Package 'otp'

May 5, 2020

Title One Time Password Generation and Verification

Version 0.1.0

Description Generating and validating One-time Password based on
Hash-based Message Authentication Code (HOTP)
and Time Based One-time Password (TOTP)
according to RFC 4226 <https://tools.ietf.org/html/rfc4226> and
RFC 6238 <https://tools.ietf.org/html/rfc6238>.

License MIT + file LICENSE

Encoding UTF-8

LazyData true

URL https://github.com/randy3k/otp

BugReports https://github.com/randy3k/otp/issues

Imports R6, base64url, openssl

RoxygenNote 7.1.0.9000

Suggests testthat (>= 2.1.0), covr

NeedsCompilation no

Author Randy Lai [aut, cre, cph]

Maintainer Randy Lai <randy.cs.lai@gmail.com>

Repository CRAN

Date/Publication 2020-05-05 15:30:09 UTC

R topics documented:

otp-package 2
HOTP 2
TOTP 3

Index **5**

otp-package *otp: One Time Password Generation and Verification*

Description

Generating and validating One-time Password based on Hash-based Message Authentication Code (HOTP) and Time Based One-time Password (TOTP) according to RFC 4226 <https://tools.ietf.org/html/rfc4226> and RFC 6238 <https://tools.ietf.org/html/rfc6238>.

Author(s)

Maintainer: Randy Lai <randy.cs.lai@gmail.com> [copyright holder]

See Also

Useful links:

- https://github.com/randy3k/otp
- Report bugs at https://github.com/randy3k/otp/issues

HOTP *HMAC based One Time Password (HOTP)*

Description

An R6 class that implements the HMAC based One Time Password (HOTP) algorithm.

Initialization

```
HOTP$new(secret, digits = 6L, algorithm = "sha1")
```

Create an One Time Password object

- **secret** a scalar character, the base32-based secret key.
- **digits** an integer, the number of digits of the password.
- **algorithm** the hash algorithm used, possible values are "sha1", "sha256" and "sha512".

Methods

```
HOTP$at(counter)
```

Generate an one time password at counter value.

- **counter** a non-negative integer.

```
HOTP$verify(code, counter, ahead = 0L)
```

Verify if a given one time password is valid. Returns the matching `counter` value if there is a match within the `ahead` window. Otherwise return `NULL`.

- **code** a string of digits.
- **counter** a non-negative integer.
- **ahead** a non-negative integer, the amount of counter ticks to look ahead.

```
HOTP$provisioning_uri(name, issuer = NULL, counter = 0L)
```

Return a provisioning uri which is compatible with google authenticator format.

- **name** account name.
- **issuer** issuer name.
- **counter** a non-negative integer, initial counter.

See Also

https://tools.ietf.org/html/rfc4226

Examples

```
p <- HOTP$new("JBSWY3DPEHPK3PXP")
p$at(8)

p$verify("964230", 8)
p$verify("964230", 7, ahead = 3)

p$provisioning_uri("Alice", issuer = "example.com", counter = 5)
```

`TOTP` *Time based One Time Password (TOTP)*

Description

An R6 class that implements the Time based One Time Password (TOTP) algorithm.

Initialization

```
TOTP$new(secret, digits = 6L, period = 30, algorithm = "sha1")
```

Create an One Time Password object

- **secret** a scalar character, the base32-based secret key.
- **digits** an integer, the number of digits of the password.
- **period** a positive number, the number of seconds in a time step.
- **algorithm** the hash algorithm used, possible values are "sha1", "sha256" and "sha512".

Methods

`TOTP$at_time(t)`

Generate an one time password at a given time value.

- **t** a POSIXct object or an integer that represents the numbers of second since UNIX epoch.

`HOTP$verify(code, t, behind = 0L)`

Verify if a given one time password is valid. Returns the beginning time of the time step window if there is a match within the `behind` window. Otherwise return `NULL`.

- **code** a string of digits.
- **t** a POSIXct object or an integer that represents the number of seconds since UNIX epoch.
- **behind** a non-negative integer, the amount of time steps to look behind. A value of `1` means to accept the code before `period` seconds ago.

`HOTP$provisioning_uri(name, issuer = NULL)`

Return a provisioning uri which is compatible with google authenticator format.

- **name** account name.
- **issuer** issuer name.

See Also

https://tools.ietf.org/html/rfc6238

Examples

```
p <- TOTP$new("JBSWY3DPEHPK3PXP")
(code <- p$now())
p$verify(code, behind = 1)

(current_time <- Sys.time())
(code <- p$at_time(current_time))
p$verify(code, current_time + 30, behind = 1)

p$provisioning_uri("Alice", issuer = "example.com")
```

Index

HOTP, 2

otp-package, 2

TOTP, 3