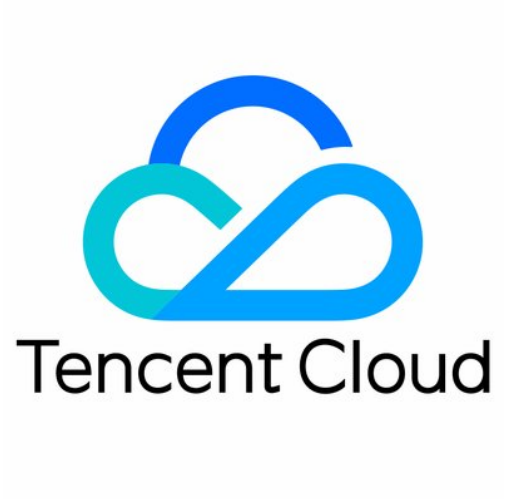
Tencent Cloud EdgeOne

Tool Guide

Product Documentation



Contents

Tool Guide
Diagnostic Tool
Self-service debugging

Last updated：2023-12-18 14:49:55

Feature Introduction

If you need to confirm whether the node cache rules, custom Cache Key, and other configurations currently configured in EdgeOne have taken effect for your resources, EdgeOne provides a self-service debugging tool to help you obtain node cache TTL, whether the resource is cacheable, Cache key, and other information, making it easy for you to debug your business configuration. After enabling self-service debugging, you can initiate a URL request from a specified client IP, carrying the `EO-Debug-Headers: all` header in the request, and view whether the resource is cached in the node, the corresponding Cache Key value, and cache time based on the returned response headers.

```
curl -voa 'http://www.example.com/test.jpg?a=1' -H 'EO-Debug-Headers: all
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  C
                                 Dload  Upload   Total   Spent    Left  S
  0     0    0     0    0     0      0      0 --:--:-- --:--:-- --:--:--
.
* TCP_NODELAY set
* Connected to www.example.com (                ) port 80 (#0)
> GET /test.jpg?a=1 HTTP/1.1
> Host: www.example.com
> User-Agent: curl/7.62.0
> Accept: */*
> EO-Debug-Headers: all
>
< HTTP/1.1 200 OK
< Last-Modified: Wed, 07 Dec 2022 12:32:32 GMT
< Etag: "639087e0-4"
< Server: nginx/1.20.2
< Date: Thu, 29 Jun 2023 12:19:58 GMT
< Content-Type: image/jpeg
< Content-Length: 4
< Accept-Ranges: bytes
< Connection: keep-alive
< EO-LOG-UUID: 6570353728066144953
< EO-Cache-Status: HIT
< EO-Debug-Status: on
< EO-Debug-CacheKey: www.example.com/test.jpg a=1
< EO-Debug-Cacheable: yes
< EO-Debug-CacheTTL: 00d00h10m00s
{ [4 bytes data]
100     4  100     4    0     0     47      0 --:--:-- --:--:-- --:--:--
* Connection #0 to host www.example.com left intact
```

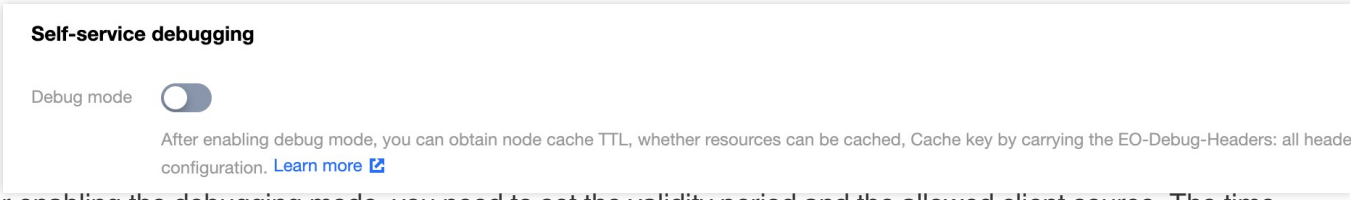
Usage Scenarios

If you have configured more complex cache rules and custom cache keys in the rule engine of the console, and need to verify whether the configuration is effective, you can use this feature for verification.

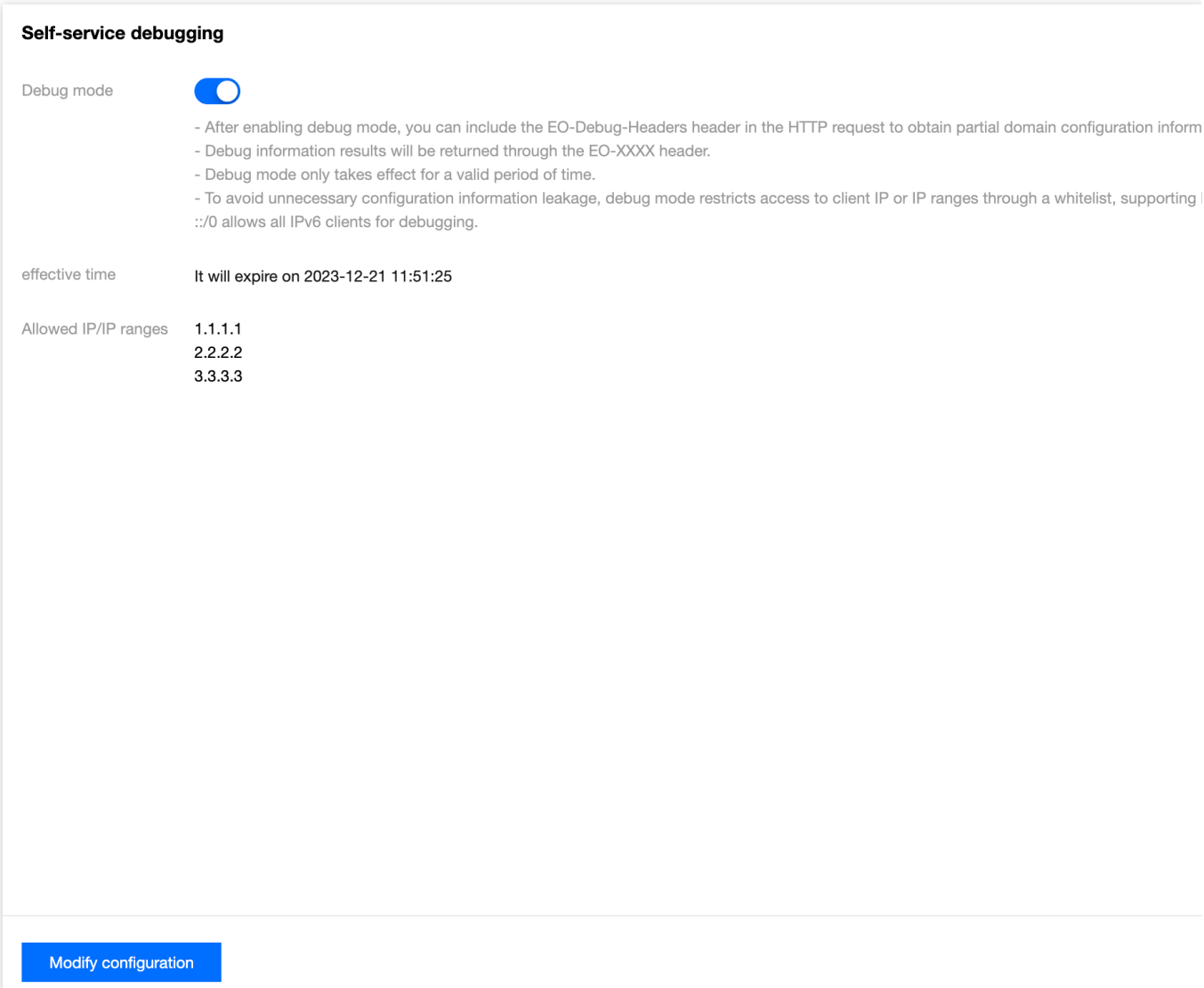
Directions

For example, the domain name `www.example.com` under the current site `example.com` has been configured to cache `.jpg` suffix files in EdgeOne nodes for 600 seconds; the cache Cache Key is configured to retain the specified parameter a as the cache key. After the configuration is completed, you need to verify whether the current configuration has taken effect, and you can follow the steps below to verify:

1. Log in to the EdgeOne console, click on the site list in the left menu bar, and click on the site to be configured in the site list.
2. On the site details page, click on **Diagnostic Tools > Self-service Debugging**.
3. On the self-service debugging page, click the "switch" to enable the self-service debugging feature.



4. After enabling the debugging mode, you need to set the validity period and the allowed client source. The time range is 1-365 days, with a default of 7 days. The client IP allows for the input of 100 entries, accommodating both IPv4 and IPv6 IP/IP segments. The notation 0.0.0.0/0 signifies the permission for all IPv4 clients to execute debugging, while ::/0 indicates the allowance for all IPv6 clients to carry out debugging.



5. Click **Save**, and the allowed client IPs can debug within the effective time.
6. Initiate a curl request for verification from the specified client IP source in a Mac/Linux environment, for example:

`curl -voa 'http://www.example.com/test.jpg?a=1' -H 'EO-Debug-Headers: all'` 。The request result is as follows：

```
curl -voa 'http://www.example.com/test.jpg?a=1' -H 'EO-Debug-Headers: all
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  C
                                 Dload  Upload   Total   Spent    Left  S
  0     0    0     0    0     0      0      0 --:--:-- --:--:-- --:--:--
.
* TCP_NODELAY set
* Connected to www.example.com (                ) port 80 (#0)
> GET /test.jpg?a=1 HTTP/1.1
> Host: www.example.com
> User-Agent: curl/7.62.0
> Accept: */*
> EO-Debug-Headers: all
>
< HTTP/1.1 200 OK
< Last-Modified: Wed, 07 Dec 2022 12:32:32 GMT
< Etag: "639087e0-4"
< Server: nginx/1.20.2
< Date: Thu, 29 Jun 2023 12:19:58 GMT
< Content-Type: image/jpeg
< Content-Length: 4
< Accept-Ranges: bytes
< Connection: keep-alive
< EO-LOG-UUID: 6570353728066144953
< EO-Cache-Status: HIT
< EO-Debug-Status: on
< EO-Debug-CacheKey: www.example.com/test.jpg a=1
< EO-Debug-Cacheable: yes
< EO-Debug-CacheTTL: 00d00h10m00s
{ [4 bytes data]
100     4  100     4    0     0     47      0 --:--:-- --:--:-- --:--:--
* Connection #0 to host www.example.com left intact
```

In the response headers, you can see the corresponding Cache Key, cache status, and cache time for this request, which is consistent with the configuration in the example, indicating that the current configuration has taken effect.

Related References

When the self-service debugging mode is enabled, the debug header explanations in the response are as follows:

Header Name	Configurations	Meaning of the returned value
EO-Debug-Status	Indicates whether the self-service debugging mode is enabled.	on: activated, and the request client IP is within the allowlist & the request time is within the validity period;

		off: Off, or activated but the request time is beyond the validity period; forbidden: activated, but the request client IP is not in the allowlist.
EO-Debug-Cacheable	The Request URL of this request, according to the configured EdgeOne node cache TTL, the final cacheable status of the Request URL resource in EdgeOne nodes.	yes：cacheable content no: non-cacheable content
EO-Debug-CacheKey	The Request URL of this request, according to the custom Cache key, the final Cache key generated for the Request URL resource in EdgeOne nodes.	For example: `www.example.com/test.jpg a=1,` indicating the Cache Key generated for the Request URL resource in EdgeOne
EO-Debug-CacheTTL	The Request URL of this request, according to the configured EdgeOne node cache TTL, the final cache TTL duration of the Request URL resource in EdgeOne nodes.	List values, including numbers and time units. d stands for days, h stands for hours, m stands for minutes, and s stands for seconds, for example: 3d0h0m0s means the cache TTL is 3 days; 0d0h5m0s means the cache is 5 minutes; 0d0h0m5s means the cache is 5 seconds.

Speed Test Tools
Real User Monitoring

Last updated：2024-01-25 11:03:32

Overview

Real User Monitoring is a feature interconnected with EdgeOne. It provides one-stop frontend monitoring solutions. You only need to install its SDK to your project and complete simple configuration, and then it will take care of the user page quality in an all-around manner by monitoring the page performance and frontend quality in real time, truly enabling cost-effective usage and non-intrusive monitoring.

Note:

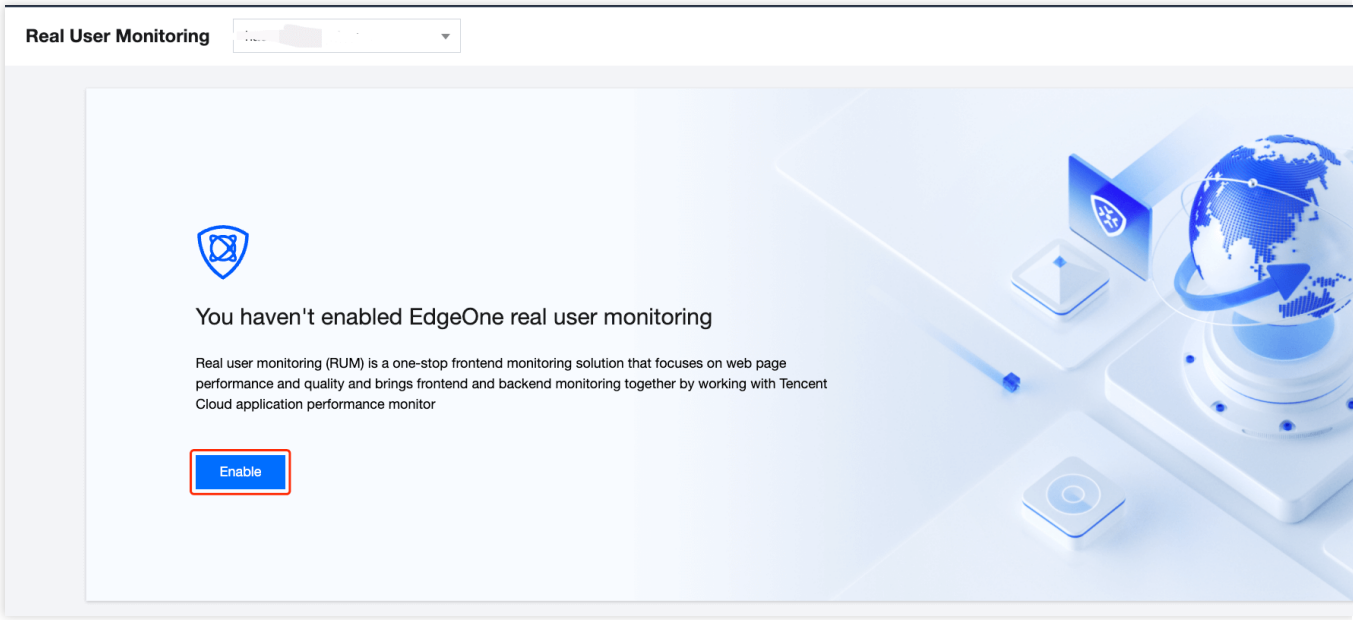
RUM provides a free tier of 500,000 reports per day for each application. Reports exceeding the free tier (500,000) will be billed. The fees are not part of your EdgeOne plan but are charged by RUM. For billing details, see Billing Overview.

Use Cases

Page performance analysis: RUM offers metrics such as firstScreenTime, TCP connection establishment duration, time to first byte (TTFB), and SSL handshake duration. In addition, it supports latest Web Vitals standards, Google's webpage loading speed and experience metrics, helping you optimize the user experience in an all-around manner.
User access analysis: RUM displays the business PV/UV and top access metrics of each page. It analyzes the user access data in various dimensions including network, browser, and region, so that you can stay on top of and analyze the user access information.
Static resource speed test: RUM supports different types of resource speed tests on image loading, CDN resource operation, etc., so you can view diverse information such as resources used on a page and loading duration of each resource.

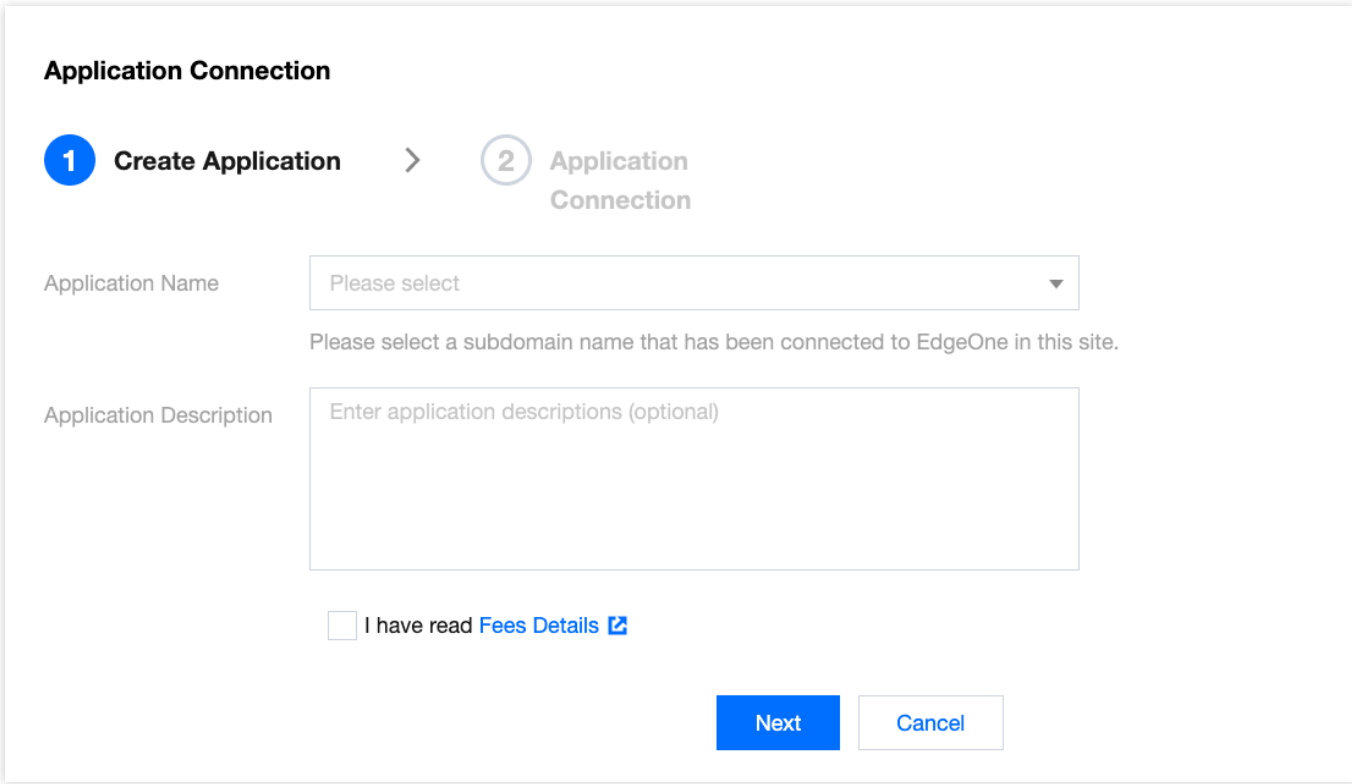
Directions

1. Log in to the EdgeOne console and click **Speed Test Tools** > **Real User Monitoring** on the left sidebar.
2. If you enter the **Real User Monitoring** page for the first time, as this feature is based on EdgeOne and RUM, you need to click **Enable** to grant the relevant permissions.



3. On the **Real User Monitoring** page, click **Application connection**.

4. In the **Application connection** window, enter the application name and description, select **I have understood the billing details**, and click **Next**.

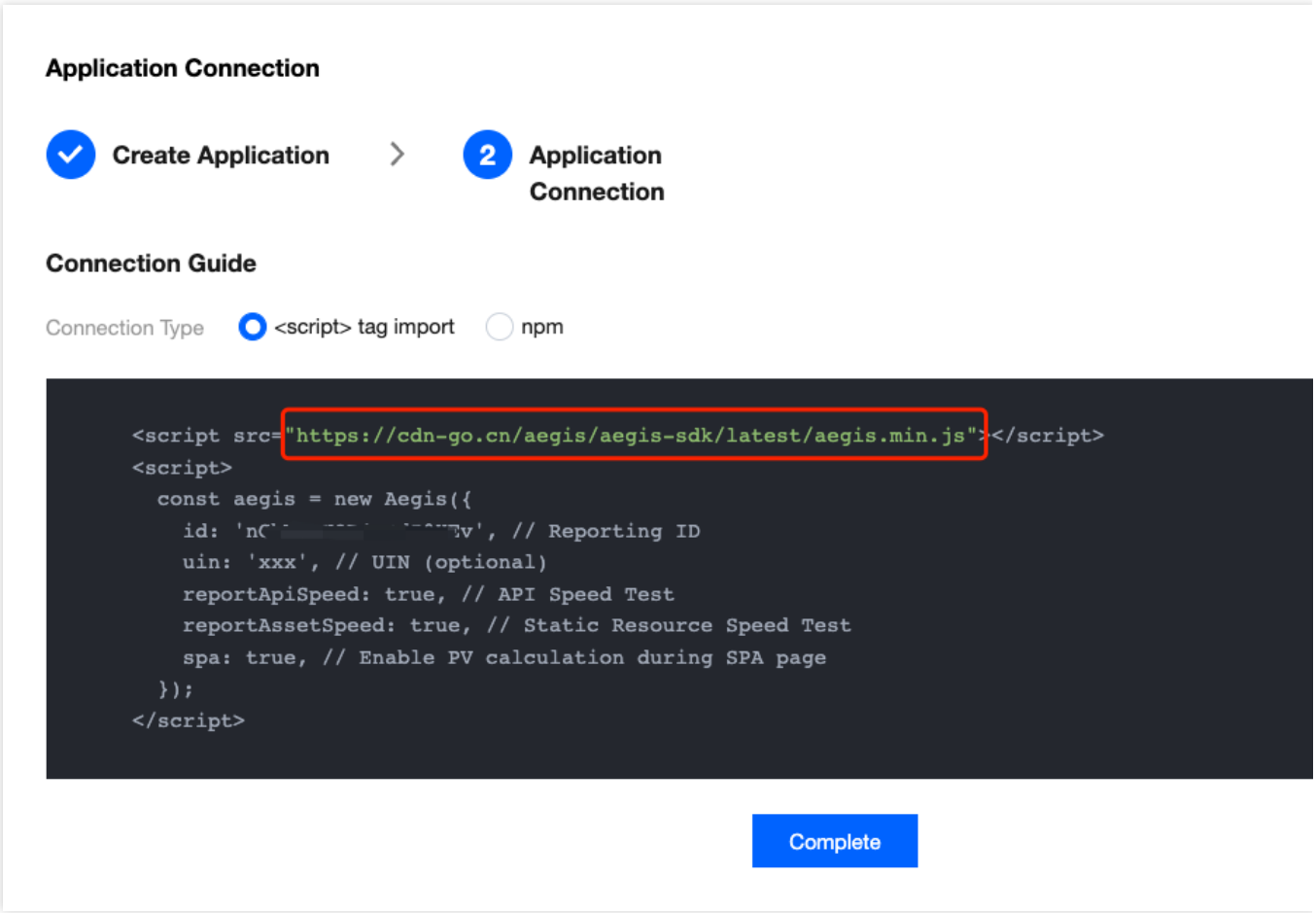


5. Install the SDK based on the connection type.

Install the SDK by importing the `<script>` tag

5.1.1 On the connection guide page, copy the provided `<script>` tag code.

5.1.2 Import the code below `<script> tag import` into the `<head></head>` tags of the site to be monitored.



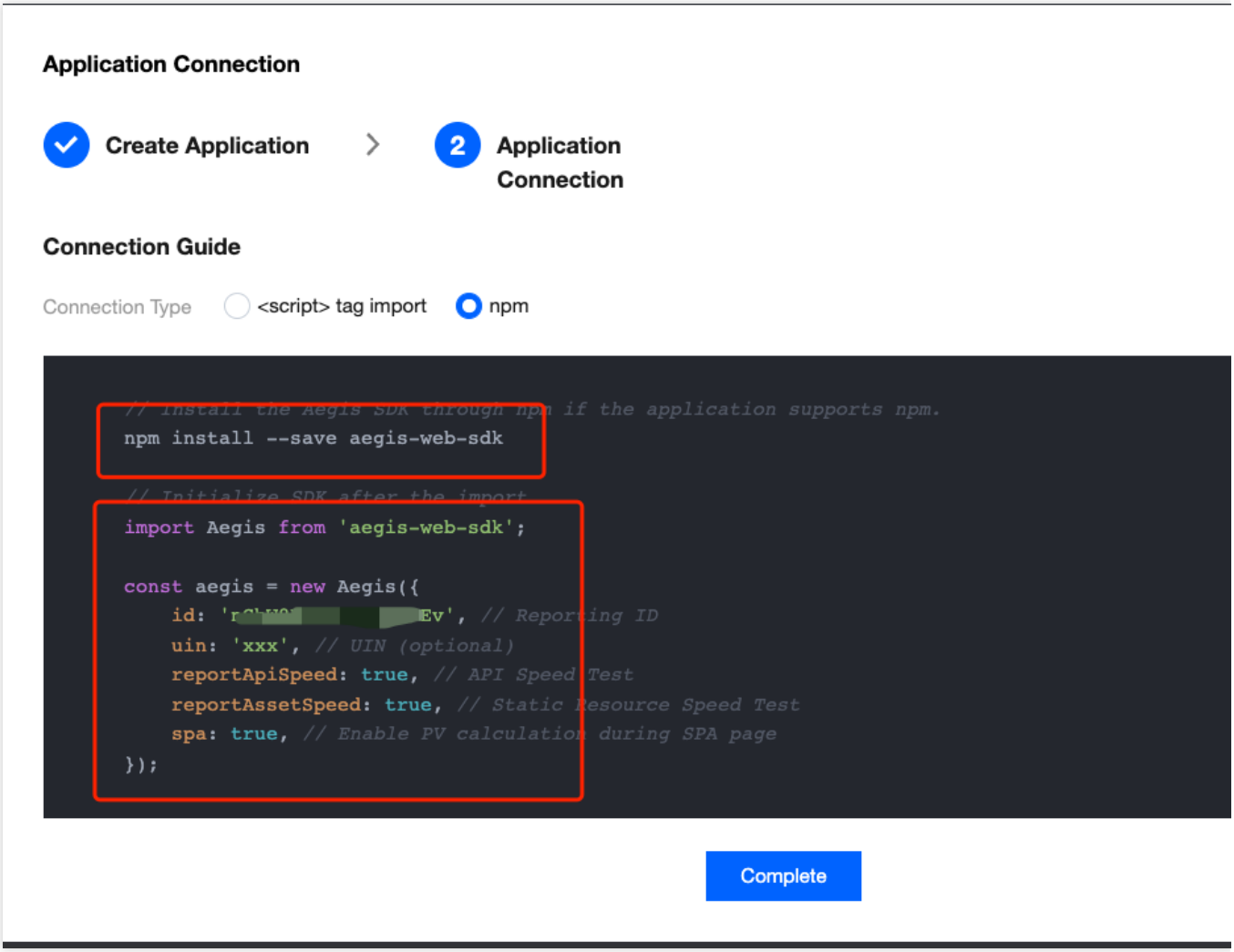
Note:

This connection method uses the “h3-Q050” protocol, where `cache-control` is `max-age=666` by default. To modify `cache-control`, you can add the `max_age` parameter, such as `<script src="https://cdn-go.cn/aegis/aegis-sdk/latest/aegis.min.js?max_age=3600"></script>`.

Install the SDK through npm

5.1 On the connection guide page, copy the first command line to import `aegis sdk` into your development environment.

5.2 Then, copy the provided code to initialize the SDK in your JavaScript code.



Data Monitoring

After performing the above connection steps, go to the **Page performance**, **Page view**, and **Static resource** pages to view the relevant data.

Page performance

The **Page performance** module supports multidimensional page performance analysis. You can analyze key page performance metrics such as firstScreenTime and request response through various views including performance change trend chart, page loading waterfall plot, and regional view. For more information, see Page Performance.

Page view

The **Page view** module displays the page view information such as UV, PV, WAU and MAU, and supports multidimensional page access analysis. For more information, see Page View.

Static resource

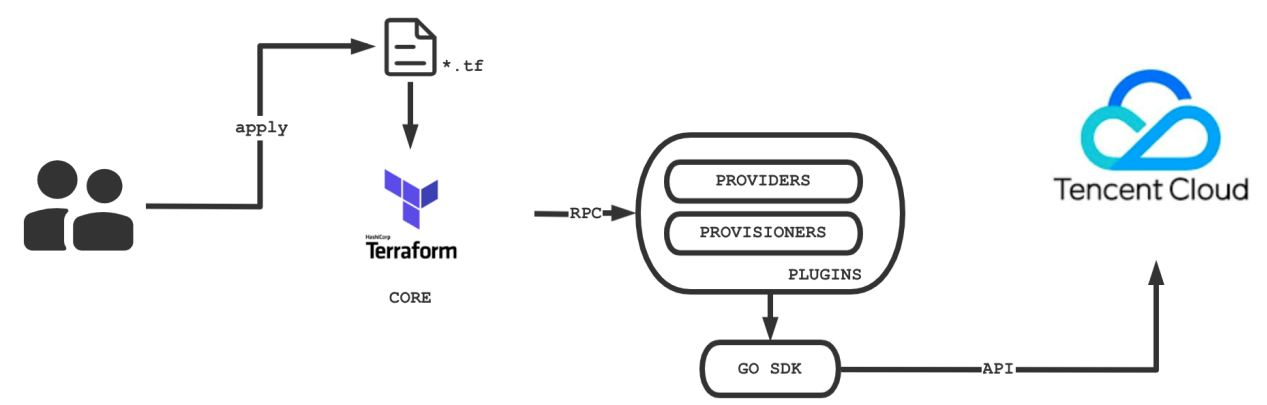
Frontend HTML pages mainly contain the following static resources: JavaScript, CSS, and image files. If such files fail to load, or it takes a long time to load them, the page will be affected or even crash. To address these problems, static resource monitoring helps you analyze the frontend static resource status. For more information, see Static Resource.

Terraform Overview

Last updated：2024-01-25 11:20:13

Terraform Overview

Terraform is an open-source resource orchestration tool written in Go and running on the client. It is highly scalable based on the HashiCorp Plugin architecture. Currently, Tencent Cloud implements the TencentCloud Provider based on Terraform plugin to manage Tencent Cloud resources through Terraform. The schematic diagram is as follows:



Based on tencentcloud-sdk-go, TencentCloud Provider offers more than 183 resources and 158 data sources across over 30 products, covering compute, storage, network, container service, load balancing, middleware, database, and cloud monitoring to meet your basic needs for cloudification.
For a quick start on Terraform, see tencentcloud_teo_zone and terraform-provider-tencentcloud/examples/tencentcloud-teo/.

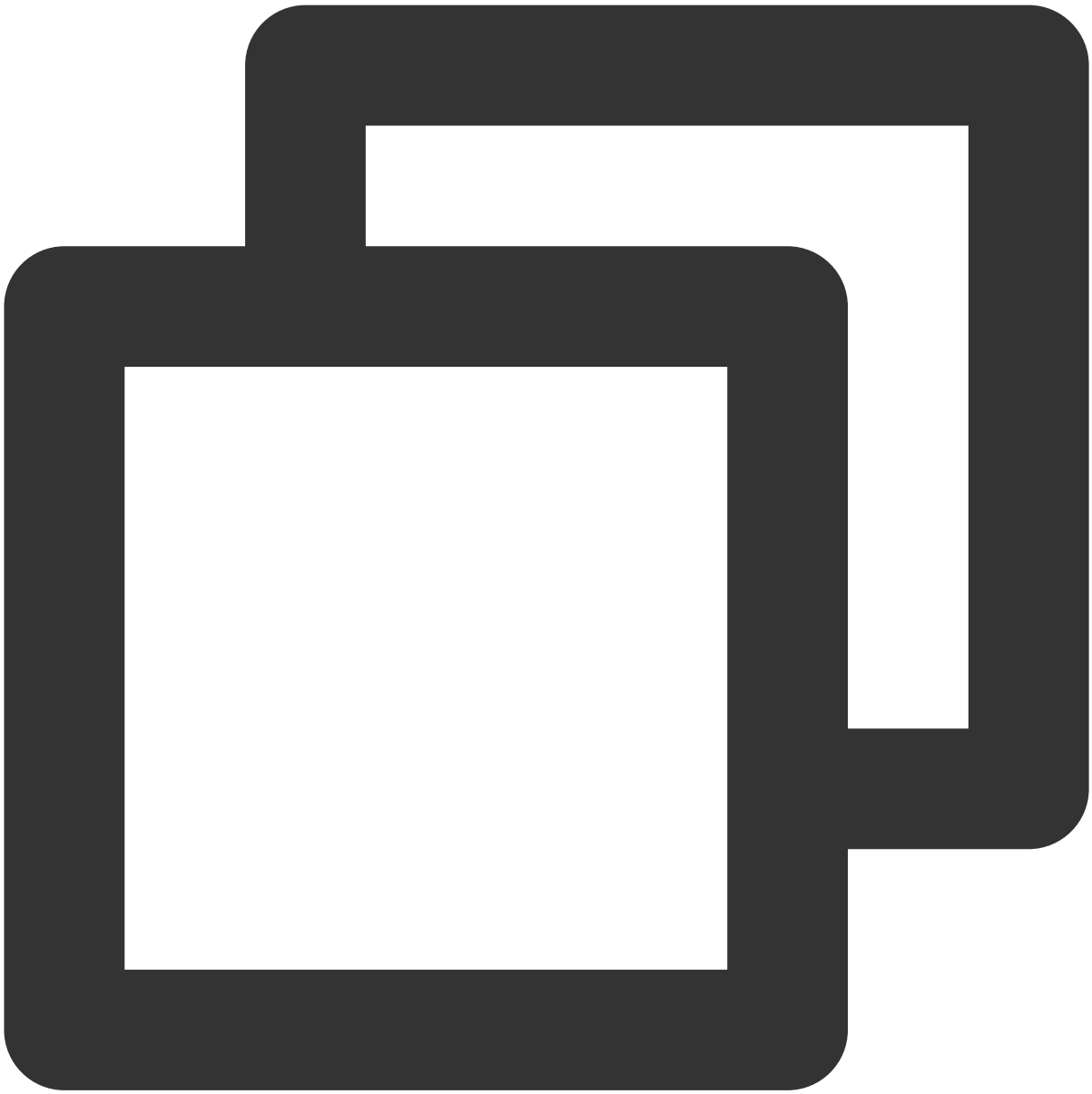
Terraform Strengths

Multi-cloud orchestration

Terraform is suitable for multi-cloud solutions where you can deploy similar infrastructures in Tencent Cloud, other cloud providers, or local IDCs. You can manage resources from different cloud providers at the same time using the same tools and similar configuration files.

Infrastructure and code

You can use the high-level configuration syntax HCL to describe an infrastructure, so that it can be codified and versioned for sharing and reuse as shown in the following example:



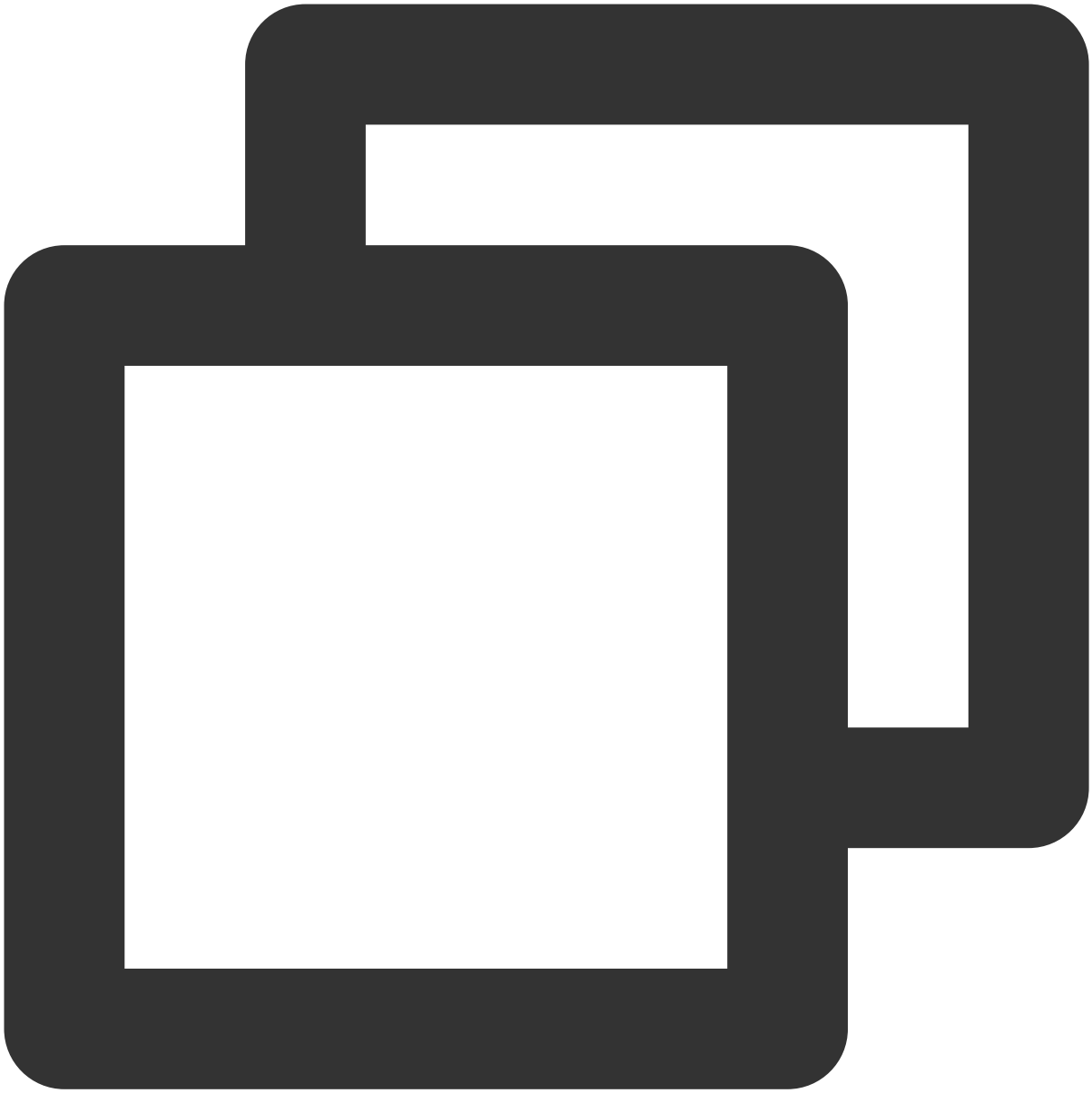
```
# Create the `example.com` site with NS access
resource "tencentcloud_teo_zone" "zone" {
  zone_name      = "example.com"
  # Query available plans by `zone_available_plans`
  plan_type      = "<your-plan-type>"
  type           = "full"
  paused         = false
  cname_speed_up = "enabled"
```

```
}

# Create the DNS record of `example.com`
resource "tencentcloud_teo_dns_record" "dns_record" {
  zone_id     = tencentcloud_teo_zone.zone.id
  type        = "A"
  name        = "example.com"
  # Enable the CDN acceleration service
  mode        = "proxied"
  content     = "<your-backend-ip>"
  ttl         = 60
}
```

Execution plan

Terraform has a "planning" step. It runs the `terraform plan` command to generate an execution plan, which shows the state of Terraform when `apply` is called. This allows you to avoid incidents when the infrastructure is manipulated on Terraform, as shown in the following example:



```
Terraform used the selected providers to generate the following execution plan. Res
  + create

Terraform will perform the following actions:

  # tencentcloud_teo_dns_record.dns_record will be created
  + resource "tencentcloud_teo_dns_record" "dns_record" {
      + cname         = (known after apply)
      + content       = "<your-backend-ip>"
      + created_on    = (known after apply)
      + domain_status = (known after apply)
```

```
      + id             = (known after apply)
      + locked         = (known after apply)
      + mode           = "proxied"
      + modified_on    = (known after apply)
      + name           = "example.com"
      + priority       = (known after apply)
      + type           = "A"
      + status         = (known after apply)
      + ttl            = 60
      + zone_id        = (known after apply)
    }

  # tencentcloud_teo_zone.zone will be created
  + resource "tencentcloud_teo_zone" "zone" {
      + area                    = (known after apply)
      + cname_speed_up          = "enabled"
      + cname_status            = (known after apply)
      + created_on              = (known after apply)
      + id                      = (known after apply)
      + modified_on             = (known after apply)
      + name                    = "example.com"
      + name_servers            = (known after apply)
      + original_name_servers   = (known after apply)
      + paused                  = false
      + plan_type               = "sta"
      + status                  = (known after apply)
      + type                    = "full"
      + vanity_name_servers_ips = (known after apply)

      + vanity_name_servers {
          + servers = (known after apply)
          + switch  = (known after apply)
        }
    }

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

------------------------------------------------------------------------

Note: You didn't use the -out option to save this plan, so Terraform can't guarante
```

Auto change

You can apply complex change sets to your infrastructure with minimal manual intervention. With the execution plan and resource topology mentioned above, you can get an accurate picture of Terraform dynamics and avoid possible human errors.

Remote State Management

Terraform introduces the concept of backend, a remote state storage mechanism. Currently, Tencent Cloud can manage your tfState files through COS to avoid storing files locally and causing file losses. In addition, remote storage makes it possible for multiple users to manage Terraform resources concurrently.

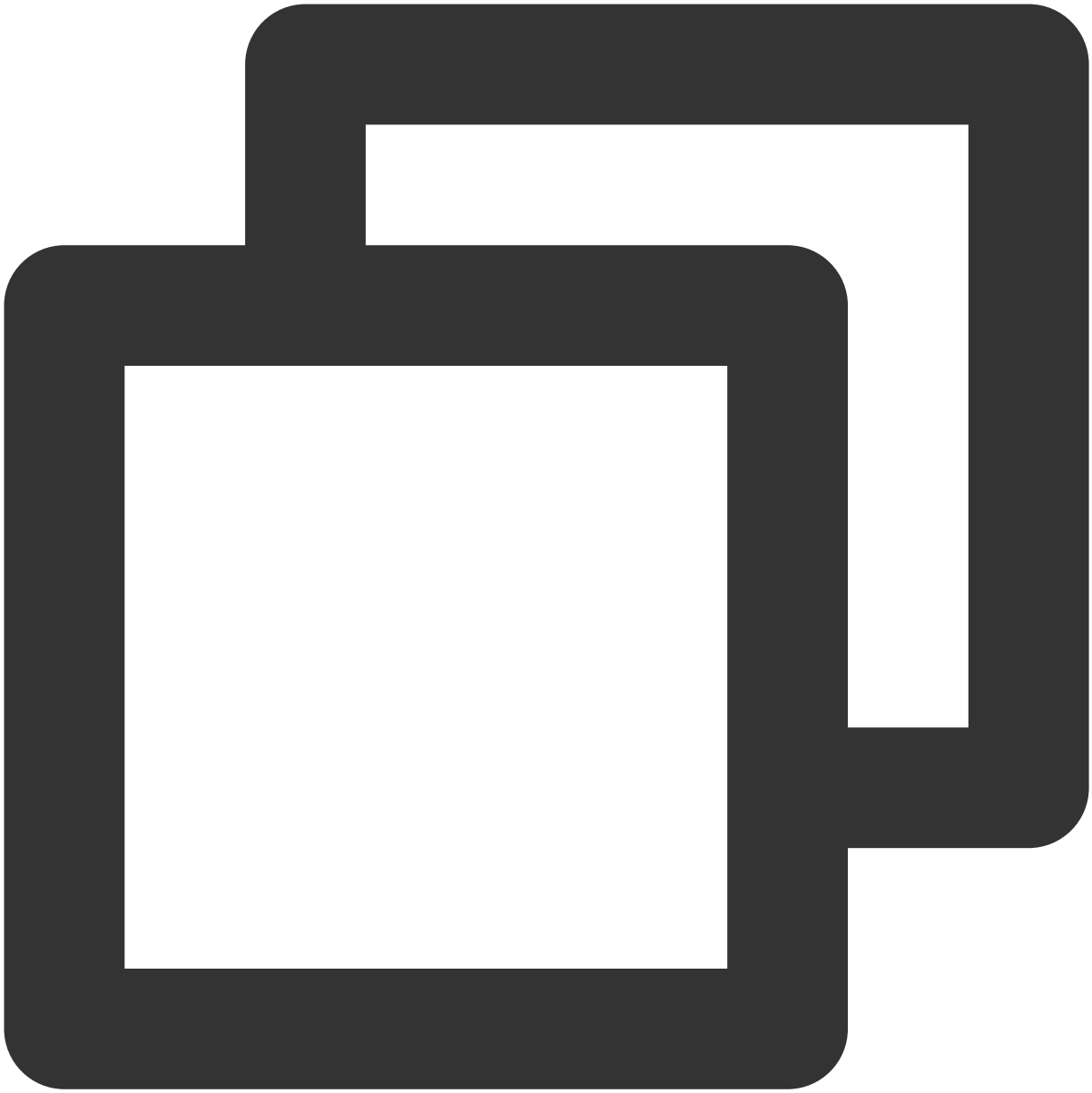
Installing and Configuring Terraform

Last updated：2024-01-25 11:20:13

This document describes how to install and configure Terraform.

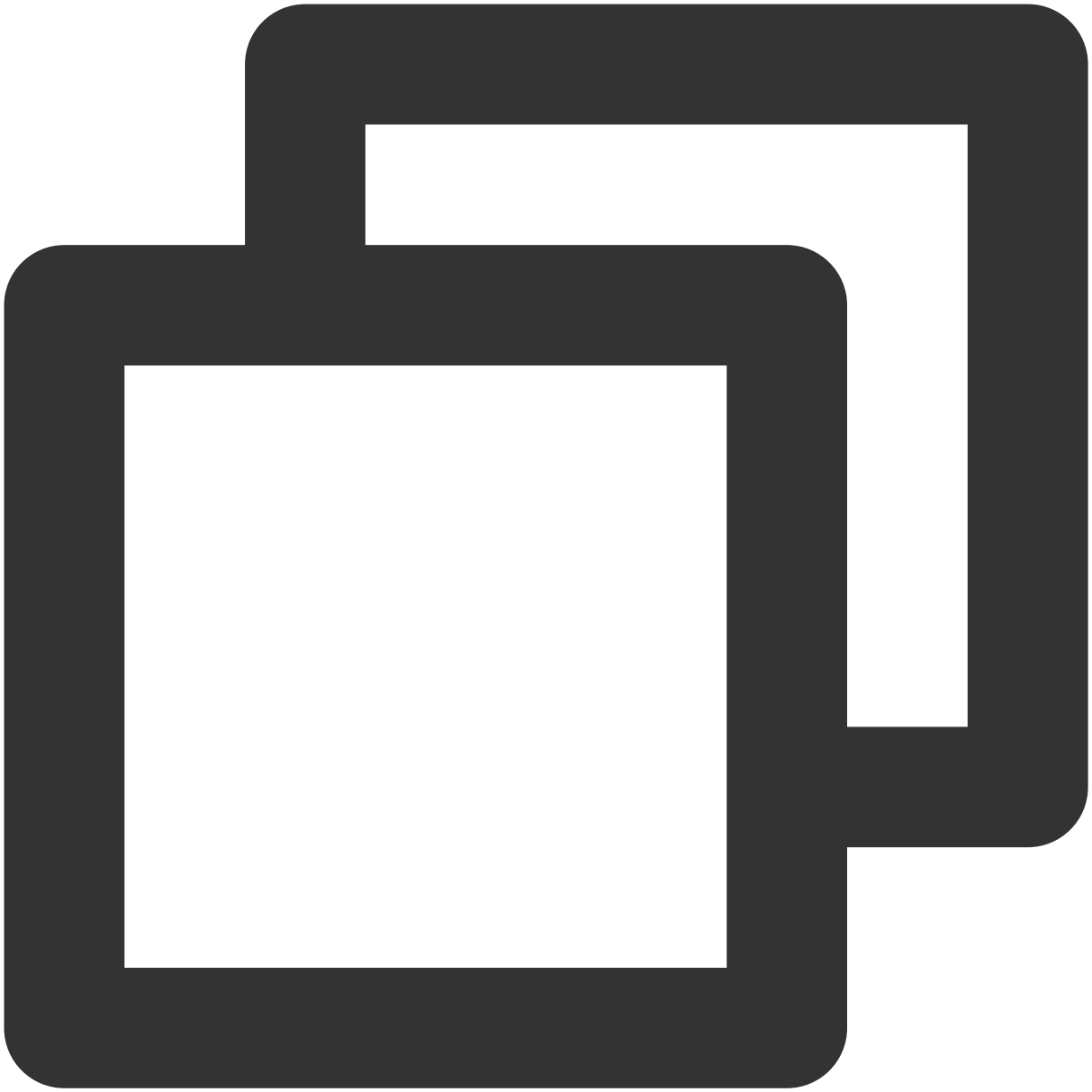
Step 1. Install Terraform

1. Go to Terraform official website and use the command line to install Terraform directly or download the binary installation file.
2. Unzip the file and configure the global path.
Skip this step if you use the command line.
Linux and macOS
Windows
1. Run the following command to unzip the file. Replace 1.x.x with the actual version number of Terraform to be installed.



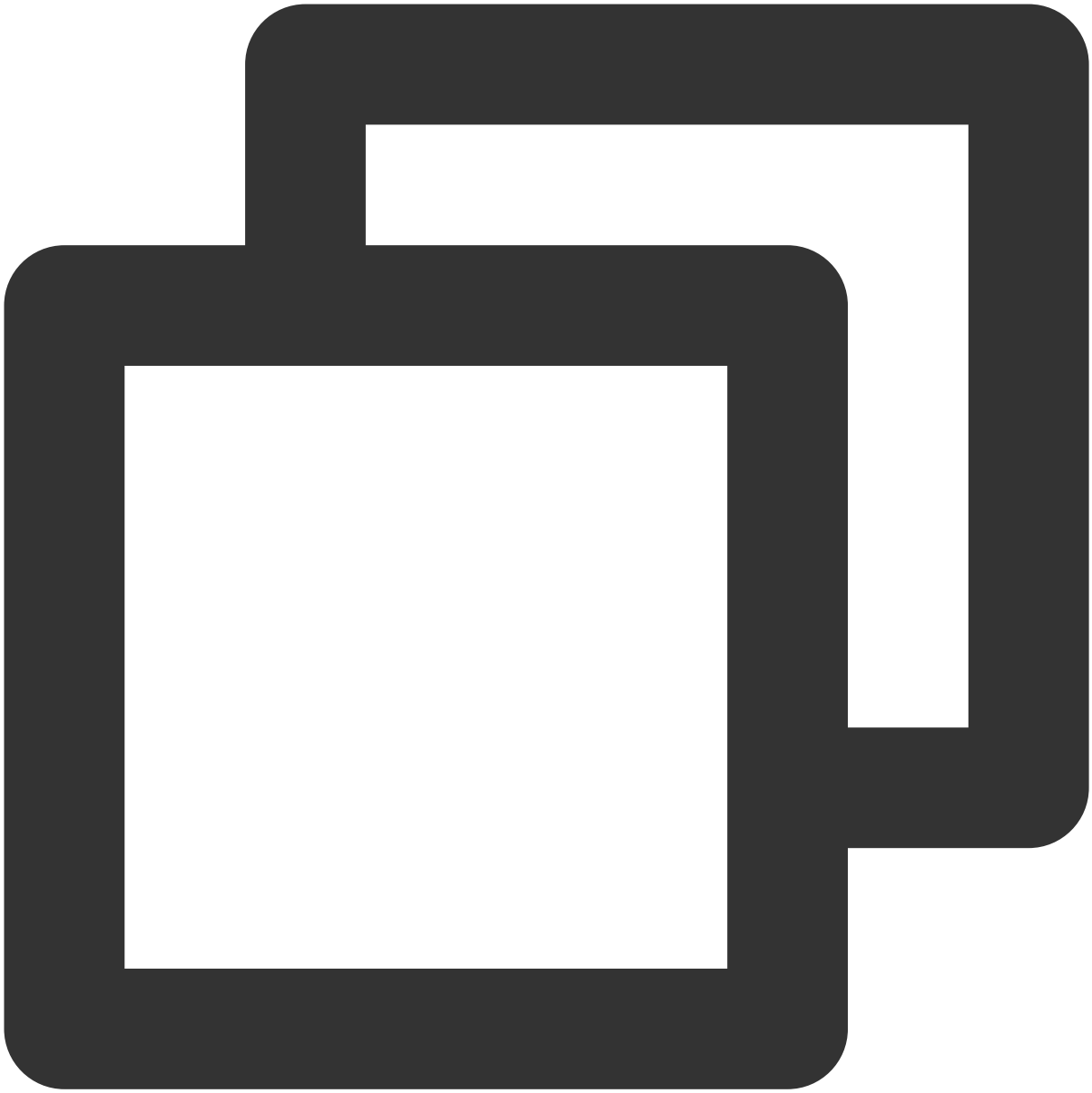
```
unzip terraform_1.x.x_linux_amd64.zip
```

2. Run the following command to add the current directory to the `~/.profile` file.



```
echo $"export PATH=\\$PATH:$(pwd)" >> ~/.bash_profile
```

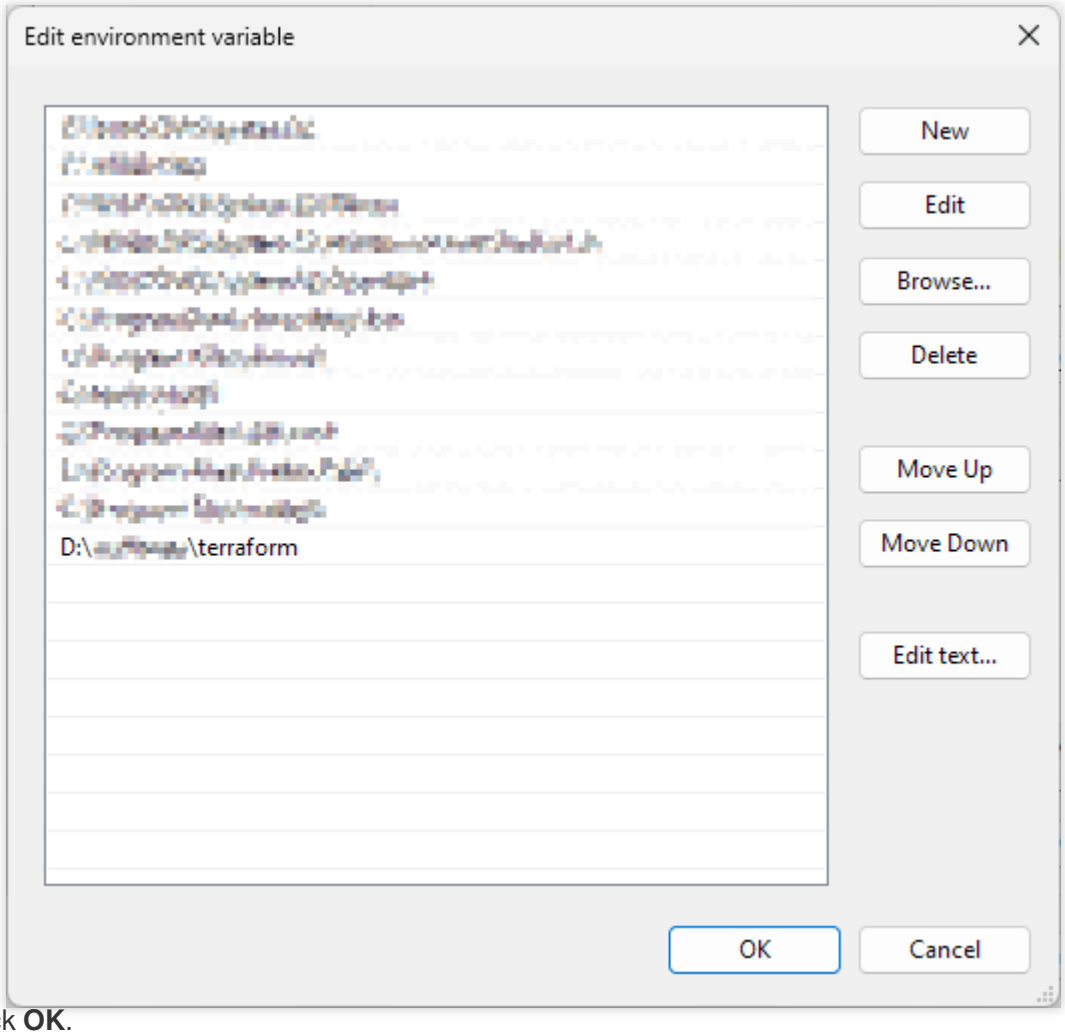
3. Run the following command to make the global path configuration take effect.



```
source ~/.bash_profile
```

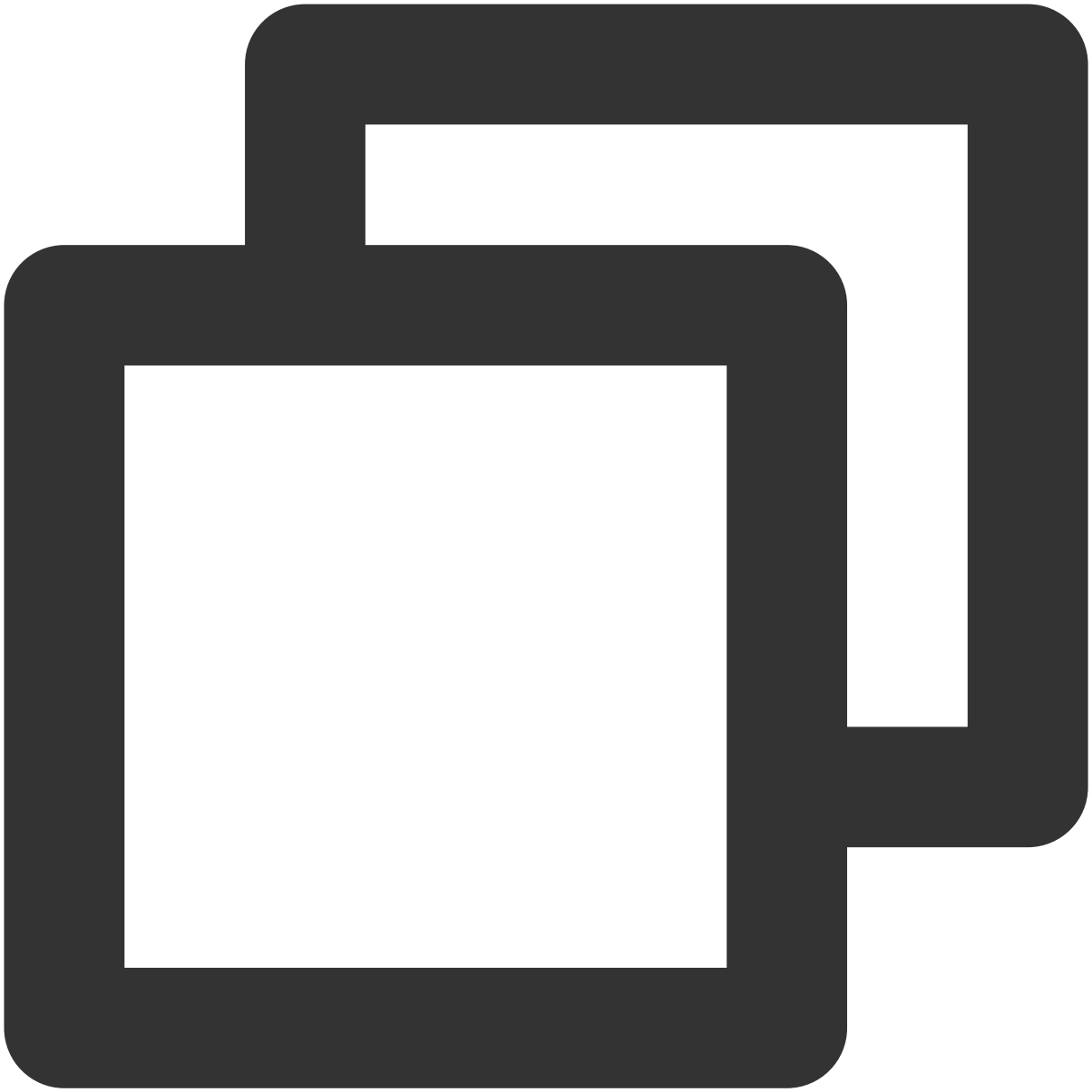
1. On the desktop, right-click the **This PC** icon and select **Properties** in the pop-up menu.
2. In the pop-up window, click **Advanced system settings**.
3. In the **System Properties** window that pops up, click **Environment Variables**.
4. In **System variables**, add the absolute path of `terraform.exe` to `Path` as shown below:

This document uses Windows 10 as an example, and `terraform.exe` is located in `D:\\xxxx\\terraform`.



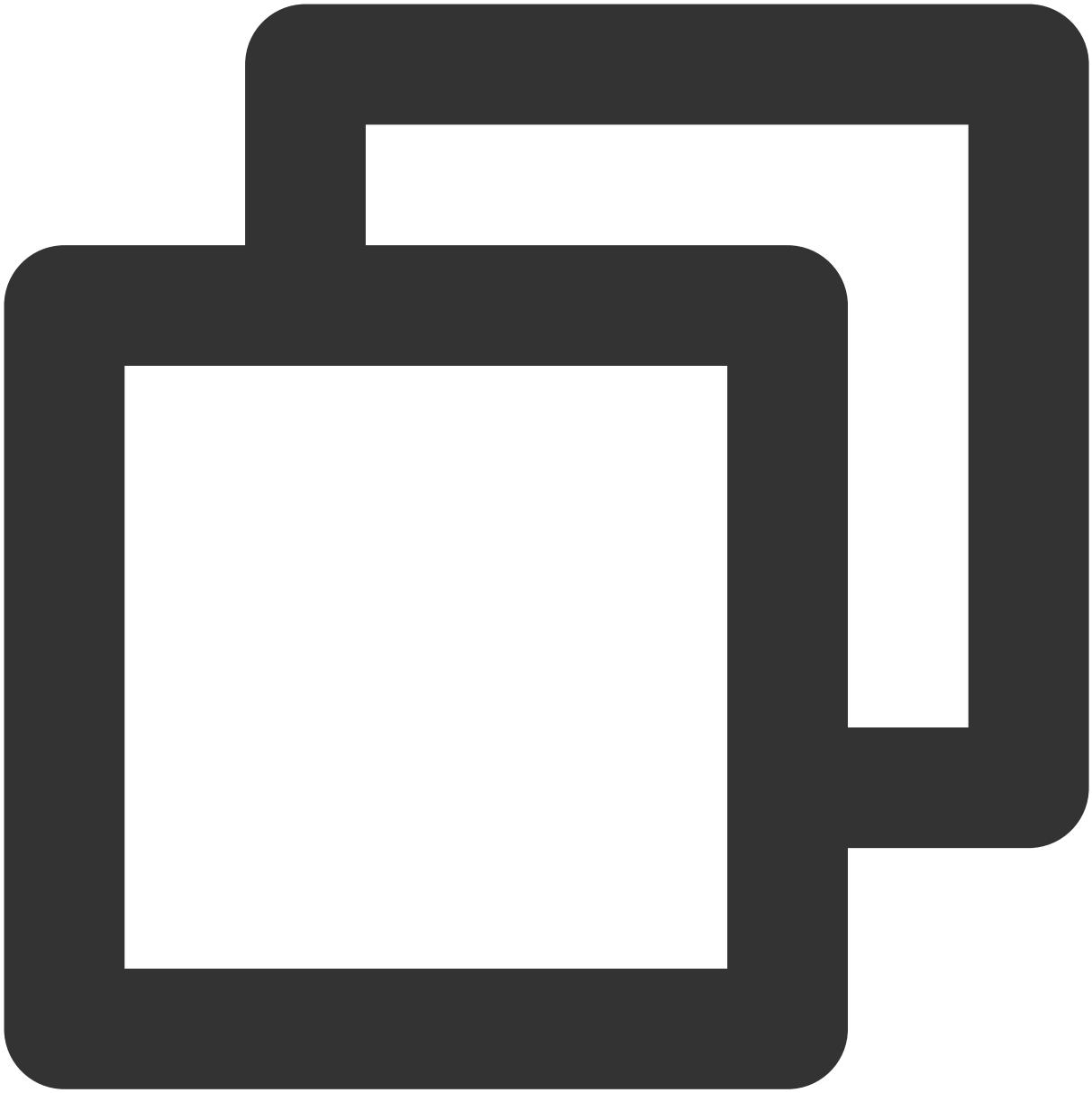
5. Click **OK**.

3. Run the following command to check whether the installation is successful.



```
terraform  -version
```

If the following information is returned (the version number may be different), the installation is successful:

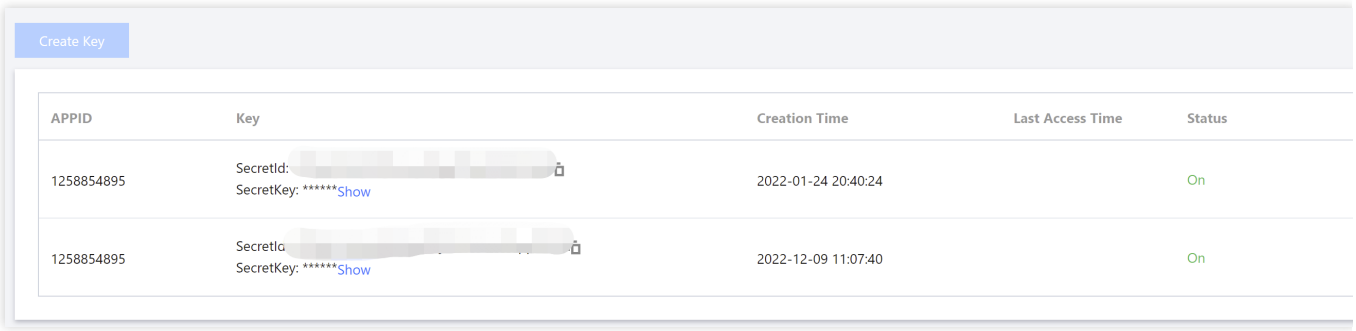


```
> Terraform v1.0.10
> on darwin_amd64
> Your version of Terraform is out of date! The latest version
> is 1.1.0. You can update by downloading from https://www.terraform.io/downloads.h
```

Step 2. Configure security credentials

Before using Terraform for the first time, go to the TencentCloud API Key page to apply for `SecretId` and `SecretKey`. If you already have them, go to step 3.

1. Log in to the CAM console and select **Access Key** > **Manage API Key** on the left sidebar.
2. On the **Manage API Key** page, click **Create Key** to create a pair of `SecretId/SecretKey`.



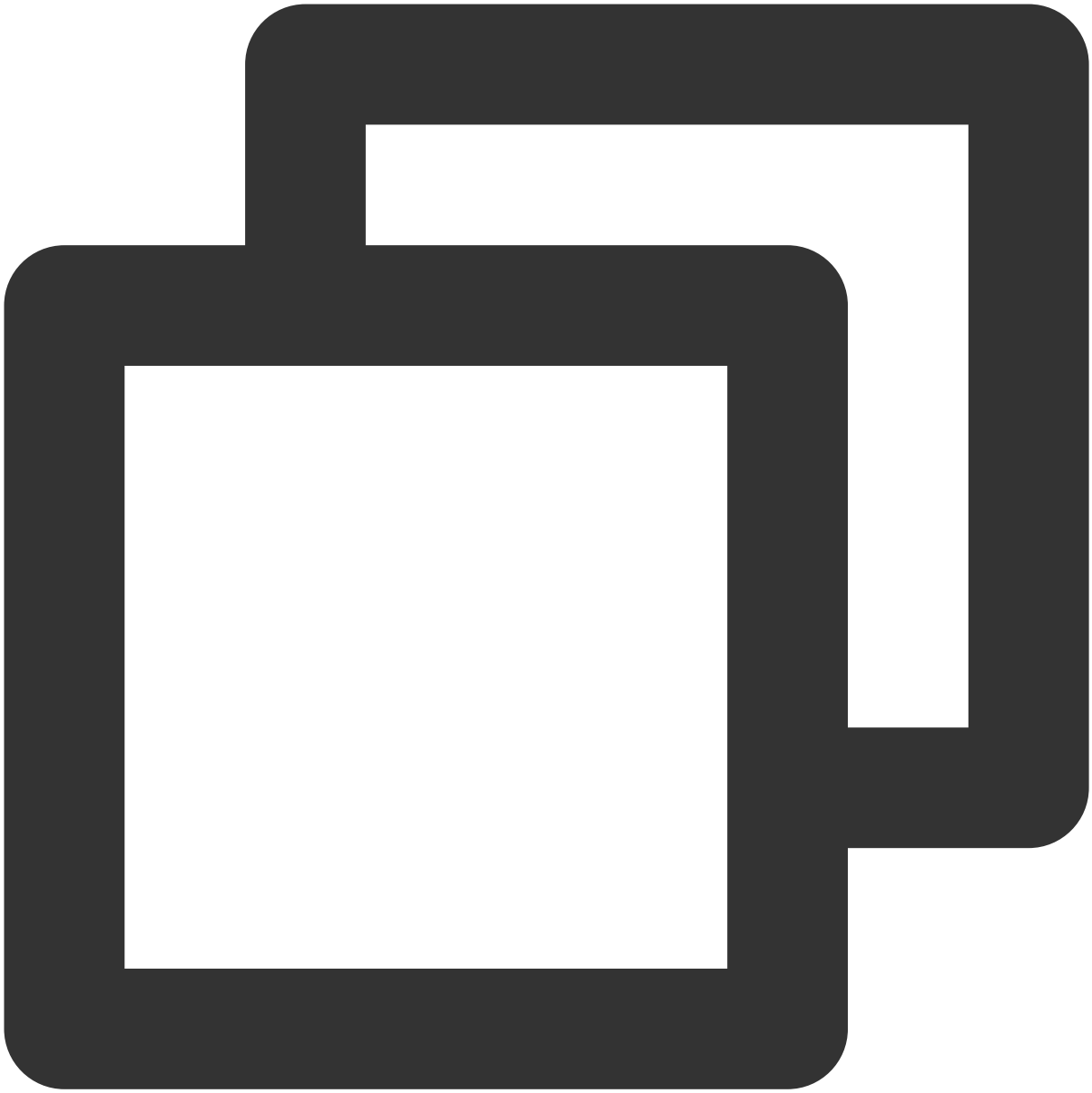
3. You can authenticate in two ways:

Authentication by environment variable

Authentication by Static Credential

Add the following content to the environment variable configuration:

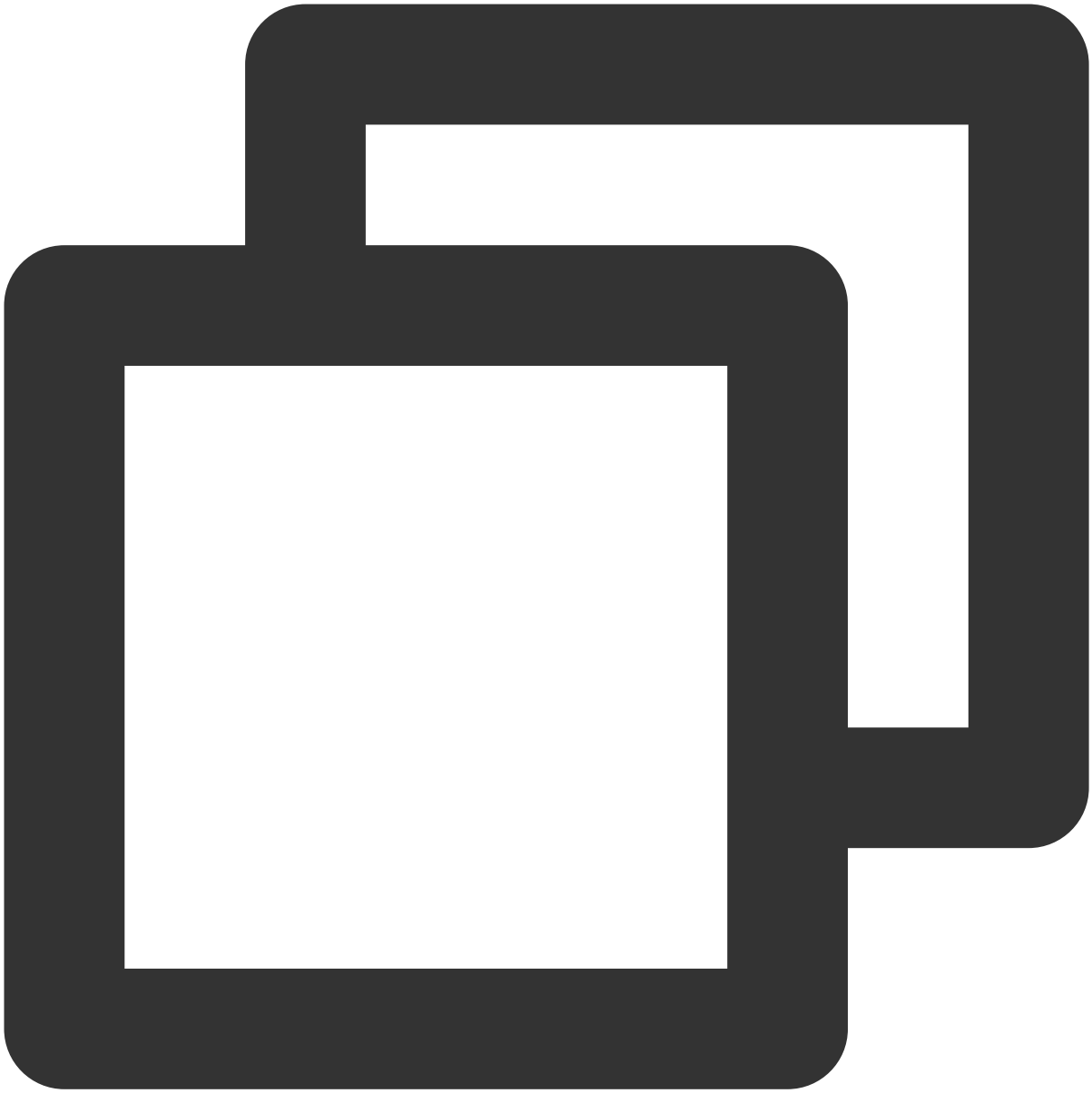
Replace `<your-secret-id>` and `<your-secret-key>` with `SecretId` and `SecretKey` obtained in the Get credentials step.



```
export TENCENTCLOUD_SECRET_ID=<your-secret-id>
export TENCENTCLOUD_SECRET_KEY=<your-secret-key>
```

Create a `provider.tf` file in the user directory and enter the following content:

Replace `<your-secret-id>` and `<your-secret-key>` with `SecretId` and `SecretKey` obtained in the Get credentials step.



```
provider "tencentcloud" {
    secret_id = "<your-secret-id>"
    secret_key = "<your-secret-key>"
}
```

4. At this point, you have installed Terraform and configured the environment variable. You can now proceed to create a site through Terraform.

Creating Site Through Terraform

Last updated：2024-01-25 11:20:13

Overview

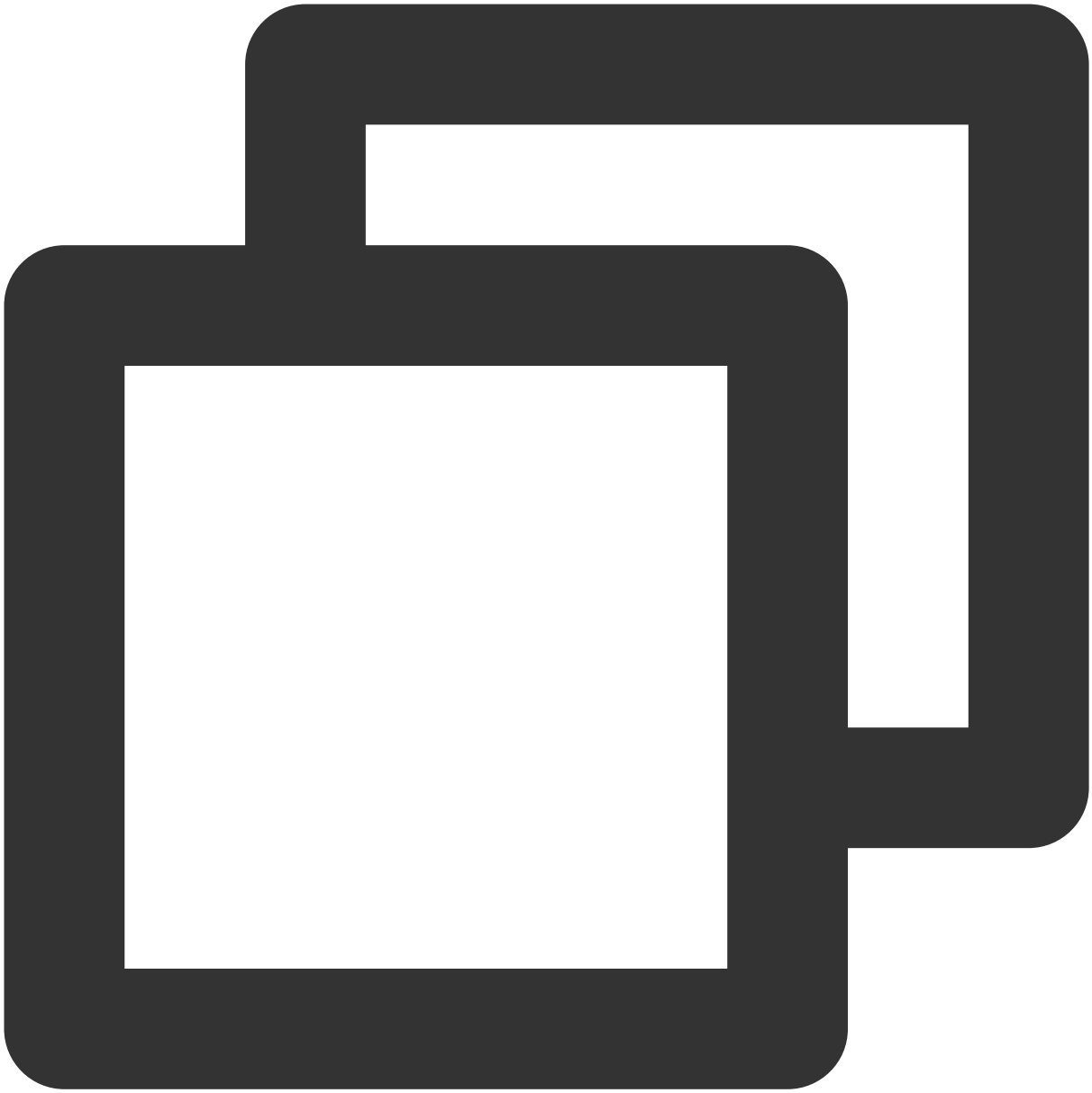
EdgeOne has been connected to Terraform to allow for quick configuration. This document describes how to use Terraform to quickly add an EdgeOne site.

Prerequisites

You have installed and configured Terraform as instructed in Installing and Configuring Terraform.

Directions

1. Define the site resources managed through Terraform.
You need to write a Terraform configuration file and save it with the `.tf` extension. You can view the parameter definitions of EdgeOne site resources on the Terraform Provider documentation page.
Below is the `tencent_teo.tf` sample configuration file:



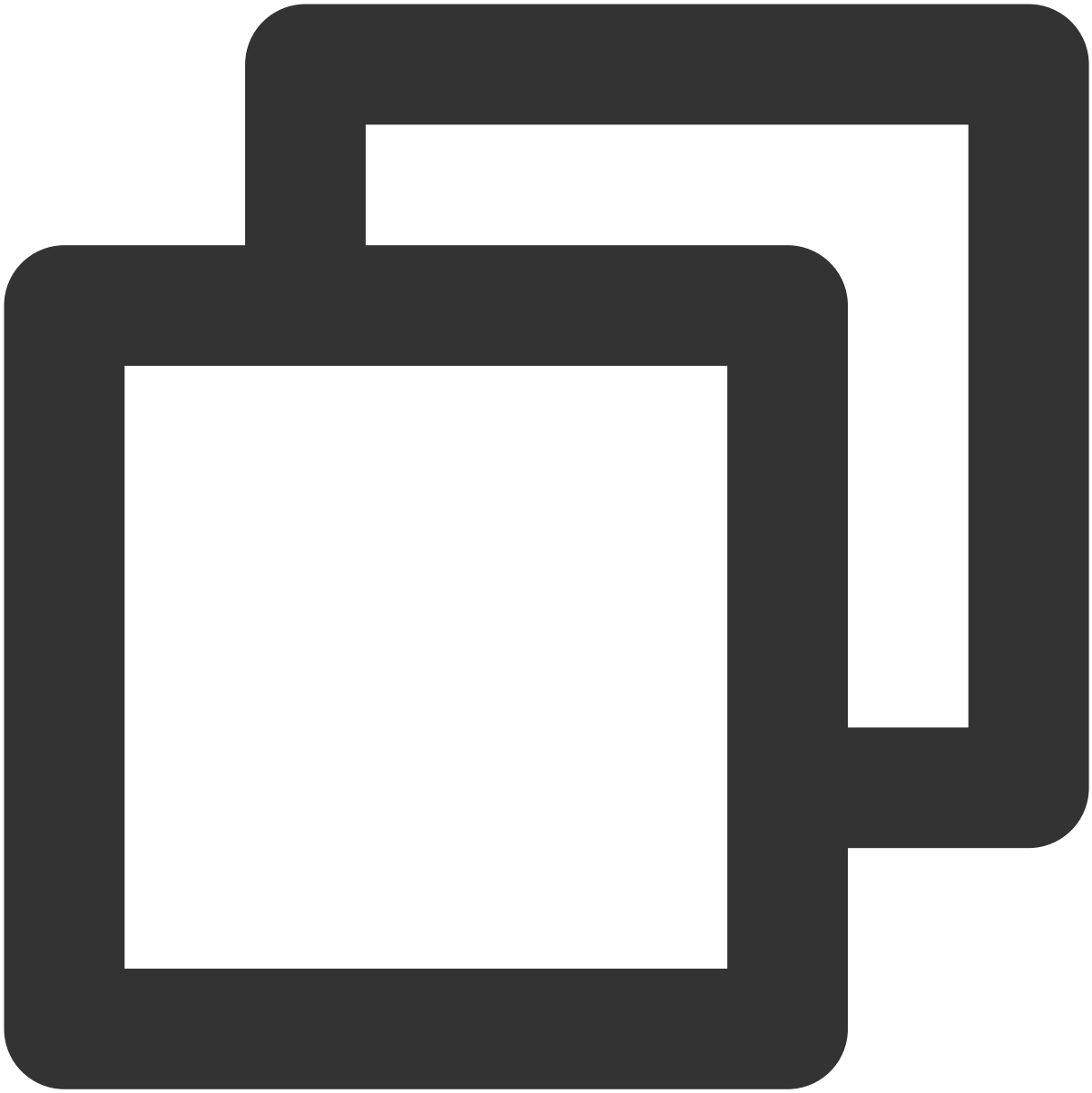
```
terraform {
  required_providers {
    tencentcloud = {
      source  = "tencentcloudstack/tencentcloud"
      version = ">= 1.78.5"
    }
  }
}
provider "tencentcloud" {
  secret_id  = "<your-secret-id>"
  secret_key = "<your-secret-key>"
```

```
  region     = "ap-guangzhou"
}
resource "tencentcloud_teo_zone" "example" {
  zone_name = "example.com"
  plan_type = "<your-plan-type>"
  tags = {
    "createdBy" = "terraform"
  }
}
```

2. In the directory of the Terraform configuration file, run the `terraform init` command to initialize the configuration.

In this step, Terraform will automatically check the `provider` field in the configuration file and download the latest module and plugin.

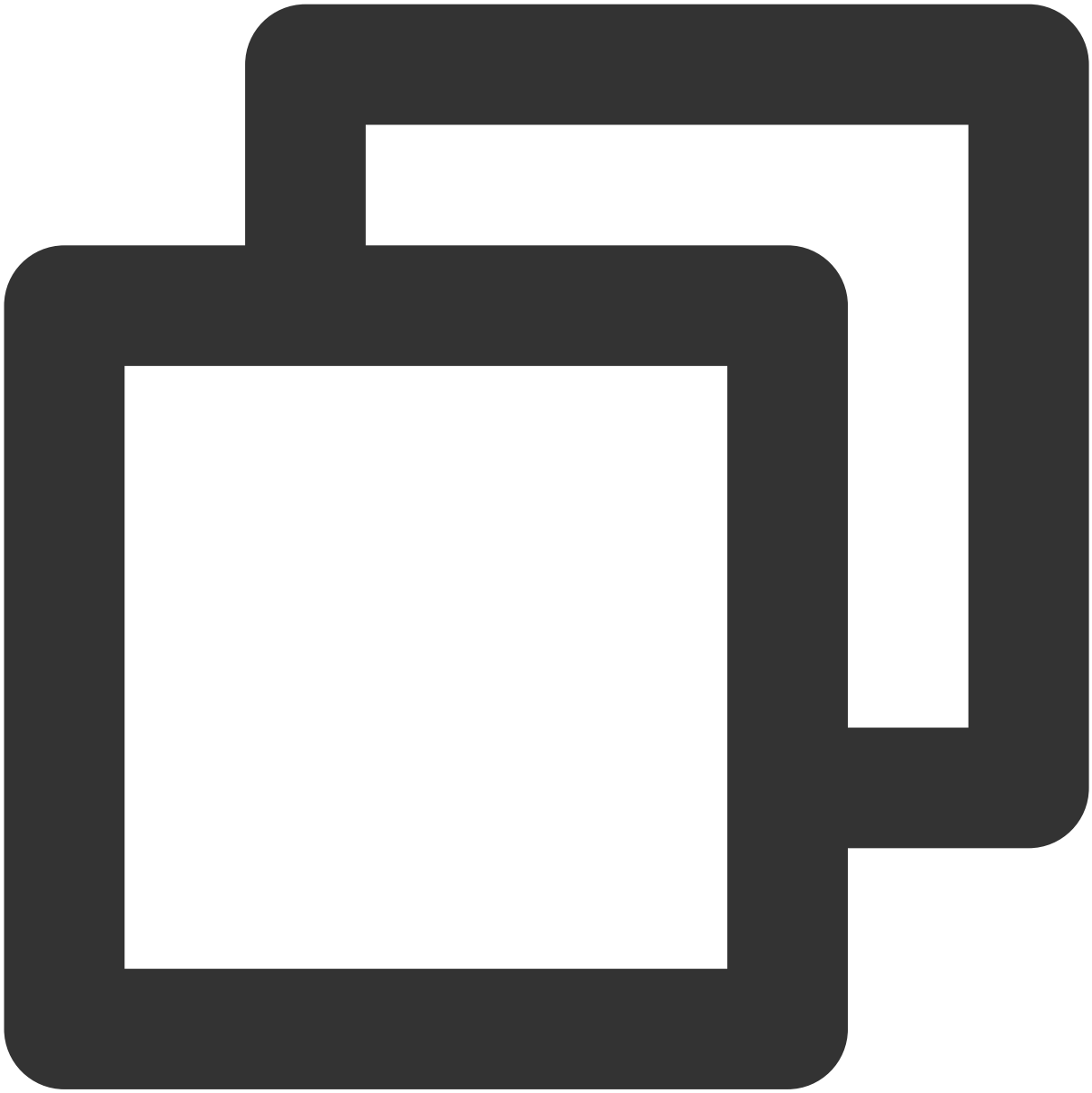
If the following message is printed, the initialization is successful.



```
Initializing the backend...
Initializing provider plugins...
- Finding tencentcloudstack/tencentcloud versions matching ">= 1.78.5"...
- Installing tencentcloudstack/tencentcloud v1.78.5...
- Installed tencentcloudstack/tencentcloud v1.78.5 (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 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.
Terraform has been successfully initialized!
```

3. Run the `terraform plan` command to preview the configuration and verify whether it is correct.



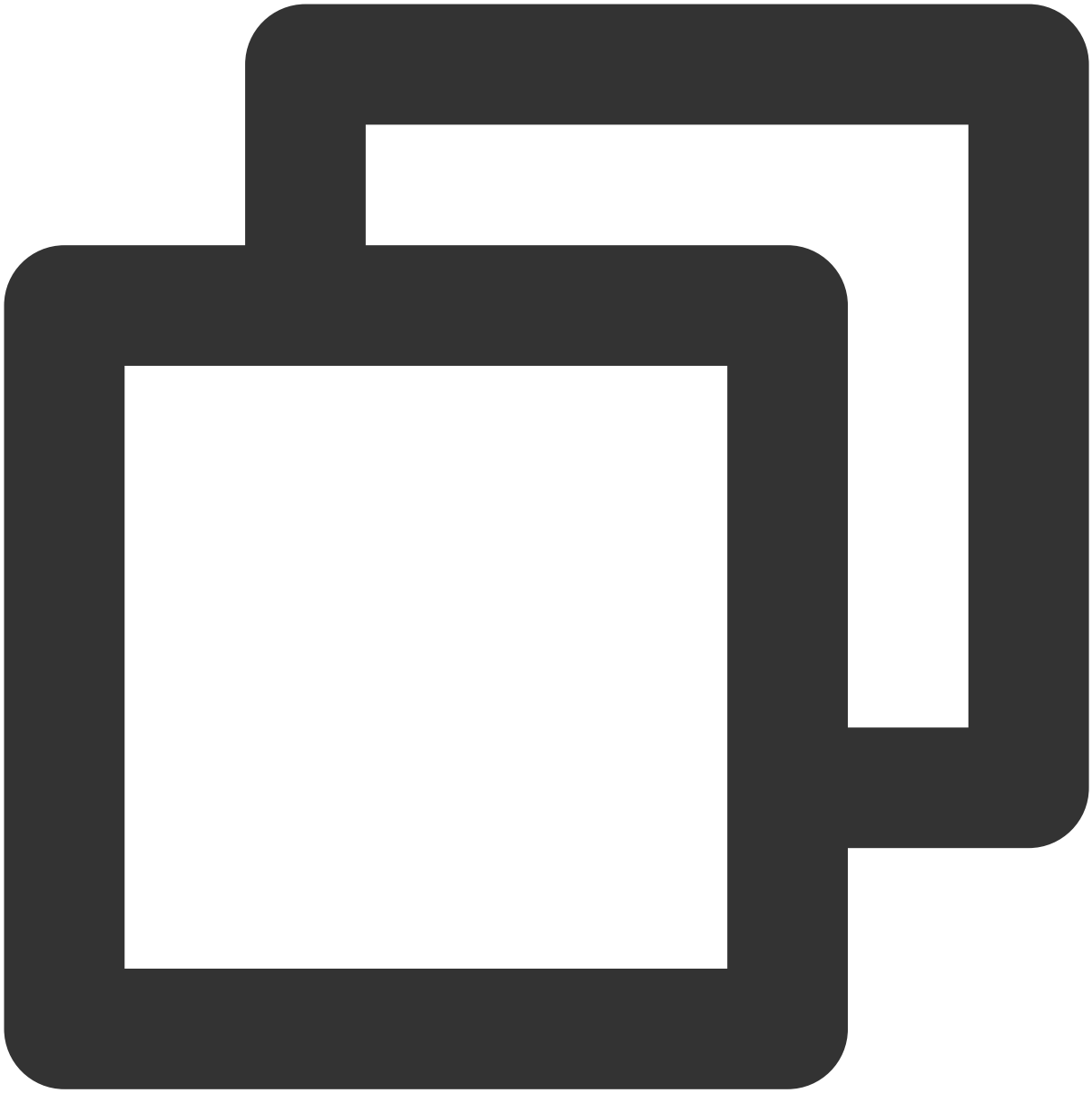
```
PS tf-doc> terraform.exe plan
Terraform used the selected providers to generate the following execution plan. Res
  + create
Terraform will perform the following actions:
  # tencentcloud_teo_zone.example will be created
  + resource "tencentcloud_teo_zone" "example" {
```

```
      + area                    = (known after apply)
      + cname_speed_up          = (known after apply)
      + cname_status            = (known after apply)
      + created_on              = (known after apply)
      + id                      = (known after apply)
      + modified_on             = (known after apply)
      + name_servers            = (known after apply)
      + original_name_servers   = (known after apply)
      + paused                  = (known after apply)
      + plan_type               = "ent"
      + resources               = (known after apply)
      + status                  = (known after apply)
      + tags                    = {
          + "createdBy" = "terraform"
        }
      + type                    = (known after apply)
      + vanity_name_servers_ips = (known after apply)
      + zone_id                 = (known after apply)
      + zone_name               = "example.com"
    }
Plan: 1 to add, 0 to change, 0 to destroy.
─────────────────────────────────────────────────────────────────────────────

Note: You didn't use the -out option to save this plan, so Terraform can't guarante
```

4. Run `terraform apply` to add an EdgeOne node.

After the `terraform apply` command is executed, Terraform will ask you to confirm the actions to be executed. After confirming that everything is correct, enter `yes` . Then, wait for the command to complete.



```
PS tf-doc> terraform apply
Terraform used the selected providers to generate the following execution plan. Res
  + create
Terraform will perform the following actions:
  # tencentcloud_teo_zone.example will be created
  + resource "tencentcloud_teo_zone" "example" {
      + area                    = (known after apply)
      + cname_speed_up          = (known after apply)
      + cname_status            = (known after apply)
      + created_on              = (known after apply)
      + id                      = (known after apply)
```

```
      + modified_on             = (known after apply)
      + name_servers            = (known after apply)
      + original_name_servers   = (known after apply)
      + paused                  = (known after apply)
      + plan_type               = "ent"
      + resources               = (known after apply)
      + status                  = (known after apply)
      + tags                    = {
          + "createdBy" = "terraform"
        }
      + type                    = (known after apply)
      + vanity_name_servers_ips = (known after apply)
      + zone_id                 = (known after apply)
      + zone_name               = "example.com"
    }
Plan: 1 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_teo_zone.example: Creating...
tencentcloud_teo_zone.example: Creation complete after 6s [id=zone-2ag9gej58j36]
Apply complete! Resources: 1 added, 0 changed, 0 destroyed.
```

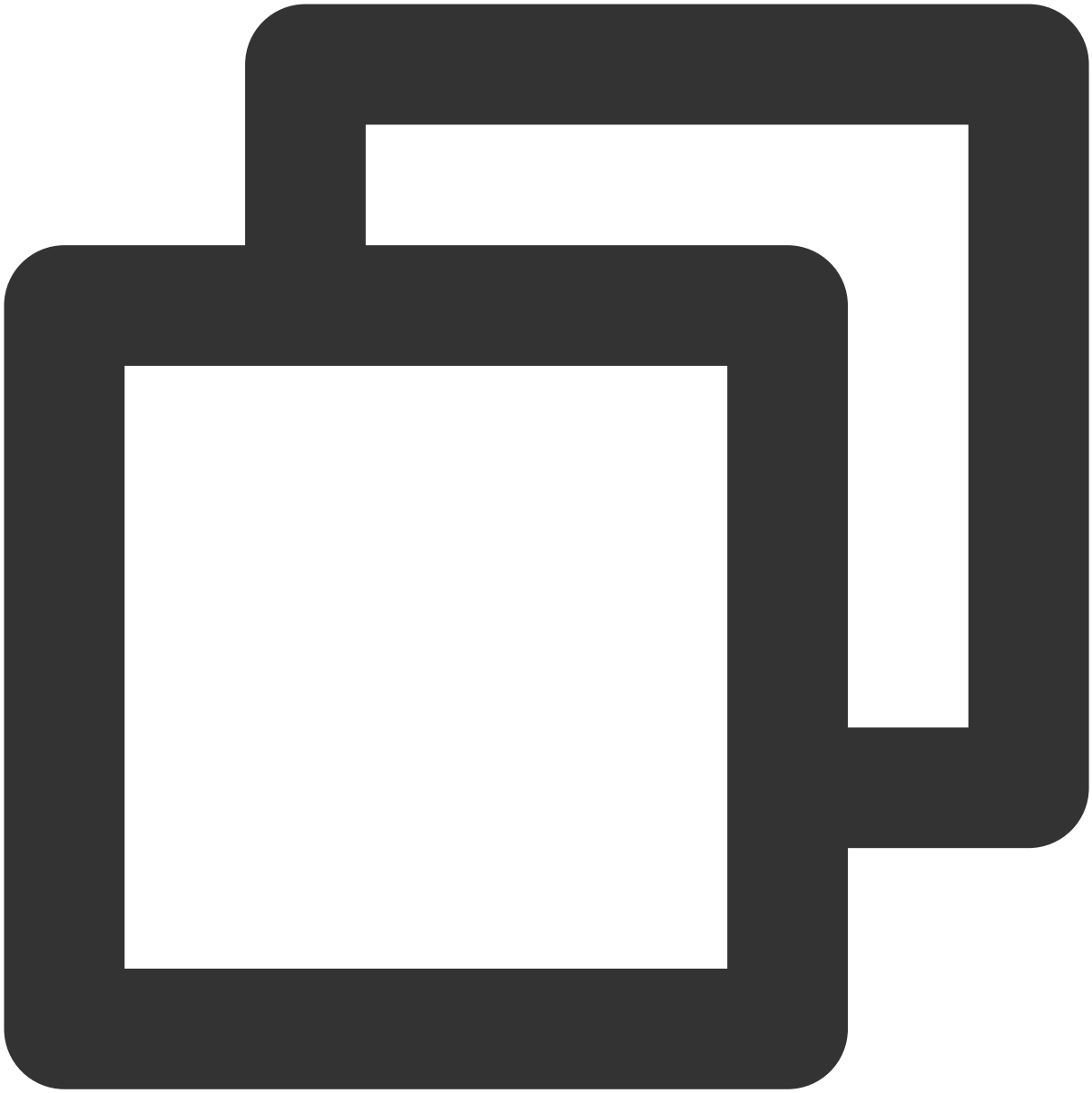
5. Modify the DNS configuration of the connected site.

To make the site effective, you need to modify the NS server of the site if you use NS access or to add a TXT record for site verification if you use CNAME access. For more information, see Connection Method.

6. Verify whether the site has taken effect.

Wait a few minutes after step 5 and refresh the resource status through `terraform refresh`. Then, run `terraform show` to check whether the site has taken effect.

If NS access is used, the value of the `status` field should be `active` after the site takes effect. If CNAME access is used, the value of the `cname_status` field should be `finished` after the site takes effect.



```
PS tf-doc> terraform refresh
tencentcloud_teo_zone.example: Refreshing state... [id=zone-2ag9gej58j36]
PS tf-doc> terraform show
# tencentcloud_teo_zone.example:
resource "tencentcloud_teo_zone" "example" {
    area                   = "overseas"
    cname_speed_up         = "enabled"
    cname_status           = "pending"
...
    original_name_servers  = []
    paused                 = false
```

```
    plan_type               = "ent"
    status                  = "active"
    tags                    = {
        "createdBy" = "terraform"
    }
    type                    = "full"
    vanity_name_servers_ips = []
    zone_id                 = "zone-2ag9gej58j36"
    zone_name               = "example.com"
}
```

Notes

1. When creating a site, you can query available plans through the data source tencentcloud_teo_zone_available_plans.
2. You can configure Terraform with environment variables to avoid writing the TencentCloud API key to the Terraform configuration file.

Configuring Site Acceleration Through Terraform

Last updated：2024-01-25 11:20:13

Overview

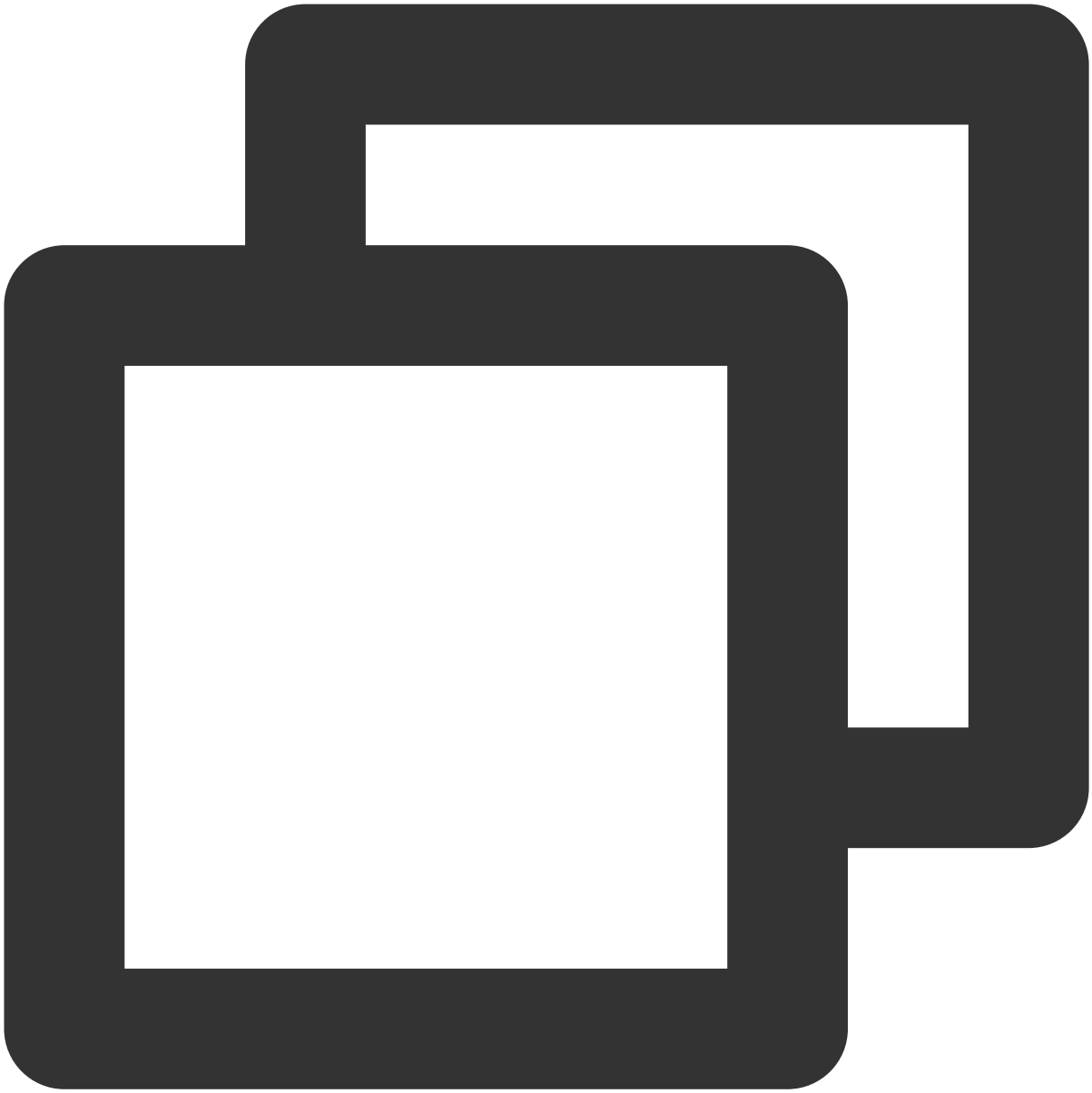
EdgeOne has been connected to Terraform to allow for quick configuration. This document describes how to use Terraform to configure site acceleration. For more information, see Tencent Cloud EdgeOne.

Prerequisites

1. You have installed and configured Terraform as instructed in Installing and Configuring Terraform.
2. You have connected to the site through Terraform as instructed in Creating Site Through Terraform.

Directions

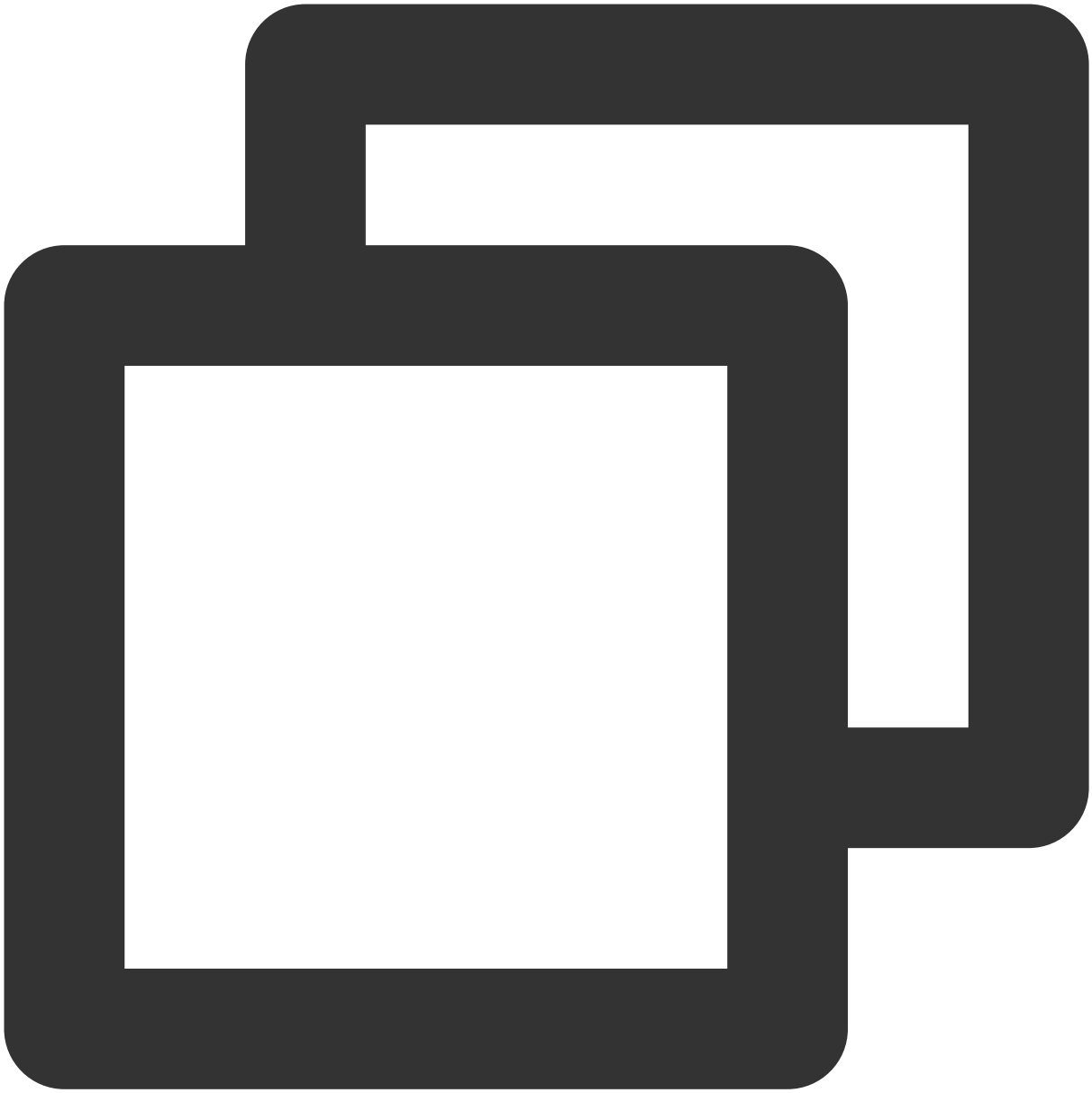
1. Modify the Terraform configuration file and add the resource definitions of site acceleration configuration.
You can view the parameter definitions of the site acceleration configuration on the Terraform Provider documentation page. Below is the `tencent_teo.tf` sample configuration file.



```
terraform {
  required_providers {
    tencentcloud = {
      source  = "tencentcloudstack/tencentcloud"
      version = ">= 1.78.5"
    }
  }
}
provider "tencentcloud" {
  secret_id  = "<your-secret-id>"
  secret_key = "<your-secret-key>"
```

```
  region     = "ap-guangzhou"
}
resource "tencentcloud_teo_zone" "example" {
  zone_name = "example.com"
  plan_type = "ent"
  tags = {
    "createdBy" = "terraform"
  }
}
resource "tencentcloud_teo_zone_setting" "example" {
  zone_id = tencentcloud_teo_zone.example.id
  # Cache rule configuration
  cache {
    follow_origin {
      switch = "on" # Follow the origin server
    }
  }
  # Cache key configuration
  cache_key {
    full_url_cache = "off" # Do not enable full path cache
    ignore_case    = "on" # Ignore the case
    query_string {
      switch = "on"
      action = "includeCustom" # Only use the specified URL parameter
      value  = ["param0", "param1"]
    }
  }
  # HTTPS acceleration configuration of the domain name
  https {
    ocsp_stapling = "on" # OCSP configuration enabled
    tls_version   = ["TLSv1.2", "TLSv1.3"] # Supported TLS protocol versions
  }
  # Smart compression configuration
  compression {
    switch     = "on"
    algorithms = ["brotli", "gzip"]
  }
  # Region of the client IP during origin-pull
  client_ip_header {
    switch      = "on"
    header_name = "EO-Client-IPCountry"
  }
}
```

2. Run the `terraform plan` command to preview the configuration and verify whether it is correct.



```
PS tf-doc> terraform.exe plan
tencentcloud_teo_zone.example: Refreshing state... [id=zone-2ag9gej58j36]
Terraform used the selected providers to generate the following execution plan. Res
  + create
Terraform will perform the following actions:
  # tencentcloud_teo_zone_setting.example will be created
  + resource "tencentcloud_teo_zone_setting" "example" {
      + area    = (known after apply)
      + id      = (known after apply)
      + zone_id = "zone-2ag9gej58j36"
      + cache {
```

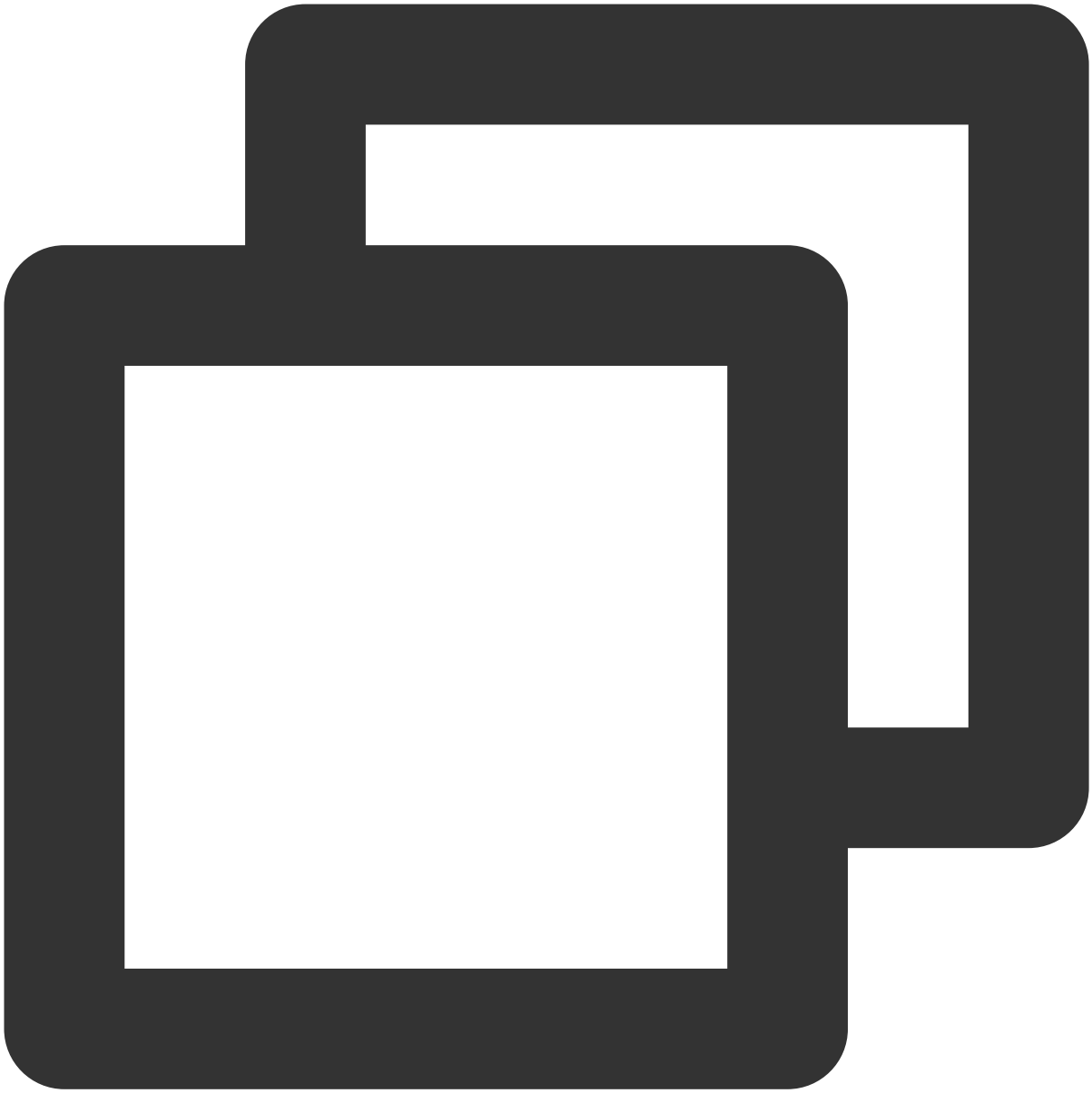
```
      + cache {
          + cache_time           = (known after apply)
          + ignore_cache_control = (known after apply)
          + switch               = (known after apply)
        }
      + follow_origin {
          + switch = "on"
        }
      + no_cache {
          + switch = (known after apply)
        }
    }
  + cache_key {
      + full_url_cache = "off"
      + ignore_case    = "on"
      + query_string {
          + action = "includeCustom"
          + switch = "on"
          + value  = [
              + "param0",
              + "param1",
            ]
        }
    }
  + cache_prefresh {
      + percent = (known after apply)
      + switch  = (known after apply)
    }
  + client_ip_header {
      + header_name = "EO-Client-IPCountry"
      + switch      = "on"
    }
  + compression {
      + algorithms = [
          + "brotli",
          + "gzip",
        ]
      + switch     = "on"
    }
  + force_redirect {
      + redirect_status_code = (known after apply)
      + switch               = (known after apply)
    }
  + https {
      + ocsp_stapling = "on"
      + tls_version   = [
          + "TLSv1.2",
```

```
                + "TLSv1.3",
              ]
        }
      + ipv6 {
          + switch = (known after apply)
        }
      + max_age {
          + follow_origin = (known after apply)
          + max_age_time  = (known after apply)
        }
      + offline_cache {
          + switch = (known after apply)
        }
      + origin {
          + backup_origins       = (known after apply)
          + cos_private_access   = (known after apply)
          + origin_pull_protocol = (known after apply)
          + origins              = (known after apply)
        }
      + post_max_size {
          + max_size = (known after apply)
          + switch   = (known after apply)
        }
      + quic {
          + switch = (known after apply)
        }
      + smart_routing {
          + switch = (known after apply)
        }
      + upstream_http2 {
          + switch = (known after apply)
        }
      + web_socket {
          + switch  = (known after apply)
          + timeout = (known after apply)
        }
    }
Plan: 1 to add, 0 to change, 0 to destroy.
─────────────────────────────────────────────────────────────────────────────

Note: You didn't use the -out option to save this plan, so Terraform can't guarante
```

3. Run `terraform apply` to create the site acceleration configuration.

After the `terraform apply` command is executed, Terraform will ask you to confirm the actions to be executed. After confirming that everything is correct, enter `yes` . Then, wait for the command to complete.



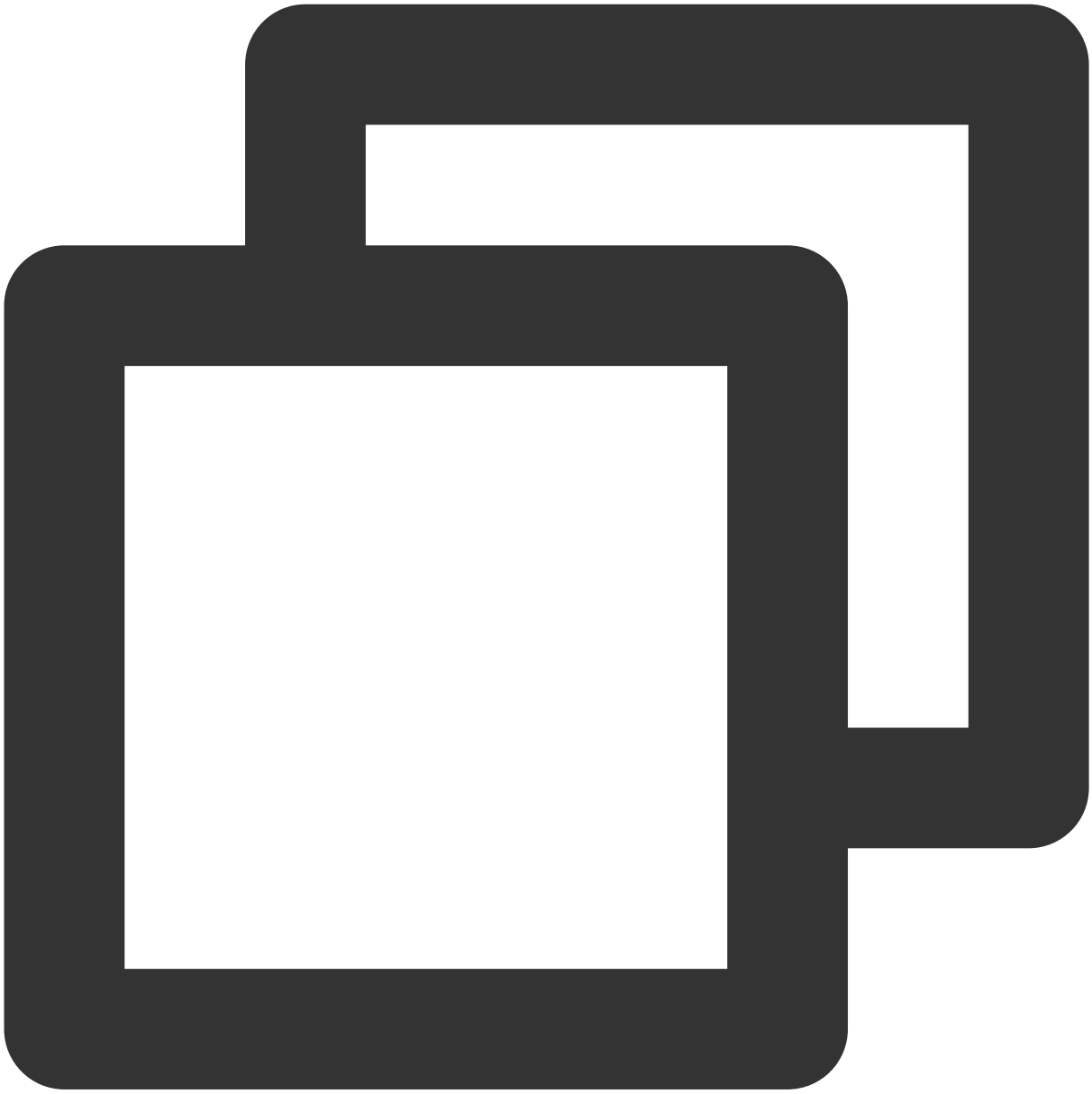
```
PS tf-doc> terraform.exe apply
tencentcloud_teo_zone.example: Refreshing state... [id=zone-2ag9gej58j36]
Terraform used the selected providers to generate the following execution plan. Res
  + create
Terraform will perform the following actions:
  # tencentcloud_teo_zone_setting.example will be created
  + resource "tencentcloud_teo_zone_setting" "example" {
      + area    = (known after apply)
      + id      = (known after apply)
      + zone_id = "zone-2ag9gej58j36"
      + cache {
```

```
      + cache {
          + cache_time           = (known after apply)
          + ignore_cache_control = (known after apply)
          + switch               = (known after apply)
        }
      + follow_origin {
          + switch = "on"
        }
      + no_cache {
          + switch = (known after apply)
        }
    }
  + cache_key {
      + full_url_cache = "off"
      + ignore_case    = "on"
      + query_string {
          + action = "includeCustom"
          + switch = "on"
          + value  = [
              + "param0",
              + "param1",
            ]
        }
    }
  + cache_prefresh {
      + percent = (known after apply)
      + switch  = (known after apply)
    }
  + client_ip_header {
      + header_name = "EO-Client-IPCountry"
      + switch      = "on"
    }
  + compression {
      + algorithms = [
          + "brotli",
          + "gzip",
        ]
      + switch     = "on"
    }
  + force_redirect {
      + redirect_status_code = (known after apply)
      + switch               = (known after apply)
    }
  + https {
      + ocsp_stapling = "on"
      + tls_version   = [
          + "TLSv1.2",
```

```
                + "TLSv1.3",
              ]
        }
      + ipv6 {
          + switch = (known after apply)
        }
      + max_age {
          + follow_origin = (known after apply)
          + max_age_time  = (known after apply)
        }
      + offline_cache {
          + switch = (known after apply)
        }
      + origin {
          + backup_origins       = (known after apply)
          + cos_private_access   = (known after apply)
          + origin_pull_protocol = (known after apply)
          + origins              = (known after apply)
        }
      + post_max_size {
          + max_size = (known after apply)
          + switch   = (known after apply)
        }
      + quic {
          + switch = (known after apply)
        }
      + smart_routing {
          + switch = (known after apply)
        }
      + upstream_http2 {
          + switch = (known after apply)
        }
      + web_socket {
          + switch  = (known after apply)
          + timeout = (known after apply)
        }
    }
Plan: 1 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_teo_zone_setting.example: Creating...
tencentcloud_teo_zone_setting.example: Creation complete after 1s [id=zone-2ag9gej5
Apply complete! Resources: 1 added, 0 changed, 0 destroyed.
```

4. Check the command execution result.

You can run `terraform show` to check whether the site acceleration configuration takes effect or log in to the EdgeOne console for confirmation.



```
PS tf-doc> terraform state show tencentcloud_teo_zone_setting.example
# tencentcloud_teo_zone_setting.example:
resource "tencentcloud_teo_zone_setting" "example" {
    area    = "overseas"
    id      = "zone-2ag9gej58j36"
    zone_id = "zone-2ag9gej58j36"
    cache {
```

```
        follow_origin {
            switch = "on"
        }
        no_cache {
            switch = "off"
        }
    }
    cache_key {
        full_url_cache = "off"
        ignore_case    = "on"
        query_string {
            action = "includeCustom"
            switch = "on"
            value  = [
                "param0",
                "param1",
            ]
        }
    }
    cache_prefresh {
        percent = 90
        switch  = "off"
    }
    client_ip_header {
        header_name = "EO-Client-IPCountry"
        switch      = "on"
    }
    compression {
        algorithms = [
            "brotli",
            "gzip",
        ]
        switch     = "on"
    }
    force_redirect {
        redirect_status_code = 302
        switch               = "off"
    }
    https {
        http2         = "on"
        ocsp_stapling = "on"
        tls_version   = [
            "TLSv1.2",
            "TLSv1.3",
        ]
        hsts {
            include_sub_domains = "off"
```

```
            max_age              = 0
            preload              = "off"
            switch               = "off"
        }
    }
    ipv6 {
        switch = "off"
    }
    max_age {
        follow_origin = "on"
        max_age_time  = 600
    }
    offline_cache {
        switch = "on"
    }
    origin {
        backup_origins       = []
        origin_pull_protocol = "follow"
        origins              = []
    }
    post_max_size {
        max_size = 524288000
        switch   = "on"
    }
    quic {
        switch = "off"
    }
    smart_routing {
        switch = "off"
    }
    upstream_http2 {
        switch = "off"
    }
    web_socket {
        switch  = "off"
        timeout = 30
    }
}
```

Configuring Rule Engine Through Terraform

Last updated：2024-01-25 11:20:13

Overview

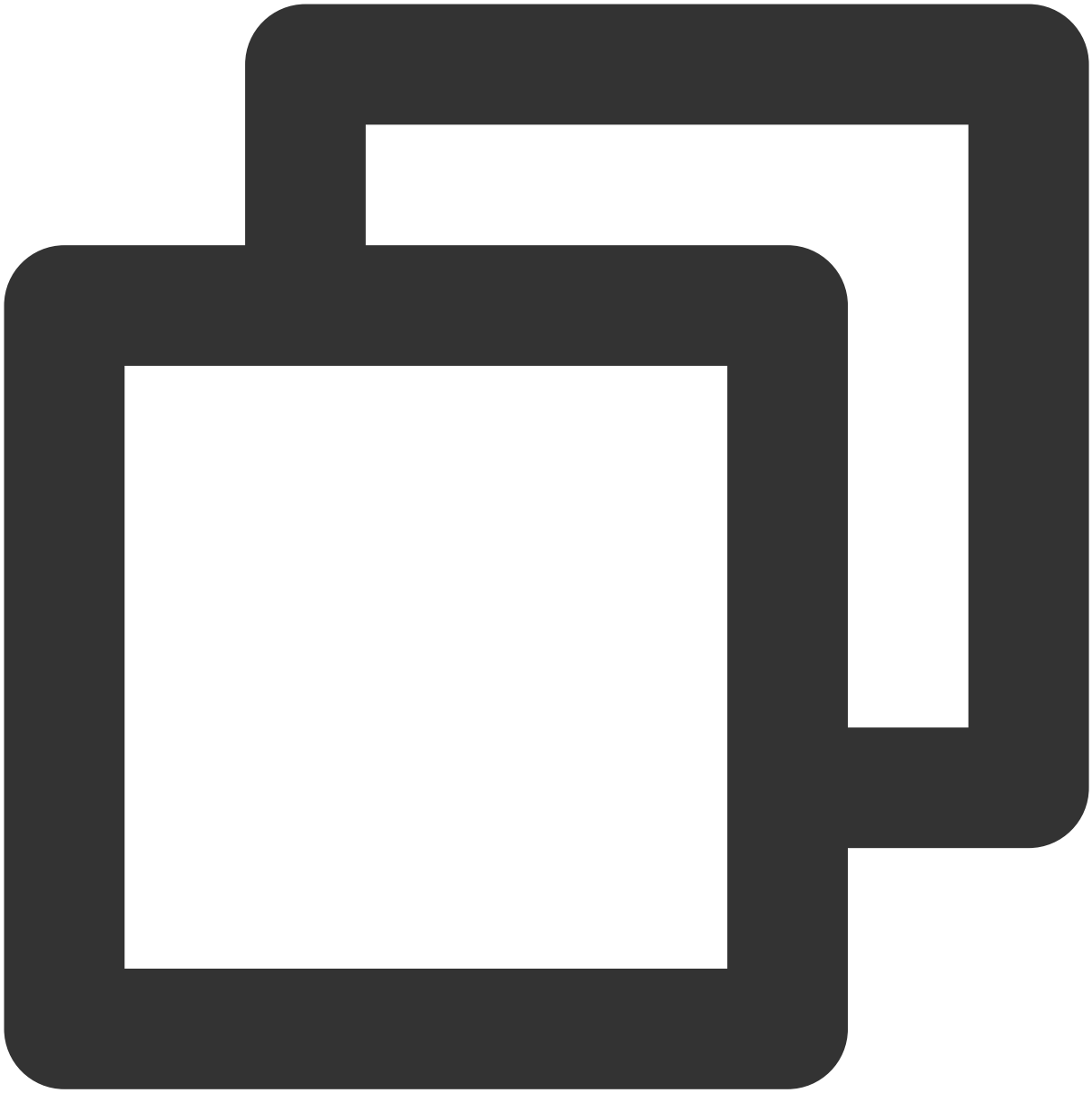
EdgeOne has been connected to Terraform to allow for quick configuration. This document describes how to use Terraform to configure the rule engine of the EdgeOne site and differentiate subdomain name configurations.

Prerequisites

1. You have installed and configured Terraform as instructed in Installing and Configuring Terraform.
2. You have connected to the site through Terraform as instructed in Creating Site Through Terraform.

Directions

1. Modify the Terraform configuration file by adding the resource definitions of the DNS record and rule engine of the subdomain name.
You can view the parameter definitions of the DNS record and rule engine on the Terraform Provider documentation page. Below is the `tencent_teo.tf` sample configuration file.

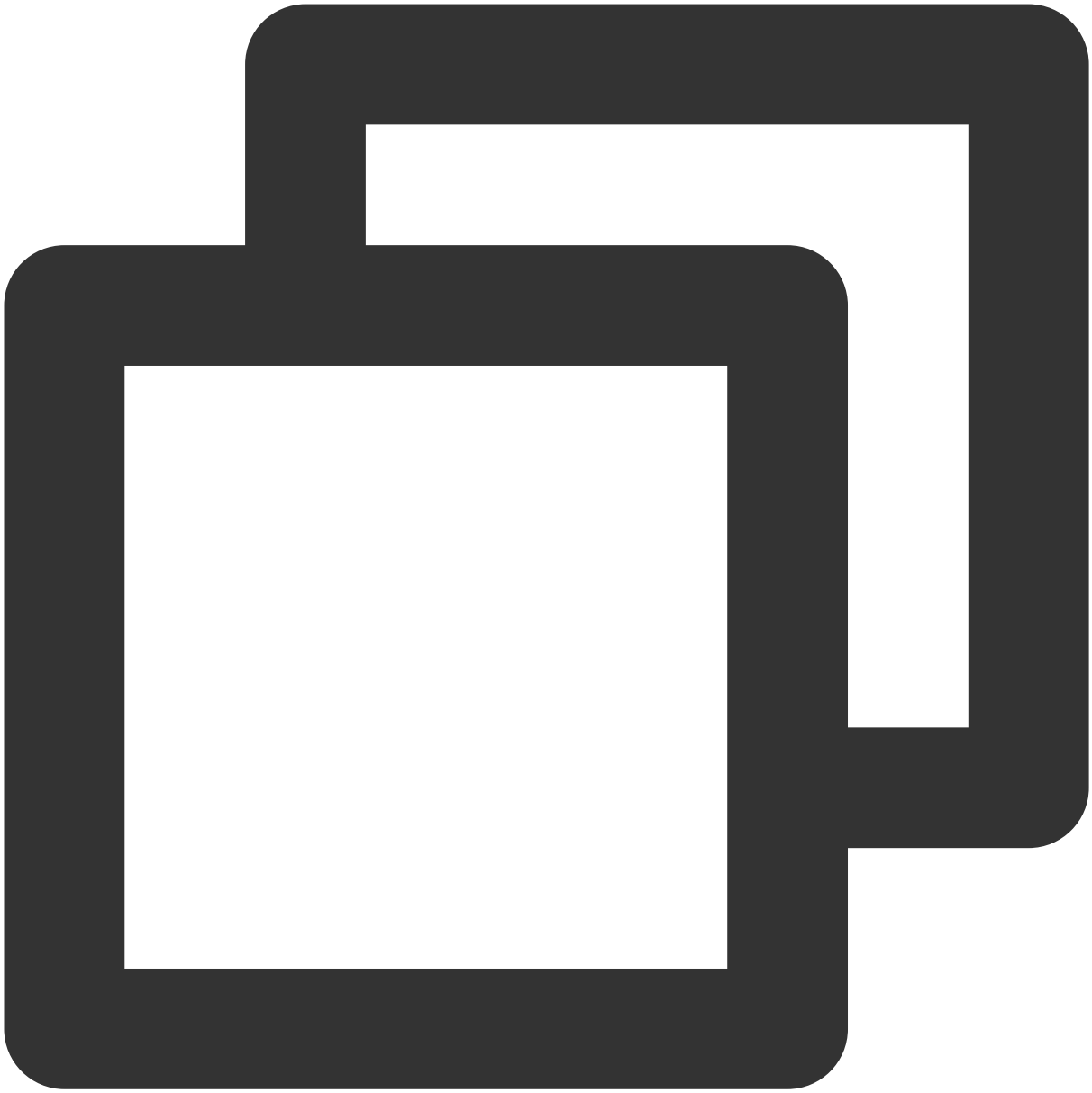


```
terraform {
  required_providers {
    tencentcloud = {
      source  = "tencentcloudstack/tencentcloud"
      version = ">= 1.78.5"
    }
  }
}
provider "tencentcloud" {
  secret_id  = "<your-secret-id>"
  secret_key = "<your-secret-key>"
```

```
  region     = "ap-guangzhou"
}
resource "tencentcloud_teo_zone" "example" {
  zone_name = "example.com"
  plan_type = "ent"
  tags = {
    "createdBy" = "terraform"
  }
}
# DNS record of the subdomain name
resource "tencentcloud_teo_dns_record" "rule_record" {
  zone_id = tencentcloud_teo_zone.example.id
  type    = "A"
  name    = "rule.example.com"
  content = "<your-backend-ip>"
  mode    = "proxied"
  ttl     = 300
}
# Differentiated configurations of the subdomain name
resource "tencentcloud_teo_rule_engine" "rule_example" {
  zone_id   = tencentcloud_teo_zone.example.id
  rule_name = "example_rule"
  status    = "enable" # Enable the rule
  rules {
    # For `rule.example.com` and requests with the `mp3` or `mp4` file extension
    or {
      and {
        target   = "host"
        operator = "equal"
        values   = [tencentcloud_teo_dns_record.rule_record.name]
      }
      and {
        target   = "extension"
        operator = "equal"
        values   = ["mp4", "mp3"]
      }
    }
    actions {
      # Use the specified `CacheKey`
      normal_action {
        action = "CacheKey"
        # `CacheKey` is case-sensitive
        parameters {
          name   = "Type"
          values = ["IgnoreCase"]
        }
        parameters {
```

```
          name   = "Switch"
          values = ["off"]
        }
        # `CacheKey` contains the `User-Agent` header
        parameters {
          name   = "Type"
          Values = ["Header"]
        }
        parameters {
          name   = "Switch"
          values = ["on"]
        }
        parameters {
          name   = "Value"
          values = "User-Agent"
        }
      }
    }
    # Add the specified response header
    actions {
      rewrite_action {
        action = "ResponseHeader"
        parameters {
          action = "add"
          name   = "Added-Header"
          values = ["Added-Value"]
        }
      }
    }
  }
}
```

2. Run the `terraform plan` command to preview the configuration and verify whether it is correct.



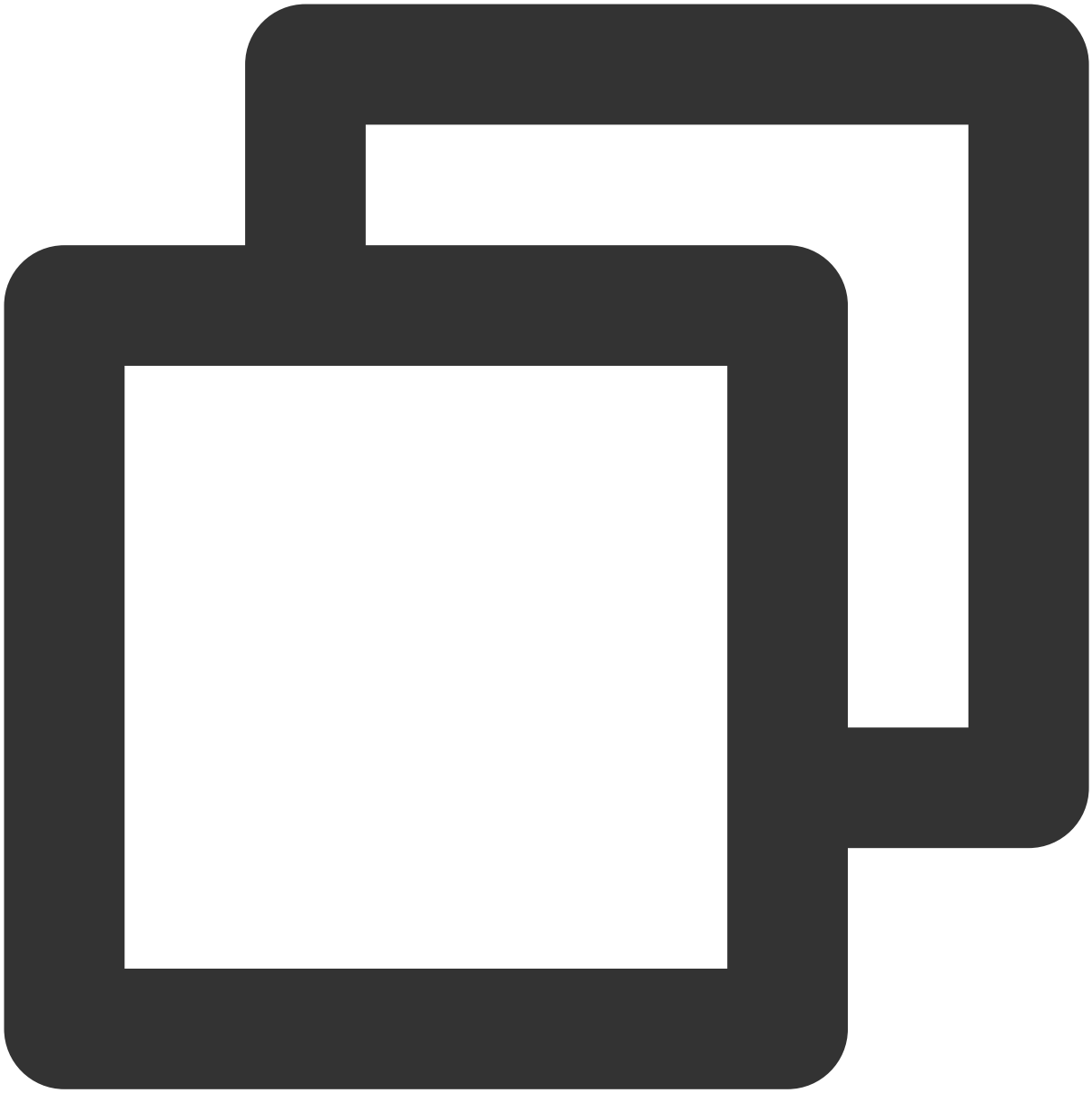
```
PS tf-doc> terraform plan
tencentcloud_teo_zone.example: Refreshing state... [id=zone-2ag9gej58j36]
Terraform used the selected providers to generate the following execution plan. Res
  + create
Terraform will perform the following actions:
  # tencentcloud_teo_dns_record.rule_record will be created
  + resource "tencentcloud_teo_dns_record" "rule_record" {
      + cname         = (known after apply)
      + content       = "<your-backend-ip>"
      + created_on    = (known after apply)
      + dns_record_id = (known after apply)
```

```
      + domain_status = (known after apply)
      + id            = (known after apply)
      + locked        = (known after apply)
      + mode          = "proxied"
      + modified_on   = (known after apply)
      + name          = "rule.example.com"
      + priority      = (known after apply)
      + status        = (known after apply)
      + ttl           = 300
      + type          = "A"
      + zone_id       = "zone-2ag9gej58j36"
    }
  # tencentcloud_teo_rule_engine.rule_example will be created
  + resource "tencentcloud_teo_rule_engine" "rule_example" {
      + id        = (known after apply)
      + rule_id   = (known after apply)
      + rule_name = "example_rule"
      + status    = "enable"
      + zone_id   = "zone-2ag9gej58j36"
      + rules {
          + actions {
              + normal_action {
                  + action = "CacheKey"
                  + parameters {
                      + name   = "Type"
                      + values = [
                          + "IgnoreCase",
                        ]
                    }
                  + parameters {
                      + name   = "Switch"
                      + values = [
                          + "off",
                        ]
                    }
                  + parameters {
                      + name   = "Type"
                      + values = [
                          + "Header",
                        ]
                    }
                  + parameters {
                      + name   = "Switch"
                      + values = [
                          + "on",
                        ]
                    }
```

```
                + parameters {
                    + name   = "Value"
                    + values = [
                        + "User-Agent",
                      ]
                  }
              }
          }
        + actions {
            + rewrite_action {
                + action = "ResponseHeader"
                + parameters {
                    + action = "add"
                    + name   = "Added-Header"
                    + values = [
                        + "Added-Value",
                      ]
                  }
              }
          }
        + or {
            + and {
                + operator = "equal"
                + target   = "host"
                + values   = [
                    + "rule.example.com",
                  ]
              }
            + and {
                + operator = "equal"
                + target   = "extension"
                + values   = [
                    + "mp3",
                    + "mp4",
                  ]
              }
          }
      }
  }
Plan: 2 to add, 0 to change, 0 to destroy.
─────────────────────────────────────────────────────────────────────────────
Note: You didn't use the -out option to save this plan, so Terraform can't guarante
```

3. Run `terraform apply` to create the DNS record and rule engine of the subdomain name.

After the `terraform apply` command is executed, Terraform will ask you to confirm the actions to be executed. After confirming that everything is correct, enter `yes` . Then, wait for the command to complete.



```
PS tf-doc> terraform apply
tencentcloud_teo_zone.example: Refreshing state... [id=zone-2ag9gej58j36]
Terraform used the selected providers to generate the following execution plan. Res
  + create
Terraform will perform the following actions:
  # tencentcloud_teo_dns_record.rule_record will be created
  + resource "tencentcloud_teo_dns_record" "rule_record" {
      + cname         = (known after apply)
      + content       = "<your-backend-ip>"
      + created_on    = (known after apply)
      + dns_record_id = (known after apply)
```

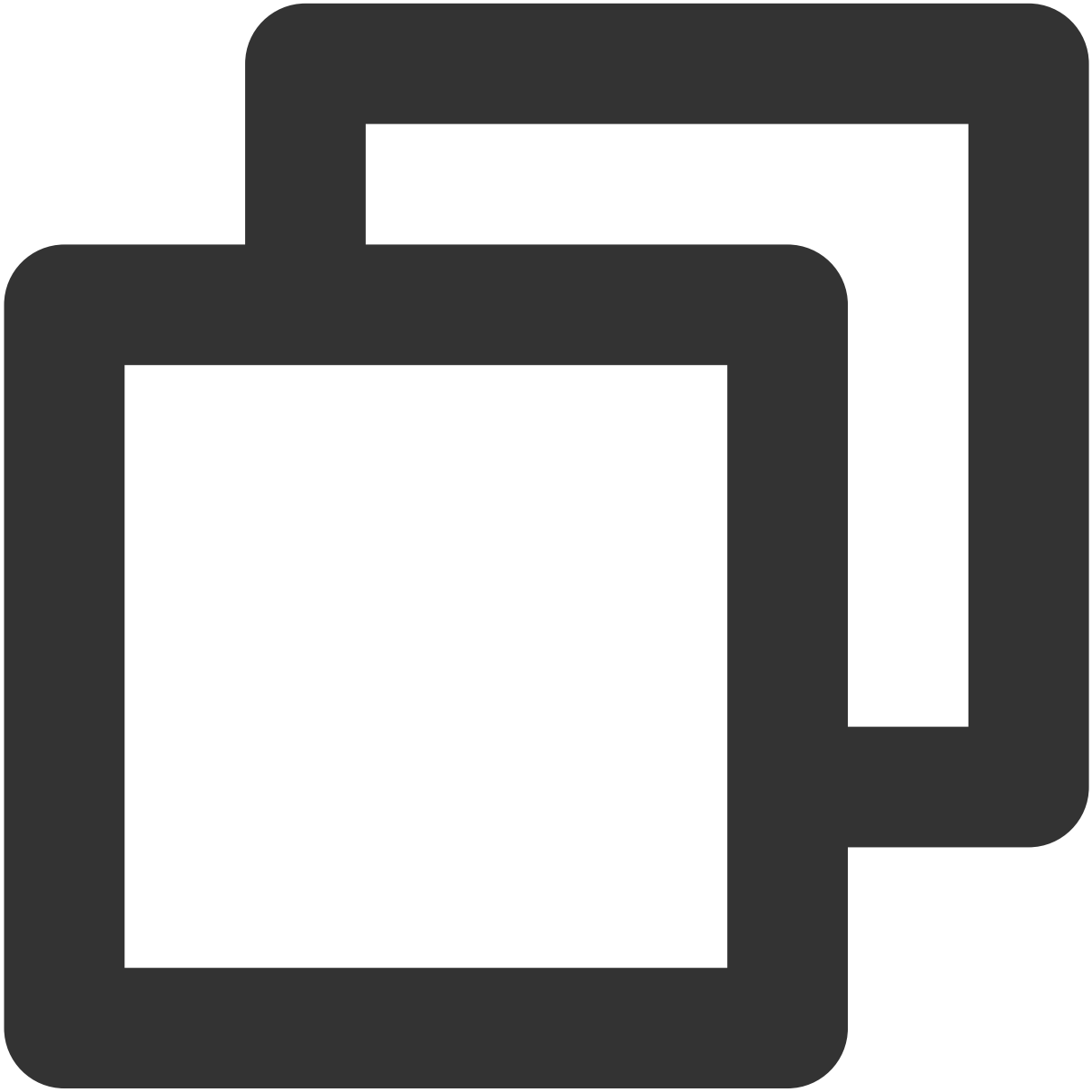
```
      + domain_status = (known after apply)
      + id            = (known after apply)
      + locked        = (known after apply)
      + mode          = "proxied"
      + modified_on   = (known after apply)
      + name          = "rule.example.com"
      + priority      = (known after apply)
      + status        = (known after apply)
      + ttl           = 300
      + type          = "A"
      + zone_id       = "zone-2ag9gej58j36"
    }
  # tencentcloud_teo_rule_engine.rule_example will be created
  + resource "tencentcloud_teo_rule_engine" "rule_example" {
      + id        = (known after apply)
      + rule_id   = (known after apply)
      + rule_name = "example_rule"
      + status    = "enable"
      + zone_id   = "zone-2ag9gej58j36"
      + rules {
          + actions {
              + normal_action {
                  + action = "CacheKey"
                  + parameters {
                      + name   = "Type"
                      + values = [
                          + "IgnoreCase",
                        ]
                    }
                  + parameters {
                      + name   = "Switch"
                      + values = [
                          + "off",
                        ]
                    }
                  + parameters {
                      + name   = "Type"
                      + values = [
                          + "Header",
                        ]
                    }
                  + parameters {
                      + name   = "Switch"
                      + values = [
                          + "on",
                        ]
                    }
```

```
                    + parameters {
                        + name   = "Value"
                        + values = [
                            + "User-Agent",
                          ]
                      }
                  }
              }
            + actions {
                + rewrite_action {
                    + action = "ResponseHeader"
                    + parameters {
                        + action = "add"
                        + name   = "Added-Header"
                        + values = [
                            + "Added-Value",
                          ]
                      }
                  }
              }
            + or {
                + and {
                    + operator = "equal"
                    + target   = "host"
                    + values   = [
                        + "rule.example.com",
                      ]
                  }
                + and {
                    + operator = "equal"
                    + target   = "extension"
                    + values   = [
                        + "mp3",
                        + "mp4",
                      ]
                  }
              }
          }
      }
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_teo_dns_record.rule_record: Creating...
tencentcloud_teo_dns_record.rule_record: Creation complete after 2s [id=zone-2ag9ge
tencentcloud_teo_rule_engine.rule_example: Creating...
```

```
tencentcloud_teo_rule_engine.rule_example: Creation complete after 1s [id=zone-2ag9
Apply complete! Resources: 2 added, 0 changed, 0 destroyed.
```

4. Check the command execution result.

You can run `terraform show` to check whether the rule engine configuration takes effect or log in to the EdgeOne console for confirmation.



```
PS tf-doc> terraform state show tencentcloud_teo_dns_record.rule_record
# tencentcloud_teo_dns_record.rule_record:
resource "tencentcloud_teo_dns_record" "rule_record" {
    content       = "<your-backend-ip>"
```

```
    created_on    = "2022-10-20T06:31:38Z"
    dns_record_id = "record-2ahmb3w7ssl8"
    domain_status = [
        "security",
    ]
    id            = "zone-2ag9gej58j36#record-2ahmb3w7ssl8"
    locked        = false
    mode          = "proxied"
    modified_on   = "2022-10-20T06:31:38Z"
    name          = "rule.example.com"
    priority      = 0
    status        = "active"
    ttl           = 300
    type          = "A"
    zone_id       = "zone-2ag9gej58j36"
}
PS tf-doc> terraform state show tencentcloud_teo_rule_engine.rule_example
# tencentcloud_teo_rule_engine.rule_example:
resource "tencentcloud_teo_rule_engine" "rule_example" {
    id        = "zone-2ag9gej58j36#rule-2ahmb5dhn9qq"
    rule_id   = "rule-2ahmb5dhn9qq"
    rule_name = "example_rule"
    status    = "enable"
    zone_id   = "zone-2ag9gej58j36"
    rules {
        actions {
            normal_action {
                action = "CacheKey"
                parameters {
                    name   = "Type"
                    values = [
                        "IgnoreCase",
                    ]
                }
                parameters {
                    name   = "Switch"
                    values = [
                        "off",
                    ]
                }
                parameters {
                    name   = "Type"
                    values = [
                        "Header",
                    ]
                }
                parameters {
```

```
                    name   = "Switch"
                    values = [
                        "on",
                    ]
                }
                parameters {
                    name   = "Value"
                    values = [
                        "User-Agent",
                    ]
                }
            }
        }
        actions {
            rewrite_action {
                action = "ResponseHeader"
                parameters {
                    action = "add"
                    name   = "Added-Header"
                    values = [
                        "Added-Value",
                    ]
                }
            }
        }
        or {
            and {
                operator = "equal"
                target   = "host"
                values   = [
                    "rule.example.com",
                ]
            }
            and {
                operator = "equal"
                target   = "extension"
                values   = [
                    "mp3",
                    "mp4",
                ]
            }
        }
    }
}
```

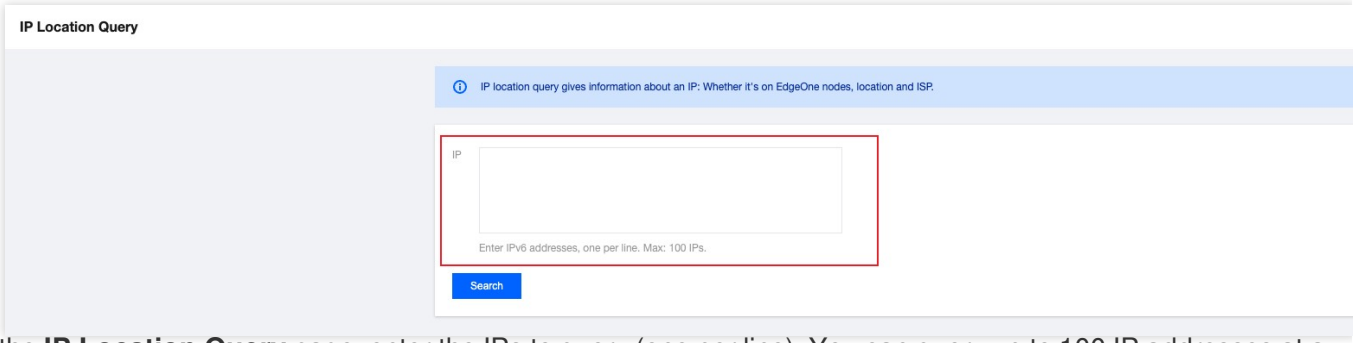
IP Location Query

Last updated：2023-04-10 18:14:58

This document describes how to verify whether an IP is owned by EdgeOne and query the IP geolocation.

Directions

1. Log in to the EdgeOne console and click **IP Location Query** in the left sidebar.



2. On the **IP Location Query** page, enter the IPs to query (one per line). You can query up to 100 IP addresses at a time. IPv6 addresses are supported.
3. Click **Search**. The **Query results** table shows the IP geolocation and whether they are owned by EdgeOne nodes. To export the query results, click the download icon

in the top-right corner of the table. The query results are exported to a CSV file.