TiDB 安装测试

TiDB 的安装部署

部署机器及服务环境

IP	services	操作系统	机器配置
192.168.1.213	TiDB, PD , TiKV	CentOS7.3	8C 8G 50G 普通磁盘（虚拟机）
192.168.1.214	TiDB, PD , TiKV	CentOS7.3	8C 8G 50G 普通磁盘（虚拟机）
192.168.1.215	TiDB, PD , TiKV	CentOS7.3	8C 8G 50G 普通磁盘（虚拟机）
192.168.1.212	monitor, haproxy	CentOS7.3	4C 4G 50G 普通磁盘（虚拟机）

NTP 时间同步

```
# 所有机器做NTP时间同步
systemctl status ntpd.service
systemctl stop ntpd.service
ntpdate pool.ntp.org
systemctl start ntpd.service
ntpstat
    synchronised to NTP server (182.92.12.11) at stratum 3
       time correct to within 43 ms
       polling server every 512 s
```

中控机部署

```
## 中控机上建立tidb用户
# useradd tidb
# passwd tidb
# su - tidb
$ ssh-keygen -t rsa

(root 执行)
# 下载tidb-ansible
    git clone -b release-2.0 https://github.com/pingcap/tidb-ansible.git
# 中控机安装 Ansible 及其依赖
    yum -y install epel-release
    yum -y install python-pip curl
    pip install -r tidb-ansible/requirements.txt
# 安装sshpass ,在批量创建tidb用户的时候使用
    yum -y install sshpass

## ansible 配置集群机器的ssh互信
vim hosts.ini

    [servers]
```

```
    192.168.1.212
    192.168.1.213
    192.168.1.214
    192.168.1.215

    [all:vars]
    username = tidb
    ntp_server = pool.ntp.org
## 中控机上执行 root 用户
# ansible-playbook -i hosts.ini create_users.yml -k
## 在所有的机器上添加tidb 用户，配置sudo 免密，配置机器之间免密互通

## 查看用户是否互通
[tidb@TiDB01 tidb-ansible]$ ansible -i hosts.ini all -m shell -a 'whoami'
192.168.1.213 | SUCCESS | rc=0 >>
tidb
192.168.1.215 | SUCCESS | rc=0 >>
tidb
192.168.1.214 | SUCCESS | rc=0 >>
tidb
[tidb@TiDB01 tidb-ansible]$ ansible -i hosts.ini all -m shell -a 'whoami' -b
192.168.1.213 | SUCCESS | rc=0 >>
root
192.168.1.215 | SUCCESS | rc=0 >>
root
192.168.1.214 | SUCCESS | rc=0 >>
root
```

monitor 机器生成PDF 依赖包

```
yum install fontconfig    ## 本例中为192.168.1.212
```

数据盘 ext4 的挂载参数

```
# xfs 亦可
# vi /etc/fstab
/dev/nvme0n1 /data1 ext4 defaults,nodelalloc,noatime 0 2
```

编辑 inventory.ini 文件

```
[tidb_servers]
192.168.1.213
192.168.1.214
192.168.1.215
[tikv_servers]
192.168.1.213
192.168.1.214
192.168.1.215
[pd_servers]

192.168.1.213
```

```
192.168.1.214
192.168.1.215
[spark_master]
[spark_slaves]
[monitoring_servers]
192.168.1.212
[grafana_servers]
192.168.1.212
[monitored_servers]
192.168.1.212
192.168.1.213
192.168.1.214
192.168.1.215
[alertmanager_servers]
192.168.1.212
[kafka_exporter_servers]
[pump_servers:children]
tidb_servers
[drainer_servers]
[pd_servers:vars]
[all:vars]
deploy_dir = /alidata/tidb/deploy   ## 配置分发目录
ansible_user = tidb         ## 本例中使用tidb 启动集群
cluster_name = GJDB-cluster
tidb_version = v2.0.1        ## 默认为 latest
process_supervision = systemd
timezone = Asia/Shanghai
set_timezone = True
enable_firewalld = False
enable_ntpd = True
set_hostname = False
enable_binlog = False
zookeeper_addrs = ""
kafka_addrs = ""
enable_slow_query_log = False
enable_tls = False
deploy_without_tidb = False
alertmanager_target = ""
grafana_admin_user = "admin"
grafana_admin_password = "admin"
```

下载TiDB binary 到中控机

```
ansible-playbook local_prepare.yml
```

初始化集群机器的环境，修改内核参数，检查机器的配置

```
## 修改检测条件
tidb-ansible/roles/check_system_optional/defaults/main.yml
    ---
```

```
    # CPU
    tidb_min_cpu: 8
    tikv_min_cpu: 8
    pd_min_cpu: 4
    monitor_min_cpu: 4

    # Mem
    tidb_min_ram: 8000      ## 默认的配置为16G
    tikv_min_ram: 8000
    pd_min_ram: 8000
    monitor_min_ram: 8000

    # Disk
    tidb_min_disk: 500000000000
    tikv_min_disk: 500000000000
    pd_min_disk: 200000000000
    monitor_min_disk: 500000000000

  tidb-ansible/bootstrap.yml
    #- name: tikv_servers machine benchmark   ## 去除磁盘性能测试
    #  hosts: tikv_servers
    #  gather_facts: false
    #  roles:
    #    - { role: machine_benchmark, when: not dev_mode }

 ## 修改系统参数，初始化集群环境
 ansible-playbook bootstrap.yml
```

部署TiDB集群

```
ansible-playbook deploy.yml
```

启动集群

```
ansible-playbook start.yml
```

haproxy 配置

```
# 安装haproxy
yum -y install haproxy
# 记录haproxy 日志
vim /etc/rsyslog.conf
    # Provides UDP syslog reception
    $ModLoad imudp
    $UDPServerRun 514

    # Provides TCP syslog reception
    $ModLoad imtcp
    $InputTCPServerRun 514
```

```
    local2.* /var/log/haproxy.log

# 配置haproxy.cfg
    global
        log         127.0.0.1 local2
        chroot      /var/lib/haproxy
        pidfile     /var/run/haproxy.pid
        maxconn     4000
        user        haproxy
        group       haproxy
        daemon
        # turn on stats unix socket
        stats socket /var/lib/haproxy/stats
    defaults
        mode                    tcp
        log                     global
        option                  tcplog
        option                  dontlognull
        retries                 3
        timeout http-request    10s
        timeout queue           1m
        timeout connect         10s
        timeout client          1m
        timeout server          1m
        timeout http-keep-alive 10s
        timeout check           10s
        maxconn                 3000
    listen stats
        mode http
        bind 0.0.0.0:1080
        stats enable
        stats hide-version
        stats uri     /haproxyadmin?stats
        stats realm   Haproxy\ Statistics
        stats auth    admin:admin
        stats admin if TRUE

    frontend mysql
        bind *:3306
        mode tcp
        log global
        default_backend tidb-servers

    backend tidb-servers
        balance leastconn
        server tidb01 192.168.1.213:4000 check port 4000 rise 1 fall 2 maxconn 3000
        server tidb02 192.168.1.214:4000 check port 4000 rise 1 fall 2 maxconn 3000
        server tidb03 192.168.1.215:4000 check port 4000 rise 1 fall 2 maxconn 3000
```

连接tidb 集群

```
mysql -u root -P 3306 -h 192.168.1.212         # 默认root密码为空
```

连接监控grafana

```
192.168.1.212:3000            # admin/admin
# 监控架构
在集群的每台机器上，有node_exporter 进程收集 tidb服务以及，机器的信息，push 到PushGateWay 服务，
Prometheus 拉取 PushGateWay 的数据进行存储，grafana 查询 Prometheus 后 在web 页面进行展示，报警由
alertManager 服务进行
```

数据的导出、导入

```
# 使用 tidb-tools 中的mydumper(或自己安装的mydumper)进行数据的导出，多线程导出
./bin/mydumper -h 192.168.1.212 -P 3306 -u root -t 16 -F 64 -B test -T t1,t2 --skip-tz-utc -o
./var/test
# 使用 tidb-tools 中的 loader 进行数据库的导入
./bin/loader -u root -P 3306 -h 192.168.1.212 -t 32 -d ./var/test ## sql 放入 ./  目录下
```

sysbench 性能测试

```
# 拉取测试 工具及脚本
 git clone https://github.com/pingcap/tidb-bench.git
# conf.sh 配置
    host=192.168.1.212
    port=3306
    user=root
    password=''
    tcount=16
    tsize=1000000
    threads=256
    dbname=sbtest

    # report interval
    interval=10

    # max time in seconds
    maxtime=600

    # just large enough to fit maxtime
    requests=2000000000

    driver=mysql

# lua 测试脚本
    [root@R730 sysbench]# ls lua-tests/db/
    bulk_insert.lua  insert.lua   Makefile.in      parallel_prepare.lua
select_random_ranges.lua
    common.lua       Makefile     oltp.lua         select.lua                update_index.lua
    delete.lua       Makefile.am  oltp_simple.lua  select_random_points.lua
update_non_index.lua

# 数据准备
```

```
    #!/bin/bash
    set -x
    . ./conf.sh

    if [[ ${password} = "" ]];
    then
        mysql -h ${host} -P ${port} -u${user} -e "CREATE DATABASE IF NOT EXISTS ${dbname}"
    else
        mysql -h ${host} -P ${port} -u${user} -p${password} -e "CREATE DATABASE IF NOT EXISTS
${dbname}"
    fi

    sysbench --test=./lua-tests/db/oltp.lua --db-driver=${driver} --mysql-host=${host} --mysql-
port=${port} \
     --mysql-user=${user} --mysql-password=${password} --mysql-db=${dbname} \
     --oltp-tables-count=${tcount} --oltp-table-size=${tsize} --rand-init=on prepare

 # oltp 基准测试
    #!/bin/bash
    set -x
    . ./conf.sh

    # run
    sysbench --test=./lua-tests/db/oltp.lua --db-driver=${driver} --mysql-host=${host} --mysql-
port=${port} \
      --mysql-user=${user} --mysql-password=${password} --mysql-db=${dbname} \
      --oltp-tables-count=${tcount} --oltp-table-size=${tsize} \
      --num-threads=${threads} --max-requests=${requests} \
      --oltp-read-only=off --report-interval=${interval} --rand-type=uniform \
      --max-time=${maxtime} --percentile=95 run

 # 查询基准测试
    #!/bin/bash
    set -x
    . ./conf.sh

    sysbench --test=./lua-tests/db/select.lua --db-driver=${driver} --mysql-host=${host} --
mysql-port=${port} \
     --mysql-user=${user} --mysql-password=${password} --mysql-db=${dbname} \
     --oltp-tables-count=${tcount} --oltp-table-size=${tsize} \
     --num-threads=${threads} --report-interval=${interval} \
     --max-requests=${requests} --percentile=95  --max-time=${maxtime} run

 # insert 测试
    #!/bin/bash
    set -x
    . ./conf.sh

    sysbench --test=./lua-tests/db/insert.lua --db-driver=${driver} --mysql-host=${host} --
mysql-port=${port} \
     --mysql-user=${user} --mysql-password=${password} --mysql-db=${dbname} \
     --oltp-tables-count=${tcount} --oltp-table-size=${tsize} \

     --num-threads=${threads} --report-interval=${interval} \
```

```
     --max-requests=${requests} --percentile=95 --max-time=${maxtime} run

 # 删除测试
    #!/bin/bash
    set -x
    . ./conf.sh

    sysbench --test=./lua-tests/db/delete.lua --db-driver=${driver} --mysql-host=${host} --
mysql-port=${port} \
     --mysql-user=${user} --mysql-password=${password} --mysql-db=${dbname} \
     --oltp-tables-count=${tcount} --oltp-table-size=${tsize} \
     --num-threads=${threads} --report-interval=${interval} \
     --max-requests=${requests} --max-time=${maxtime} --percentile=95 run
```

TPCC 测试

```
# 设置TiDB
stmt-count-limit = 100000000
# 编译
cd /usr/local/sysbench/tidb-bench/tpcc/src ; make
# 重启tidb
 ansible-playbook stop.yml --tags tidb
 ansible-playbook start.yml --tags tidb
# 创建数据库
mysqladmin create tpcc
mysql tpcc < create_table.sql
mysql tpcc < add_fkey_idx.sql
# 填充数据
./tpcc_load -h192.168.1.212 -P3306 -d tpcc -u root -p "" -w 1000
|hostname| |port| |dbname| |user| |password| |WAREHOUSES|
# 进行测试
./tpcc_start -h192.168.1.212 -P3306 -d tpcc -uroot -p "" -w1000 -c32 -r10 -l10800
|hostname| |port| |dbname| |user| |password| |WAREHOUSES| |CONNECTIONS| |WARMUP TIME| |BENCHMARK
TIME|
```