hmac

package | standard library

Version: go1.20.1 Latest | Published: Feb 14, 2023 | License: BSD-3-Clause | Imports: 3 | Imported by: 30,654

Details Valid go.mod file | Redistributable license | Tagged version | Stable version

Learn more

Repository cs.opensource.google/go/go

Links Report a Vulnerability

Documentation

Documentation

Overview

Package hmac implements the Keyed-Hash Message Authentication Code (HMAC) as defined in U.S. Federal Information Processing Standards Publication 198. An HMAC is a cryptographic hash that uses a key to sign a message. The receiver verifies the hash by recomputing it using the same key.

Receivers should be careful to use Equal to compare MACs in order to avoid timing side-channels:

```
// ValidMAC reports whether messageMAC is a valid HMAC tag for message.
func ValidMAC(message, messageMAC, key []byte) bool {
    mac := hmac.New(sha256.New, key)
    mac.Write(message)
    expectedMAC := mac.Sum(nil)
    return hmac.Equal(messageMAC, expectedMAC)
}
```

Index

Constants

This section is empty.

Variables

This section is empty.

Functions

func Equal

added in go1.1

```
func Equal(mac1, mac2 []byte) bool
```

Equal compares two MACs for equality without leaking timing information.

func New

```
func New(h func() hash.Hash, key []byte) hash.Hash
```

New returns a new HMAC hash using the given hash.Hash type and key. New functions like sha256.New from crypto/sha256 can be used as h. h must return a new Hash every time it is called. Note that unlike other hash implementations in the standard library, the returned Hash does not implement encoding.BinaryMarshaler or encoding.BinaryUnmarshaler.

Types

This section is empty.

Source Files

View all

hmac.go

GitHub

Slack

r/golang

Meetup

Golang Weekly

Report an Issue

Google