Connecting with Go

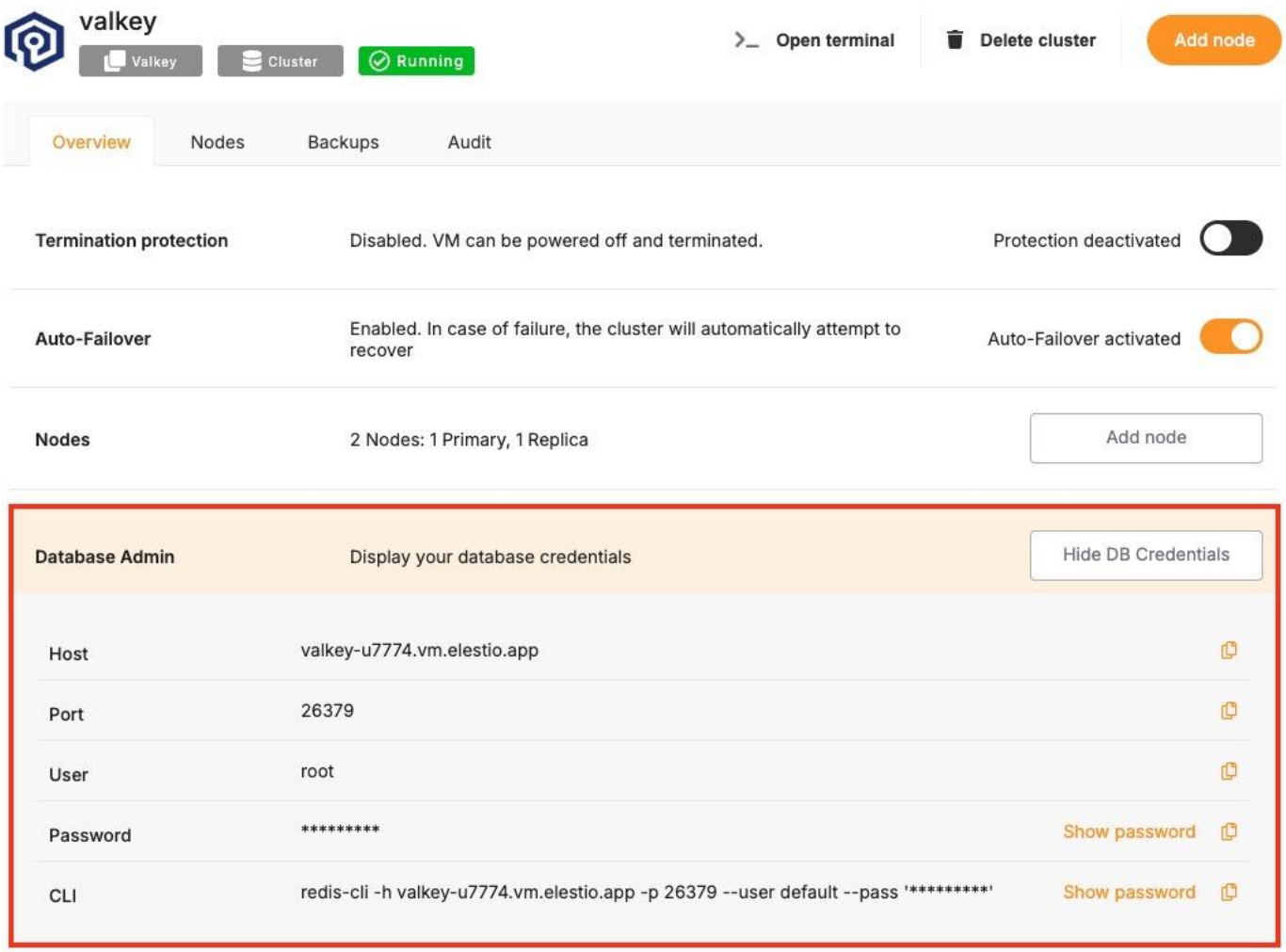
This guide explains how to establish a connection between a Go application and a Valkey database using the go-redis package. It walks through the necessary setup, configuration, and execution of a simple Valkey command.

Variables

Certain parameters must be provided to establish a successful connection to a Valkey database. Below is a breakdown of each required variable, its purpose, and where to find it. Here’s what each variable represents:

Variable	Description	Purpose
`HOST`	Valkey hostname, from the Elestio service overview page	The address of the server hosting your Valkey instance.
`PORT`	Port for Valkey connection, from the Elestio service overview page	The network port used to connect to Valkey. The default port is 6379.
`PASSWORD`	Valkey password, from the Elestio service overview page	The authentication key required to connect securely to Valkey.

These values can usually be found in the Elestio service overview details as shown in the image below, make sure to take a copy of these details and add it to the code moving ahead.



Prerequisites

Install Go

Check if Go is installed by running:

```
go version
```

If not installed, download it from golang.org and install.

Install the go-redis Package

The go-redis package enables Go applications to interact with Valkey. Install it using:

```
go get github.com/redis/go-redis/v9
```

Code

Once all prerequisites are set up, create a new file named `valkey.go` and add the following code:

```
package main

import (
	"context"
	"fmt"
	"time"

	"github.com/redis/go-redis/v9"
)

func main() {
	opt := &redis.Options{
		Addr:     "HOST:PORT",
		Password: "PASSWORD",
		DB:       0,
	}

	valkey := redis.NewClient(opt)
	ctx, cancel := context.WithTimeout(context.Background(), 5*time.Second)
	defer cancel()

	err := valkey.Set(ctx, "testKey", "Hello Valkey", 0).Err()
	if err != nil {
		fmt.Println("Valkey operation failed:", err)
		return
	}

	val, err := valkey.Get(ctx, "testKey").Result()
	if err != nil {
		fmt.Println("Valkey operation failed:", err)
		return
	}

	fmt.Println("Connected to Valkey")
	fmt.Println("Retrieved value:", val)
```

```
    if err := valkey.Close(); err != nil {
        fmt.Println("Error closing connection:", err)
    }
}
```

To execute the script, open the terminal or command prompt and navigate to the directory where `valkey.go` is located. Once in the correct directory, run the script with the command:

```
go run valkey.go
```

If the connection is successful, the terminal will display output similar to:

```
Connected to Valkey
Retrieved value: Hello Valkey
```

Revision #1
Created 2025-07-04 10:49:40 UTC
Updated 2025-07-04 10:51:33 UTC