age Simple file encryption

CONCEPTS

Encrypt with the public key (recipient, age1...); decrypt with the secret key in an identity file.

Plaintext stdin or file
↓ `age -r / -p`
Encrypted .age file
↓ `age -d -i`
Plaintext stdout or -o

age reads stdin and writes stdout by default, so it slots into pipes.

Keep the identity file secret and backed up: without it the files cannot be decrypted.

KEYS

`age-keygen -o` *`<file>`*
Generate a key pair into a file

`age-keygen -y` *`<file>`*
Print the public key of an identity

`age-keygen -pq -o` *`<file>`*
Generate a post-quantum key

`age-keygen | age -p > key.age`
Generate a key, protect it with a passphrase

`age-inspect` *`<file>`*
Show recipient types without decrypting

`age-inspect --json` *`<file>`*
Machine-readable inspection

ENCRYPT

`age -r` *`<recipient>`* **`-o`** *`<out>`* *`<file>`*
Encrypt to a public key

`age -r` *`<recipient>`* **`<`** *`<file>`* **`>`** *`<out>`*
Encrypt stdin to stdout

ENCRYPT (cont.)

`age -r` *`<r1>`* **`-r`** *`<r2>`* *`<file>`*
Encrypt to several recipients

`age -R` *`<recipients.txt>`* *`<file>`*
Encrypt to recipients from a file

`age -a -r` *`<recipient>`* *`<file>`*
Armored (PEM text) output

`age -p` *`<file>`*
Encrypt with a passphrase

`age -e -i` *`<identity>`* *`<file>`*
Encrypt to an identity file

`age -R -` *`<file>`*
Read recipients from stdin

DECRYPT

`age -d -i` *`<identity>`* *`<file>`**`.age`*
Decrypt with an identity file

`age -d -i` *`<identity>`* **`-o`** *`<out>`* *`<file>`**`.age`*
Decrypt to a named file

`age -d` *`<file>`**`.age`*
Decrypt a passphrase file

`age -d -i` *`<id1>`* **`-i`** *`<id2>`* *`<file>`**`.age`*
Try several identity files

`age -d -i -` *`<file>`**`.age`*
Read the identity from stdin

`age -d -i` *`<key.age>`* *`<file>`**`.age`*
Use a passphrase-protected identity

SSH KEYS

`age -R` *`<id_ed25519.pub>`* *`<file>`*
Encrypt to an SSH public key

`age -d -i` *`<id_ed25519>`* *`<file>`**`.age`*
Decrypt with the SSH private key

SSH KEYS (cont.)

`age -r "ssh-ed25519 AAAA..."` *`<file>`*
Give an SSH key inline

`curl` *`<url>`***`/`***`<user>`***`.keys | age -R -`** *`<file>`*
Encrypt to a GitHub user's keys

PIPELINES

`tar cz` *`<dir>`* **`| age -r`** *`<recipient>`* **`>`** *`<out>`***`.tar.gz.age`**
Archive and encrypt in one go

`age -d -i` *`<identity>`* *`<f>`***`.tar.gz.age | tar xz`**
Decrypt and unpack

`age -p <` *`<file>`* **`| base64`**
Passphrase-encrypt to pasteable text

`age -d -i` *`<identity>`* *`<file>`**`.age | less`**
Read an encrypted file without saving it

RECIPES

Set up and round-trip

```
age-keygen -o key.txt
# note the printed age1... public key
age -r age1... -o notes.txt.age notes.txt
age -d -i key.txt -o notes.txt notes.txt.age
```

Encrypted backup to two people

```
tar cz ~/data | age -R team.txt > data.tar.gz.age
age -d -i key.txt data.tar.gz.age | tar xz
```

Send a file to a GitHub user

```
curl https://github.com/<user>.keys | age -R - -a report.pdf > report.pdf.age
```

Passphrase-protected secret key

```
age-keygen | age -p > key.age
age -d key.age | grep 'public key'
age -d -i key.age secrets.age
```