+++ title = "Vault" description = "" keywords = ["grafana", "vault", "configuration"] weight = 1200 +++

Vault integration

> *Only available in Grafana Enterprise v7.1+.*

If you manage your secrets with Hashicorp Vault, you can use them for [Configuration]({{< relref "../administration/configuration.md" >}}) and [Provisioning]({{< relref "../administration/provisioning.md" >}}).

> ***Note:*** *If you have Grafana [set up for high availability]({{< relref "../administration/set-up-for-high-availability.md" >}}), then we advise not to use dynamic secrets for provisioning files. Each Grafana instance is responsible for renewing its own leases. Your data source leases might expire when one of your Grafana servers shuts down.*

Configuration

Before using Vault, you need to activate it by providing a URL, authentication method (currently only token), and a token for your Vault service. Grafana automatically renews the service token if it is renewable and set up with a limited lifetime.

If you're using short-lived leases, then you can also configure how often Grafana should renew the lease and for how long. We recommend keeping the defaults unless you run into problems.

```
[keystore.vault]
# Location of the Vault server
;url =
# Vault namespace if using Vault with multi-tenancy
;namespace =
# Method for authenticating towards Vault. Vault is inactive if this option is not set
# Possible values: token
;auth_method =
# Secret token to connect to Vault when auth_method is token
;token =
# Time between checking if there are any secrets which needs to be renewed.
;lease_renewal_interval = 5m
# Time until expiration for tokens which are renewed. Should have a value higher than lease_renewal_interval
;lease_renewal_expires_within = 15m
# New duration for renewed tokens. Vault may be configured to ignore this value and impose a stricter limit.
;lease_renewal_increment = 1h
```

Example for `vault server -dev`:

```
[keystore.vault]
url = http://127.0.0.1:8200 # HTTP should only be used for local testing
auth_method = token
token = s.sAZLyI0r7sFLMPq6MWtoOhAN # replace with your key
```

Using the Vault expander

After you configure Vault, you must set the configuration or provisioning files you wish to use Vault. Vault configuration is an extension of configuration's [variable expansion]({{< relref "../administration/configuration.md#variable-expansion" >}}) and follows the `$__vault{<argument>}` syntax.

The argument to Vault consists of three parts separated by a colon:

- The first part specifies which secrets engine should be used.
- The second part specifies which secret should be accessed.
- The third part specifies which field of that secret should be used.

For example, if you place a Key/Value secret for the Grafana admin user in *secret/grafana/admin_defaults* the syntax for accessing it's *password* field would be `$__vault{kv:secret/grafana/admin_defaults:password}`.

Secrets engines

Vault supports many secrets engines which represents different methods for storing or generating secrets when requested by an authorized user. Grafana supports a subset of these which are most likely to be relevant for a Grafana installation.

Key/Value

Grafana supports Vault's K/V version 2 storage engine which is used to store and retrieve arbitrary secrets as `kv`.

```
$__vault{kv:secret/grafana/smtp:username}
```

Databases

The Vault databases secrets engines is a family of secret engines which shares a similar syntax and grants the user dynamic access to a database. You can use this both for setting up Grafana's own database access and for provisioning data sources.

```
$__vault{database:database/creds/grafana:username}
```

Examples

The following examples show you how to set your [configuration]({{< relref "../administration/configuration.md" >}}) or [provisioning]({{< relref "../administration/provisioning.md" >}}) files to use Vault to retrieve configuration values.

Configuration

The following is a partial example for using Vault to set up a Grafana configuration file's email and database credentials. Refer to [Configuration]({{< relref "../administration/configuration.md" >}}) for more information.

```
[smtp]
enabled = true
host = $__vault{kv:secret/grafana/smtp:hostname}:587
user = $__vault{kv:secret/grafana/smtp:username}
password = $__vault{kv:secret/grafana/smtp:password}

[database]
type = mysql
```

```
host = mysqlhost:3306
name = grafana
user = $__vault{database:database/creds/grafana:username}
password = $__vault{database:database/creds/grafana:password}
```

Provisioning

The following is a full examples of a provisioning YAML file setting up a MySQL data source using Vault's database secrets engine. Refer to [Provisioning]({{< relref "../administration/provisioning.md" >}}) for more information.

provisioning/custom.yaml

```
apiVersion: 1

datasources:
  - name: statistics
    type: mysql
    url: localhost:3306
    database: stats
    user: $__vault{database:database/creds/ro/stats:username}
    secureJsonData:
      password: $__vault{database:database/creds/ro/stats:password}
```