

v8.5

>

文档中心

> 关于 TiDB

> 快速上手

> 应用开发

> 部署标准集群

硬件环境需求

环境与系统配置检查

> 规划集群拓扑

使用 TiUP 部署

使用 TiUP 部署 TiDB 集群



10 Contributors

本指南介绍如何在生产环境中使用 [TiUP](#) 部署 TiDB 集群。

TiUP 是在 TiDB v4.0 中引入的集群运维工具，提供了使用 Golang 编写的集群管理组件 [TiUP cluster](#)。通过使用 TiUP cluster 组件，你可以轻松执行日常的数据库运维操作，包括部署、启动、关闭、销毁、弹性扩缩容、升级 TiDB 集群，以及管理 TiDB 集群参数。

TiUP 还支持部署 TiDB、TiFlash、TiCDC 以及监控系统。本指南介绍了如何部署不同拓扑的 TiDB 集群。

第 1 步：软硬件环境需求及前置检查

务必阅读以下文档：

- [软硬件环境需求](#)
- [环境与系统配置检查](#)

此外，建议阅读了解 [TiDB 安全配置最佳实践](#)。

第 2 步：在中控机上部署 TiUP 组件

在中控机上部署 TiUP 组件有两种方式：在线部署和离线部署。

在线部署

以普通用户身份登录中控机。以 `tidb` 用户为例，后续安装 TiUP 及集群管理操作均通过该用户完成：

- 执行如下命令安装 TiUP 工具：

```
curl --proto '=https' --tlsv1.2 -sSf https://tiup-mirrors.pingcap.com/install.sh | sh
```

- 按如下步骤设置 TiUP 环境变量：

- 重新声明全局环境变量：

```
source .bash_profile
```

- 确认 TiUP 工具是否安装：

```
which tiup
```

- 安装 TiUP 集群组件：

```
tiup cluster
```

- 如果已经安装，则更新 TiUP 集群组件至最新版本：

```
tiup update --self && tiup update cluster
```

预期输出 “Updated successfully!” 字样。

- 验证当前 TiUP 集群版本信息。执行如下命令查看 TiUP 集群组件版本：

```
tiup --binary cluster
```

离线部署

离线部署 TiUP 组件的操作步骤如下。

准备 TiUP 离线组件包

方式一：在 [官方下载页面](#) 选择对应版本的 TiDB server 离线镜像包（包含 TiUP 离线组件包）。需要同时下载 TiDB-community-server 软件包和 TiDB-community-toolkit 软件包。

方式二：使用 `tiup mirror clone` 命令手动打包离线组件包。步骤如下：

- 在在线环境中安装 TiUP 包管理工具
 - 执行如下命令安装 TiUP 工具：

-  下载 PDF
-  文档反馈
-  社区交流

本页导航

- 第 1 步：软硬件环境需求及前置检查
- 第 2 步：在中控机上部署 TiUP 组件
 - 在线部署
 - 离线部署
 - 准备 TiUP 离线组件包
 - 部署离线环境 TiUP 组件
 - 合并离线包
- 第 3 步：初始化集群拓扑文件
- 第 4 步：执行部署命令
- 第 5 步：查看 TiUP 管理的集群情况
- 第 6 步：检查部署的 TiDB 集群情况


```
curl --proto '=https' --tlsv1.2 -sSf https://tiup-mirrors.pingcap.com/install.sh |
```

ii. 重新声明全局环境变量：

```
source .bash_profile
```

iii. 确认 TiUP 工具是否安装：

```
which tiup
```

2. 使用 TiUP 制作离线镜像

i. 在一台和外网相通的机器上拉取需要的组件：

```
tiup mirror clone tidb-community-server-${version}-linux-amd64 ${version} --os=linux
```

该命令会在当前目录下创建一个名叫 `tidb-community-server-${version}-linux-amd64` 的目录，里面包含 TiUP 管理的组件包。

ii. 通过 `tar` 命令将该组件包打包然后发送到隔离环境的中控机：

```
tar czvf tidb-community-server-${version}-linux-amd64.tar.gz tidb-community-server-
```

此时， `tidb-community-server-${version}-linux-amd64.tar.gz` 就是一个独立的离线环境包。

3. 自定义制作的离线镜像，或调整已有离线镜像中的内容

如果从官网下载的离线镜像不满足你的具体需求，或者希望对已有的离线镜像内容进行调整，例如增加某个组件的新版本等，可以采取以下步骤进行操作：

i. 在制作离线镜像时，可通过参数指定具体的组件和版本等信息，获得不完整的离线镜像。例如，要制作一个只包括 v1.12.3 版本 TiUP 和 TiUP Cluster 的离线镜像，可执行如下命令：

```
tiup mirror clone tiup-custom-mirror-v1.12.3 --tiup v1.12.3 --cluster v1.12.3
```

如果只需要某一特定平台的组件，也可以通过 `--os` 和 `--arch` 参数来指定。

ii. 参考上文“使用 TiUP 制作离线镜像”第 2 步的方式，将此不完整的离线镜像传输到隔离环境的中控机。

iii. 在隔离环境的中控机上，查看当前使用的离线镜像路径。较新版本的 TiUP 可以直接通过命令获取当前的镜像地址：

```
tiup mirror show
```

以上命令如果提示 `show` 命令不存在，可能当前使用的是较老版本的 TiUP。此时可以通过查看 `$HOME/.tiup/tiup.toml` 获得正在使用的镜像地址。将此镜像地址记录下来，后续步骤中将以变量 `${base_mirror}` 指代此镜像地址。

iv. 将不完整的离线镜像合并到已有的离线镜像中：

首先将当前离线镜像中的 `keys` 目录复制到 `$HOME/.tiup` 目录中：

```
cp -r ${base_mirror}/keys $HOME/.tiup/
```

然后使用 TiUP 命令将不完整的离线镜像合并到当前使用的镜像中：

```
tiup mirror merge tiup-custom-mirror-v1.12.3
```

v. 上述步骤完成后，通过 `tiup list` 命令检查执行结果。在本文例子中，使用 `tiup list tiup` 和 `tiup list cluster` 均应能看到对应组件的 `v1.12.3` 版本出现在结果中。

部署离线环境 TiUP 组件

将离线包发送到目标集群的中控机后，执行以下命令安装 TiUP 组件：

```
tar xzvf tidb-community-server-${version}-linux-amd64.tar.gz && \
sh tidb-community-server-${version}-linux-amd64/local_install.sh && \
source /home/tidb/.bash_profile
```

`local_install.sh` 脚本会自动执行 `tiup mirror set tidb-community-server-${version}-linux-amd64` 命令将当前镜像地址设置为 `tidb-community-server-${version}-linux-amd64` 。

合并离线包

如果是通过[官方下载页面](#)下载的离线软件包，需要将 TiDB-community-server 软件包和 TiDB-community-toolkit 软件包合并到离线镜像中。如果是通过 `tiup mirror clone` 命令手动打包的离线组件包，不需要执行此步骤。

执行以下命令合并离线组件到 server 目录下。

```
tar xf tidb-community-toolkit-${version}-linux-amd64.tar.gz
ls -ld tidb-community-server-${version}-linux-amd64 tidb-community-toolkit-${version}-linux
cd tidb-community-server-${version}-linux-amd64/
cp -rp keys ~/.tiup/
tiup mirror merge ../tidb-community-toolkit-${version}-linux-amd64
```

若需将镜像切换到其他目录，可以通过手动执行 `tiup mirror set <mirror-dir>` 进行切换。如果需要切换到在线环境，可执行 `tiup mirror set https://tiup-mirrors.pingcap.com`。

第 3 步：初始化集群拓扑文件

执行如下命令，生成集群初始化配置文件：

```
tiup cluster template > topology.yaml
```

针对两种常用的部署场景，也可以通过以下命令生成建议的拓扑模板：

- 混合部署场景：单台机器部署多个实例，详情参见[混合部署拓扑架构](#)。

```
tiup cluster template --full > topology.yaml
```

- 跨机房部署场景：跨机房部署 TiDB 集群，详情参见[跨机房部署拓扑架构](#)。

```
tiup cluster template --multi-dc > topology.yaml
```

执行 `vi topology.yaml`，查看配置文件的内容：

```
global:
  user: "tidb"
  ssh_port: 22
  deploy_dir: "/tidb-deploy"
  data_dir: "/tidb-data"
server_configs: {}
pd_servers:
  - host: 10.0.1.4
  - host: 10.0.1.5
  - host: 10.0.1.6
tidb_servers:
  - host: 10.0.1.7
  - host: 10.0.1.8
  - host: 10.0.1.9
tikv_servers:
  - host: 10.0.1.1
  - host: 10.0.1.2
  - host: 10.0.1.3
monitoring_servers:
  - host: 10.0.1.4
grafana_servers:
  - host: 10.0.1.4
alertmanager_servers:
  - host: 10.0.1.4
```

下表列出了常用的 6 种场景，请根据链接中的拓扑说明以及配置文件模板配置 `topology.yaml`。如果有其他组合场景的需求，请根据多个模板自行调整。

场景	配置任务	配置文件模板	拓扑说明
OLTP 业务	部署最小拓扑架构	简单最小配置模板 详细最小配置模板	最小集群拓扑，包括 tidb-server、tikv-server、pd-server。
HTAP 业务	部署 TiFlash 拓扑架构	简单 TiFlash 配置模版 详细 TiFlash 配置模版	在最小拓扑的基础上部署 TiFlash。TiFlash 是列式存储引擎，已经逐步成为集群拓扑的标配。
使用 TiCDC 进行增量同步	部署 TiCDC 拓扑架构	简单 TiCDC 配置模板 详细 TiCDC 配置模板	在最小拓扑的基础上部署 TiCDC。TiCDC 支持多种下游：TiDB、MySQL、Kafka、MQ、Confluent 和存储服务。
使用 Spark 的 OLAP 业务	部署 TiSpark 拓扑架构	简单 TiSpark 配置模板 详细 TiSpark 配置模板	在最小拓扑的基础上部署 TiSpark 组件。TiSpark 是 PingCAP 为解决用户复杂 OLAP 需求而推出的产品。TiUP cluster 组件对 TiSpark 的支持目前为实验特性。
单台机器，多个实例	混合部署拓扑架构	简单混部配置模板 详细混部配置模板	也适用于单机多实例需要额外增加目录、端口、资源配比、label 等配置的场景。
跨机房部署 TiDB 集群	跨机房部署拓扑架构	跨机房配置模板	以典型的两地三中心架构为例，介绍跨机房部署架构，以及需要注意的关键设置。

- ⓘ

注意
- 对于需要全局生效的参数，请在配置文件中 `server_configs` 的对应组件下配置。
 - 对于需要某个节点生效的参数，请在具体节点的 `config` 中配置。
 - 配置的层次结构使用 `.` 表示。如：`log.slow-threshold`。更多格式参考 [TiUP 配置参数模版](#)。
 - 如果需要指定在目标机创建的用户组名，可以参考[这个例子](#)。

更多参数说明，请参考：

- TiDB [config.toml.example](#)
- TiKV [config.toml.example](#)
- PD [config.toml.example](#)
- TiFlash [config.toml.example](#)

第 4 步：执行部署命令

- ⓘ

注意
- 通过 TiUP 部署集群时用于初始化的用户（通过 `--user` 指定），可以使用密钥或者交互密码的方式进行安全认证：
- 如果使用密钥方式，可以通过 `-i` 或者 `--identity_file` 指定密钥的路径。
 - 如果使用密码方式，可以通过 `-p` 进入密码交互窗口。
 - 如果已经配置免密登录目标机，则不需填写认证。
- TiUP 用于实际执行相关进程的用户和组（通过 `topology.yaml` 指定，默认值为 `tidb`），一般情况下会在目标机器上自动创建，但以下情况例外：
- `topology.yaml` 中设置的用户名在目标机器上已存在。
 - 在命令行上使用了参数 `--skip-create-user` 明确指定跳过创建用户的步骤。
- 无论 `topology.yaml` 中约定的用户和组是否被自动创建，TiUP 都会自动生成一对 ssh key，并为每台机器的该用户设置免密登录。在此后的操作中都会使用这个用户和 ssh key 去管理机器，而用于初始化的用户和密码在部属完成后不再被使用。

执行部署命令前，先使用 `check` 及 `check --apply` 命令检查和自动修复集群存在的潜在风险：

1. 检查集群存在的潜在风险：

```
tiup cluster check ./topology.yaml --user root [-p] [-i /home/root/.ssh/gcp_rsa]
```

2. 自动修复集群存在的潜在风险：

```
tiup cluster check ./topology.yaml --apply --user root [-p] [-i /home/root/.ssh/gcp_rsa]
```

3. 部署 TiDB 集群：

```
tiup cluster deploy tidb-test v8.5.1 ./topology.yaml --user root [-p] [-i /home/root/.s
```

以上部署示例中：

- `tidb-test` 为部署的集群名称。
- `v8.5.1` 为部署的集群版本，可以通过执行 `tiup list tidb` 来查看 TiUP 支持的最新可用版本。
- 初始化配置文件为 `topology.yaml`。

- `--user root` 表示通过 `root` 用户登录到目标主机完成集群部署，该用户需要有 `ssh` 到目标机器的权限，并且在目标机器有 `sudo` 权限。也可以用其他有 `ssh` 和 `sudo` 权限的用户完成部署。
- `[-i]` 及 `[-p]` 为可选项，如果已经配置免密登录目标机，则不需填写。否则选择其一即可，`[-i]` 为可登录到目标机的 `root` 用户（或 `--user` 指定的其他用户）的私钥，也可使用 `[-p]` 交互式输入该用户的密码。

预期日志结尾输出 `Deployed cluster `tidb-test` successfully` 关键词，表示部署成功。

第 5 步：查看 TiUP 管理的集群情况

```
tiup cluster list
```

TiUP 支持管理多个 TiDB 集群，该命令会输出当前通过 TiUP cluster 管理的所有集群信息，包括集群名称、部署用户、版本、密钥信息等。

第 6 步：检查部署的 TiDB 集群情况

例如，执行如下命令检查 `tidb-test` 集群情况：

```
tiup cluster display tidb-test
```

预期输出包括 `tidb-test` 集群中实例 ID、角色、主机、监听端口和状态（由于还未启动，所以状态为 `Down/inactive`）、目录信息。

第 7 步：启动集群

安全启动是 TiUP cluster 从 v1.9.0 起引入的一种新的启动方式，采用该方式启动数据库可以提高数据库安全性。推荐使用安全启动。

安全启动后，TiUP 会自动生成 TiDB root 用户的密码，并在命令行界面返回密码。

注意

- 使用安全启动方式后，不能通过无密码的 `root` 用户登录数据库，你需要记录命令行返回的密码进行后续操作。
- 该自动生成的密码只会返回一次，如果没有记录或者忘记该密码，请参照[忘记 root 密码修改密码](#)。

方式一：安全启动

```
tiup cluster start tidb-test --init
```

预期结果如下，表示启动成功。

```
Started cluster `tidb-test` successfully.
The root password of TiDB database has been changed.
The new password is: 'y_+3Hwp=*Awz8971s6'.
Copy and record it to somewhere safe, it is only displayed once, and will not be stored.
The generated password can NOT be got again in future.
```

方式二：普通启动

```
tiup cluster start tidb-test
```

预期结果输出 `Started cluster `tidb-test` successfully`，表示启动成功。使用普通启动方式后，可通过无密码的 `root` 用户登录数据库。

第 8 步：验证集群运行状态

```
tiup cluster display tidb-test
```

预期结果输出：各节点 `Status` 状态信息为 `Up` 说明集群状态正常。

探索更多

如果你已同时部署了 [TiFlash](#)，接下来可参阅以下文档：

- [使用 TiFlash](#)
- [TiFlash 集群运维](#)
- [TiFlash 报警规则与处理方法](#)
- [TiFlash 常见问题](#)

如果你已同时部署了 [TiCDC](#)，接下来可参阅以下文档：

- [Changefeed 概述](#)
- [管理 Changefeed](#)
- [TiCDC 故障处理](#)

- [TiCDC 常见问题](#)

如果你想在不断线上服务的情况下扩容或缩容 TiDB 集群，请参阅[使用 TiUP 扩容缩容 TiDB 集群](#)。

使用 TiUP 部署 TiDB 集群 更新于 2025/1/22 上午11:34:56: `tiup: add '--user' (#15664) (#19650)`

文档内容是否有帮助?

☐ 是

☐ 否

产品

TiDB

TiDB Cloud Serverless

TiDB Cloud Dedicated

生态

TiKV

TiFlash

OSS Insight

资源

TiDB 路线图

常见问题解答

开发者手册

博客

Education

支持

社区

联系我们


公司


关于我们


招贤纳士


新闻报道


Stay Connected

















© 2025 PingCAP. All Rights Reserved. / [隐私政策](#) / [安全合规](#)