云资源自动化 for Terraform

常见问题

产品文档

文档目录

常见问题

签名错误

最近更新时间：2023-03-07 10:35:48

现象描述

下载或更新 Provider 出现如下报错信息：

```
Initializing the backend...

Initializing provider plugins...
- Checking for available provider plugins on https://releases.hashicorp.com...

Error installing provider "tencentcloud": openpgp: signature made by unknown enti
ty.

Terraform analyses the configuration and state and automatically downloads
plugins for the providers used. However, when attempting to download this
plugin an unexpected error occured.

This may be caused if for some reason Terraform is unable to reach the
plugin repository. The repository may be unreachable if access is blocked
by a firewall.

If automatic installation is not possible or desirable in your environment,
you may alternatively manually install plugins by downloading a suitable
distribution package and placing the plugin's executable file in the
following directory:
terraform.d/plugins/darwin_amd64
```

问题定位

参考 官网说明，可得知是 Terraform 版本过低导致更新失败。

处理步骤

更新 Terraform 版本大于等于0.11.15版本即可。

Init 加速

最近更新时间：2023-05-30 10:06:25

现象描述

下载或更新 Provider 出现如下报错信息：

```
Initializing the backend...
Initializing provider plugins...
- Finding latest version of tencentcloudstack/tencentcloud...
╷
│ Error: Failed to install provider
│
│ Error while installing tencentcloudstack/tencentcloud v1.78.5: could not query pr
│ "https://github.com/tencentcloudstack/terraform-provider-tencentcloud/releases/do
╵
```

或者下载很慢：

```
Initializing the backend...

Initializing provider plugins...
- Finding latest version of tencentcloudstack/tencentcloud...
╷
│ Error: Failed to query available provider packages
│
│ Could not retrieve the list of available versions for provider tencentcloudstack/
│ (Client.Timeout exceeded while awaiting headers)
╵
```

问题定位

Terraform 的默认镜像源 `registry.terraform.io` 部署在国外，在国内拉取镜像时可能存在网络问题导致拉取速度慢、甚至无法成功拉取。为解决此问题，您可以使用腾讯云提供的 镜像源。

操作步骤

创建 Terraform CLI 配置文件

按需创建 `.terraformrc` 或 `terraform.rc` 配置文件，您可以将其与其他配置文件放在一个文件夹内，位置取决于主机操作系统：
在 Windows 上，该文件必须命名 `terraform.rc` ，并放置在相关用户的 `%APPDATA%` 目录中。该目录的物理位置取决于您的 Windows 版本和系统配置；可以在 PowerShell 中使用 `$env:APPDATA` 查找其在系统上的位置。
在所有其他系统上，文件必须命名 `.terraformrc` ，并直接放在相关用户的根目录中。Terraform CLI 配置文件的位置也可以使用 `TF_CLI_CONFIG_FILE` 环境变量来指定。任何此类文件都应遵循命名模式 `*.tfrc` ，详情见 Terraform 官网指引。
以 MacOS 为例，创建 `.terraformrc` 文件在用户根目录下，内容如下：

在您保存 Terraform 配置文件的目录下，执行 `terraform init` 命令初始化配置。

```
provider_installation {
    network_mirror {
        url = "https://mirrors.tencent.com/terraform/"
    }
}
```

运行 terraform init

在您保存 Terraform 配置文件的目录下，执行 `terraform init` 命令初始化配置。

此步骤中，Terraform 会自动检测配置文件中的 provider 字段，并下载最新版本的模块和插件。若打印如下信息，则表示初始化成功。

```
Initializing the backend...

Initializing provider plugins...
- Finding latest version of tencentcloudstack/tencentcloud...
- Installing tencentcloudstack/tencentcloud v1.78.5...
- Installed tencentcloudstack/tencentcloud v1.78.5 (verified checksum)

Terraform has created a lock file .terraform.lock.hcl to record the provider
```

```
selections it made above. Include this file in your version control repository
so that Terraform can guarantee to make the same selections by default when
you run "terraform init" in the future.

╷
│ Warning: Incomplete lock file information for providers
│
│ Due to your customized provider installation methods, Terraform was forced to cal
│   - tencentcloudstack/tencentcloud
│
│ The current .terraform.lock.hcl file only includes checksums for darwin_amd64, so
│
│ To calculate additional checksums for another platform, run:
│   terraform providers lock -platform=linux_amd64
│ (where linux_amd64 is the platform to generate)
╵

Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.

If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
```

开启日志追踪

最近更新时间：2023-05-29 15:17:03

操作场景

本文介绍如何在本地开启日志追踪，以获取更详细的日志，便于自查及协助工单处理。

操作步骤

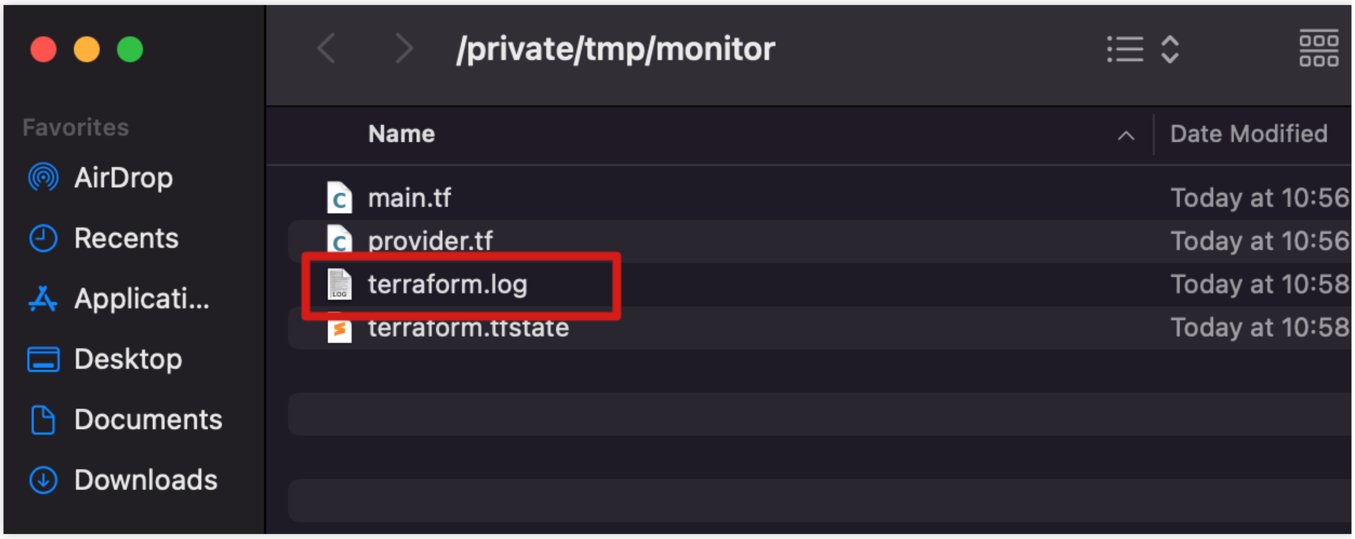
1. 在 CLI 中执行 `terraform apply` 前，可以使用以下命令开启本地日志跟踪：

```
export TF_LOG=DEBUG
export TF_LOG_PATH=./terraform.log
```

2. 开启后，再次执行以下命令。

```
terraform apply/destroy
```

执行完毕后，可查看 Terraform 本地文件夹会生成一个 terraform.log 的文件。文件记录了 tencentcloud provider 定义的日志输出。如下图所示：



Name	Date Modified
main.tf	Today at 10:56
provider.tf	Today at 10:56
terraform.log	Today at 10:58
terraform.tfstate	Today at 10:58

示例

以下为执行出错示例，及分析定位问题的过程。

本例中创建了一个 K8S cluster 并挂载一台已经存在的 CVM 作节点。

```
➜ terraform apply

2021/12/09 17:53:02 [WARN] Log levels other than TRACE are currently unreliable, an
  Use TF_LOG=TRACE to see Terraform's internal logs.
  ----
data.tencentcloud_instance_types.default: Refreshing state...
data.tencentcloud_cbs_storages.storages: Refreshing state...
data.tencentcloud_vpc_subnets.vpc2: Refreshing state...
data.tencentcloud_images.default: Refreshing state...
data.tencentcloud_vpc_subnets.vpc: Refreshing state...
An execution plan has been generated and is shown below.
```

```
Resource actions are indicated with the following symbols:
  + create

Terraform will perform the following actions:

  # tencentcloud_kubernetes_cluster.managed_cluster will be created
  + resource "tencentcloud_kubernetes_cluster" "managed_cluster" {
      + certification_authority      = (known after apply)
      + claim_expired_seconds        = 300
      + cluster_as_enabled           = false
      + cluster_cidr                 = "10.1.0.0/16"
      + cluster_deploy_type          = "MANAGED_CLUSTER"
      + cluster_desc                 = "test cluster desc"
      + cluster_external_endpoint    = (known after apply)
      + cluster_internet             = false
      + cluster_intranet             = false
      + cluster_ipvs                 = true
      + cluster_max_pod_num          = 32
      + cluster_max_service_num      = 32
      + cluster_name                 = "keep"
      + cluster_node_num             = (known after apply)
      + cluster_os                   = "ubuntu16.04.1 LTSx86_64"
      + cluster_os_type              = "GENERAL"
      + cluster_version              = "1.10.5"
      + container_runtime            = "docker"
      + deletion_protection          = false
      + domain                       = (known after apply)
      + id                           = (known after apply)
      + ignore_cluster_cidr_conflict = false
      + is_non_static_ip_mode        = false
      + kube_config                  = (known after apply)
      + network_type                 = "GR"
      + node_name_type               = "lan-ip"
      + password                     = (known after apply)
      + pgw_endpoint                 = (known after apply)
      + security_policy              = (known after apply)
      + user_name                    = (known after apply)
      + vpc_id                       = "vpc-h70b6b49"
      + worker_instances_list        = (known after apply)

      + worker_config {
          + availability_zone                       = "ap-guangzhou-3"
          + count                                   = 1
          + enhanced_monitor_service                = false
          + enhanced_security_service               = false
          + instance_charge_type                    = "POSTPAID_BY_HOUR"
          + instance_charge_type_prepaid_period     = 1
          + instance_charge_type_prepaid_renew_flag = "NOTIFY_AND_MANUAL_RENEW"
          + instance_name                           = "sub machine of tke"
          + instance_type                           = "S1.SMALL1"
          + internet_charge_type                    = "TRAFFIC_POSTPAID_BY_HOUR"
          + internet_max_bandwidth_out              = 100
          + password                                = (sensitive value)
          + public_ip_assigned                      = true
          + subnet_id                               = "subnet-1uwh63so"
          + system_disk_size                        = 60
          + system_disk_type                        = "CLOUD_SSD"
          + user_data                               = "dGVzdA=="

          + data_disk {
              + disk_size = 50
              + disk_type = "CLOUD_PREMIUM"
            }
        }
    }

  # tencentcloud_kubernetes_cluster_attachment.test_attach will be created
  + resource "tencentcloud_kubernetes_cluster_attachment" "test_attach" {
      + cluster_id      = (known after apply)
      + hostname        = "user"
      + id              = (known after apply)
      + instance_id     = "ins-lmnl6t1g"
      + labels          = {
          + "test1" = "test1"
          + "test2" = "test2"
        }
      + password        = (sensitive value)
      + security_groups = (known after apply)
      + state           = (known after apply)

      + worker_config {
          + docker_graph_path = "/var/lib/docker"
          + is_schedule       = true

          + data_disk {
              + auto_format_and_mount = false
              + disk_size             = 50
              + disk_type             = "CLOUD_PREMIUM"
            }
        }
    }

Plan: 2 to add, 0 to change, 0 to destroy.
```

```
Do you want to perform these actions?
  Terraform will perform the actions described above.
  Only 'yes' will be accepted to approve.

  Enter a value: yes

tencentcloud_kubernetes_cluster.managed_cluster: Creating...

Error: [TencentCloudSDKError] Code=InternalError.CidrConflictWithOtherCluster, Mess

  on main.tf line 424, in resource "tencentcloud_kubernetes_cluster" "managed_clust
 424: resource "tencentcloud_kubernetes_cluster" "managed_cluster" {

```

CLI 提示错误如下：

```
[TencentCloudSDKError] Code=InternalError.CidrConflictWithOtherCluster, Message=Das
```

问题分析及定位：

1. 找到 `requestId: d7dfb178-f081-480a-9bc3-89efc5fb1db5`
2. 打开 `terraform.log` ，搜索该 requestId，找到上下文如下所示：

```
2021-12-09T17:53:20.222+0800 [DEBUG] plugin.terraform-provider-tencentcloud.exe: 20

_CONFLICT_WITH_OTHER_CLUSTER[cidr 10.1.0.0/16 is conflict with cluster id: cls-1zc0

6 is conflict with cluster id: cls-1zc0kpyo], err : CheckCIDRWithVPCClusters failed

2021-12-09T17:53:20.593+0800 [DEBUG] plugin.terraform-provider-tencentcloud.exe: 20
```

3. 分析日志，可定位是创建 K8s cluster 过程中出现问题。示例中是因为 cidr 与已存在的其他 K8s cluster 有冲突造成的。

说明：

若 CLI 提示错误信息不够清晰，或无 requestID 的报错造成定位有困难，可将 tf 项目文件，CLI 提示错误以及其产生的日志 terraform.log 文件通过 提交工单 请求协助。

管理现存资源

最近更新时间：2023-05-29 16:18:12

操作场景

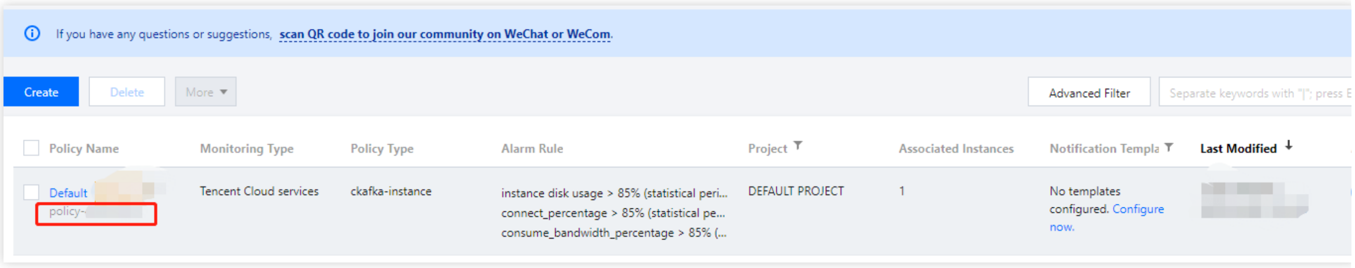
若您在使用 Terraform 管理云资源之前，已经通过腾讯云控制台创建了资源，则可参考本文进行操作，将现存资源使用 Terraform 管理。

操作步骤

使用 Terraform 接管已经存在的资源，实际上在 Terraform 源文件和状态文件里都反映出该资源的状态即可。本文以使用 Terraform 管理现存 PostgreSQL 告警策略为例。

获取资源 ID

1. 登录云监控控制台，选择左侧导航栏中的告警配置 > 告警策略。
2. 找到并记录该策略 ID。如下图所示：


If you have any questions or suggestions, scan QR code to join our community on WeChat or WeCom.
Create
Delete
More
Advanced Filter
Separate keywords with "|"; press E

Policy Name	Monitoring Type	Policy Type	Alarm Rule	Project	Associated Instances	Notification Templa	Last Modified
Default [redacted] policy-[redacted]	Tencent Cloud services	ckafka-instance	instance disk usage > 85% (statistical peri... connect_percentage > 85% (statistical pe... consume_bandwidth_percentage > 85% (...	DEFAULT PROJECT	1	No templates configured. Configure now.	[redacted]

导入资源文件

1. 进入 Terraform 工作目录，执行以下命令，查看 main.tf 内容。

```
tencent-cloud cat main.tf
```

返回结果如下所示：

```
resource "tencentcloud_monitor_alarm_policy" "policy" {}
```

2. 在文件所在目录下执行以下命令，完成初始化工作。

```
terraform init --upgrade
```

返回结果如下所示：

```
Initializing the backend...

Initializing provider plugins...
- Finding latest version of tencentcloudstack/tencentcloud...
- Installing tencentcloudstack/tencentcloud v1.60.22...
- Installed tencentcloudstack/tencentcloud v1.60.22 (signed by a HashiCorp partner,

Partner and community providers are signed by their developers.
If you'd like to know more about provider signing, you can read about it here:
https://www.terraform.io/docs/cli/plugins/signing.html
```

```
Terraform has made some changes to the provider dependency selections recorded
in the .terraform.lock.hcl file. Review those changes and commit them to your
version control system if they represent changes you intended to make.

Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.

If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
```

3. 初始化完成后，执行以下命令，将资源导入状态文件。

```
terraform import tencentcloud_monitor_alarm_policy.policy policy-vor9w72r
```

返回信息如下所示：

```
➜  terraform import tencentcloud_monitor_alarm_policy.policy policy-vor9w72r

tencentcloud_monitor_alarm_policy.policy: Importing from ID "policy-vor9w72r"...
tencentcloud_monitor_alarm_policy.policy: Import prepared!
  Prepared tencentcloud_monitor_alarm_policy for import
tencentcloud_monitor_alarm_policy.policy: Refreshing state... [id=policy-vor9w72r]

Import successful!

The resources that were imported are shown above. These resources are now in
your Terraform state and will henceforth be managed by Terraform.
```

4. 执行以下命令，查看状态文件。

```
cat terraform.tfstate
```

可查看如下所示资源相关信息：

```
{
  "version": 4,
  "terraform_version": "1.1.0",
  "serial": 1,
  "lineage": "35791a73-d371-db51-5871-bfee13426217",
  "outputs": {},
  "resources": [
    {
      "mode": "managed",
      "type": "tencentcloud_monitor_alarm_policy",
      "name": "policy",
        "provider": "provider[\\"registry.terraform.io/tencentcloudstack/tencentcloud
        "instances": [
          {
            "schema_version": 0,
            "attributes": {
              "conditions": [
                {
                  "is_union_rule": 0,
                  "rules": [
                    {
                      "continue_period": 5,
                      "description": "cpu",
                      "filter": [],
                      "is_power_notice": 0,
                      "metric_name": "Cpu",
                      "notice_frequency": 86400,
                      "operator": "gt",
                      "period": 60,
                      "rule_type": "STATIC",
                      "unit": "%",
                      "value": "90"
                    }
                  ]
                }
              ],
              "conditon_template_id": null,
              "create_time": null,
              "enable": 1,
              "event_conditions": [
                {
                  "continue_period": 0,
                  "description": "HASwitch",
                  "filter": [],
                  "is_power_notice": 0,
                  "metric_name": "ha_switch",
                  "notice_frequency": 0,
                  "operator": "",
                  "period": 0,
                  "rule_type": "",
                  "unit": "",
                  "value": ""
                }
              ],
              "id": "policy-vor9w72r",
              "monitor_type": "MT_QCE",
              "namespace": "POSTGRESQL",
              "notice_ids": [
                "notice-l9ziyxw6"
              ],
              "policy_name": "PgSql",
              "project_id": 0,
              "remark": "",
              "trigger_tasks": [],
              "update_time": null
            },
            "sensitive_attributes": [],
            "private": "eyJzY2hlbWFfdmVyc2lvbiI6IjAifQ=="
          }
        ]
      }
    ]
  }
```

更新源文件

1. 执行以下命令，打印资源信息。

```
terraform show
```

返回信息如下所示：

```
# tencentcloud_monitor_alarm_policy.policy:
resource "tencentcloud_monitor_alarm_policy" "policy" {
    enable       = 1
    id           = "policy-vor9w72r"
    monitor_type = "MT_QCE"
    namespace    = "POSTGRESQL"
    notice_ids   = [
        "notice-l9ziyxw6",
    ]
    policy_name  = "PgSql"
    project_id   = 0
    conditions {
        is_union_rule = 0

        rules {
            continue_period  = 5
            description      = "cpu"
            is_power_notice  = 0
            metric_name      = "Cpu"
            notice_frequency = 86400
            operator         = "gt"
            period           = 60
            rule_type        = "STATIC"
            unit             = "%"
            value            = "90"
        }
    }

    event_conditions {
        continue_period  = 0
        description      = "HASwitch"
        is_power_notice  = 0
        metric_name      = "ha_switch"
        notice_frequency = 0
        period           = 0
    }
}
```

2. 将资源代码拷贝至 Terraform 源文件 `tencentcloud.tf` 中。需删除其中不可设置的选项，例如 ID。完成编辑后， `tencentcloud.tf` 文件内容如下所示：

```
provider tencentcloud {}
resource "tencentcloud_monitor_alarm_policy" "policy" {
  enable       = 1
#  id            = "policy-vor9w72r"
  monitor_type = "MT_QCE"
  namespace    = "POSTGRESQL"
  notice_ids   = [
    "notice-l9ziyxw6",
  ]
  policy_name  = "PgSql"
  project_id   = 0
```

```
    conditions {
      is_union_rule = 0

      rules {
        continue_period  = 5
        description      = "cpu"
        is_power_notice  = 0
        metric_name      = "Cpu"
        notice_frequency = 86400
        operator         = "gt"
        period           = 60
        rule_type        = "STATIC"
        unit             = "%"
        value            = "90"
      }
    }

    event_conditions {
      continue_period  = 0
      description      = "HASwitch"
      is_power_notice  = 0
      metric_name      = "ha_switch"
      notice_frequency = 0
      period           = 0
    }
  }
```

校验

执行以下命令，使用当前工作目录下的代码和状态文件执行 refresh。

```
terraform plan
```

返回信息如下所示，可查看 Terraform 已成功接管。

```
tencentcloud_monitor_alarm_policy.policy: Refreshing state... [id=policy-vor9w72r]

No changes. Your infrastructure matches the configuration.

Terraform has compared your real infrastructure against your configuration and foun
```

至此，Terraform 已成功接管现存资源。可通过 destory 来删除该资源，也可通过修改代码的方式对资源进行修改。例如，修改告警阈值后，执行以下命令进行更新。

```
terraform plan
```

返回信息如下所示，可查看修改的 value 会导致 Terraform 提示将会更新这个告警策略。

```
tencentcloud_monitor_alarm_policy.policy: Refreshing state... [id=policy-vor9w72r]

Terraform used the selected providers to generate the following execution plan. Res
  ~ update in-place

Terraform will perform the following actions:

  # tencentcloud_monitor_alarm_policy.policy will be updated in-place
  ~ resource "tencentcloud_monitor_alarm_policy" "policy" {
        id            = "policy-vor9w72r"
        # (6 unchanged attributes hidden)
```

```
      ~ conditions {
            # (1 unchanged attribute hidden)

          ~ rules {
              ~ value            = "90" -> "99"
                # (9 unchanged attributes hidden)
            }
        }

        # (1 unchanged block hidden)
    }

Plan: 0 to add, 1 to change, 0 to destroy.
```

状态锁

最近更新时间：2023-05-29 16:25:34

背景信息

使用 Backend 配置 Terraform 会涉及执行环境和远端状态同步的问题，Terraform 引入了锁的机制避免多个执行环境操作同一个 backend 导致状态不同步，不同厂商均可通过 Terraform 提供的 lock interface 实现自己的状态锁。本文介绍 Cos backend 锁的实现步骤和异常处理。

状态锁与文件写入锁

根据 源码 可知，Cos Backend 执行时会涉及2种锁。
状态锁，以 terraform.tfstate.tflock 文件形式出现，记录当前进程的信息并写入 Backend 存储桶：

}

```
{
    "ID":"1234abcd-1234-cdef-5678-1234567890ab",
    "Operation":"OperationTypePlan",
    "Info":"",
    "Who":"UserName@Host",
    "Version":"1.3.0",
    "Created":"2006-01-02T15:04:05Z07:00",
    "Path":"terraform.tfstate.tflock"
}
```

文件写锁，用来避免多个进程同时对存储桶写入 terraform.tfstate.tflock 文件。Cos Backend 以腾讯云标签形式来实现，格式为：tencentcloud-terraform-lock:xxxxxxxxx（xxxxxxxxx 为 bucket:lockfile 的 MD5），当文件写入成功后自动释放。

上锁步骤

通常情况下，执行 terraform 命令会进行如下步骤：

1. terraform 开始执行（plan / apply 等命令）。
2. Backend 服务对当前账号添加腾讯云标签作为文件写锁。
3. 存储桶检查并写入 terraform.tfstate.tflock 作为状态锁。
4. 删除标签，释放文件写锁。
5. 等待 terraform 执行完毕。
6. 存储桶检查并删除 terraform.tfstate.tflock 取消状态锁。
7. terraform 执行完毕。

异常处理

场景1

当有进程执行 上锁步骤 中的步骤4、5、6、7时，如果其他进程尝试操作同一 Backend，则会抛出 Lock Error，示例如下：

```
│
│ Error: Error acquiring the state lock
│
│ Error message: lock file terraform.tfstate.tflock exists
│ Lock Info:
│   ID:        1234abcd1234cdef56781234567890ab
│   Path:      terraform.tfstate.tflock
│   Operation: OperationTypePlan
│   Who:       UserName@Host
│   Version:   1.1.2
│   Created:   2006-01-02T15:04:05Z07:00
│   Info:
│
│
│ Terraform acquires a state lock to protect the state from being written
│ by multiple users at the same time. Please resolve the issue above and try
│ again. For most commands, you can disable locking with the "-lock=false"
│ flag, but this is not recommended.
╵
```

此时其他执行环境需要等待当前进程执行完毕，或您可按照文中的英文提示加入 `-lock=false` 忽略状态锁重新执行**（不推荐该方式）**。
在此过程中，Terraform 进程意外退出而没有及时释放状态锁，需要您手动解锁：
以上文为例，根据锁 ID 执行以下强行解锁命令，即可解锁。

```
terraform force-unlock -force 1234abcd1234cdef56781234567890ab
```

删除存储桶下的 terraform.tfstate.tflock 文件也可解锁。

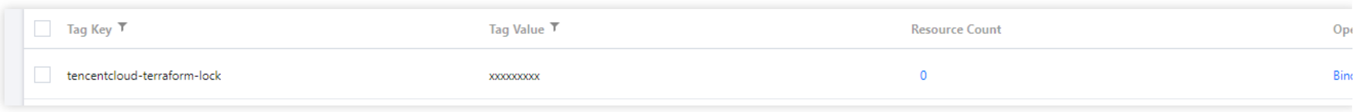
场景2

如在执行 上锁步骤 的步骤3、4时，由于进程中断而退出，标签锁未解开，此时再次进行 terraform 操作会报标签无法创建的异常：

```
│ Error: Error acquiring the state lock
│
│ Error message: 2 errors occurred:
│       * failed to create tag: tencentcloud-terraform-lock -> xxxxxxxxx: [TencentC
│ RequestId=47c57b0b-2491-42cd-9f29-0b14802681e5
│       * lock file terraform/state/terraform.tfstate.tflock not exists
│
│
│
│ Terraform acquires a state lock to protect the state from being written
│ by multiple users at the same time. Please resolve the issue above and try
│ again. For most commands, you can disable locking with the "-lock=false"
│ flag, but this is not recommended.
```

此时，Terraform 没有相关命令释放这个锁，您需要手动删除 tencentcloud-terraform-lock:xxxxxxxxx 标签。您可通过控制台，或调用云 API 接口 DeleteTag 进行删除。控制台操作步骤如下：

1. 登录标签控制台，选择左侧导航栏中的 标签列表。
2. 单击如下图所示标签所在行右侧的删除，并在弹出窗口中单击确定即可。



Tag Key	Tag Value	Resource Count	Op
tencentcloud-terraform-lock	xxxxxxxxx	0	Bin