Venafi Provider

Venafi (https://www.venafi.com) is the enterprise platform for Machine Identity Protection. The Venafi provider streamlines the process of acquiring SSL/TLS keys and certificates from Venafi services giving assurance of compliance with Information Security policies. It provides resources that allow private keys and certficates to be created as part of a Terraform deployment.

Use the navigation to the left to read about the available resources.

Example Usage for Venafi Cloud

You can sign up for a Venafi Cloud account by visiting https://www.venafi.com/platform/cloud/devops (https://www.venafi.com/platform/cloud/devops). Once registered, find your API key by clicking your name in the top right of the web interface. You will also need to specify the ID of a `zone` to use when requesting certificates. Zones are the part of a Venafi Cloud project that define the machine identity policy applied to certificate requests and the certificate authority that will issue certificates.

```
# Configure the Venafi provider
provider "venafi" {
    api_key = "xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx"
    zone    = "zzzzzzzz-zzzz-zzzz-zzzz-zzzzzzzzzzzz"
}

# Generate a key pair and request a certificate
resource "venafi_certificate" "webserver" {
    # ...
}
```

Example Usage for Venafi Trust Protection Platform

Your Venafi administrator can provide you with the URL for the Trust Protection Platform REST API and grant you permission to use it. At the same time they'll provide you with the Distinguished Name of a policy folder to specify for the `zone`. Policy folders define the machine identity policy applied to certificate requests and the certificate authority that will issue certificates. You may also need to ask them for a root CA certificate for your `trust_bundle` if the Venafi Platform URL is secured by a certificate your Terraform computer does not already trust.

```
# Configure the Venafi provider
provider "venafi" {
    url          = "https://tpp.venafi.example:443/vedsdk"
    trust_bundle = "${file("/opt/venafi/bundle.pem")}"
    tpp_username = "local:terraform"
    tpp_password = "password"
    zone         = "DevOps\\Terraform"
}

# Generate a key pair and request a certificate
resource "venafi_certificate" "webserver" {
    # ...
}
```

Argument Reference

The following arguments are supported:

- `zone` - (Required, string) Zone ID for Venafi Cloud or policy folder for Venafi Platform.
- `url` - (Optional, string) Venafi URL (e.g. "https://tpp.venafi.example:443/vedsdk" (https://tpp.venafi.example:443/vedsdk%22)).
- `tpp_username` - (Optional, string) WebSDK account username for authentication (applies only to Venafi Platform).
- `tpp_password` - (Optional, string) WebSDK account password for authentication (applies only to Venafi Platform).
- `api_key` - (Optional, string) REST API key for authentication (applies only to Venafi Cloud).
- `trust_bundle` - (Optional, string) PEM trust bundle for Venafi Platform server certificate (e.g. "${file("bundle.pem")}").
- `dev_mode` - (Optional, boolean) When "true" will test the provider without connecting to Venafi Platform or Venafi Cloud.

Environment Variables

The following environment variables can also be used to specify provider argument values:

- VENAFI_ZONE
- VENAFI_URL
- VENAFI_USER
- VENAFI_PASS
- VENAFI_API
- VENAFI_DEVMODE

venafi_certificate

Provides access to TLS key and certificate data enrolled using Venafi. This can be used to define a certificate.

Example Usage

```
resource "venafi_certificate" "webserver" {
    common_name = "web.venafi.example"
    algorithm = "RSA"
    rsa_bits = "2048"
    san_dns = [
        "web01.venafi.example",
        "web02.venafi.example"
    ]
    key_password = "${var.pk_pass}"
}
```

Argument Reference

The following arguments are supported:

- `common_name` - (Required, string) The common name of the certificate.
- `algorithm` - (Optional, string) Key encryption algorithm, either RSA or ECDSA. Defaults to "RSA".
- `rsa_bits` - (Optional, integer) Number of bits to use when generating an RSA key. Applies when algorithm=RSA. Defaults to 2048.
- `ecsa_curve` - (Optional, string) Elliptic curve to use when generating an ECDSA key pair. Applies when algorithm=ECDSA. Defaults to "P521".
- `san_dns` - (Optional, set of strings) List of DNS names to use as alternative subjects of the certificate.
- `san_email` - (Optional, set of strings) List of email addresses to use as alternative subjects of the certificate.
- `san_ip` - (Optional, set of strings) List of IP addresses to use as alternative subjects of the certificate.
- `key_password` - (Optional, string) The password used to encrypt the private key.
- `expiration_window` - (Optional, integer) Number of hours before certificate expiry to request a new certificate.

Attributes Reference

The following attributes are exported:

- `private_key_pem` - The private key in PEM format.
- `chain` - The trust chain of X509 certificate authority certificates in PEM format concatenated together.

- `certificate` - The X509 certificate in PEM format.