ACME Certificate and Account Provider

The Automated Certificate Management Environment (ACME) is an evolving standard for the automation of a domain-validated certificate authority. Clients register themselves on an authority using a private key and contact information, and answer challenges for domains that they own by supplying response data issued by the authority via either HTTP or DNS. Via this process, they prove that they own the domains in question, and can then request certificates for them via the CA. No part of this process requires user interaction, a traditional blocker in obtaining a domain validated certificate.

Currently the major ACME CA is Let's Encrypt (https://letsencrypt.org), but the ACME support in Terraform can be configured to use any ACME CA, including an internal one that is set up using Boulder (https://github.com/letsencrypt/boulder), or another CA that implements the ACME standard with Let's Encrypt's divergences (https://github.com/letsencrypt/boulder/blob/master/docs/acme-divergences.md).

For more detail on the ACME process, see here (https://letsencrypt.org/how-it-works/). For the ACME spec, click here (https://ietf-wg-acme.github.io/acme/draft-ietf-acme-acme.html). Note that as mentioned in the last paragraph, the ACME provider may diverge (https://github.com/letsencrypt/boulder/blob/master/docs/acme-divergences.md) from the current ACME spec to account for the real-world divergences that are made by CAs such as Let's Encrypt.

NOTE: The upstream version of the ACME provider supports ACME v2 only. For ACME v1 endpoints, version 0.6.0 is required, which can be found here (https://github.com/vancluever/terraform-provider-acme/releases/tag/v0.6.0). Note that this version is a 3rd party plugin (/docs/configuration/providers.html#third-party-plugins) and needs to be installed as such.

Basic Example

The following example can be used to create an account using the `acme_registration` (/docs/providers/acme/r/registration.html) resource, and a certificate using the `acme_certificate` (/docs/providers/acme/r/certificate.html) resource. The initial private key is created using the `tls_private_key` (/docs/providers/tls/r/private_key.html) resource, but can be supplied via other means. DNS validation is performed by using Amazon Route 53 (https://aws.amazon.com/route53/), for which appropriate credentials are assumed to be in your environment.

NOTE: The directory URLs in all examples in this provider reference Let's Encrypt's staging server endpoint. For production use, change the directory URLs to the production endpoints, which can be found here (https://letsencrypt.org/docs/acme-protocol-updates/).

```
provider "acme" {
  server_url = "https://acme-staging-v02.api.letsencrypt.org/directory"
}

resource "tls_private_key" "private_key" {
  algorithm = "RSA"
}

resource "acme_registration" "reg" {
  account_key_pem = "${tls_private_key.private_key.private_key_pem}"
  email_address   = "nobody@example.com"
}

resource "acme_certificate" "certificate" {
  account_key_pem           = "${acme_registration.reg.account_key_pem}"
  common_name               = "www.example.com"
  subject_alternative_names = ["www2.example.com"]

  dns_challenge {
    provider = "route53"
  }
}
```

Argument Reference

The following arguments are required:

- `server_url` - (Required) The URL to the ACME endpoint's directory.

> Note that the account key is not a provider-level config value at this time to allow the management of accounts and certificates within the same provider.

acme_certificate DNS Challenge Providers

This subsection documents all of the DNS challenge providers that can be used with the `acme_certificate` (/docs/providers/acme/r/certificate.html) resource.

For complete information on how to use these providers with the `acme_certifiate` resource, see here (/docs/providers/acme/r/certificate.html#using-dns-challenges).

Refer to a specific provider on the left sidebar for more details.

Relation to Terraform provider configuration

The DNS provider configuration specified in the `acme_certificate` (/docs/providers/acme/r/certificate.html) resource is separate from any that you supply in a corresponding provider whose functionality overlaps with the certificate's DNS providers. This ensures that there are no hard dependencies between any of these providers and the ACME provider, but it is important to note so that configuration is supplied correctly.

As an example, if you specify manual configuration for the AWS provider (/docs/providers/aws/index.html) via the `provider` (/docs/configuration/providers.html) block instead of the environment, you will still need to supply the configuration explicitly in the `config` block of the `dns_challenge` (/docs/providers/acme/r/certificate.html#dns_challenge) argument.

Note that some of Terraform's providers have environment variable settings that overlap with the settings here, generally depending on whether or not these variables are supported by the corresponding provider's SDK.

We alias certain provider environment variables so the same settings can be supplied to both ACME and the respective native cloud provider. For specific details, see the page for the provider in question.

AuroraDNS DNS Challenge Provider

The `auroradns` DNS challenge provider can be used to perform DNS challenges for the `acme_certificate` (/docs/providers/acme/r/certificate.html) resource with AuroraDNS (https://auroradns.microsoft.com/en-ca/).

For complete information on how to use this provider with the `acme_certifiate` resource, see here (/docs/providers/acme/r/certificate.html#using-dns-challenges).

Example

```
resource "acme_certificate" "certificate" {
  ...

  dns_challenge {
    provider = "auroradns"
  }
}
```

Argument Reference

The following arguments can be either passed as environment variables, or directly through the `config` block in the `dns_challenge` (/docs/providers/acme/r/certificate.html#dns_challenge) argument in the `acme_certificate` (/docs/providers/acme/r/certificate.html) resource. For more details, see here (/docs/providers/acme/r/certificate.html#using-dns-challenges).

- `AURORA_USER_ID` - The user ID to use.
- `AURORA_KEY` - The key to use.

Azure DNS Challenge Provider

The `azure` DNS challenge provider can be used to perform DNS challenges for the `acme_certificate` (/docs/providers/acme/r/certificate.html) resource with Microsoft Azure (https://azure.microsoft.com/en-ca/).

For complete information on how to use this provider with the `acme_certifiate` resource, see here (/docs/providers/acme/r/certificate.html#using-dns-challenges).

Example

```
resource "acme_certificate" "certificate" {
  ...

  dns_challenge {
    provider = "azure"
  }
}
```

Argument Reference

The following arguments can be either passed as environment variables, or directly through the `config` block in the `dns_challenge` (/docs/providers/acme/r/certificate.html#dns_challenge) argument in the `acme_certificate` (/docs/providers/acme/r/certificate.html) resource. For more details, see here (/docs/providers/acme/r/certificate.html#using-dns-challenges).

- `AZURE_CLIENT_ID` - The Client ID of the Service Principal. Can also be supplied with `ARM_CLIENT_ID`.
- `AZURE_CLIENT_SECRET` - The Client Secret associated with the Service Principal. Can also be supplied with `ARM_CLIENT_SECRET`.
- `AZURE_SUBSCRIPTION_ID` - The ID of the Azure Subscription. Can also be supplied with `ARM_SUBSCRIPTION_ID`.
- `AZURE_TENANT_ID` - The Tenant ID to use. Can also be supplied with `ARM_TENANT_ID`.
- `AZURE_RESOURCE_GROUP` - The resource group to use to place the DNS records in. Can also be supplied with `ARM_RESOURCE_GROUP`.

Bluecat DNS Challenge Provider

The `bluecat` DNS challenge provider can be used to perform DNS challenges for the `acme_certificate` (/docs/providers/acme/r/certificate.html) resource with Bluecat Address Manager (https://www.bluecatnetworks.com/platform/management/bluecat-address-manager/).

For complete information on how to use this provider with the `acme_certifiate` resource, see here (/docs/providers/acme/r/certificate.html#using-dns-challenges).

Example

```
resource "acme_certificate" "certificate" {
  ...

  dns_challenge {
    provider = "bluecat"
  }
}
```

Argument Reference

The following arguments can be either passed as environment variables, or directly through the `config` block in the `dns_challenge` (/docs/providers/acme/r/certificate.html#dns_challenge) argument in the `acme_certificate` (/docs/providers/acme/r/certificate.html) resource. For more details, see here (/docs/providers/acme/r/certificate.html#using-dns-challenges).

- `BLUECAT_SERVER_URL` - The URL for the address manager to use.
- `BLUECAT_USER_NAME` - The user name to use.
- `BLUECAT_PASSWORD` - The password to use for the supplied user name.
- `BLUECAT_CONFIG_NAME` - The configuration name to use.
- `BLUECAT_DNS_VIEW` - The DNS view to use.

Cloudflare DNS Challenge Provider

The `cloudflare` DNS challenge provider can be used to perform DNS challenges for the `acme_certificate` (/docs/providers/acme/r/certificate.html) resource with Cloudflare DNS (https://www.cloudflare.com/dns/).

For complete information on how to use this provider with the `acme_certifiate` resource, see here (/docs/providers/acme/r/certificate.html#using-dns-challenges).

Example

```
resource "acme_certificate" "certificate" {
  ...

  dns_challenge {
    provider = "cloudflare"
  }
}
```

Argument Reference

The following arguments can be either passed as environment variables, or directly through the `config` block in the `dns_challenge` (/docs/providers/acme/r/certificate.html#dns_challenge) argument in the `acme_certificate` (/docs/providers/acme/r/certificate.html) resource. For more details, see here (/docs/providers/acme/r/certificate.html#using-dns-challenges).

- `CLOUDFLARE_EMAIL` - The email address to use.
- `CLOUDFLARE_API_KEY` - The API key to use.

CloudXNS DNS Challenge Provider

The `cloudxns` DNS challenge provider can be used to perform DNS challenges for the `acme_certificate` (/docs/providers/acme/r/certificate.html) resource with CloudXNS (https://www.cloudxns.net/).

For complete information on how to use this provider with the `acme_certifiate` resource, see here (/docs/providers/acme/r/certificate.html#using-dns-challenges).

Example

```
resource "acme_certificate" "certificate" {
  ...

  dns_challenge {
    provider = "cloudxns"
  }
}
```

Argument Reference

The following arguments can be either passed as environment variables, or directly through the `config` block in the `dns_challenge` (/docs/providers/acme/r/certificate.html#dns_challenge) argument in the `acme_certificate` (/docs/providers/acme/r/certificate.html) resource. For more details, see here (/docs/providers/acme/r/certificate.html#using-dns-challenges).

- `CLOUDXNS_API_KEY` - The API key to use.
- `CLOUDXNS_SECRET_KEY` - The secret key to use.

DigitalOcean DNS Challenge Provider

The `digitalocean` DNS challenge provider can be used to perform DNS challenges for the `acme_certificate` (/docs/providers/acme/r/certificate.html) resource with DigitalOcean (https://www.digitalocean.com/).

For complete information on how to use this provider with the `acme_certifiate` resource, see here (/docs/providers/acme/r/certificate.html#using-dns-challenges).

Example

```
resource "acme_certificate" "certificate" {
  ...

  dns_challenge {
    provider = "digitalocean"
  }
}
```

Argument Reference

The following arguments can be either passed as environment variables, or directly through the `config` block in the `dns_challenge` (/docs/providers/acme/r/certificate.html#dns_challenge) argument in the `acme_certificate` (/docs/providers/acme/r/certificate.html) resource. For more details, see here (/docs/providers/acme/r/certificate.html#using-dns-challenges).

- `DO_AUTH_TOKEN` - The auth token to use.

DNSimple DNS Challenge Provider

The `dnsimple` DNS challenge provider can be used to perform DNS challenges for the `acme_certificate` (/docs/providers/acme/r/certificate.html) resource with DNSimple (https://dnsimple.com/).

For complete information on how to use this provider with the `acme_certifiate` resource, see here (/docs/providers/acme/r/certificate.html#using-dns-challenges).

Example

```
resource "acme_certificate" "certificate" {
  ...

  dns_challenge {
    provider = "dnsimple"
  }
}
```

Argument Reference

The following arguments can be either passed as environment variables, or directly through the `config` block in the `dns_challenge` (/docs/providers/acme/r/certificate.html#dns_challenge) argument in the `acme_certificate` (/docs/providers/acme/r/certificate.html) resource. For more details, see here (/docs/providers/acme/r/certificate.html#using-dns-challenges).

- `DNSIMPLE_OAUTH_TOKEN` - The OAuth token to use.

DNS Made Easy DNS Challenge Provider

The `dnsmadeeasy` DNS challenge provider can be used to perform DNS challenges for the `acme_certificate` (/docs/providers/acme/r/certificate.html) resource with DNS Made Easy (https://dnsmadeeasy.com/).

For complete information on how to use this provider with the `acme_certifiate` resource, see here (/docs/providers/acme/r/certificate.html#using-dns-challenges).

Example

```
resource "acme_certificate" "certificate" {
  ...

  dns_challenge {
    provider = "dnsmadeeasy"
  }
}
```

Argument Reference

The following arguments can be either passed as environment variables, or directly through the `config` block in the `dns_challenge` (/docs/providers/acme/r/certificate.html#dns_challenge) argument in the `acme_certificate` (/docs/providers/acme/r/certificate.html) resource. For more details, see here (/docs/providers/acme/r/certificate.html#using-dns-challenges).

- `DNSMADEEASY_API_KEY` - The API key to use.

- `DNSMADEEASY_API_SECRET` - The secret key to use.

DNSPod DNS Challenge Provider

The `dnspod` DNS challenge provider can be used to perform DNS challenges for the `acme_certificate` (/docs/providers/acme/r/certificate.html) resource with DNSPod (https://www.dnspod.cn/).

For complete information on how to use this provider with the `acme_certifiate` resource, see here (/docs/providers/acme/r/certificate.html#using-dns-challenges).

Example

```
resource "acme_certificate" "certificate" {
  ...

  dns_challenge {
    provider = "dnspod"
  }
}
```

Argument Reference

The following arguments can be either passed as environment variables, or directly through the `config` block in the `dns_challenge` (/docs/providers/acme/r/certificate.html#dns_challenge) argument in the `acme_certificate` (/docs/providers/acme/r/certificate.html) resource. For more details, see here (/docs/providers/acme/r/certificate.html#using-dns-challenges).

- `DNSPOD_API_KEY` - The API key to use.

DuckDNS DNS Challenge Provider

The `duckdns` DNS challenge provider can be used to perform DNS challenges for the `acme_certificate` (/docs/providers/acme/r/certificate.html) resource with DuckDNS (http://www.duckdns.org/).

For complete information on how to use this provider with the `acme_certifiate` resource, see here (/docs/providers/acme/r/certificate.html#using-dns-challenges).

Example

```
resource "acme_certificate" "certificate" {
  ...

  dns_challenge {
    provider = "duckdns"
  }
}
```

Argument Reference

The following arguments can be either passed as environment variables, or directly through the `config` block in the `dns_challenge` (/docs/providers/acme/r/certificate.html#dns_challenge) argument in the `acme_certificate` (/docs/providers/acme/r/certificate.html) resource. For more details, see here (/docs/providers/acme/r/certificate.html#using-dns-challenges).

- `DUCKDNS_TOKEN` - The auth token to use.

Dyn DNS Challenge Provider

The `dyn` DNS challenge provider can be used to perform DNS challenges for the `acme_certificate` (/docs/providers/acme/r/certificate.html) resource with Dyn (https://dyn.com/).

For complete information on how to use this provider with the `acme_certifiate` resource, see here (/docs/providers/acme/r/certificate.html#using-dns-challenges).

Example

```
resource "acme_certificate" "certificate" {
  ...

  dns_challenge {
    provider = "dyn"
  }
}
```

Argument Reference

The following arguments can be either passed as environment variables, or directly through the `config` block in the `dns_challenge` (/docs/providers/acme/r/certificate.html#dns_challenge) argument in the `acme_certificate` (/docs/providers/acme/r/certificate.html) resource. For more details, see here (/docs/providers/acme/r/certificate.html#using-dns-challenges).

- `DYN_CUSTOMER_NAME` - The customer name to use.
- `DYN_USER_NAME` - The user name to use.
- `DYN_PASSWORD` - The password for the supplied user.

Exec DNS Challenge Provider

The `exec` DNS challenge provider can be used to perform DNS challenges for the `acme_certificate` (/docs/providers/acme/r/certificate.html) resource, using a custom external script.

For complete information on how to use this provider with the `acme_certifiate` resource, see here (/docs/providers/acme/r/certificate.html#using-dns-challenges).

Example

```
resource "acme_certificate" "certificate" {
  ...

  dns_challenge {
    provider = "exec"

    config = {
      "EXEC_PATH" = "./update-dns.sh"
    }
  }
}
```

Usage Details

The file name of the external script is specified in the environment variable `EXEC_PATH`. When it is run by Terraform, four command-line parameters are passed to it: The action ("present" or "cleanup"), the fully-qualified domain name, the value for the record, and the TTL.

In the above basic example, the `update-dns.sh` script would be called in the following fashion:

```
./update-dns.sh "present" "_acme-challenge.foo.example.com." "MsijOYZxqyjGnFGwhjrhfg-Xgbl5r68WPda0J9EgqqI
" "120"
```

If the script returns a non-zero return code, the execution of the update is considered to have failed, and Terraform will return an error.

When the record is to be removed, the script is called again, with the first command-line parameter set to "cleanup" instead of "present".

Using raw values

If you want to use the raw domain, token, and keyAuth values with your script, you can set `EXEC_MODE` to `RAW`. When used like this, `update-dns.sh` will be called in the following way:

```
./update-dns.sh "present" "foo.example.com." "--" "some-token" "KxAy-J3NwUmg9ZQuM-gP_Mq1nStaYSaP9tYQs5_-Y
sE.ksT-qywTd8058G-SHHWA3RAN72Pr0yWtPYmmY5UBpQ8"
```

Argument Reference

The following arguments can be either passed as environment variables, or directly through the `config` block in the `dns_challenge` (/docs/providers/acme/r/certificate.html#dns_challenge) argument in the `acme_certificate` (/docs/providers/acme/r/certificate.html) resource. For more details, see here (/docs/providers/acme/r/certificate.html#using-dns-challenges).

> **NOTE:** Due to the nature of the `exec` provider, it's recommended that these be supplied as explicit `config` values.

- `EXEC_MODE` - Send the raw domain, token, and keyAuth values to the external script. The only usable value here is `RAW`.
- `EXEC_PATH` - The path to the external script to call.

Exoscale DNS Challenge Provider

The `exoscale` DNS challenge provider can be used to perform DNS challenges for the `acme_certificate` (/docs/providers/acme/r/certificate.html) resource with Exoscale (https://www.exoscale.com/dns/).

For complete information on how to use this provider with the `acme_certifiate` resource, see here (/docs/providers/acme/r/certificate.html#using-dns-challenges).

Example

```
resource "acme_certificate" "certificate" {
  ...

  dns_challenge {
    provider = "exoscale"
  }
}
```

Argument Reference

The following arguments can be either passed as environment variables, or directly through the `config` block in the `dns_challenge` (/docs/providers/acme/r/certificate.html#dns_challenge) argument in the `acme_certificate` (/docs/providers/acme/r/certificate.html) resource. For more details, see here (/docs/providers/acme/r/certificate.html#using-dns-challenges).

- `EXOSCALE_API_KEY` - The API key to use.
- `EXOSCALE_API_SECRET` - The API secret to use.
- `EXOSCALE_ENDPOINT` - The API endpoint to use.

Akamai FastDNS DNS Challenge Provider

The `fastdns` DNS challenge provider can be used to perform DNS challenges for the `acme_certificate` (/docs/providers/acme/r/certificate.html) resource with Akamai FastDNS (https://www.akamai.com/us/en/products/cloud-security/fast-dns.jsp).

For complete information on how to use this provider with the `acme_certifiate` resource, see here (/docs/providers/acme/r/certificate.html#using-dns-challenges).

Example

```
resource "acme_certificate" "certificate" {
  ...

  dns_challenge {
    provider = "fastdns"
  }
}
```

Argument Reference

The following arguments can be either passed as environment variables, or directly through the `config` block in the `dns_challenge` (/docs/providers/acme/r/certificate.html#dns_challenge) argument in the `acme_certificate` (/docs/providers/acme/r/certificate.html) resource. For more details, see here (/docs/providers/acme/r/certificate.html#using-dns-challenges).

- `AKAMAI_HOST` - The host to use.
- `AKAMAI_CLIENT_TOKEN` - The client token to use.
- `AKAMAI_CLIENT_SECRET` - The client secret to use.
- `AKAMAI_ACCESS_TOKEN` - The access token to use.

Gandi DNS Challenge Provider

The `gandi` DNS challenge provider can be used to perform DNS challenges for the `acme_certificate` (/docs/providers/acme/r/certificate.html) resource with Gandi (https://www.gandi.net/en).

> **NOTE:** This provider is for the Gandi V4 API. For the V5 API and higher (aka LiveDNS), use the `gandiv5` (/docs/providers/acme/dns_providers/gandiv5.html) provider.

For complete information on how to use this provider with the `acme_certifiate` resource, see here (/docs/providers/acme/r/certificate.html#using-dns-challenges).

Example

```
resource "acme_certificate" "certificate" {
  ...

  dns_challenge {
    provider = "gandi"
  }
}
```

Argument Reference

The following arguments can be either passed as environment variables, or directly through the `config` block in the `dns_challenge` (/docs/providers/acme/r/certificate.html#dns_challenge) argument in the `acme_certificate` (/docs/providers/acme/r/certificate.html) resource. For more details, see here (/docs/providers/acme/r/certificate.html#using-dns-challenges).

- `GANDI_API_KEY` - The API key to use.

Gandi LiveDNS Challenge Provider

The `gandiv5` DNS challenge provider can be used to perform DNS challenges for the `acme_certificate` (/docs/providers/acme/r/certificate.html) resource with Gandi LiveDNS (https://doc.livedns.gandi.net/).

NOTE: For the legacy Gandi DNS service, use the use the `gandi` (/docs/providers/acme/dns_providers/gandi.html) provider.

For complete information on how to use this provider with the `acme_certifiate` resource, see here (/docs/providers/acme/r/certificate.html#using-dns-challenges).

Example

```
resource "acme_certificate" "certificate" {
  ...

  dns_challenge {
    provider = "gandiv5"
  }
}
```

Argument Reference

The following arguments can be either passed as environment variables, or directly through the `config` block in the `dns_challenge` (/docs/providers/acme/r/certificate.html#dns_challenge) argument in the `acme_certificate` (/docs/providers/acme/r/certificate.html) resource. For more details, see here (/docs/providers/acme/r/certificate.html#using-dns-challenges).

- `GANDIV5_API_KEY` - The API key to use.

Google Cloud DNS DNS Challenge Provider

The `gcloud` DNS challenge provider can be used to perform DNS challenges for the `acme_certificate` (/docs/providers/acme/r/certificate.html) resource with Google Cloud DNS (https://cloud.google.com/dns/docs/).

For complete information on how to use this provider with the `acme_certifiate` resource, see here (/docs/providers/acme/r/certificate.html#using-dns-challenges).

Example

```
resource "acme_certificate" "certificate" {
  ...

  dns_challenge {
    provider = "gcloud"
  }
}
```

Argument Reference

The following arguments can be either passed as environment variables, or directly through the `config` block in the `dns_challenge` (/docs/providers/acme/r/certificate.html#dns_challenge) argument in the `acme_certificate` (/docs/providers/acme/r/certificate.html) resource. For more details, see here (/docs/providers/acme/r/certificate.html#using-dns-challenges).

- `GCE_PROJECT` - The project name.
- `GCE_SERVICE_ACCOUNT_FILE` - The path to the service account file. This is the same file referenced by the `credentials` (/docs/providers/google/index.html#credentials) option in the Terraform Google provider (/docs/providers/google/index.html).

GleSYS DNS Challenge Provider

The glesys DNS challenge provider can be used to perform DNS challenges for the acme_certificate (/docs/providers/acme/r/certificate.html) resource with GleSYS (https://glesys.com/).

For complete information on how to use this provider with the acme_certifiate resource, see here (/docs/providers/acme/r/certificate.html#using-dns-challenges).

Example

```
resource "acme_certificate" "certificate" {
  ...

  dns_challenge {
    provider = "glesys"
  }
}
```

Argument Reference

The following arguments can be either passed as environment variables, or directly through the config block in the dns_challenge (/docs/providers/acme/r/certificate.html#dns_challenge) argument in the acme_certificate (/docs/providers/acme/r/certificate.html) resource. For more details, see here (/docs/providers/acme/r/certificate.html#using-dns-challenges).

- GLESYS_API_USER - The API user to use.
- GLESYS_API_KEY - The API key to use.

GoDaddy DNS Challenge Provider

The `godaddy` DNS challenge provider can be used to perform DNS challenges for the `acme_certificate` (/docs/providers/acme/r/certificate.html) resource with GoDaddy (https://godaddy.com/).

For complete information on how to use this provider with the `acme_certifiate` resource, see here (/docs/providers/acme/r/certificate.html#using-dns-challenges).

Example

```
resource "acme_certificate" "certificate" {
  ...

  dns_challenge {
    provider = "godaddy"
  }
}
```

Argument Reference

The following arguments can be either passed as environment variables, or directly through the `config` block in the `dns_challenge` (/docs/providers/acme/r/certificate.html#dns_challenge) argument in the `acme_certificate` (/docs/providers/acme/r/certificate.html) resource. For more details, see here (/docs/providers/acme/r/certificate.html#using-dns-challenges).

- `GODADDY_API_KEY` - The API key to use.
- `GODADDY_API_SECRET` - The API secret to use.

Amazon Lightsail DNS Challenge Provider

The `lightsail` DNS challenge provider can be used to perform DNS challenges for the `acme_certificate` (/docs/providers/acme/r/certificate.html) resource with Amazon Lightsail (https://aws.amazon.com/lightsail/).

For complete information on how to use this provider with the `acme_certifiate` resource, see here (/docs/providers/acme/r/certificate.html#using-dns-challenges).

Example

```
resource "acme_certificate" "certificate" {
  ...

  dns_challenge {
    provider = "lightsail"
  }
}
```

Argument Reference

The following arguments can be either passed as environment variables, or directly through the `config` block in the `dns_challenge` (/docs/providers/acme/r/certificate.html#dns_challenge) argument in the `acme_certificate` (/docs/providers/acme/r/certificate.html) resource. For more details, see here (/docs/providers/acme/r/certificate.html#using-dns-challenges).

> **NOTE:** Several other options exist for configuring the AWS credential chain. For more details, see the AWS SDK documentation (https://docs.aws.amazon.com/sdk-for-go/v1/developer-guide/configuring-sdk.html).

- `AWS_ACCESS_KEY_ID` - The AWS access key ID.
- `AWS_SECRET_ACCESS_KEY` - The AWS secret access key.
- `AWS_SESSION_TOKEN` - The session token to use, if necessary.
- `DNS_ZONE` - The hosted zone ID to use.

Linode DNS Challenge Provider

The `linode` DNS challenge provider can be used to perform DNS challenges for the `acme_certificate` (/docs/providers/acme/r/certificate.html) resource with Linode (https://www.linode.com/).

For complete information on how to use this provider with the `acme_certifiate` resource, see here (/docs/providers/acme/r/certificate.html#using-dns-challenges).

Example

```
resource "acme_certificate" "certificate" {
  ...

  dns_challenge {
    provider = "linode"
  }
}
```

Argument Reference

The following arguments can be either passed as environment variables, or directly through the `config` block in the `dns_challenge` (/docs/providers/acme/r/certificate.html#dns_challenge) argument in the `acme_certificate` (/docs/providers/acme/r/certificate.html) resource. For more details, see here (/docs/providers/acme/r/certificate.html#using-dns-challenges).

- `LINODE_API_KEY` - The API key to use.

Namecheap DNS Challenge Provider

The `namecheap` DNS challenge provider can be used to perform DNS challenges for the `acme_certificate` (/docs/providers/acme/r/certificate.html) resource with Namecheap (https://www.namecheap.com/).

For complete information on how to use this provider with the `acme_certifiate` resource, see here (/docs/providers/acme/r/certificate.html#using-dns-challenges).

Example

```
resource "acme_certificate" "certificate" {
  ...

  dns_challenge {
    provider = "namecheap"
  }
}
```

Argument Reference

The following arguments can be either passed as environment variables, or directly through the `config` block in the `dns_challenge` (/docs/providers/acme/r/certificate.html#dns_challenge) argument in the `acme_certificate` (/docs/providers/acme/r/certificate.html) resource. For more details, see here (/docs/providers/acme/r/certificate.html#using-dns-challenges).

- `NAMECHEAP_API_USER` - The API user to use.
- `NAMECHEAP_API_KEY` - The API key to use.

Name.com DNS Challenge Provider

The `namedotcom` DNS challenge provider can be used to perform DNS challenges for the `acme_certificate` (/docs/providers/acme/r/certificate.html) resource with Name.com (https://www.name.com/).

For complete information on how to use this provider with the `acme_certifiate` resource, see here (/docs/providers/acme/r/certificate.html#using-dns-challenges).

Example

```
resource "acme_certificate" "certificate" {
  ...

  dns_challenge {
    provider = "namedotcom"
  }
}
```

Argument Reference

The following arguments can be either passed as environment variables, or directly through the `config` block in the `dns_challenge` (/docs/providers/acme/r/certificate.html#dns_challenge) argument in the `acme_certificate` (/docs/providers/acme/r/certificate.html) resource. For more details, see here (/docs/providers/acme/r/certificate.html#using-dns-challenges).

- `NAMECOM_USERNAME` - The user name to use.
- `NAMECOM_API_TOKEN` - The API token to use.

NS1 DNS Challenge Provider

The `ns1` DNS challenge provider can be used to perform DNS challenges for the `acme_certificate` (/docs/providers/acme/r/certificate.html) resource with NS1 (https://ns1.com/).

For complete information on how to use this provider with the `acme_certifiate` resource, see here (/docs/providers/acme/r/certificate.html#using-dns-challenges).

Example

```
resource "acme_certificate" "certificate" {
  ...

  dns_challenge {
    provider = "ns1"
  }
}
```

Argument Reference

The following arguments can be either passed as environment variables, or directly through the `config` block in the `dns_challenge` (/docs/providers/acme/r/certificate.html#dns_challenge) argument in the `acme_certificate` (/docs/providers/acme/r/certificate.html) resource. For more details, see here (/docs/providers/acme/r/certificate.html#using-dns-challenges).

- `NS1_API_KEY` - The API key to use.

Open Telekom Cloud DNS Challenge Provider

The `otc` DNS challenge provider can be used to perform DNS challenges for the `acme_certificate` (/docs/providers/acme/r/certificate.html) resource with Open Telekom Cloud (https://cloud.telekom.de/).

For complete information on how to use this provider with the `acme_certifiate` resource, see here (/docs/providers/acme/r/certificate.html#using-dns-challenges).

Example

```
resource "acme_certificate" "certificate" {
  ...

  dns_challenge {
    provider = "otc"
  }
}
```

Argument Reference

The following arguments can be either passed as environment variables, or directly through the `config` block in the `dns_challenge` (/docs/providers/acme/r/certificate.html#dns_challenge) argument in the `acme_certificate` (/docs/providers/acme/r/certificate.html) resource. For more details, see here (/docs/providers/acme/r/certificate.html#using-dns-challenges).

- `OTC_USER_NAME` - The user name to use.
- `OTC_DOMAIN_NAME` - The domain name to use.
- `OTC_PASSWORD` - The password for the supplied user.
- `OTC_PROJECT_NAME` - The project name.
- `OTC_IDENTITY_ENDPOINT` - The identity endpoint to use.

OVH DNS Challenge Provider

The `ovh` DNS challenge provider can be used to perform DNS challenges for the `acme_certificate` (/docs/providers/acme/r/certificate.html) resource with OVH (https://www.ovh.com/).

For complete information on how to use this provider with the `acme_certifiate` resource, see here (/docs/providers/acme/r/certificate.html#using-dns-challenges).

Example

```
resource "acme_certificate" "certificate" {
  ...

  dns_challenge {
    provider = "ovh"
  }
}
```

Argument Reference

The following arguments can be either passed as environment variables, or directly through the `config` block in the `dns_challenge` (/docs/providers/acme/r/certificate.html#dns_challenge) argument in the `acme_certificate` (/docs/providers/acme/r/certificate.html) resource. For more details, see here (/docs/providers/acme/r/certificate.html#using-dns-challenges).

- `OVH_ENDPOINT` - The API endpoint to use. Can be one of `ovh-eu` or `ovh-ca`.
- `OVH_APPLICATION_KEY` - The application key to use.
- `OVH_APPLICATION_SECRET` - The application secret to use.
- `OVH_CONSUMER_KEY` - The consumer key to use.

PowerDNS DNS Challenge Provider

The `powerdns` DNS challenge provider can be used to perform DNS challenges for the `acme_certificate` (/docs/providers/acme/r/certificate.html) resource with a PowerDNS (https://www.powerdns.com/) name server.

For complete information on how to use this provider with the `acme_certifiate` resource, see here (/docs/providers/acme/r/certificate.html#using-dns-challenges).

Example

```
resource "acme_certificate" "certificate" {
  ...

  dns_challenge {
    provider = "powerdns"
  }
}
```

Argument Reference

The following arguments can be either passed as environment variables, or directly through the `config` block in the `dns_challenge` (/docs/providers/acme/r/certificate.html#dns_challenge) argument in the `acme_certificate` (/docs/providers/acme/r/certificate.html) resource. For more details, see here (/docs/providers/acme/r/certificate.html#using-dns-challenges).

- `PDNS_API_URL` - The API URL to use.
- `PDNS_API_KEY` - The API key to use.

Rackspace DNS Challenge Provider

The `rackspace` DNS challenge provider can be used to perform DNS challenges for the `acme_certificate` (/docs/providers/acme/r/certificate.html) resource with Rackspace (https://www.rackspace.com/cloud/dns).

For complete information on how to use this provider with the `acme_certifiate` resource, see here (/docs/providers/acme/r/certificate.html#using-dns-challenges).

Example

```
resource "acme_certificate" "certificate" {
  ...

  dns_challenge {
    provider = "rackspace"
  }
}
```

Argument Reference

The following arguments can be either passed as environment variables, or directly through the `config` block in the `dns_challenge` (/docs/providers/acme/r/certificate.html#dns_challenge) argument in the `acme_certificate` (/docs/providers/acme/r/certificate.html) resource. For more details, see here (/docs/providers/acme/r/certificate.html#using-dns-challenges).

- `RACKSPACE_USER` - The user to use.
- `RACKSPACE_API_KEY` - The API key to use.

RFC 2136 DNS Challenge Provider

The `rfc2136` DNS challenge provider can be used to perform DNS challenges for the `acme_certificate` (/docs/providers/acme/r/certificate.html) resource with an RFC 2136 (https://tools.ietf.org/html/rfc2136)-compatible DNS server.

For complete information on how to use this provider with the `acme_certifiate` resource, see here (/docs/providers/acme/r/certificate.html#using-dns-challenges).

Example

```
resource "acme_certificate" "certificate" {
  ...

  dns_challenge {
    provider = "rfc2136"
  }
}
```

Argument Reference

The following arguments can be either passed as environment variables, or directly through the `config` block in the `dns_challenge` (/docs/providers/acme/r/certificate.html#dns_challenge) argument in the `acme_certificate` (/docs/providers/acme/r/certificate.html) resource. For more details, see here (/docs/providers/acme/r/certificate.html#using-dns-challenges).

> To disable TSIG authentication, leave the specific TSIG variables unset.

- `RFC2136_NAMESERVER` - The network address of the DNS server to send the updates to. Can be in the form of `HOST` or `HOST:PORT`.

- `RFC2136_TSIG_ALGORITHM` - The TSIG algorithm to use. Can be one of `hmac-md5.sig-alg.reg.int.` (HMAC-MD5), `hmac-sha1.` (HMAC-SHA1), `hmac-sha256.` (HMAC-SHA256), or `hmac-sha512.` (HMAC-SHA512). Default: `hmac-md5.sig-alg.reg.int.`

- `RFC2136_TSIG_KEY` - The TSIG secret key name.

- `RFC2136_TSIG_SECRET` - The TSIG secret key payload.

- `RFC2136_TIMEOUT` - The DNS propagation timeout.

Amazon Route 53 DNS Challenge Provider

The `route53` DNS challenge provider can be used to perform DNS challenges for the `acme_certificate` (/docs/providers/acme/r/certificate.html) resource with Amazon Route 53 (https://route53.microsoft.com/en-ca/).

For complete information on how to use this provider with the `acme_certifiate` resource, see here (/docs/providers/acme/r/certificate.html#using-dns-challenges).

Example

```
resource "acme_certificate" "certificate" {
  ...

  dns_challenge {
    provider = "route53"
  }
}
```

Argument Reference

The following arguments can be either passed as environment variables, or directly through the `config` block in the `dns_challenge` (/docs/providers/acme/r/certificate.html#dns_challenge) argument in the `acme_certificate` (/docs/providers/acme/r/certificate.html) resource. For more details, see here (/docs/providers/acme/r/certificate.html#using-dns-challenges).

> **NOTE:** Several other options exist for configuring the AWS credential chain. For more details, see the AWS SDK documentation (https://docs.aws.amazon.com/sdk-for-go/v1/developer-guide/configuring-sdk.html).

- `AWS_ACCESS_KEY_ID` - The AWS access key ID.
- `AWS_SECRET_ACCESS_KEY` - The AWS secret access key.
- `AWS_SESSION_TOKEN` - The session token to use, if necessary.
- `AWS_HOSTED_ZONE_ID` - The hosted zone ID to use. This can be used to override ACME's default domain discovery and force the provider to use a specific hosted zone.

Vultr DNS Challenge Provider

The `vultr` DNS challenge provider can be used to perform DNS challenges for the `acme_certificate` (/docs/providers/acme/r/certificate.html) resource with Vultr (https://www.vultr.com/).

For complete information on how to use this provider with the `acme_certifiate` resource, see here (/docs/providers/acme/r/certificate.html#using-dns-challenges).

Example

```
resource "acme_certificate" "certificate" {
  ...

  dns_challenge {
    provider = "vultr"
  }
}
```

Argument Reference

The following arguments can be either passed as environment variables, or directly through the `config` block in the `dns_challenge` (/docs/providers/acme/r/certificate.html#dns_challenge) argument in the `acme_certificate` (/docs/providers/acme/r/certificate.html) resource. For more details, see here (/docs/providers/acme/r/certificate.html#using-dns-challenges).

- `VULTR_API_KEY` - The API key to use.

acme_certificate

The `acme_certificate` resource can be used to create and manage an ACME TLS certificate.

> **NOTE:** As the usage model of Terraform generally sees it as being run on a different server than a certificate would normally be placed on, the `acme_certifiate` resource only supports DNS challenges.

Example

The below example is the same example that can be found on the index page (/docs/providers/acme/index.html), and creates both an account and certificate within the same configuration. The account is created using the `acme_registration` (/docs/providers/acme/r/registration.html) resource.

> **NOTE:** When creating accounts and certificates within the same configuration, ensure that you reference the `account_key_pem` (/docs/providers/acme/r/registration.html#account_key_pem) argument in the `acme_registration` (/docs/providers/acme/r/registration.html) resource as the corresponding `account_key_pem` argument in the `acme_certificate` resource. This will ensure that the account gets created before the certificate and avoid errors.

```
provider "acme" {
  server_url = "https://acme-staging-v02.api.letsencrypt.org/directory"
}

resource "tls_private_key" "private_key" {
  algorithm = "RSA"
}

resource "acme_registration" "reg" {
  account_key_pem = "${tls_private_key.private_key.private_key_pem}"
  email_address   = "nobody@example.com"
}

resource "acme_certificate" "certificate" {
  account_key_pem           = "${acme_registration.reg.account_key_pem}"
  common_name               = "www.example.com"
  subject_alternative_names = ["www2.example.com"]

  dns_challenge {
    provider = "route53"
  }
}
```

Using an external CSR

The `acme_certificate` resource can also take an external CSR. In this example, we create one using `tls_cert_request` (/docs/providers/tls/r/cert_request.html) first, before supplying it to the `certificate_request_pem` argument.

> **NOTE:** Some current ACME CA implementations (including Let's Encrypt) strip most of the organization information out of a certificate request subject. You may wish to confirm with the CA what behavior to expect when using the `certificate_request_pem` argument with this resource.

NOTE: It is not a good practice to use the same private key for both your account and your certificate. Make sure you use different keys.

```
provider "acme" {
  server_url = "https://acme-staging-v02.api.letsencrypt.org/directory"
}

resource "tls_private_key" "reg_private_key" {
  algorithm = "RSA"
}

resource "acme_registration" "reg" {
  account_key_pem = "${tls_private_key.reg_private_key.private_key_pem}"
  email_address   = "nobody@example.com"
}

resource "tls_private_key" "cert_private_key" {
  algorithm = "RSA"
}

resource "tls_cert_request" "req" {
  key_algorithm   = "RSA"
  private_key_pem = "${tls_private_key.cert_private_key.private_key_pem}"
  dns_names       = ["www.example.com", "www2.example.com"]

  subject {
    common_name = "www.example.com"
  }
}

resource "acme_certificate" "certificate" {
  account_key_pem         = "${acme_registration.reg.account_key_pem}"
  certificate_request_pem = "${tls_cert_request.req.cert_request_pem}"

  dns_challenge {
    provider = "route53"
  }
}
```

Argument Reference

The resource takes the following arguments:

NOTE: All arguments in `acme_certificate`, other than `min_days_remaining`, force a new resource when changed.

- `account_key_pem` (Required) - The private key of the account that is requesting the certificate.
- `common_name` - The certificate's common name, the primary domain that the certificate will be recognized for. Required when not specifying a CSR.
- `subject_alternative_names` - The certificate's subject alternative names, domains that this certificate will also be recognized for. Only valid when not specifying a CSR.
- `key_type` - The key type for the certificate's private key. Can be one of: `P256` and `P384` (for ECDSA keys of respective

length) or 2048, 4096, and 8192 (for RSA keys of respective length). Required when not specifying a CSR. The default is 2048 (RSA key of 2048 bits).

- certificate_request_pem - A pre-created certificate request, such as one from tls_cert_request (/docs/providers/tls/r/cert_request.html), or one from an external source, in PEM format. Either this, or the in-resource request options (common_name, key_type, and optionally subject_alternative_names) need to be specified.
- dns_challenge (Required) - The DNS challenge to use in fulfilling the request.
- must_staple (Optional) Enables the OCSP Stapling Required (https://letsencrypt.org/docs/integration-guide/#implement-ocsp-stapling) TLS Security Policy extension. Certificates with this extension must include a valid OCSP Staple in the TLS handshake for the connection to succeed. Defaults to false. Note that this option has no effect when using an external CSR - it must be enabled in the CSR itself.

> **NOTE:** OCSP stapling requires specific webserver configuration to support the downloading of the staple from the CA's OCSP endpoints, and should be configured to tolerate prolonged outages of the OCSP service. Consider this when using must_staple, and only enable it if you are sure your webserver or service provider can be configured correctly.

- min_days_remaining (Optional) - The minimum amount of days remaining on the expiration of a certificate before a renewal is attempted. The default is 7. A value of less than 0 means that the certificate will never be renewed.

Using DNS challenges

As the usage model of Terraform generally sees it as being run on a different server than a certificate would normally be placed on, the acme_certifiate resource only supports DNS challenges. This method authenticates certificate domains by requiring the requester to place a TXT record on the FQDNs in the certificate.

The ACME provider responds to DNS challenges automatically by utilizing one of the supported DNS challenge providers. Most providers take credentials as environment variables, but if you would rather use configuration for this purpose, you can by specifying config blocks within a dns_challenge block, along with the provider parameter.

For a full list of providers, click here (/docs/providers/acme/dns_providers/index.html).

Example with the Route 53 provider (/docs/providers/acme/dns_providers/route53.html):

```
resource "acme_certificate" "certificate" {
  ...

  dns_challenge {
    provider = "route53"

    config {
      AWS_ACCESS_KEY_ID     = "${var.aws_access_key}"
      AWS_SECRET_ACCESS_KEY = "${var.aws_secret_key}"
      AWS_DEFAULT_REGION    = "us-east-1"
    }
  }

  ...
}
```

Relation to Terraform provider configuration

The DNS provider configuration specified in the `acme_certificate` resource is separate from any that you supply in a corresponding provider whose functionality overlaps with the certificate's DNS providers. This ensures that there are no hard dependencies between any of these providers and the ACME provider, but it is important to note so that configuration is supplied correctly.

As an example, if you specify manual configuration for the AWS provider (/docs/providers/aws/index.html) via the `provider` (/docs/configuration/providers.html) block instead of the environment, you will still need to supply the configuration explicitly as per above.

Some of these providers have environment variable settings that overlap with the ones found here, generally depending on whether or not these variables are supported by the corresponding provider's SDK.

Check the DNS provider page (/docs/providers/acme/dns_providers/index.html) of a specific provider for more details on exactly what variables are supported.

Certificate renewal

The `acme_certificate` resource handles automatic certificate renewal so long as a plan or apply is done within the number of days specified in the `min_days_remaining` resource parameter. During refresh, if Terraform detects that the certificate is within the expiry range specified in `min_days_remaining`, or is already expired, Terraform will mark the certificate to be renewed on the next apply.

Note that a value less than `0` supplied to `min_days_remaining` will cause renewal checks to be bypassed, and the certificate will never renew.

Attribute Reference

The following attributes are exported:

- `id` - The full URL of the certificate within the ACME CA.
- `certificate_url` - The full URL of the certificate within the ACME CA. Same as `id`.
- `certificate_domain` - The common name of the certificate.
- `account_ref` - The URI of the account for this certificate.
- `private_key_pem` - The certificate's private key, in PEM format, if the certificate was generated from scratch and not with `certificate_request_pem`. If `certificate_request_pem` was used, this will be blank.
- `certificate_pem` - The certificate in PEM format.
- `issuer_pem` - The intermediate certificate of the issuer.

acme_registration

The `acme_registration` resource can be used to create and manage accounts on an ACME server. Once registered, the same private key that has been used for registration can be used to request authorizations for certificates.

> This resource is named `acme_registration` for historical reasons - in the ACME v1 spec, a *registration* referred to the account entity. This resource name is stable and more than likely will not change until a later major version of the provider, if at all.

Example

The following creates an account off of a private key generated with the `tls_private_key` (/docs/providers/tls/r/private_key.html) resource.

```
provider "acme" {
  server_url = "https://acme-staging-v02.api.letsencrypt.org/directory"
}

resource "tls_private_key" "private_key" {
  algorithm = "RSA"
}

resource "acme_registration" "reg" {
  account_key_pem = "${tls_private_key.private_key.private_key_pem}"
  email_address   = "nobody@example.com"
}
```

Argument Reference

> **NOTE:** All arguments in `acme_registration` force a new resource if changed.

The resource takes the following arguments:

- `account_key_pem` (Required) - The private key used to identity the account.
- `email_address` (Required) - The contact email address for the account.

Attribute Reference

The following attributes are exported:

- `id`: The original full URL of the account.
- `registration_url`: The current full URL of the account.

> `id` and `registration_url` will usually be the same and will usually only diverge when migrating protocols, ie: ACME v1 to v2.