+++ title = “Google Cloud KMS” description = “Using Google Cloud KMS to encrypt database secrets” keywords = [“grafana”, “Google Cloud KMS integration”] weight = 3 +++

Using Google Cloud KMS to encrypt database secrets

You can use an encryption key from Google Cloud Key Management Service to encrypt secrets in the Grafana database.

Prerequisites:

- A Google Cloud account with permission to list and create KMS keys and service accounts to access those keys
- Access to the Grafana [configuration]({{< relref “../../administration/configuration/#config-file-locations” >}}) file

1. Create a key ring in Google Cloud KMS.
2. Create a symmetric encryption key in the key ring.
3. Create a service account and assign it a role: it can be a predefined role or custom role with permissions to encrypt and decrypt secrets with Key Management Service.
4. Create a service account key and save its JSON file to you computer, for example, as `~/.config/gcloud/sample-project-credentials.json`.
5. From within Grafana, turn on [envelope encryption]({{< relref “../../administration/database-encryption.md” >}}).
6. Add your Google Cloud KMS details to the Grafana configuration file; depending on your operating system, is usually named `grafana.ini`: a. Add a new section to the configuration file, with a name in the format of `[security.encryption.azurekv.<KEY-NAME>]`, where `<KEY-NAME>` is any name that uniquely identifies this key among other provider keys. b. Fill in the section with the following values:

   - `key_id`: encryption key ID, refer to Getting the ID for a Key.
   - `credentials_file`: full path to service account key JSON file on your computer.

   An example of a Google Cloud KMS provider section in the `grafana.ini` file is as follows:

```
# Example of Google Cloud KMS provider setup
;[security.encryption.googlekms.example-encryption-key]
# Google Cloud KMS key ID
key_id = 1234abcd-12ab-34cd-56ef-1234567890ab
# Full path to a JSON file with a service account key
```

```
credentials_file = ~/.config/gcloud/sample-project-credentials.json
```

7. Update the `[security]` section of the `grafana.ini` configuration file with the new Encryption Provider key that you created:

   ```
   [security]
   # previous encryption key, used for legacy alerts, decrypting existing secrets or used
   secret_key = AaaaAaaa
   # encryption provider key in the format <PROVIDER>.<KEY-NAME>
   encryption_provider = googlekms.example-encryption-key
   # list of configured key providers, space separated
   available_encryption_providers = googlekms.example-encryption-key
   ```

   > **Note:** The encryption key stored in the `secret_key` field is still used by Grafana's legacy alerting system to encrypt secrets. Do not change or remove that value.

8. Restart Grafana.

9. (Optional) From the command line and the root directory of Grafana Enterprise, re-encrypt all of the secrets within the Grafana database with the new key using the following command:

   ```
   grafana-cli admin secrets-migration re-encrypt
   ```

   If you do not re-encrypt existing secrets, then they will remain encrypted by the previous encryption key. Users will still be able to access them.

   > **Note:** This process could take a few minutes to complete, depending on the number of secrets (such as data sources or alert notification channels) in your database. Users might experience errors while this process is running, and alert notifications might not be sent.

   > **Note:** If you are updating this encryption key during the initial setup of Grafana before any data sources, alert notification channels, or dashboards have been created, then this step is not necessary because there are no secrets in Grafana to migrate.