certbot-dns-google Documentation

Release 0

Certbot Project

Jun 01, 2026

CONTENTS:

The `dns_google` plugin automates the process of completing a `dns-01` challenge (`DNS01`) by creating, and subsequently removing, TXT records using the Google Cloud DNS API.

Note

The plugin is not installed by default. It can be installed by heading to certbot.eff.org, choosing your system and selecting the Wildcard tab.

CHAPTER

ONE

NAMED ARGUMENTS

`--dns-google-credent`	Google Cloud Platform *credentials* JSON file. (Required if not using Application Default Credentials.)
`--dns-google-project`	The ID of the Google Cloud project that the Google Cloud DNS managed zone(s) reside in. (Default: project that the Google *credentials* belong to)
`--dns-google-propaga`	The number of seconds to wait for DNS to propagate before asking the ACME server to verify the DNS record. (Default: 60)

CHAPTER

TWO

CREDENTIALS

Use of this plugin requires Google Cloud Platform credentials with the ability to modify the Cloud DNS managed zone(s) for which certificates are being issued.

In most cases, configuring credentials for Certbot will require creating a service account, and then either *granting permissions with predefined roles* or *granting permissions with custom roles* using IAM.

2.1 Providing Credentials

The preferred method of providing credentials is to set up Application Default Credentials (ADC) in the environment that Certbot is running in.

If you are running Certbot on Google Cloud then a service account can be assigned directly to most types of workload, including Compute Engine VMs, Kubernetes Engine Pods, Cloud Run jobs, Cloud Functions, and Cloud Builds.

If you are not running Certbot on Google Cloud then a credentials file should be provided using the `--dns-google-credentials` command-line argument. Google provides documentation for creating service account keys, which is the most common method of using a service account outside of Google Cloud.

Listing 1: Example service account key file:

```
{
   "type": "service_account",
   "project_id": "...",
   "private_key_id": "...",
   "private_key": "...",
   "client_email": "...",
   "client_id": "...",
   "auth_uri": "https://accounts.google.com/o/oauth2/auth",
   "token_uri": "https://accounts.google.com/o/oauth2/token",
   "auth_provider_x509_cert_url": "https://www.googleapis.com/oauth2/v1/certs",
   "client_x509_cert_url": "..."
}
```

Caution

You should protect these credentials as you would a password. Users who can read this file can use these credentials to issue some types of API calls on your behalf, limited by the permissions assigned to the account. Users who can cause Certbot to run using these credentials can complete a `dns-01` challenge to acquire new certificates or revoke existing certificates for domains these credentials are authorized to manage.

Certbot will emit a warning if it detects that the credentials file can be accessed by other users on your system. The warning reads "Unsafe permissions on credentials configuration file", followed by the path to the credentials file. This warning will be emitted each time Certbot uses the credentials file, including for renewal, and cannot be silenced except by addressing the issue (e.g., by using a command like `chmod 600` to restrict access to the file).

If you are running Certbot within another cloud platform, a CI platform, or any other platform that supports issuing OpenID Connect Tokens, then you may also have the option of securely authenticating with workload identity federation. Instructions are generally available for most platforms, including AWS or Azure, GitHub Actions, and GitLab CI.

2.2 Granting Permissions with Predefined Roles

The simplest method of granting the required permissions to the user or service account that Certbot is authenticating with is to use either of these predefined role strategies:

- dns.admin against the *DNS zone(s)* that Certbot will be issuing certificates for.
- dns.reader against the *project* containing the relevant DNS zones.

or

- dns.admin against the *project* containing the relevant DNS zones

For instructions on how to grant roles, please read the Google provided documentation for granting access roles against a project and granting access roles against zones.

Caution

Granting the `dns.admin` role at the project level can present a significant security risk. It provides full administrative access to all DNS zones within the project, granting the ability to perform any action up to and including deleting all zones within a project.

2.3 Granting Permissions with Custom Roles

Custom roles are an alternative to predefined roles that provide the ability to define fine grained permission sets for specific use cases. They should generally be used when it is desirable to adhere to the principle of least privilege, such as within production or other security sensitive workloads.

The following is an example strategy for granting granular permissions to Certbot using custom roles. If you are not already familiar with how to do so, Google provides documentation for creating a custom IAM role.

Firstly, create a custom role containing the permissions required to make DNS record updates. We suggest naming the custom role `Certbot - Zone Editor` with the ID `certbot.zoneEditor`. The following permissions are required:

- `dns.changes.create`
- `dns.changes.get`
- `dns.changes.list`
- `dns.resourceRecordSets.create`
- `dns.resourceRecordSets.delete`
- `dns.resourceRecordSets.list`
- `dns.resourceRecordSets.update`

Next, create a custom role granting Certbot the ability to discover DNS zones. We suggest naming the custom role `Certbot - Zone Lister` with the ID `certbot.zoneLister`. The following permissions are required:

- `dns.managedZones.get`
- `dns.managedZones.list`

Finally, grant the custom roles to the user or service account that Certbot is authenticating with:

- Grant your custom `Certbot - Zone Editor` role against the *DNS zone(s)* that Certbot will be issuing certificates for.
- Grant your custom `Certbot - Zone Lister` role against the *project* containing the relevant DNS zones.

For instructions on how to grant roles, please read the Google provided documentation for granting access roles against a project and granting access roles against zones.

CHAPTER

THREE

EXAMPLES

Listing 1: To acquire a certificate for `example.com`, providing a credentials file

```
certbot certonly \
  --dns-google \
  --dns-google-credentials ~/.secrets/certbot/google.json \
  -d example.com
```

Listing 2: To acquire a certificate for `example.com`, where ADC is available and a credentials file is not required

```
certbot certonly \
  --dns-google \
  -d example.com
```

Listing 3: To acquire a single certificate for both `example.com` and `www.example.com`

```
certbot certonly \
  --dns-google \
  -d example.com \
  -d www.example.com
```

Listing 4: To acquire a certificate for `example.com`, where the managed DNS zone resides in another Google Cloud project.

```
certbot certonly \
  --dns-google \
  --dns-google-credentials ~/.secrets/certbot/google-project-test-foo.json \
  --dns-google-project test-bar \
  -d example.com
```

Listing 5: To acquire a certificate for `example.com`, waiting 120 seconds for DNS propagation

```
certbot certonly \
  --dns-google \
  --dns-google-propagation-seconds 120 \
  -d example.com
```

CHAPTER

FOUR

API DOCUMENTATION

Certbot plugins implement the Certbot plugins API, and do not otherwise have an external API.

CHAPTER

FIVE

INDICES AND TABLES

- genindex
- modindex
- search

PYTHON MODULE INDEX

C

INDEX

C

M