

## Table of Contents

前言	1.1
TiDB 运维手册 SOP 系列	1.2
01 Release-2.1 升级到 Release-3.0线上集群升级	1.2.1
02 Prometheus 等监控组件迁移	1.2.2
TroubleShooting 系列	1.3
POC 测试系列	1.4
CaseStudy 系列	1.5

## 前言

作为 TiDB 分布式数据库的使用者，在 POC 测试阶段、运维阶段，一定会遇到不少的问题，官方联合 TUG 的核心成员以及热心的社区用户，将日常的经验进行总结，一起完成了几大系列文章，希望能够帮助更多的新老 TiDB 用户。

## TiDB 运维手册 SOP 系列

SOP 的定位，是基于上线后的集群，将常见的运维操作标准化的手册；  
如果你有需求，希望得到更多大咖的操作建议，可以提 issue；如果你有  
类似运维操作的 SOP 经验，欢迎提 pr，经过官方 review 后，可以发  
布。

# 01 Release-2.1 升级到 Release-3.0 线上集群升级

李仲舒 2020 年 2 月 10 日

## 一、背景 / 目的

分布式数据库集群运维过程有一定的复杂性和繁琐性，3.0 版本是目前被广泛使用的版本，相比 2.1 有大幅度增加性能，以及很多新增的功能和特性，整体架构、配置也有较大的优化。该篇根据广大用户的升级经验，尽可能将 Release-2.1 升级到 Release-3.0 的准备工作、升级过程中注意事项、升级后重点关注列举详细，做到防患于未然。为 Release-3.0 版本的优秀特性和产品性能在业务场景中广泛使用提供文档依托。

适用人群默认为熟悉 2.1 版本的使用，但是没有做过大版本升级。

## 二、操作前的 Check 项

### 2.1 备份原集群修改过的 TiDB、TiKV 和 PD 参数

创建临时目录，备份升级前的参数配置

```
mkdir -p /tmp/tidb_update_3.0/conf  
mkdir -p /tmp/tidb_update_3.0/group_vars
```

确认并备份 tidb-ansible/conf/tidb.yml 中的参数

```
$cat tidb.yml |grep -v "#" |grep -v '^$' > /tmp/tidb_update_3.0/conf/tidb.yml  
$cat /tmp/tidb_update_3.0/conf/tidb.yml  
... 配置展示省略 ...
```

确认并备份 tidb-ansible/conf/tikv.yml 中的参数

```
$cat tikv.yml |grep -v "#" |grep -v '^$' > /tmp/tidb_update_3.0/conf/tikv.yml  
$cat /tmp/tidb_update_3.0/conf/tikv.yml  
... 配置展示省略 ...
```

确认并备份 tidb-ansible/conf/pd.yml 中的参数

```
$cat pd.yml |grep -v "#" |grep -v ^$ > /tmp/tidb_update_3.0/conf/pd.yml  
$cat /tmp/tidb_update_3.0/conf/pd.yml  
... 配置展示省略 ...
```

确认并备份 PD 集群中 etcd 记录的 PD 配置信息

```
$cd tidb-ansible/resource/bin/  
$./pd-ctl -u "http://{pd-ip}:{pd_client_port}" config show  
$cat /tmp/tidb_update_3.0/conf/pd.json  
... 配置展示省略 ...
```

确认并备份 TiDB、TiKV、PD、Grafana、Prometheus 等组件的组参数的变化，尤其是端口的变化

```
$cd tidb-ansible/group_vars  
$ll |awk '{print $9}'|grep -v ^$  
alertmanager_servers.yml  
all.yml  
drainer_servers.yml  
grafana_servers.yml  
importer_server.yml  
lightning_server.yml  
monitored_servers.yml  
monitoring_servers.yml  
pd_servers.yml  
pump_servers.yml  
tidb_servers.yml  
tikv_servers.yml  
$cp *.yml /tmp/tidb_update_3.0/group_vars/
```

## 三、升级前的注意事项

### 3.1 查看 TiDB 的 Release Notes

以 v3.0.9 [Release Notes](#) 为例，通过Release Notes了解做了那些优化或者修复了哪些 bug。

### 3.2 升级 3 个注意、2 个不支持、1 个用户

- 注意通知业务，升级期间可能会有偶尔的性能抖动，PD leader 升级可能会有 3s 的影响；
- 注意预估升级时间 = tikv 个数 \* 5 min (默认是 transfer leader 时间) + 10 min，滚动升级 TiKV 时间较长；

- 注意通知业务升级过程禁止操作 DDL，最好过一次完整的数据冷备份，通过 MyDumper 导出业务库；
- 整个升级过程不支持版本回退，目前未出现升级失败需要回退案例；
- 升级过程中不支持 DDL 操作，否则会有未定义问题，最终导致升级异常，影响业务；
- 升级操作通过中控机的 TiDB 管理用户完成，默认是“tidb”用户。

## 四、操作步骤

### 4.1 下载最新版的 v3.0.x 版本的 tidb-ansible

对应的 TAG 可以查看 Github 中的 [tidb-ansible 项目](#)，以 v3.0.9 为例，注意设置目录别名 “tidb-ansible-v3.0.9”

```
git clone -b v3.0.9 https://github.com/pingcap/tidb-ansible
```

tidb-ansible-v3.0.9 几个特殊的地方：

- 关于新增的 `excessive_rolling_update.yml` 和 `rolling_update.yml` 的关系。
  - 如果部署采用（默认）`systemd` 模式，使用 `excessive_rolling_update.yml` 来进行滚动升级操作，原因是涉及到 [PD 滚动升级](#)（以 v3.0.9 为例）的代码变动，该脚本仅本次升级使用一次，以后再次升级到后续版本均由 `rolling_update.yml` 来完成。

```
$ cat inventory.ini|grep supervision  
  
# process supervision, [systemd, supervise]  
  
process_supervision = systemd
```

- 如果采用 `supervise` 模式，依然使用 `rolling_update.yml` 来进行滚动升级操作。

```
$ cat inventory.ini|grep supervision  
# process supervision, [systemd, supervise]  
process_supervision = supervise
```

- 新增 `config check`（以 v3.0.9 为例），主要检查参数配置语法正确性。

```
## 代码块
- name: Pre-check PD configuration
  hosts: pd_servers[0]
  tags:
    - pd
  roles:
    - check_config_pd

- name: Pre-check TiKV configuration
  hosts: tikv_servers[0]
  tags:
    - tikv
  roles:
    - check_config_tikv

- name: Pre-check TiDB configuration
  hosts: tidb_servers[0]
  tags:
    - tidb
  roles:
    - check_config_tidb
```

## 4.2 更新 ansible 及依赖组件版本

```
# 验证版本是否符合要求
# 版本要求
$cat ./requirements.txt
ansible==2.7.11
jinja2>=2.9.6
jmespath>=0.9.0
# 检查版本
$ansible --version
$pip show jinja2
$pip show jmespath
# 卸载重新安装
# 卸载组件
$sudo pip uninstall ansible -y
$sudo pip uninstall jinja2 -y
$sudo pip uninstall jmespath -y
# 安装组件
$sudo pip install -r ./requirements.txt
```

## 4.3 编辑新 ansible 的 inventory.ini 和 配置

参考原始 inventory.ini 编辑新 inventory.ini

```
$cd tidb-ansible-v3.0.9
# 按照 tidb-ansible/inventory.ini 配置
$vi inventory.ini
```

### 注意

不要直接 cp 或者 mv 方式覆盖新的 inventory.ini 文件，建议将备份的原 tidb-ansible 的 inventory.ini 的配置参数通过复制、黏贴的方式将参数填写到新的 tidb-ansible 的 inventory.ini 配置文件中。另外注意以下几点：

- TiKV 单机多实例
  - TiKV 多实例部署时，必须添加 `tikv_status_port` 参数，同机上的多个实例注意区分端口，否则会造成监控缺失。  
【功能介绍】监控数据到 Prometheus 从 push 模式调整为 pull 模式，TiKV 的启动脚本 `run_tikv.sh` 需要配置 `tikv_status_port`，需要在 inventory.ini 的 `[tikv_servers]` 主机组下的目标实例配置中添加参数 `tikv_status_port`，通过滚动升级才会将参数写入到 `run_tikv.sh` 中。

```
```yml
vi tidb-ansible/inventory.ini
...省略上部分参数...

[tikv_servers]
TiKV1-1 ansible_host=172.16.10.4 deploy_dir=/data1/deploy
TiKV1-2 ansible_host=172.16.10.4 deploy_dir=/data2/deploy
...省略下部分参数...
```

```

- `label` 设置是否正确：
  - `[tikv_servers]` 主机组配置中同台主机多实例设置为相同的 `host label`；
  - `[pd_servers:vars]` 下的 `locations_labels` 是否正确设置为 `locations_labels = ["host"]`；
- `enable_binlog` 是否开启，如果设置为 `True`，升级过程中 Pump Server 会跟滚动升级操作一起升级，Drainer Server 需要单独完成升级操作。
- `deploy_dir` 和指定的参数（例如：端口、IP、别名）配置是否正确。

## 4.4 编辑 group\_vars 文件

### 注意

不要直接 cp 或者 mv 方式覆盖新的 group\_vars 目录下的 yml 文件，建议将原来 group\_vars 下的 yml 文件分别和新的 group\_vars 目录下的 yml 文件的配置参数对比，如果之前有做过参数调整，修改新的配置文件；如果默认的配置，可以忽略这个步骤。

```
#按照 /tmp/tidb_update_3.0/group_vars 手动修改过的参数进行修改,
$cd tidb-ansible-v3.0.9/group_vars
```

## 4.5 编辑 conf 文件

### 注意

不要直接 cp 或者 mv 方式覆盖新的 conf 目录下的 yml 文件，建议对比 /tmp/tidb\_update\_3.0/conf 下的 yml 配置文件 和 新的 conf 目录下的 yml 配置文件，将变化的参数进行修改，如果没有变动，按照默认的配置。另外，需要再额外关注以下几个变化的功能点的相关参数。

- 【功能介绍】开启共享 block-cache
  - storage.block-cache.capacity 默认开启共享池来自动调整，自动调整范围涵盖以下参数：rocksdb.defaultcf.block-cache-size, rocksdb.writecf.block-cache-size, rocksdb.lockcf.block-cache-size, raftdb.defaultcf.block-cache-size，只需要在 tidb-ansible/conf/tikv.toml 设置 capacity，其他的 block-cache-size 参数不再需要手动设置。
  - 计算方法：capacity = (MEM\_TOTAL \* 0.5 / TiKV 实例数量)

```
vi tidb-ansible/conf/tikv.yml
...省略上部分参数...
storage:
  block-cache:
    capacity: "1GB"
...省略下部分参数...
```

- 【功能介绍】开启静默 region，降低 region 心跳对 CPU 消耗 50%。

```
vi tidb-ansible/conf/tikv.yml
...省略上部分参数...
raftstore:
  hibernate-regions: true
...省略下部分参数...
```

## 4.6 下载 TiDB binary

### 注意

该操作会从互联网中的 PingCAP 的介质服务器下载相应的版本 tar 包或者 binary 文件，下载的介质和对应的 tidb-ansible 版本一一对应，不建议版本差异化下载，例如使用 v3.0.9 的 tidb-ansible，修改 inventory.ini 中的 tidb-version 配置，下载其他版本的 tidb binary 文件。

```
$cd tidb-ansible-v3.0.9
$ansible-playbook local_prepare.yml
```

## 4.7 滚动升级

### 注意

可以通过修改代码延长 Transfer leader 时间，来减少滚动 TiKV 过程的性能抖动。可以将 retries: 18 调整至 1800。如果提前 transfer leader 完成以后，会停止 check，继续下面的工作。延长 check 时间，也意味着升级时间会拉长，可以根据真实需求平衡参数。

```
# 以非 tls 模式为例
cd tidb-ansible-v3.0.9/common_tasks
vi add_evict_leader_scheduler.yml
...上部分代码忽略...
- name: check tikv's leader count
  uri:
    url: "http://{{ pd_addr }}/pd/api/v1/store/{{ store_id }}"
    method: GET
    return_content: yes
    body_format: json
    status_code: 200
  register: store_info
  until: (store_info.json.status.leader_count is defined and
           store_info.json.status.leader_count > 0)
  retries: 18
  delay: 10
  failed_when: false
  when: not enable_tls|default(false)
...下部分代码忽略...
```

- 执行滚动升级脚本，顺序为 PD、TiKV、Pump、TiDB。
  - systemd 模式（默认）

#### 注意

`excessive_rolling_update.yml` 仅限滚动升级 TiDB 集群使用一次，后面再次升级到其他版本均通过 `rolling_update.yml` 来完成。

```
$cd tidb-ansible-v3.0.9
$ansible-playbook excessive_rolling_update.yml
```

- supervise 模式

```
$cd tidb-ansible-v3.0.9
$ansible-playbook rolling_update.yml
```

- 升级如果中间步骤报错退出，处理好问题后，可以利用 `--tags` 和 `-l` 跳过前面升级完的组件，继续后面的升级。举例：在升级 TiKV1-2 时报错，依次执行执行：

```
ansible-playbook excessive_rolling_update.yml --tags=tikv  
ansible-playbook excessive_rolling_update.yml --tags=pump  
ansible-playbook excessive_rolling_update.yml --tags=tidb
```

- 升级后会开启 [Region Merge](#), 减少空 Region 和小 Region 对 CPU 消耗, 由于刚升级, 之前堆积的需要 merge 的量会比较多, 建议先调低并发到 2:

```
./pd-ctl -u "http://{pd-ip}:{pd_client_port}"  
» config set merge-schedule-limit 2  
Success!
```

- 等观察 region 数量趋于稳定后, 再逐渐调大到 8

```
./pd-ctl -u "http://{pd-ip}:{pd_client_port}"  
» config set merge-schedule-limit 8  
Success!  
» config set replica-schedule-limit 16  
Success!
```

## 4.8. 滚动升级 TiDB 监控组件

注意 该操作会将

`Prometheus`、`Grafana`、`blackbox_exporter`,`node_exporter` 监控组件滚动升级, 升级过程中 `Grafana` 页面会有一段时间不可用状态或者数据为空, 看不到监控数据或者显示 “no data” 是预期的。

## 五、操作后 Check 监控项

登录 `Grafana` 页面 `http://{grafana-ip}:{grafana-port}` 用户名和密码: `inventory.ini` 有配置

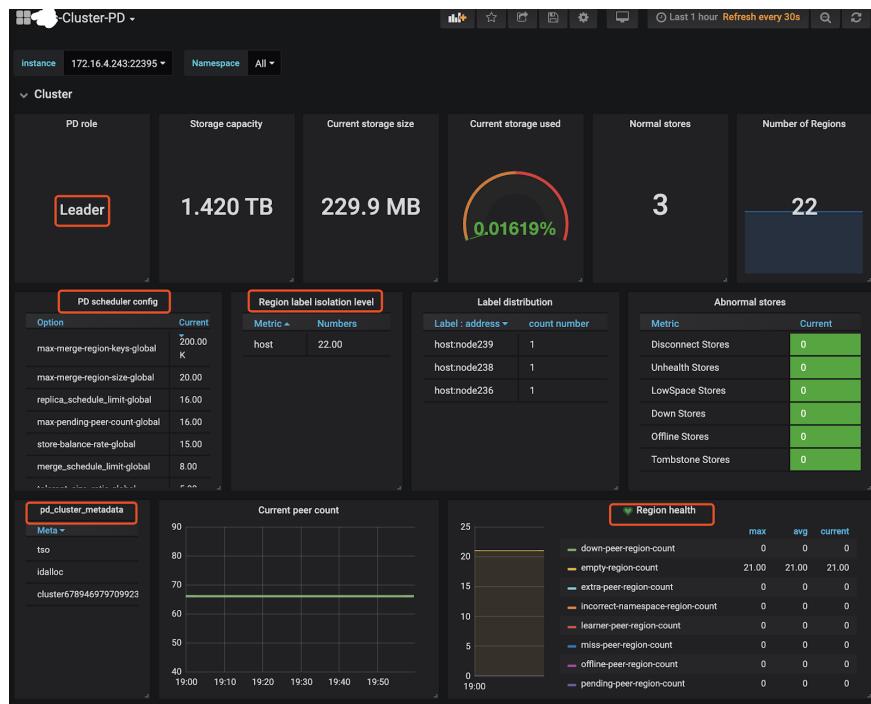
- 查看 `overview` 页面, `Overview` 页面的 `Services Port Status` 状态是否均为绿色的 `up` 状态;
- 查看 `TiDB` 页面, `Query Summary` 监控栏的 `Duration`、`QPS`、`Statement OPS`、`QPS By Instance`、`Failed Query OPM` 监控项是否正常, 在每个监控项左上 都会有一个“i”光标放在那里会描述监控项的解释和预期情况;

## 01 Release-2.1 升级到 Release-3.0线上集群升级



- 查看 TiKV-Details 页面，TiKV 页面已经失效，新的监控数据主要会展示在 TiKV-Details 、 TiKV-Summary 、 TiKV-Trouble-Shooting 中。可以通过 TiKV-Details 页面，通过 Cluster 、 Error 、 Server 确认 TiKV 实例的状态的负载以及错误情况。
- 查看 PD 页面，查看 Cluster 监控栏中的 Storage capacity 、 Current storage size 、 Current storage used 、 Normal stores 、 Number of Regions 确认当前集群存

储数据和 Region 的情况，另外添加了 PD 参数展示和 Label 状态展示的监控。



- 出现以下监控告警可能是预期的 Region Merge 操作导致的，因为 v3.0 版本的 Region Merge 效果比 v2.1 要好，所以如果开启 Region Merge 并发调度，可能短时间会出现 `tikvclient_backoff_count error` 告警，同时 `values` 会逐渐减少。对业务几乎没有影响，如担心业务延迟可以现将并发调度调整为 0，等到业务低峰时候，再加大 Region Merge 并发度。

#### 【样例】

```
TiDB tikvclient_backoff_count error
状态: 问题
cluster: xxx-cluster, instance: 172.16.1.1:10081, values:2
2000000
2020-02-11 13:04
```

## 六、周边工具升级

### 6.1 tidb-binlog 升级

- tidb-binlog 按照 tidb-ansible 部署
  - pump 会与 TiDB Cluster 的滚动升级操作一同升级，升级过程对于正在增量同步的数据没有影响，升级完成以后，可以登录到目标的 pump 节点，确认 pump 的版本。

```
$ ./pump -V
Release Version: v3.0.9
Git Commit Hash: c97e501d5054ed63c325c02b581a7c1a661cbc
Build TS: 2020-01-14 12:53:37
Go Version: go1.13
Go OS/Arch: linux/amd64
```

- drainer 升级需要先将 drainer 的 binary 上传至目标的节点 \${deploy\_dir}/bin 目录下，注意不要覆盖原 drainer，可以参考下面的操作。

```
$tidb-ansible/resources/bin
$ ./drainer -V
Release Version: v3.0.9
Git Commit Hash: c97e501d5054ed63c325c02b581a7c1a661cbc
Build TS: 2020-01-14 12:53:50
Go Version: go1.13
Go OS/Arch: linux/amd64
$cp drainer drainer309
$scp drainer309 tidb@{drainer-ip}:${deploy_dir}/bin/
$ssh {drainer-ip}
$cd ${deploy_dir}/scripts
$./stop_drainer.sh
$mv ${deploy_dir}/bin/drainer ${deploy_dir}/bin/drainer
$mv ${deploy_dir}/bin/drainer309 ${deploy_dir}/bin/drai
$./start_drainer.sh
```

## 6.2 Tispark 升级

- 目前 Tispark 不能直接通过 ansible 升级
- local\_prepare.yml 会将 Tispark 和 spark binary 下载到中控机本地目录
  - 代码块

```
tispark_packages:
  - name: spark-2.4.3-bin-hadoop2.7.tgz
    version: 2.4.3
    url: http://download.pingcap.org/spark-2.4.3-bin
    checksum: "sha256:80a4c564ceff0d9aff82b7df610"
  - name: tispark-latest.tar.gz
    version: latest
    url: http://download.pingcap.org/tispark-assembl
  - name: tispark-sample-data.tar.gz
    version: latest
    url: http://download.pingcap.org/tispark-sample-
    checksum: "sha256:bd0368a9d8663a4a8de89e39cc4"
```

- 下载目录

```
$cd tidb-ansible-v3.0.9/downloads
$ ll *spark*|awk '{print $9}'
spark-2.4.3-bin-hadoop2.7.tgz
tispark-core-2.1.8-spark_2.4-jar-with-dependencies.
tispark-sample-data.tar.gz
$pwd
/home/tidb/lzs/ansible/
```

- 升级操作和 drainer 升级操作类似
  - 先停掉 Spark 服务
  - 将 Tispark 的 jar 包拷贝到spark安装目录下面的 jars 子目录
  - 启动 Spark 服务
  - 打开master web ui: `http://${master ip}:8080` , 确认可以访问, 确认work个数、core个数、内存数是否符合预期
  - 简单测试: `./bin/spark-shell --master spark://${master ip}:7077` 运行一些简单的测试命令, 例如看看是否能访问tidb

## 七、升级常见问题

### 7.1 tidb-ansible 操作常见问题排查

- 通过 git 下载 tidb-ansible 分支和升级 TiDB 集群版本不匹配问题
  - 不建议多版本混用 tidb-ansible , 目前很多逻辑支持的都是当前版本, 使用 tidb-ansible 版本和 tidb 集群版本一致。通过 git

clone 下载指定的 tag 分支, inventory.ini 中的 tidb-version 参数不要擅自修改。

- 执行 `local_prepare.yml` 报错
  - 网络延迟高, 导致部分 curl get 下载介质文件不完整或者中断报错;
  - 下载的介质文件异常, 可以根据报错找到对应的脚本未知和下载的地址, 通过 curl 命令进行下载验证。
- 执行 `rolling_update.yml` 报错
  - tikv 滚动过程中报错退出, 确认具体的报错位置, 查看对应的 ansible-playbook 逻辑

## 八、相关案例

### Asktug 问题

- <https://asktug.com/t/topic/2126>
- <https://asktug.com/t/v2-1-4-v3-0-0-tikv-up/415>
- <https://asktug.com/t/topic/2280>
- <https://asktug.com/t/v2-1-9-v3-0-5-fatal-schema-1146-table-mysql-tidb-doesnt-exist/1616>
- <https://asktug.com/t/tidbv2-1-1-v3-0-1/1638>

## 02 Prometheus 等监控组件迁移

李仲舒 2020 年 2 月 10 日

### 一、背景

TiDB Cluster 主要通过 Prometheus 方案解决线上监控、问题排查等问题，是 TiDB Cluster 自动化运维中必不可少的运维组件。Prometheus 的关联组件包括：Grafana、pushgateway、node\_exporter、blackbox\_exporter 以及 alertmanager。在我们运维过程可能会遇到由于机器故障、机房搬迁、机型替换等原因，需要迁移 Prometheus 服务，本文将描述如何对 Prometheus 监控数据进行迁移操作。

一般 node\_exporter、blackbox\_exporter 随集群其他节点部署，不需要单独迁移，需要迁移的通常为 Grafana + Prometheus + pushgateway + alertmanager。

### 二、操作前了解相关配置

Prometheus 监控数据迁移之前，先确认一下原监控是否有调整过重要的 IP、端口、目录、配置参数、版本等等信息。

#### 2.1 查看 Prometheus 拓扑结构

- Prometheus 的拓扑结构主要通过 `tidb-ansible/inventory.ini` 来查看，端口、目录。

```
## Monitoring Part
# prometheus and pushgateway servers
[monitoring_servers] -- 提供 Prometheus 和 Pushgateway 服务的
172.16.4.236 prometheus_port=9597

[grafana_servers] -- 提供 Grafana 服务的节点 IP 和 端口
172.16.4.236 grafana_port=3000

# node_exporter and blackbox_exporter servers
[monitored_servers] -- 提供被监控的节点的 node_exporter 和 bl
172.16.4.240
172.16.4.242
172.16.4.243
172.16.4.239
172.16.4.238
172.16.4.236
172.16.5.90
[alertmanager_servers] -- 提供 alertmanager 服务的告警服务 IP
#172.16.4.236
```

## 2.2 了解 Prometheus 服务的配置情况

```
$ cat monitoring_servers.yml
---
prometheus_port: 9090
pushgateway_port: 9091

# How long to retain samples in the storage
prometheus_storage_retention: "30d"
```

## 2.3 了解 Grafana、Alertmanager、blackbox exporter、nodebox exporter 服务的目录结构及配置情况。

- Grafana 服务的目录结构及配置

```
$ cat grafana_servers.yml
---
grafana_port: 3000

grafana_api_keys_dir: "{{ playbook_dir }}/conf/keys"
```

- node\_exporter 和 blackbox\_exporter 服务的目录及配置

```
$ 目录配置
$ cat monitored_servers.yml
---

node_exporter_log_dir: "{{ deploy_dir }}/log"
$端口配置
$ cat all.yml
... 忽略部分参数配置 ...
# default configuration for multiple host groups and roles
node_exporter_port: 9170
blackbox_exporter_port: 9715
kafka_exporter_port: 9307
... 忽略部分参数配置 ...
```

- alertmanager 服务的目录及配置

```
$ cat alertmanager_servers.yml
---

alertmanager_port: 9093
alertmanager_cluster_port: 9094
```

- 检查 tidb-ansible 下载版本是否为预期版本，以 v3.0.7 为例。这里注意一定要使用和集群一致的 ansible 版本，这是由于不同小版本 Prometheus 的一些抓取监控数据的度量值会有变化。

```
cd /tidb-ansible
$ git branch -v
* (no branch) f4902bb bump version to v3.0.7
```

## 三、操作步骤

### 3.1 更新 inventory.ini 配置，部署新的 Prometheus 等监控服务

- 在 inventory.ini 添加新的 prometheus 组件配置，同时注释掉原 prometheus 配置。如果对于端口和目录没有特殊要求的情况，只配置新 Prometheus 的 IP 就可以了。否则也需要在 inventory.ini 的位置添加对应的端口参数和目录参数 (deploy\_dir) 。
- prometheus 和 pushgateway 一般都是一起部署的，所以默认将 pushgateway 也重新部署。

- grafana 和 alertmanager 可以选择迁移或者不迁移，如果迁移也要同步修改 IP

```
$ cat inventory.ini
---- 忽略上半部分代码 ----
[monitoring_servers] # 记得注释掉之前的 配置，同时也指定 Prometheus 的端口
#172.16.4.236 prometheus_port=9597
172.16.4.242 prometheus_port=9090 pushgateway_port=9091
[grafana_servers] # 记得注释掉之前的 配置，同时也指定 grafana 的端口
#172.16.4.236 grafana_port=3000
172.16.4.242 grafana_port=3000
# node_exporter and blackbox_exporter servers
[monitored_servers] # 如果新节点确认是否已经配置在受监控的主机中。
172.16.4.240
172.16.4.242
172.16.4.243
172.16.4.239
172.16.4.238
172.16.4.236
172.16.5.90
[alertmanager_servers]
#172.16.4.236
172.16.4.242 alertmanager_port=9093 alertmanager_cluster_port=9094
---- 忽略上半部分代码 ----
```

- 部署开始

```
$# 配置节点系统配置
$ansible-playbook bootstrap.yml -l 172.16.4.242
$# 部署 prometheus
$ansible-playbook deploy.yml -l 172.16.4.242 -t prometheus
$# 部署 pushgateway
$ansible-playbook deploy.yml -l 172.16.4.242 -t pushgateway
$# 部署 grafana
$ansible-playbook deploy.yml -l 172.16.4.242 -t grafana
$# 部署 altermanger
$ansible-playbook deploy.yml -l 172.16.4.242 -t altermanger
$# 部署 node_exporter、blackbox_exporter，如果该节点通过 tidb-ansible-deploy.yml -l 172.16.4.242 -t node_exporter
```

### 3.2 启动 Prometheus 服务

- 拉启 Prometheus 和 pushgateway 服务

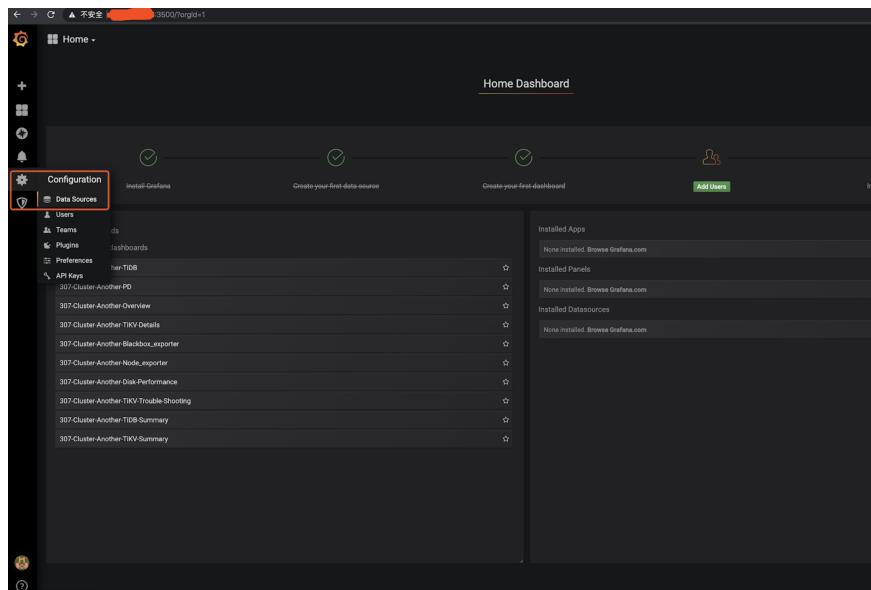
```
$# 确保已经停掉原 prometheus 和 pushgateway  
$# 启动 prometheus 和 pushgateway  
$ansible-playbook start.yml -l 172.16.4.242 -t prometheus  
$ansible-playbook start.yml -l 172.16.4.242 -t pushgateway  
$ansible-playbook start.yml -l 172.16.4.242 -t grafana  
$ansible-playbook start.yml -l 172.16.4.242 -t altermanager  
$ansible-playbook start.yml -l 172.16.4.242 -t node_exporter
```

- 滚动整个监控集群

```
$# 滚动整个监控集群  
$ansible-playbook rolling_update_monitor.yml
```

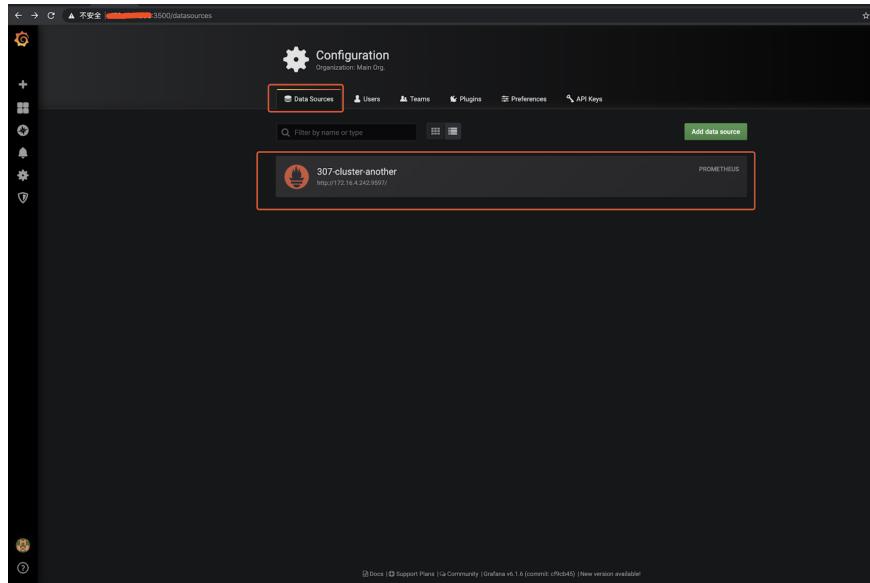
- 配置 Grafana -- 如果 Grafana 也已经迁移，则跳过此步骤

- 设置中找到 Data Sources

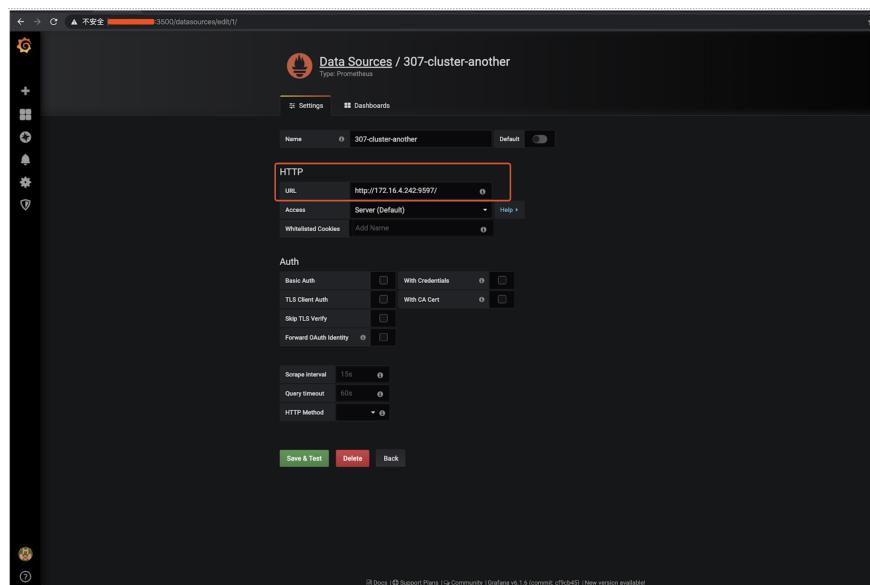


- 选择要调整的 Data Sources

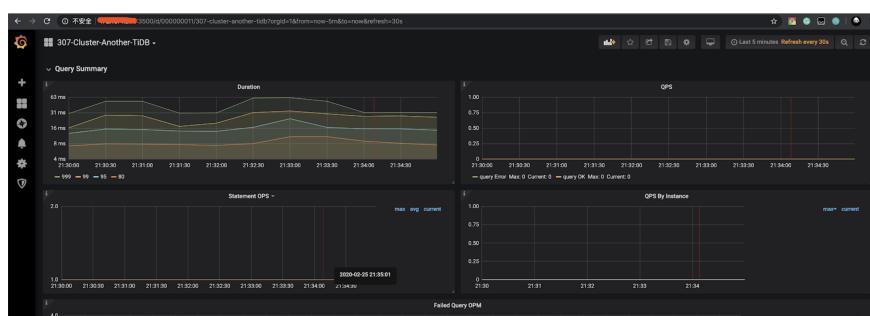
## 01 Release-2.1 升级到 Release-3.0线上集群升级



- 将“HTTP”的“URL”进行调整设置为最新的 Prometheus 的监控平台地址配置“172.16.4.242:9090”，并且点击“save & test”，保存。



## 四、操作后 Check 监控项



## 五、迁移可能会涉及的 FAQ

## 5.1 如何导入原 Prometheus 的监控数据

如果希望保留原 Prometheus 的监控数据，可以将其导入到新监控中，有全量导入和截取一段导入两种方式。

- 全量导出 Prometheus 数据
  - 全量数据正常会非常大，尤其是保留启动参数配置了 `storage.tsdb.retention="30d"` 保留 30 天的情况，可以通过 tar 方式压缩

```

## 通过 ssh 进入原 Prometheus 服务节点
$ssh ${prometheus-ip}
## 进入 Prometheus 的部署目录下的 scripts 目录
$cd ${deploy_dir}/scripts
## 查看 Prometheus 的启动配置项，“--storage.tsdb.path” 配置就是
$cat run_prometheus.sh
#!/bin/bash
set -e
ulimit -n 1000000

DEPLOY_DIR=/home/tidb/deploy_3.0.7
cd "${DEPLOY_DIR}" || exit 1

# WARNING: This file was auto-generated. Do not edit!
#           All your edit might be overwritten!
exec > >(tee -i -a "/home/tidb/deploy_3.0.7/log/prometheus.conf"
exec 2>&1

exec bin/prometheus \
    --config.file="/home/tidb/deploy_3.0.7/conf/prometheus.conf" \
    --web.listen-address=:9597" \
    --web.external-url="http://172.16.4.236:9597/" \
    --web.enable-admin-api \
    --log.level="info" \
    --storage.tsdb.path="/home/tidb/deploy_3.0.7/prometheus" \
    --storage.tsdb.retention="30d"
$cd /home/tidb/deploy_3.0.7/
$ls -lart prometheus2.0.0.data.metric
$total 4
drwxr-xr-x. 11 tidb tidb 150 Dec 20 12:04 ..
-rw-r--r--. 1 tidb tidb 0 Dec 20 13:57 lock
drwxr-xr-x. 3 tidb tidb 68 Feb 25 16:04 01DZJF2G2BYG7JW6
drwxr-xr-x. 3 tidb tidb 68 Feb 25 16:04 01DZR8F29SV2TNQ8
drwxr-xr-x. 3 tidb tidb 68 Feb 25 16:04 01DZY1VT0K4NVQPE
drwxr-xr-x. 3 tidb tidb 68 Feb 25 16:04 01E03V8DZRMSSYX1E
drwxr-xr-x. 3 tidb tidb 68 Feb 25 16:04 01E09MN09R3T9BB8
drwxr-xr-x. 3 tidb tidb 68 Feb 25 16:04 01E0FE1HVQ5GNDB8
drwxr-xr-x. 3 tidb tidb 68 Feb 25 16:04 01E0N7E8VXC6TVB8
drwxr-xr-x. 3 tidb tidb 68 Feb 25 16:04 01E0V0TTKJ4BT3T8
drwxr-xr-x. 3 tidb tidb 68 Feb 25 16:04 01E10T7GWVAXD918
drwxr-xr-x. 3 tidb tidb 68 Feb 25 16:04 01E16KM1TE5R4V2C
drwxr-xr-x. 3 tidb tidb 68 Feb 25 16:04 01E1CD0N9DZEJBPE
drwxr-xr-x. 3 tidb tidb 68 Feb 25 16:04 01E1J6DAM2BJTG1V
drwxr-xr-x. 3 tidb tidb 68 Feb 25 16:04 01E1QZSYNC18JXT8
drwxr-xr-x. 3 tidb tidb 68 Feb 25 16:04 01E1SXK95Y7TC24F
drwxr-xr-x. 3 tidb tidb 68 Feb 25 16:04 01E1VVCJDV9F6MV8
drwxr-xr-x. 3 tidb tidb 68 Feb 25 16:04 01E1VVCP5D79302
drwxr-xr-x. 3 tidb tidb 68 Feb 25 16:04 01E1XMAN6TKGTWV8

```

```
drwxr-xr-x. 20 tidb tidb 4096 Feb 25 16:04 .
drwxr-xr-x. 3 tidb tidb 95 Feb 25 16:10 wal
$查看 prometheus2.0.0.data.metrics 大小
$du -sh prometheus2.0.0.data.metrics
8.9G  prometheus2.0.0.data.metrics/
$# 压缩监控数据文件，时间较长建议放在后台执行。
$tar -cvzf prometheus2.0.0.data.metrics.tar.gz prometheus2.
```

- 截取一段 Prometheus 数据导出
  - 可以直接按照创建日期直接截取一部分监控数据文件

```

$ cd 01E1VVCJDV9F6MVW1XQ0NB0KRA
$ ls -lart
total 3868
drwxr-xr-x. 2 tidb tidb      20 Feb 24 23:00 chunk
-rw-r--r--. 1 tidb tidb 3945252 Feb 24 23:00 index
-rw-r--r--. 1 tidb tidb      9 Feb 24 23:00 tomb
-rw-r--r--. 1 tidb tidb    305 Feb 25 16:04 meta.
drwxr-xr-x. 3 tidb tidb     68 Feb 25 16:04 .
drwxr-xr-x. 20 tidb tidb   4096 Feb 25 16:04 ..
$ cd ..../01E1SXK95Y7TC24KMQGFCE789J
$ ls -lart
total 14836
drwxr-xr-x. 2 tidb tidb      20 Feb 24 05:00 chun
-rw-r--r--. 1 tidb tidb 15179425 Feb 24 05:00 inde
-rw-r--r--. 1 tidb tidb      9 Feb 24 05:00 tomb
-rw-r--r--. 1 tidb tidb    929 Feb 25 16:04 meta
drwxr-xr-x. 3 tidb tidb     68 Feb 25 16:04 .
drwxr-xr-x. 20 tidb tidb   4096 Feb 25 16:04 ..
$ cd ..../01E1QZSYNC18JXTYMTD8NK3WRC
$ ls -lart
total 37744
drwxr-xr-x. 2 tidb tidb      34 Feb 23 11:00 chun
-rw-r--r--. 1 tidb tidb 38636384 Feb 23 11:00 inde
-rw-r--r--. 1 tidb tidb      9 Feb 23 11:00 tomb
-rw-r--r--. 1 tidb tidb    1523 Feb 25 16:04 meta
drwxr-xr-x. 3 tidb tidb     68 Feb 25 16:04 .
drwxr-xr-x. 20 tidb tidb   4096 Feb 25 16:04 ..
$ cd ..../01E1J6DAM2BJTG1WVM1ETTJ1VV
$ ls -lat
total 37024
drwxr-xr-x. 20 tidb tidb   4096 Feb 25 16:04 ..
drwxr-xr-x. 3 tidb tidb     68 Feb 25 16:04 .
-rw-r--r--. 1 tidb tidb    1523 Feb 25 16:04 meta
-rw-r--r--. 1 tidb tidb      9 Feb 21 05:00 tomb
-rw-r--r--. 1 tidb tidb 37896841 Feb 21 05:00 inde
drwxr-xr-x. 2 tidb tidb     34 Feb 21 05:00 chun
$ 将导出数据打包备份
$ tar -cvzf prometheus2.0.0.data.metrics_2125.tar.g

```

- 可以通过一些开源工具来截取指定的监控数据文件

- 工具下载

```

$wget --http-user=TiDB-DBA --http-password=areyouhungry http://127.0.0.1:9090/metrics
$ chmod 755 export-data

```

- 时间戳转换 <https://tool.lu/timestamp/>



- 加入我们希望截取 2020-02-21 00:00:01 到 2020-02-25 23:59:59

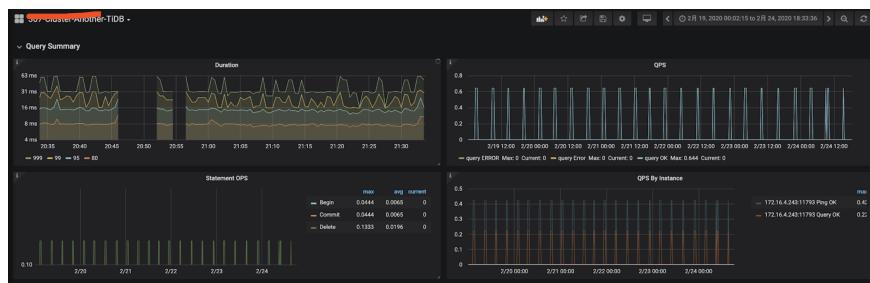
```
$ 导出监控数据操作
$ ./export-data dump --dump-dir=./ --min-time=1582214461000
$ 查看导出结果
$ ls -lart
total 12604
-rwxr-xr-x  1 tidb tidb 12906459 Feb 25 17:07 export-data
drwxr-xr-x.  5 tidb tidb      51 Feb 25 17:08 ..
drwxrwxr-x  3 tidb tidb     68 Feb 25 17:29 01E1J6DAM2B
drwxrwxr-x  3 tidb tidb     68 Feb 25 17:29 01E1SXK95Y7
drwxrwxr-x  3 tidb tidb     68 Feb 25 17:29 01E1QZSYNC18
drwxrwxr-x  3 tidb tidb     68 Feb 25 17:29 01E1VVCJDV9F
drwxrwxr-x  3 tidb tidb     68 Feb 25 17:29 01E1XMAN6TK
drwxrwxr-x  3 tidb tidb     68 Feb 25 17:29 01E1VVCPT5D
drwxrwxr-x  3 tidb tidb     68 Feb 25 17:29 01E1XS63NHQ
drwxrwxr-x  3 tidb tidb     68 Feb 25 17:29 01E1XTW4B4H
drwxrwxr-x 10 tidb tidb    297 Feb 25 17:29 .

$ 查看导出大小
$ du -sh *
713M  01E1J6DAM2BJTG1WVM1ETTJ1VV
745M  01E1QZSYNC18JXTYMTD8NK3WRC
256M  01E1SXK95Y7TC24KM0GFCE789J
17M   01E1VVCJDV9F6MVW1XQ0NB0KRA
263M  01E1VVCPT5D79304RW41GDMTYS
6.6M  01E1XMAN6TKGTWV2SMWCKY7P4J
664K  01E1XS63NHQMH82X4NNJ0J2PAN
123M  01E1XTW4B4HQGVGB25VEDP8W0T
13M   export-data
$ 将导出数据打包备份
$ i=`ls |grep -v "export-data"`
$ echo $i
01E1J6DAM2BJTG1WVM1ETTJ1VV 01E1QZSYNC18JXTYMTD8NK3WRC 01E1S
$ tar -cvzf prometheus_2125.tar.gz ${i}
```

- 将监控数据导入到目标节点

```
$# 原节点
$scp prometheus_2125.tar.gz tidb@172.16.4.242:/home/tidb/deploy/
$# 查看新节点的监控数据存放位置，在进程中会有“--storage.tsdb.path”
$$ ps -ef |grep prometheus|grep "storage.tsdb.path"
tidb      2118      1  6 Feb25 ?          01:18:00 bin/prometheus
$# 新节点，目标目录解压
$cd /home/tidb/deploy_3.0.7_new/prometheus2.0.0.data.metrics
$tar -xvf prometheus_2125.tar.gz
```

- 通过 Grafana 验证数据是否导入成功 - TiDB 监控看到的 2.21-2.25 的监控数据



## 5.2 tidb-ansible 中使用的 Prometheus 组件的版本信息和默认配置的情况查看

### 1. 版本信息

#### 注意

目前 Prometheus 版本和 TiDB Cluster 版本，TiDB 版本都是按照 Release notes 里面的版本打版的，基本上每个独立的小版本的 Prometheus 抓取的监控度量数据都有一些变化，所以建议是完全按照同版本的 tidb-ansible 新建新的 prometheus 服务。

- Prometheus 及其他组件版本

```
$ pwd
/tidb-ansible/roles/local/templates
$ cat binary_packages.yml.j2
---

third_party_packages:
  - name: prometheus
    version: 2.8.1
    url: "https://github.com/prometheus/prometheus/releases/download/v2.8.1/prometheus-v2.8.1.linux-amd64.tar.gz"
  - name: alertmanager
    version: 0.17.0
    url: "https://github.com/prometheus/alertmanager/releases/download/v0.17.0/alertmanager-v0.17.0.linux-amd64.tar.gz"
  - name: node_exporter
    version: 0.17.0
    url: "https://github.com/prometheus/node_exporter/releases/download/v0.17.0/node_exporter-v0.17.0.linux-amd64.tar.gz"
  - name: blackbox_exporter
    version: 0.12.0
    url: "https://github.com/prometheus/blackbox_exporter/releases/download/v0.12.0/blackbox_exporter-v0.12.0.linux-amd64.tar.gz"
  - name: pushgateway
    version: 0.7.0
    url: "https://github.com/prometheus/pushgateway/releases/download/v0.7.0/pushgateway-v0.7.0.linux-amd64.tar.gz"
  - name: grafana
    version: 6.1.6
    url: "https://dl.grafana.com/oss/release/grafana-6.1.6.linux-amd64.tar.gz"

third_party_packages_under_gfw:
  - name: prometheus
    version: 2.8.1
    url: "https://download.pingcap.org/prometheus-2.8.1.linux-amd64.tar.gz"
  - name: alertmanager
    version: 0.17.0
    url: "http://download.pingcap.org/alertmanager-0.17.0.linux-amd64.tar.gz"
  - name: node_exporter
    version: 0.17.0
    url: "http://download.pingcap.org/node_exporter-0.17.0.linux-amd64.tar.gz"
  - name: pushgateway
    version: 0.7.0
    url: "http://download.pingcap.org/pushgateway-0.7.0.linux-amd64.tar.gz"
  - name: grafana
    version: 6.1.6
    url: "https://download.pingcap.org/grafana-6.1.6.linux-amd64.tar.gz"
  - name: blackbox_exporter
    version: 0.12.0
    url: "http://download.pingcap.org/blackbox_exporter-0.12.0.linux-amd64.tar.gz"

----- 忽略下半部分代码 -----
```

- TiDB 版本

```
$ cat inventory.ini |grep version  
tidb_version = v3.0.7
```

1. 配置文件

2. Prometheus.yml 文件会配置 Grafana、pushgateway、node\_exporter、blackbox\_exporter 以及 altermanager 的 IP 和端口信息，一般通过 tidb-ansible 来完成所有的配置，可以自动配置，以下为具体的配置文件和配置参数。

- prometheus.yml

```
$ cat run_prometheus.sh|grep "config.file"  
--config.file="/home/tidb/deploy_3.0.7/conf/prometheus.
```

- Grafana 会单独有一个数据源的 json 配置，用来读取 Prometheus 的监控数据
  - grafana 的数据源配置

```
$ pwd  
/home/tidb/deploy_3.0.7/data.grafana  
$ cat data_source.json  
{  
    "name": "307-cluster-another",  
    "type": "prometheus",  
    "access": "proxy",  
    "url": "http://172.16.4.236:9597/",  
    "basicAuth": false
```

1. 监控数据

2. 监控数据采用的 binary 格式的，建议不要随意修改，一个完整的监控数据文件目录包括 chunks 目录、index、tombstones、meta.json 文件。

```
$ ls -lart  
total 34396  
drwxr-xr-x.  2 tidb tidb      34 Jan 27 11:00 chunks  
-rw-r--r--.  1 tidb tidb 35205230 Jan 27 11:00 index  
-rw-r--r--.  1 tidb tidb       9 Jan 27 11:00 tombstones  
drwxr-xr-x. 21 tidb tidb    4096 Feb 25 17:00 ..  
-rw-r--r--.  1 tidb tidb    1523 Feb 25 17:35 meta.json  
drwxr-xr-x.  3 tidb tidb      68 Feb 25 17:35 .
```

1. 监控模版

2. 如 3.1 所说，建议跟 TiDB Cluster 版本要匹配，不然可能会出现监控数据无法使用的情况。

```
$ cat /home/tidb/deploy_3.0.7/conf/prometheus.yml
---
global:
  scrape_interval:      15s # By default, scrape targets every 15s.
  evaluation_interval: 15s # By default, scrape targets evaluate every 15s.
  # scrape_timeout is set to the global default (10s).
  external_labels:
    cluster: '307-cluster-another'
    monitor: "prometheus"

# Load and evaluate rules in this file every 'evaluation_interval'.
rule_files:
  - 'node.rules.yml'
  - 'blacker.rules.yml'
  - 'bypass.rules.yml'
  - 'pd.rules.yml'
  - 'tidb.rules.yml'
  - 'tikv.rules.yml'
----- 忽略下半部分代码 -----
```

## **TroubleShooting 系列**

## **POC 测试系列**

## **CaseStudy 系列**