有一个/etc/ansible/hosts文件，编辑/etc/ansible/hosts，在文件的最后添加需要执行剧本的目标主机组，添加的内容如下：

## db-[99:101]-node.example.com

[hosts]

node1

node2

node3

编辑/etc/hosts文件，添加ip与主机名的映射，如下所示：

172.30.11.13 node1

172.30.11.14 node2

172.30.11.15 node3

##### （2）编辑剧本入口文件

install\_gpmall\_cluster.yaml文件为执行剧本的入口文件，需要将调用roles的顺序及哪些主机调用哪些roles在这个文件中体现出来，install\_gpmall\_cluster.yaml文件的具体内容如下：

---

- hosts: hosts

remote\_user: root

roles:

- init

- mariadb

- redis

- zookeeper

- kafka

- jar

- nginx

##### （3）执行剧本

当所有准备工作都完成之后，使用ansible-playbook命令执行剧本，首先使用--syntax-check参数检测脚本的语法，命令如下：

[root@ansible gpmall\_ansible]# ansible-playbook install\_gpmall\_cluster.yaml --syntax-check

playbook: install\_gpmall\_cluster.yaml

直接返回文件名，表示脚本没有语法错误。执行剧本，命令如下：

[root@ansible gpmall\_ansible]# ansible-playbook install\_gpmall\_cluster.yaml

PLAY [hosts] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

TASK [Gathering Facts] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [node2]

ok: [node1]

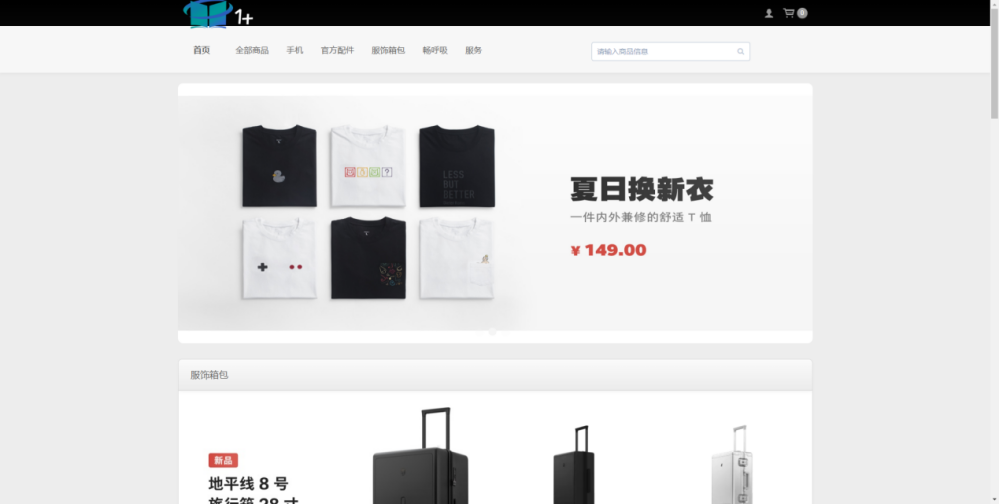
ok: [node3]

...

...

在等待一段时间之后，剧本执行完毕，若没有报错，访问node1节点的ip地址可以访问集群商城界面。

访问商城界面如下：



至此，使用Ansible一键部署gpmall集群商城完成。在企业项目的全生命管理周期中，使用合适的工具（Ansible）部署企业应用。