# age Simple file encryption

**START HERE**

**age-keygen -o key.txt**
Make a key pair

**age -R** *<recipients.txt>* *<file>*
Encrypt to many keys

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

**age -a -r** *<recipient>* *<file>*
Text-safe armored output

**age -d -i key.txt** *<file>***.age**
Decrypt with your key

**age -R ~/.ssh/id_ed25519.pub** *<file>*
Encrypt to an SSH key

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

**age-keygen -y key.txt**
Show the public key

**Plaintext** stdin or file → **age -r / -p** → **Encrypted** .age file → **age -d -i** → **Plaintext** stdout or -o

## CONCEPTS

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

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
*The public key is printed to the terminal and kept in the file as a comment.*

**age-keygen -y** *<file>*
Print the public key of an identity
*Redirect to a file to build a recipients list.*

**age-keygen -pq -o** *<file>*
Generate a post-quantum key
*Needs age 1.3 or later; recipients are about 2000 characters long.*

**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

**age -r** *<r1>* **-r** *<r2>* *<file>*
Encrypt to several recipients
*Any one of the listed keys can decrypt.*

**age -R** *<recipients.txt>* *<file>*
Encrypt to recipients from a file
*One recipient per line; blank lines and # comments are ignored.*

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

**age -p** *<file>*
Encrypt with a passphrase
*Leave the prompt empty and age generates a strong passphrase.*

**age -e -i** *<identity>* *<file>*
Encrypt to an identity file
*Needs the explicit -e flag.*

**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

## DECRYPT (cont.)

**age -d** *<file>***.age**
Decrypt a passphrase file
*age detects passphrase files and prompts.*

**age -d -i** *<id1>* **-i** *<id2>* *<file>***.age**
Try several identity files
*Unused identities are ignored.*

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

**age -d -i** *<key.age>* *<file>***.age**
Use a passphrase-protected identity
*age asks for the passphrase while decrypting.*

## SSH KEYS

**age -R** *<id_ed25519.pub>* *<file>*
Encrypt to an SSH public key
*ssh-ed25519 and ssh-rsa keys work.*

**age -d -i** *<id_ed25519>* *<file>***.age**
Decrypt with the SSH private key
*ssh-agent is not supported; keys on YubiKeys cannot decrypt.*

**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
*Use https://github.com/<user>.keys as the URL.*

## 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
```