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

ENCRYPT

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

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

`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

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

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

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