Encrypting data files

v5 (checked by Cyber Security)

Encryption

Encryption is a technical tool to protect data. You can only encrypt a **single** file. In case you want to encrypt a directory containing multiple files that logically belong together, you first has to create an archive file (zip/tar file). How to make an archive file, see Archiving datasets in Areda: a guide.

There is a distinction between **symmetrical** and **asymmetrical** encryption.

Symmetrical encryption

Symmetrical encryption means that you generate a key (password) and only the person(s) who knows this key can decrypt the file. If you lose this encryption key there is no way to decrypt. If available, check research group policies regarding key sharing.

In some cases it is desired or needed to give access to encrypted data files without sharing the key. This is possible by means of asymmetrical encryption.

Asymmetrical encryption

Asymmetrical encryption is based on two different keys: a public and a private one. The combination of public and private key is unique. The public key is used when encrypting the file, the private key when decrypting it.

A key pair can be generated by means of a specific tool, such as AGE, Actually Good Encryption (see below). From this key pair only the public key should be shared. People who shared their public key with the person who encrypts the file, can decrypt it by using their private, non-shared, key.

For further explanation of asymmetrical encryption, see the section below about encryption tool AGE or this Wikipedia website.

Recommended encryption tools

Always use a program that is designed for encryption and nothing else, which is open source, and available for multiple platforms. **Do not use** a standard built-in encryption functionality offered, for instance, by 7-zip, as this is known to be very weak and can easily be cracked if you know the type of the files.

GnuPG (for experienced users)

GnuPG (GNU Privacy Guard) supports both symmetrical and asymmetrical encryption.

If you are already a GnuPG user, you can also use it for encryption of datasets. Make sure that people who have to decrypt your archive, in some cases many years later, are also used to GnuPG.

Please note that GnuPG is not easy for beginners. Below we discuss two more user-friendly encryption tools.

AES Crypt (symmetrical encryption)

A user-friendly open source cross platform encryption tool is AES Crypt. From the website you can download the version for your platform and install it on your computer (home directory, e.g. c:\users\bob). The website offers clear guidelines for each platform.

AGE (asymmetrical encryption)

Asymmetrical encryption is recommended when sharing of a single encryption key is not desired or permitted, or when the risk of losing the key for decryption is higher, often in case of long-term archiving of data.

AGE is an asymmetrical encryption tool which means that it generates a key pair. AGE can be freely downloaded here. Scroll to 'Assets', expand, and download the file for your platform.

In the downloaded zip/tar file you will find the folder 'age' containing two files called (in case of Windows) age.exe and age-keygen.exe. Extract this folder to your home directory, e.g. c:\users\bob. Be sure that you use exactly this path to your home directory in the procedure described below.

How to generate a key pair

To generate a key pair, use the command prompt.

```
cd c:\users\bob\age
.\age-keygen -o privkey.txt
```

This command generates a file privkey.txt that contains your private key. Store this file in a secret place (e.g. on a USB stick saved in a vault), and securely remove the file from your hard disk. You will only need the private key when decrypting the encrypted file.

The command also prints your public key. Save this key in a text file called pubkeys.txt (use a text editor like notepad, not Word!). You do not need to keep this file secret.

Giving right to decrypt

When encrypting a file you can give someone, *who should have generated a key pair as well*, the right to decrypt by adding his or her public key in the file pubkeys.txt. It is highly recommended to add not only *your* public key but also the public key of *at least one colleague in the group* who should have the right to decrypt the file (with his/her own private key). Check research group policies regarding key sharing, if available.

If you give Alice the right to decrypt an AGE encrypted file, the file pubkeys.txt could look like this:

```
# My (Bob's) public key
age1gu7d9xqu6f48a3ldr9csrayds249nnjxsjgmhsla8saz3yx3d5escjeplq
# Alice's public key
age1l0x67n9gvz6r6x0hd48u4tlvk6wd3rwamw9fqxk7s8z609nnxqmstcsxm6
```

Note: It is not possible to add public keys to an already encrypted file. If any change in decryption rights is needed, you need to decrypt the file and encrypt it again with a new pubkeys.txt file.

How to encrypt a file

The next command encrypts the tar file stored at drive d, directory 'data', d:\data\mydataset.tgz.

```
d:

cd \data

c:\users\bob\age\age -e -R c:\users\bob\age\pubkeys.txt -o mydataset.tgz.age mydataset.tgz
```

The encrypted tar file has the extension .age.

The file pubkeys.txt. file should contain the public keys from those who have the right to decrypt (see below).

How to decrypt a file

An AGE encrypted file can be decrypted with the command

```
c:\users\bob\age\age -d -i c:\users\bob\age\privkey.txt -o mydataset.tgz mydataset.tgz.age
```

Encrypting a large dataset

In case of large datasets (> 100GB) it is recommended to execute the encryption simultaneously with the creation of an archive file (zip or tar), avoiding the storage of the intermediate unencrypted zip/tar file on disk. Such a one-step execution can be done by using the command prompt.

For instance, you have installed AES Crypt or AGE in your home directory (e.g. c:\users\bob), and you want to archive 'mydataset' with tar and encrypt in one step:

AES Crypt (using your chosen key):

```
tar -czvf - mydataset | c:\users\bob\aescrypt\aescrypt -e –p [key] mydataset.tgz.aes
```

AGE:

```
tar -czvf - mydataset | c:\users\bob\age\age -e -R c:\users\bob\age\pubkeys.txt -o mydataset.tgz.age
```

The encrypted archive file (.aes or .age) can be uploaded to Areda (see Archiving datasets in Areda: a guide).

Decryption and extraction of the encrypted archive file can be done with the command:

AES Crypt:

```
c:\users\bob\aescrypt\aescrypt -d -p [key] mydataset.tgz.aes | tar -xzvf –
```

AGE:

```
c:\users\bob\age\age -d -i c:\users\bob\age\privkey.txt -o - mydataset.tgz.age | tar -xzvf –
```