Package 'agecrypt'

August 5, 2026

Title File Encryption with the 'age' Format

Version 0.1.0

Description An idiomatic R interface to the 'age' file encryption format (<https://age-encryption.org/v1>), backed by a vendored copy of the 'agec' C implementation (<https://git.sr.ht/~min/agec>). Encrypt and decrypt raw vectors, files, and strings for one or more X25519 recipients or with a passphrase, with optional ASCII armor. Cryptography is vendored and randomness is drawn from the operating system, so the package has no external library dependencies.

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 8.0.0

Suggests askpass, openssl, testthat (>= 3.0.0), withr

Config/testthat/edition 3

URL https://github.com/pedrobtz/agecrypt

BugReports https://github.com/pedrobtz/agecrypt/issues

NeedsCompilation yes

Author Pedro Baltazar [aut, cre, cph],
agec authors [ctb, cph] (Authors of the vendored 'agec' C implementation, https://git.sr.ht/~min/agec)

Maintainer Pedro Baltazar <pedrobtz@gmail.com>

Repository CRAN

Date/Publication 2026-08-05 08:00:07 UTC

Contents

`age_file` *Encrypt and decrypt files*

Description

File-to-file encryption. Large files are streamed in constant memory in C (they are not loaded into an R vector).

Usage

```
age_encrypt_file(
  input,
  output = NULL,
  recipients,
  armor = FALSE,
  overwrite = FALSE
)

age_decrypt_file(input, output = NULL, identities, overwrite = FALSE)
```

Arguments

`input`	Path to the source file.
`output`	Destination path. If `NULL` (default), `age_encrypt_file()` appends `.age` to `input`; `age_decrypt_file()` strips a trailing `.age`. If `input` does not end in `.age`, `age_decrypt_file()` requires `output` to be given explicitly rather than guess.
`recipients`	Character vector of `"age1..."` recipient strings.
`armor`	If `TRUE`, produce ASCII-armored output.
`overwrite`	If `FALSE` (default), error when `output` already exists.
`identities`	An `age_identity`, or character input accepted by `age_identity()`.

Value

The output path, invisibly.

Examples

```
id <- age_keygen()
rec <- age_pubkey(id)
f <- tempfile(fileext = ".txt")
writeLines("hello", f)
enc <- age_encrypt_file(f, recipients = rec)
dec <- age_decrypt_file(enc, output = tempfile(), identities = id)
readLines(dec)
```

`age_identity` *Parse or load age identities*

Description

Parse or load age identities

Usage

```
age_identity(x)
```

Arguments

`x`	A character vector of inline `"AGE-SECRET-KEY-1..."` secret-key strings and/or paths to (plaintext) key files. Auto-detected per element. An `age_identity` is returned unchanged.

Value

An `age_identity` object.

Examples

```
id <- age_keygen()
# round-trip through a key file
f <- tempfile()
age_keygen(f)
id2 <- age_identity(f)
```

age_identity_free *Explicitly scrub an identity's secret key material*

Description

Zeroes and frees the secret bytes immediately rather than waiting for garbage collection. The identity becomes unusable afterwards.

Usage

```
age_identity_free(identity)
```

Arguments

identity	An age_identity object.

Value

`invisible(NULL)`.

Examples

```
id <- age_keygen()
age_identity_free(id)
```

age_keygen *Generate a new age identity*

Description

Creates a fresh X25519 identity (key pair). The secret key is generated in C and never returned to R; to persist it, pass `path` so it is written directly to a key file in the standard age format.

Usage

```
age_keygen(path = NULL, overwrite = FALSE)
```

Arguments

path	Optional file path. If given, the identity is written as a key file (`# created: / # public key: / AGE-SECRET-KEY-1...`) and the object is returned invisibly. If `NULL` (default), nothing is written.
overwrite	If `FALSE` (default) and `path` already exists, error rather than clobbering an existing key.

Value

An age_identity object. Its print method shows only the public key; the secret is never printed.

Examples

```
id <- age_keygen()
id
age_pubkey(id)
```

---

age_passphrase *Encrypt and decrypt with a passphrase*

Description

Passphrase-based (scrypt) encryption, a separate mode from the recipient-based age_encrypt_raw() family. A file encrypted this way is decrypted with the matching age_decrypt_*_passphrase() function and the same passphrase — no key pair is involved.

Usage

```
age_encrypt_raw_passphrase(x, passphrase = NULL, armor = FALSE, log_n = 18)

age_decrypt_raw_passphrase(x, passphrase = NULL)

age_encrypt_file_passphrase(
  input,
  output = NULL,
  passphrase = NULL,
  armor = FALSE,
  overwrite = FALSE,
  log_n = 18
)

age_decrypt_file_passphrase(
  input,
  output = NULL,
  passphrase = NULL,
  overwrite = FALSE
)
```

Arguments

x	For age_encrypt_raw_passphrase(), a raw vector of plaintext. For age_decrypt_raw_passphrase(), a raw vector of ciphertext or a length-1 armored string.
passphrase	A length-1 character string. If NULL (the default) and the session is interactive, it is prompted for securely via askpass::askpass() (the suggested **askpass** package must be installed). Avoid hard-coding passphrases in scripts.

armor	If TRUE, produce ASCII-armored output.
log_n	scrypt work factor, as the base-2 logarithm of the parameter N. Higher is slower and more brute-force resistant. Defaults to 18; values above 22 are rejected (also on decrypt) to bound work.
input, output	File paths. output = NULL appends/strips .age as in age_encrypt_file().
overwrite	If FALSE (default), error when output already exists.

Value

The raw functions return a raw vector; the file functions return the output path invisibly.

Examples

```
ct <- age_encrypt_raw_passphrase(
  charToRaw("secret"),
  passphrase = "hunter2",
  log_n = 8
)
rawToChar(age_decrypt_raw_passphrase(ct, passphrase = "hunter2"))
```

---

age_pubkey *Derive public recipient strings from identities*

Description

Derive public recipient strings from identities

Usage

```
age_pubkey(identities)
```

Arguments

identities	An age_identity, or character input accepted by age_identity() (inline secret keys and/or key-file paths).

Value

A character vector of "age1..." recipient strings, one per identity.

Examples

```
id <- age_keygen()
age_pubkey(id)
```

age_raw *Encrypt and decrypt raw vectors*

Description

The core buffer transforms: no file I/O, nothing written to disk. All other verbs (_file, _text) build on these.

Usage

```
age_encrypt_raw(x, recipients, armor = FALSE)

age_decrypt_raw(x, identities)
```

Arguments

x	For age_encrypt_raw(), a raw vector of plaintext. For age_decrypt_raw(), a raw vector of ciphertext, or a length-1 character string of ASCII-armored ciphertext. Armor is auto-detected.
recipients	Character vector of "age1..." recipient strings. Give several to encrypt to multiple recipients; each can decrypt.
armor	If TRUE, produce ASCII-armored (PEM) output. The return value is still a raw vector (of the armored ASCII bytes).
identities	An age_identity, or character input accepted by age_identity(). Tried in order; the first that matches wins.

Value

A raw vector: ciphertext for age_encrypt_raw(), plaintext for age_decrypt_raw().

Examples

```
id <- age_keygen()
rec <- age_pubkey(id)
ct <- age_encrypt_raw(charToRaw("hello"), recipients = rec)
rawToChar(age_decrypt_raw(ct, identities = id))
```

age_text *Encrypt and decrypt a single string*

Description

Convenience wrappers over age_encrypt_raw() / age_decrypt_raw() for one string, ASCII-armored by default so the result is copy-pasteable. Encoding is forced to UTF-8 on the way in and marked on the way out.

Usage

```
age_encrypt_text(x, recipients, armor = TRUE)

age_decrypt_text(x, identities)
```

Arguments

x	For age_encrypt_text(), a length-1 character string (errors on longer input rather than silently collapsing lines). For age_decrypt_text(), ciphertext as a raw vector or armored string.
recipients	Character vector of "age1..." recipient strings.
armor	Must be TRUE (the default): age_encrypt_text() always armors its output. Use age_encrypt_raw() for unarmored binary output.
identities	An age_identity, or character input accepted by age_identity().

Value

age_encrypt_text() returns a length-1 character string; age_decrypt_text() returns a length-1 UTF-8 string. age_decrypt_text() signals age_error_decrypt if the plaintext is not valid UTF-8 text (for binary payloads, use age_decrypt_raw()).

Examples

```
id <- age_keygen()
rec <- age_pubkey(id)
ct <- age_encrypt_text("db_password_123", recipients = rec)
age_decrypt_text(ct, identities = id)
```

Index