Come generare le chiavi VAPID con Go

Le chiavi VAPID (Voluntary Application Server Identification for Web Push) sono utilizzate nel contesto delle notifiche push per autenticare il server di origine e consentire l'invio di messaggi push ai client. In questo articolo, vedremo come generare le chiavi VAPID utilizzando il linguaggio di programmazione Go.

Prerequisiti

1. Installazione di Go: Assicurati di avere Go installato sul tuo sistema. Puoi scaricarlo da golang.org.
2. Modulo `github.com/cespare/xxhash`: Utilizzeremo questo modulo per l'hashing.
3. Familiarità con i concetti di base delle notifiche push e la libreria `crypto/elliptic`.

Crea una nuova directory per il tuo progetto e inizializza un nuovo modulo Go:

```
mkdir vapid-keygen
cd vapid-keygen
go mod init vapid-keygen
```

Anche se per questo esempio non utilizzeremo molte dipendenze esterne, è buona norma installare quelle necessarie:

```
go get -u github.com/cespare/xxhash
```

Crea un file `main.go` e aggiungi il seguente codice:

```
package main

import (
    "crypto/elliptic"
    "crypto/ecdsa"
    "crypto/rand"
    "encoding/base64"
    "fmt"
    "log"
)

func generateVAPIDKeys() (string, string, error) {
    // Curva ellittica P-256
    curve := elliptic.P256()
    privateKey, err := ecdsa.GenerateKey(curve, rand.Reader)
    if err != nil {
        return "", "", err
    }

    // Estrazione della chiave pubblica
    publicKey := append(privateKey.PublicKey.X.Bytes(), privateKey.PublicKey.Y.Bytes()...)

    // Codifica in Base64 delle chiavi
```

```
    publicKeyBase64 := base64.RawURLEncoding.EncodeToString(publicKey)
    privateKeyBase64 := base64.RawURLEncoding.EncodeToString(privateKey.D.Bytes())

    return publicKeyBase64, privateKeyBase64, nil
}

func main() {
    pubKey, privKey, err := generateVAPIDKeys()
    if err != nil {
        log.Fatalf("Error generating VAPID keys: %v", err)
    }

    fmt.Printf("Public Key: %s\n", pubKey)
    fmt.Printf("Private Key: %s\n", privKey)
}
```

Questo codice genera una coppia di chiavi usando la curva ellittica P-256 e le codifica in formato base64 URL-safe.

Una volta generate le chiavi, possono essere utilizzate per autenticare le notifiche push. Di seguito è riportato un esempio di come utilizzare queste chiavi con la libreria `web-push` per inviare notifiche push:

```
package main

import (
    "github.com/SherClockHolmes/webpush-go"
    "log"
)

func main() {
    sub := &webpush.Subscription{
        Endpoint: "https://fcm.googleapis.com/fcm/send/...",
        Keys: webpush.Keys{
            Auth: "your_auth_key",
            P256dh: "your_p256dh_key",
        },
    }

    resp, err := webpush.SendNotification([]byte("Test message"), sub, &webpush.Options{
        VAPIDPublicKey:  "your_public_key",
        VAPIDPrivateKey: "your_private_key",
        VAPIDSubject:    "mailto:example@yourdomain.org",
    })
    if err != nil {
        log.Fatal(err)
    }
    defer resp.Body.Close()
}
```

Conclusione

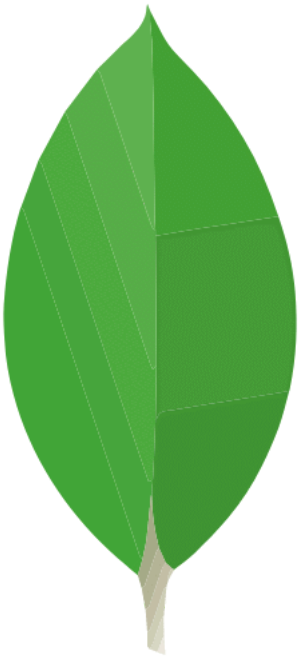
In questo articolo abbiamo visto come generare le chiavi VAPID utilizzando Go. Queste chiavi sono essenziali per autenticare le notifiche push e garantire la sicurezza della comunicazione tra il server e il client. Con le chiavi generate, puoi ora procedere a implementare la tua soluzione di notifiche push.

Applicazioni Correlate

-

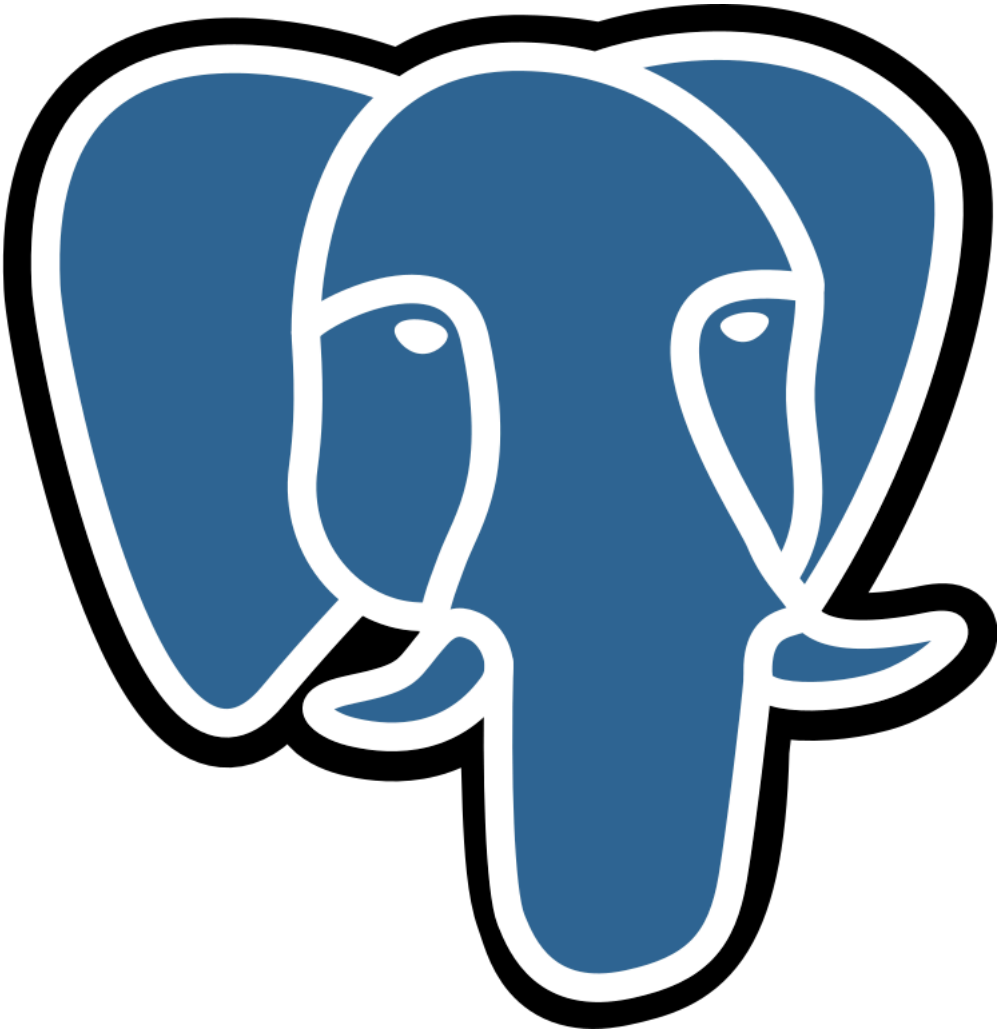
  Go Placeholder Image

  Applicazione in Go per la creazione di immagini segnaposto.
  DockerDocker ComposeGoJavaScript



Go MongoDB App

Applicazione basata su MongoDB ed implementata in Go con il driver ufficiale.
DockerDocker ComposeGoMongoDB



Go PostgreSQL App

Applicazione basata su PostgreSQL e sviluppata in Go.
DockerDocker ComposeGoPostgreSQL