

Description

Using PGP:

1. Generate public and private 1024-bit RSA keys.
2. Generate an ASCII version of your public key.
3. Pass your public-key to some else.
4. Produce a text file, and encrypt it with the other person's public key.
5. Get the other person to decrypt the encrypted message, using their private key.

Generating keys

Both the public and the private keys are generated with:

```
pgp -kg
```

Initially, the user is asked about the key sizes. The larger the key the more secure it is. A 1024 bit key is very secure.

```
C:> pgp -kg
Pretty Good Privacy(tm) Version 6.5.8
(c) 1999 Network Associates Inc.
Uses the RSAREF(tm) Toolkit, which is copyright RSA Data Security, Inc.
Export of this software may be restricted by the U.S. government.
```

```
Choose the public-key algorithm to use with your new key
1) DSS/DH (a.k.a. DSA/ElGamal) (default)
2) RSA
Choose 1 or 2: 2
Pick your RSA key size:
1) 1024 bits- High commercial grade, secure for many years
2) 2048 bits- "Military" grade, secure for foreseeable future
Choose 1, 2, or enter desired number of bits: 1
Generating a 1024-bit RSA key.
```

Next, the program asks for a user ID, which is normally the users name and his/her password. This ID helps other users to find the required public key.

```
You need a user ID for your public key. The desired form for this
user ID is your name, followed by your E-mail address enclosed in
<angle brackets>, if you have an E-mail address.
For example: John Q. Smith <jqsmith@nai.com>
Enter a user ID for your public key: Fred Smith <fred@home.com>
```

```
Enter the validity period of your signing key in days from 0 - 10950
0 is forever (the default is 0): 0
```

Next PGP also asks for a pass phrase, which is used to protect the private key if another person gets hold of it. No person can use the secret key file, unless they know the pass phrase. Thus the pass phrase is like a password but is typically much longer. The phase is also required when the user is encrypting a message with his/her private key.

```
You need a pass phrase to protect your RSA secret key.
Your pass phrase can be any sentence or phrase and may have many
words, spaces, punctuation, or any other printable characters.
```

```
Enter pass phrase: fred
```

Enter same pass phrase again: **fred**

The public and private keys are randomly derived from measuring the intervals between keystrokes. For this the software asks for the user to type a number of keys.

Note that key generation is a lengthy process.

```
PGP needs to generate some random data. This is done by measuring
the time intervals between your keystrokes. Please enter some
random text on your keyboard until the indicator reaches 100%.
Press ^D to cancel
100% of required data
Enough, thank you.
.....***** .....*****
Make this the default signing key? (Y/n) Y
```

Key generation completed.

This has created a public and a secret keyring (pubring.rkr and secring.skr).

Generating a text file of your public key

The `-kx` option can be used to extract the new public key from the public key ring and place it in a separate public key file, which can be sent to people who want to send an encrypted message to the user.

```
C:>pgp -kx fred
Pretty Good Privacy(tm) Version 6.5.8
(c) 1999 Network Associates Inc.
Uses the RSAREF(tm) Toolkit, which is copyright RSA Data Security, Inc.
Export of this software may be restricted by the U.S. government.
```

```
Extracting from keyring 'c:\windows\pubring.pkr', userid "fred".
```

```
Extract the above key(s) into which file? fred.pgp
```

```
Output file 'fred.pgp' already exists. Overwrite (y/N)? y
```

```
Key extracted to file 'fred.pgp'.
```

The public key file (`fred.pgp`) can be sent to other users, and can be added to their public key rings. Care must be taken never to send anyone a private key, but even if it is sent then it is still protected by the pass phase. An example public key is:

```
-----BEGIN PGP PUBLIC KEY BLOCK-----
Version: PGP 6.5.8

mQCNazwKsS8AAABEAM82ZVzbZEWwEluK6a2ZX5vv+KiyPvDGEEnnb2Ypv20caIc2T
Am3lUqKaXlGv1IEqAzbZ/mWK44U0tBDJZQ0ORW6n3HSVXb+dNdkMvrs+GZNXoal/
dJjU1WnA5xIkory9JQ3sQHbGoDkOEHEI0ecWfTik5yjk9alVotAxb0ckFVUvAAUR
tBpGcmVkiFNtaXR0IDxmcMvkQGhvbWUuY29tPokAlQMFEDwKsS/QMW9HJBVVLwEB
B6AEAK6dLuai0cQz7RHL3DntWR05HtSVPSTrYvDO5JXA/bk6NW9+fY42WWLD/Z5
cDV/BpuUHdhJ49I+eTbV9IO2JxEkkwN5X9S0dUA3d8AeWuH/SoAb9J3B8ePindXb
GrcC/xDDu0AsGFZl1VjNK78N/pdnPPKuCcYlwt9qnL0k458N
=sSxK
-----END PGP PUBLIC KEY BLOCK-----
```

Add a public key to your public key ring

Next, another user, say Bert, adds Fred's public key to their public keyring. This is achieved using the `-ka` option, as given next:

```

C:> pgp -ka fred.pgp
Pretty Good Privacy(tm) Version 6.5.8
(c) 1999 Network Associates Inc.
Uses the RSAREF(tm) Toolkit, which is copyright RSA Data Security, Inc.
Export of this software may be restricted by the U.S. government.

Looking for new keys...
RSA 1024      0x2415552F 2001/12/02 Fred Smith <fred@home.com>
sig?         0x2415552F          (Unknown signator, can't be checked)

keyfile contains 1 new keys. Add these keys to keyring ? (Y/n) Y

New userid: "Fred Smith <fred@home.com>".
New signature from keyID 0x2415552F on userid Fred Smith <fred@home.com>

Keyfile contains:
  1 new key(s)
  1 new signatures(s)
  1 new user ID(s)

Summary of changes :

New userid: "Fred Smith <fred@home.com>".
New signature from keyID 0x2415552F on userid Fred Smith <fred@home.com>

Added :
  1 new key(s)
  1 new signatures(s)
  1 new user ID(s)

```

Fred's key has been added to Bert's public key ring. This ring can be listed with the `–kv`, as given next:

```

C: >pgp -kv
Pretty Good Privacy(tm) Version 6.5.8
(c) 1999 Network Associates Inc.
Uses the RSAREF(tm) Toolkit, which is copyright RSA Data Security, Inc.
Export of this software may be restricted by the U.S. government.

Type bits      keyID      Date      User ID
RSA 1024      0xDB2936DB 2001/12/02 *** DEFAULT SIGNING KEY ***
                                           Bert <bert@home.com>
RSA 1024      0x2415552F 2001/12/02 Fred Smith <fred@home.com>
2 matching keys found.

```

Encrypting a file

Next, a message can be send to Fred, using his public key.

```

C: >edit hello.txt
<add some text here>

```

and then encrypted using:

```

C: >pgp -ea hello.txt
Pretty Good Privacy(tm) Version 6.5.8
(c) 1999 Network Associates Inc.
Uses the RSAREF(tm) Toolkit, which is copyright RSA Data Security, Inc.
Export of this software may be restricted by the U.S. government.

Recipients' public key(s) will be used to encrypt.

A user ID is required to select the Recipient's public key.
Enter the Recipient's user ID: fred

```

```
Key for user ID: Fred Smith <fred@home.com>
1024-bit RSA key, Key ID 0x2415552F, created 2001/12/02
WARNING: Because this public key is not certified with a trusted
signature, it is not known with high confidence that this public key
actually belongs to: "Fred Smith <fred@home.com>".
```

Are you sure you want to use this public key (y/N)?**y**

Transport armor file: hello.txt.asc

An example of the file produced is:

```
-----BEGIN PGP MESSAGE-----Version: PGP 6.5.8
hQCMA9Axb0ckFVUvAQP+PHvEd5kKte1TdN/zRTQoKcMnjdDhh+HUFjTPwNRXKMLAj
BqqPS2KF17AfqxQegscleU7RSBThOW/ORrN6lnnWxvm/aaLgJ32Cs8U+eFUOzn8P
Y/1YciNPx8hZ89SII0fVxO6YHTXWkn2gmfTW8EQRgrvy/9rOnY1qlTgl1313Ijak
L/nQCfyL/GiE904gW9O92KEYk57hfsViQ10ZuV8eUxQvUMschtfV5Vpewc/UMxaj =6rIG
-----END PGP MESSAGE-----
```

Decrypting a message

Fred can then decrypt the message with the `-da` option:

```
c:> pgp -da message.txt.asc
Stripped transport armour from 'message.txt.asc' producing 'message.txt.pgp'
```

and then with:

```
C:> pgp message.txt.pgp

File is encrypted. Secret key is required to read it.
Key for user ID: Bert Smith <Bert_s.otherserver.com>
1024-bit key, key ID 770CA60D, created 1998/12/30

You need a pass phrase to unlock your RSA secret key.
Enter pass phrase: Fred
Pass phrase is good. Just a moment.....
Plaintext filename: message.txt
```