

## Documentation

[Hummingbird Document...](#) / One Time Passwords

Article

# One Time Passwords

A one time password (OTP) valid for only one login session.



## Overview

OTPs avoid a number of shortcomings that are associated with traditional (static) password-based authentication. OTP generation algorithms typically make use of pseudo-randomness or randomness, making prediction of successor OTPs by an attacker difficult, and also cryptographic hash functions, which can be used to derive a value but are hard to reverse and therefore difficult for an attacker to obtain the data that was used for the hash. This is necessary because otherwise it would be easy to predict future OTPs by observing previous ones.

HummingbirdAuth provides support for both time based (TOTP) and counter based (HOTP) one time passwords.

## Usage

To setup one time password authentication you need a shared secret for each user. Store the shared secret with your user in a database. You can generate an authentication URL to supply to the user which includes a base32 encoded version of the shared secret.

```
// create shared secret
let sharedSecret = "random string"
// store shared secret in database alongside user
storeSecretWithUser(secret: sharedSecret)
// create TOTP and generate authentication URL
```

```
let totp = TOTP(secret: sharedSecret)
let authenticationURL = totp.createAuthenticatorURL(label: "MyURL")
```

Generally this is provided to the user via a QR Code. Most phones will automatically open up an Authenticator app to store the URL when they scan the QR Code.

## Authenticating

Compute the time based one time password as follows

```
let password = TOTP(secret: sharedSecret).compute()
```

Compare it with the password provided by the user to verify the user credentials.

---

## See Also

### Related Documentation



`struct TOTP`

A time based one time password (OTP)

`struct HOTP`

A counter based one time password (OTP)

## Authentication

-  **Authenticator Middleware**  
Request authentication middleware
-  **Sessions**  
Session based authentication

