BSc in Computer Engineering

CMP4103

Computer Systems and Network Security

Lecture 3b
Pretty Good Privacy (PGP) / GNU Privacy Guard (GPG)

Eng Diarmuid O'Briain, CEng, CISSP

Department of Electrical and Computer Engineering,
College of Engineering, Design, Art and Technology,
Makerere University

Table of Contents

This page is intentionally blank

1. GNU Privacy Guard

GnuPG is a complete and free implementation of the OpenPGP standard as defined by RFC4880 (also known as PGP). GnuPG allows to encrypt and sign data and communication, features a versatile key management system as well as access modules for all kinds of public key directories. GnuPG, also known as GPG, is a command line tool with features for easy integration with other applications. A wealth of frontend applications and libraries are available. Version 2 of GnuPG also provides support for S/MIME and Secure Shell (ssh).

GnuPG is Free Software (meaning that it respects your freedom). It can be freely used, modified and distributed under the terms of the GNU General Public License .

Project Gpg4win provides a Windows version of GnuPG stable. It is nicely integrated into an installer and features several front-ends as well as English and German manuals.

1.1 Generate a private key

Most people make their keys valid until infinity, which is the default option. If this is done don't forget to revoke the key when it is no longer in use.

Make sure that the name on the key is not a pseudonym, and that it matches the name in the users passport, or other government issued photo-identification! Additional e-mail addresses can be added to the key later.

A passphrase will be asked for twice. Usually, a short sentence or phrase that isn't easy to guess can be used. Next a request will be made to tap on the keyboard or do any of the things normally done on the computer in order for randomisation to take place. This is done so that the encryption algorithm has more human-entered elements, which, combined with the passphrase entered above, will result in the user's private key.

Key-ID of the created key is: **6E64AF4C**

```
$ gpg --gen-key

gpg (GnuPG) 1.4.16; Copyright (C) 2013 Free Software Foundation, Inc.
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.

gpg: directory `/home/alovelace/.gnupg' created
gpg: new configuration file `/home/alovelace/.gnupg/gpg.conf' created
gpg: WARNING: options in `/home/alovelace/.gnupg/gpg.conf' are not yet active during
this run
gpg: keyring `/home/alovelace/.gnupg/secring.gpg' created
gpg: keyring `/home/alovelace/.gnupg/pubring.gpg' created
Please select what kind of key you want:
   (1) RSA and RSA (default)
   (2) DSA and Elgamal
   (3) DSA (sign only)
   (4) RSA (sign only)
Your selection? 1
RSA keys may be between 1024 and 4096 bits long.
What keysize do you want? (2048) 2048
Requested keysize is 2048 bits
Please specify how long the key should be valid.
         0 = key does not expire
      <n>  = key expires in n days
      <n>w = key expires in n weeks
      <n>m = key expires in n months
      <n>y = key expires in n years
Key is valid for? (0) 1w
Key expires at Wed 16 Mar 2016 14:04:20 EAT
Is this correct? (y/N) y

You need a user ID to identify your key; the software constructs the user ID
from the Real Name, Comment and Email Address in this form:
    "Heinrich Heine (Der Dichter) <heinrichh@duesseldorf.de>"

Real name: Ada Lovelace
Email address: alovelace@mak.ac.ug
Comment: March Key
You selected this USER-ID:
    "Ada Lovelace (March Key) <alovelace@mak.ac.ug>"

Change (N)ame, (C)omment, (E)mail or (O)kay/(Q)uit? O
You need a Passphrase to protect your secret key.

Enter passphrase: babbage
Re-enter passphrase: babbage

We need to generate a lot of random bytes. It is a good idea to perform
some other action (type on the keyboard, move the mouse, utilize the
disks) during the prime generation; this gives the random number
generator a better chance to gain enough entropy.

gpg: /home/alovelace/.gnupg/trustdb.gpg: trustdb created
gpg: key 6E64AF4C marked as ultimately trusted
public and secret key created and signed.

gpg: checking the trustdb
gpg: 3 marginal(s) needed, 1 complete(s) needed, PGP trust model
gpg: depth: 0  valid:   1  signed:   0  trust: 0-, 0q, 0n, 0m, 0f, 1u
gpg: next trustdb check due at 2016-03-16
pub   2048R/6E64AF4C 2016-03-09 [expires: 2016-03-16]
      Key fingerprint = E150 F5AC F0E5 A492 6891  0903 F315 80E3 6E64 AF4C
uid                  Ada Lovelace (March Key) <alovelace@mak.ac.ug>
sub   2048R/DC6CB630 2016-03-09 [expires: 2016-03-16]
```

1.2 Generate a public key

```
$ gpg --armor --output pubkey.txt --export 'Ada Lovelace'
```

```
$ cat pubkey.txt
```

```
-----BEGIN PGP PUBLIC KEY BLOCK-----
Version: GnuPG v1

mQENBFbgA5sBCAC12JYl3KtCjH7jOtfZWBw7gISy5Zl8Y4WMQnEsD7F/na1xQqWB
2kN8ka2MzBCyUFlWQ6sJu/F8jfkzS3YGy4eCa7OZeAdQgZ4EQU+eC1rIo8cLhPA+
jyL5EacQ+jG4kBDsLD+kD8AA55whmAGoapK2lZNyX8tuoW7Ex94BkATB3EgwJ/0Q
53aGrsH93BhIEesc32duvpS0uRe9xY+iEnU9ZquCE6hdCqJBXHo2HdCqs2nN8os8
AlwAfLvN7uRc4Yv7qi5tpWM5+9L30lZlm2/Ydkl2WPxotkCp6mqp+RvZmL7w6hh/
dmflZ/Ts+SzrPMdt9QtE/hJA/J/j8uOedc8jABEBAAG0K0FkYSBMb3ZlbGFjZSAo
TWFyY2ggS2V5KSA8YWxvdmVsYWNlQGMycy5pZT6JAT4EEwECACgFAlbgA5sCGwMF
CQAJOoAGCwkIBwMCBhUIAgkKCwQWAgMBAh4BAheAAAoJEPMVgONuZK9MqIEH/1B3
BHTjJGsXbWsobJnKMJuJeHNaL+9ibmHkgPK3r1K77D8n0xyO4/k0rpS/BNbp2e9V
hEbIpk3/tgIJOGgEhm7ckSmTZSfOyTBi5YO0c/GzhAptNKDbk+qSRVgoV+qtIaeE
PBvUWthZzpa6qRO8b24hdi7QwR354Lf2ZOnU/+WY5/DBGxl4+NGJ3BIN5wB7yL8L
PeHTAyseftgN0wR6C9AwEXx7mW0kBLLFVEmwAB6sJzXvgOOsRpQ5Wr5bF5C1CWf5
MG+VQ0abjnteP3I6YibQcDEExqDXfo1nebVzKu1Nke7bCef6jvrEJ6W2BlxAGf0L
e8c2+SB5QsJFEAGmBSG5AQ0EVuADmwEIAMGIj2BqX2OP2kl7GYXaiaV+xSxndNR0
Rtv7yvehKbJ9YhmplxyHLOIXBqoFWC9YUPcoy40HmhfPrhXrvIA5Uiema6dm/BFB
OtnrpHM0JWHgBwmk3GOQWH2WKHwlRjIhc36l93wesuJYENskIy/WtLEfiuvkS4ZA
fiEhw91VVp0LFTrBahprOgYzcOhra78RhvxI8/vTS03a1GYryuOQoArCjj+TFNMh
GIVgxIY2XWxOlKO7hh75VRRrbM6dhZJkemLKPiKzqbpPfpQaCN11kytQLtJ+r0KH
Lrv3GjhGaChRehDSkVoSltPzsYpSslj/bG5jK0BqmTRzNnOUXQjWXvsAEQEAAYkB
JQQYAQIADwUCVuADmwIbDAUJAAk6gAAKCRDzFYDjbmSvTBUtB/0cxtlyK9jB82rl
QVCNViJIsfnKYC+wZ4h84HhoCpzyTBweRm1nnSNu06paps+rS/GXQ0yOOfT4b/NA
Lv5iJwKANRqkShH4LsxbGYd8Ps/jMEc8lRnSTNwlHnKGzjSco9wGnF/A2omqc2gd
LAFlHZPUJDnzhG2H5jHvgwJs6Oncgs5FyjtA/FnUUqMzy+ITQCbYQEnDQn8CmNyB
Vnxz04u+Td2ajRznD3V29UvXgTaG7gU5842CNsLiezrfqqPgnNnRISxpAboy3xCp
UbqBG/i8z6hNwGDBZRGuwKROAYC5dNDE+SBugYub16SDkhe2dR5tGbURFPSeOdLp
f17ONf3/
=JYlA
-----END PGP PUBLIC KEY BLOCK-----
```

The public key can be freely distributed, posted on a web site, or otherwise distributed. It can also be registered with public keyservers so that others can retrieve the key without having to contact the owner directly.

```
$ gpg --send-keys 'Ada Lovelace' --keyserver hkp://subkeys.pgp.net
```

1.3 Encrypting a file for personal use

Taking a file **MyFile.txt** as a target to experiment with.

```
$ ls -la | grep MyFile
-rw-r--r--  1 dobriain dobriain     74 Mar  9 14:23 MyFile.txt

$ cat MyFile.txt
MyFile
======

This is a file used as part of the exercise on encryption.
```

Now encrypt the file. The argument to the **--recipient** is the name used when generating the private key. Note the output is a new file **MyFile.txt.gpg**.

```
$ gpg --encrypt --recipient 'Ada Lovelace' MyFile.txt

$ ls -la | grep MyFile
-rw-r--r--  1 dobriain dobriain     74 Mar  9 14:23 MyFile.txt
-rw-r--r--  1 dobriain dobriain    406 Mar  9 14:23 MyFile.txt.gpg
```

Use the file command to interrogate the file type and the cat command to view the contents.

```
$ file MyFile.txt
MyFile.txt: ASCII text

$ file MyFile.txt.gpg
MyFile.txt.gpg: data

$ cat MyFile.txt
MyFile
======

This is a file used as part of the exercise on encryption.

$ cat MyFile.txt.gpg
�#
�!�l�0#�P�#ͻa��E��#Ɽ��6�VЦ+��vU`<��#<"�.�i}|�����]
                                                         vm��#Tq<�&b���j��|
#�ħ������G�H�-d��g�� r#���$�O�I��UB�
        �2%���#�����#)0���O�f﮼J�O�^#Wp#��0+<�
                                                     ��[��#6��X�#K?M#�Hb��#
    ��~##)�4��0  !e�8'�&�@�Db�Ӽn�'�9ن`v�#��sNc���Z�F$^�?��P�tUL#�^�o#_�
%�#����A��~��h�f�#�##�_���
����Rc2
       �������@#�\#�2���&�=#,#D
�6�#N���#����c��#Z#6I
                      ���,N�r�=(0����#�#�
```

1.4 Decrypting the file for personal use

Decrypt the file to a new file called **MyFile2.txt**.

```
$ gpg --output MyFile2.txt --decrypt MyFile.txt.gpg
You need a passphrase to unlock the secret key for
user: "Ada Lovelace (March Key) <alovelace@mak.ac.ug>"
2048-bit RSA key, ID DC6CB630, created 2016-03-09 (main key ID 6E64AF4C)

gpg: encrypted with 2048-bit RSA key, ID DC6CB630, created 2016-03-09
      "Ada Lovelace (March Key) <alovelace@mak.ac.ug>"

Enter passphrase: babbage
```

Check the new file and use the **diff** command to confirm it is identical to the original file **MyFile.txt**.

```
$ cat MyFile2.txt
MyFile
======

This is a file used as part of the exercise on encryption.

$ diff MyFile.txt MyFile2.txt
```

1.5 Passing encrypted files to another person

Consider two computers “A” and “B” which have Internet connectivity between them. On ***CmpA*** create a private key and from it a public key for Ada Lovelace.

```
ada@cmpA~$ gpg --gen-key

gpg (GnuPG) 1.4.18; Copyright (C) 2014 Free Software Foundation, Inc.
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.

gpg: directory `/root/.gnupg' created
gpg: new configuration file `/root/.gnupg/gpg.conf' created
gpg: WARNING: options in `/root/.gnupg/gpg.conf' are not yet active during
this run
gpg: keyring `/root/.gnupg/secring.gpg' created
gpg: keyring `/root/.gnupg/pubring.gpg' created
Please select what kind of key you want:
   (1) RSA and RSA (default) 1
   (2) DSA and Elgamal
   (3) DSA (sign only)
   (4) RSA (sign only)
Your selection?
RSA keys may be between 1024 and 4096 bits long.
What keysize do you want? (2048) 2048
Requested keysize is 2048 bits
Please specify how long the key should be valid.
         0 = key does not expire
      <n>  = key expires in n days
      <n>w = key expires in n weeks
      <n>m = key expires in n months
      <n>y = key expires in n years
Key is valid for? (0) 1
Key expires at Thu 10 Mar 2016 13:02:39 GMT
Is this correct? (y/N) y

You need a user ID to identify your key; the software constructs the user
ID
from the Real Name, Comment and Email Address in this form:
    "Heinrich Heine (Der Dichter) <heinrichh@duesseldorf.de>"

Real name: Ada Lovelace
Email address: alovelace@cedat.mak.ac.ug
Comment: Ada Lovelace March key
You selected this USER-ID:
    "Ada Lovelace (Ada Lovelace March key) <alovelace@cedat.mak.ac.ug>"

Change (N)ame, (C)omment, (E)mail or (O)kay/(Q)uit? O
You need a Passphrase to protect your secret key.

Enter passphrase: babbage
Re-enter passphrase: babbage
```

```
We need to generate a lot of random bytes. It is a good idea to perform
some other action (type on the keyboard, move the mouse, utilize the
disks) during the prime generation; this gives the random number
generator a better chance to gain enough entropy.

gpg: /root/.gnupg/trustdb.gpg: trustdb created
gpg: key 3AAB4367 marked as ultimately trusted
public and secret key created and signed.

gpg: checking the trustdb
gpg: 3 marginal(s) needed, 1 complete(s) needed, PGP trust model
gpg: depth: 0  valid:   1  signed:   0  trust: 0-, 0q, 0n, 0m, 0f, 1u
gpg: next trustdb check due at 2016-03-10
pub   2048R/3AAB4367 2016-03-09 [expires: 2016-03-10]
      Key fingerprint = 7F0F EE48 BA1D 5DF9 F7F0  A3D9 D78C 0277 3AAB 4367
uid                                Ada  Lovelace  (Ada  Lovelace  March  key)
<alovelace@cedat.mak.ac.ug>
sub   2048R/E5DE8209 2016-03-09 [expires: 2016-03-10]
```

Now create a public key from this.

```
ada@cmpA~$ gpg --armor --output pubkey.txt --export 'Ada Lovelace'

ada@cmpA~$ cat pubkey.txt
-----BEGIN PGP PUBLIC KEY BLOCK-----
Version: GnuPG v1

mQENBFbgHzkBCADASnR2j4651YHPINsvpY9DIjX7MomAxzjpHJmQHdxp8m5jLyDT
ild4zttzJgtk6QVE97+ozAo0Artk/uh3nXLGFKJwe7h491T7hePjMT3T5bfrAx9p
Wy8rP8XhOsw4kwv2x6MXMg+9fAxZbL5sZXS/4NUUsSl5XVaPsDwHBz2vHyxQ9RKA
uQARFR88xeMXpY3Z6So4YGhsKkDsMa/K6u/SBf26QlVtaRTBZx/urzrlzeBbYtgz
6LA+4RevDs7PcbxhRUq2w/TV4CA7oL/36K/LmdwjrrE2QBd93qQaGYeklGnBQOhT
sctAmGrMKZiT3uuj+4hrC6ludbU7YqIJcQFrABEBAAG0QUFkYSBMb3ZlbGFjZSAo
QWRhIExvdmVsYWNlIE1hcmNoIGtleSkgPGFsb3ZlbGFjZUBjZWRhdC5tYWsuYWMu
dWc+iQE+BBMBAgAoBQJW4B85AhsDBQkAAVGABgsJCAcDAgYVCAIJCgsEFgIDAQIe
AQIXgAAKCRDXjAJ3OqtDZxzPB/9zGZD3TJJ/RyFcZeyMVfpVskVSBc01FBt562GI
VFrwTYvf+k6WPX/B2KXsebNx2bJBvCXosQDotIDM6yTNlP8AUUM2pVdhbhuDpZMC
XYIz3PmdCHQ+0si1vpNC8AN22stCtZe7qMWMzk7MGEblE2Ie/WQRPmo9xwQ+tPEq
d5UKK9GqSOG1PdyZLxzZ5ERhMDYpszjt7oTx6cy4Uag7OrusAwDhMjufqxFWLuJu
u/1CfN+QVFvrlhuip80TB2cjLGCqMpjL+sKhj5d3I6CD9TKtQ8MX9FL1QudgVhVg
W+Qm6aNj5N748dgt3LseiUurVjNs0VZ6dtMFpcR5GxQ022MduQENBFbgHzkBCACw
xPCjRq6br7yCqCcXopJ1tKwe5vxvXKyBIADvUVX97evQaxj5eTczgvNrn9fIlb7X
9NTH+yKxize+bFky8IKbwCUmgur2uBKEhXVcyDOJapkgEIGS6KYK4Su8ucTyGBXS
+l88LNxYCcxBDiQlWPCWxt2czws3AZNCTES3LQXgdr9jBYSYRh9Kif7VxH5Aqgyx
vrGYdnq5j9CxQxioHRrgrpA6A6rIOo/DVZ8IONnTHMWT0Y37k4cMT6Gv0ieNu6aW
d6XXljFv+EDcAxGk3DzH8JAJxnVIjXXjUqAm1yreRt0ylgcZWemQwzY1FXsH8UI2
RzHUH5EgvUaGZMFOSXa9ABEBAAGJASUEGAECAA8FAlbgHzkCGwwFCQABUYAACgkQ
14wCdzqrQ2czZggAoWR6Yuiry8k3S/GnkMDh3/jX5aLjf1QEWpvi5SkXGeH3GXAl
5AFOYzmgqp5zqcKIp9gqt1VQi9uKefRGpKRlth8A9WSRxzEOyYB1BrSkuXLtnGmF
PV8CegDUlZDqINkVNb8RObXmEcEq4JZRvHyJneTbsSTAXSkE7eUWEh7z8Sm+M6qi
dxEy8yvHmFZSkz/vYcYeGTvaM8og5JWv1Iw9bdSF7kVbs9GljWI4VnAQ1q/xjFjz
ulKfO/Tp6eShduYdebgQ6n8T8rwYSGrKl//emIZi6VfdL1U/CM/7Ia0Yc36yMQXG
4FjxE5dcfrS+y7K7ZaLG1is8oP+aF5tcPcWEYQ==
=BAof
-----END PGP PUBLIC KEY BLOCK-----
```

1.5.1 Public key

On the ***cmpB*** get the public key from the ***cmpA***.

```
alan@cmpB~$ sftp ada@10.0.2.10
ada@10.0.2.10's password: adapasswd
Connected to 10.0.2.10.

sftp> ls
pubkey.txt

sftp> get pubkey.txt
Fetching /home/ada/pubkey.txt to pubkey.txt
/home/ada/pubkey.txt 100% 1763     1.7KB/s   00:00

sftp> quit
```

The cmpB now has the public key from cmpA.

```
alan@cmpB~$ ls
pubkey.txt
alan@cmpB~$
```

1.5.2 Generate a secret file and encrypt

```
alan@cmpB~$ echo "This is my little secret" > secretFile.txt

alan@cmpB~$ cat secretFile.txt
This is my little secret
```

Import Ada Lovelace's public key just copied over from ***cmpA***.

```
alan@cmpB~$ gpg --import pubkey.txt
gpg:    key    3AAB4367:    "Ada    Lovelace    (Ada    Lovelace    March    key)
<alovelace@cedat.mak.ac.ug>" not changed
gpg: Total number processed: 1
gpg:              unchanged: 1
```

Search for Ada Lovelace's key in the keyring.

```
alan@cmpB~$ gpg --list-keys
/root/.gnupg/pubring.gpg
------------------------
pub   2048R/3AAB4367 2016-03-09 [expires: 2016-03-10]
uid                  Ada Lovelace (Ada Lovelace March key) <alovelace@cedat.mak.ac.ug>
sub   2048R/E5DE8209 2016-03-09 [expires: 2016-03-10]
```

Encrypt the secret file **pubkey.txt** with Ada Lovelace's public key.

```
alan@cmpB~$ gpg --encrypt --recipient 'Ada Lovelace' secretFile.txt
```

There is now a new encrypted file in the local directory. Confirm it is encrypted.

```
alan@cmpB~$ ls secretFile*
secretFile.txt   secretFile.txt.gpg

alan@cmpB~$ file secretFile.txt.gpg
secretFile.txt.gpg: PGP RSA encrypted session key - keyid: 9EBC8885 982DEE5
RSA (Encrypt or Sign) 2048b .
```

Send the new encrypted file to ***cmpA***.

```
alan@cmpB~$ sftp ada@10.0.2.10
root@10.0.2.10's password: adapasswd
Connected to 10.0.2.10.

sftp> put secretFile.txt.gpg
Uploading secretFile.txt.gpg to /home/ada/secretFile.txt.gpg
secretFile.txt.gpg 100%  366     0.4KB/s   00:00

sftp> quit
```

1.6 Decrypt secret file on cmpA

Confirm secret file is on ***cmpA***.

```
ada@cmpA~$ ls secret*
secretFile.txt.gpg
```

Confirm secret file is on ***cmpA***.

```
ada@cmpA~$ gpg --output cmpB_secret.txt --decrypt secretFile.txt.gpg

You need a passphrase to unlock the secret key for
user: "Ada Lovelace (Ada Lovelace March key) <alovelace@cedat.mak.ac.ug>"
2048-bit RSA key, ID E5DE8209, created 2016-03-09 (main key ID 3AAB4367)

Enter passphrase: babbage

gpg: encrypted with 2048-bit RSA key, ID E5DE8209, created 2016-03-09
      "Ada Lovelace (Ada Lovelace March key) <alovelace@cedat.mak.ac.ug>"
```

Check the decrypted file.

```
ada@cmpA~$ cat cmpB_secret.txt
This is my little secret
```

- To summarise, ***cmpB*** received the public key for Ada Lovelace on ***cmpA***.
- ***cmpB*** used this public key to encrypt a file.
- ***cmpB*** sent the file to ***cmpA***.
- ***cmpA*** decrypted the file using Ada Lovelace's private key.

1.7 Digitally signing a file

GPG also provides a mechanism to digitally sign a file. Ada Lovelace wishes to have a file signed so those with her public key can confirm that she did indeed create the file, i.e. non repudiation.

```
ada@cmpA~$ echo 'This file will be signed" > sign.txt

ada@cmpA~$ gpg --armor --detach-sign sign.txt

You need a passphrase to unlock the secret key for
user: "Ada Lovelace (Ada Lovelace March key) <alovelace@cedat.mak.ac.ug>"
2048-bit RSA key, ID 3AAB4367, created 2016-03-09

Enter passphrase: babbage


ada@cmpA~$ ls -la sign*
-rw-rw-rw- 1 root root  25 Mar  9 19:18 sign.txt
-rw-rw-rw- 1 root root 473 Mar  9 19:20 sign.txt.asc

ada@cmpA~$ file sign.txt.asc
sign.txt.asc: PGP signature Signature (old)


ada@cmpA~$ cat sign.txt.asc
-----BEGIN PGP SIGNATURE-----
Version: GnuPG v1

iQEcBAABAgAGBQJW4HdvAAoJENeMAnc6q0NnQekH/3bck0fGF3FSblpQSeVfrZLJ
sCpGNvLeDv+PpyPCLRtPqwHJlCGMFsP6FCh9d07EYJIYnpHuvnwSPaJCsXqS6OcX
f11vbSo24BLWYDN/T9v7Kt3ui7jEhUYqQNZQXMzlciVrRpYqU5F4vQClTChQXZ2l
9R71QuOGi98AsAZfitAXU3L3SLPxHwieJefqsuWgLqI75uuB2atoy+FvrFSQ7gdv
nW9ylvehHFLtyXwKMQUZ5OSGW/DUl0M6CRVofu4aY9BsIHV5z9yiMiQG3Vi2t5Kl
/4YAzN34jy0YHPDTFrv3qCdgGtuB/Zv9/6CkYYRjP4XyhBtJM74483lDF+hJzjU=
=SS9+
-----END PGP SIGNATURE-----
```

1.7.1 Verifying a signed file

On cmpB get the two files, the ***sign.txt*** and ***sign.txt.asc*** from ***cmpA***.

```
alan@cmpB~$ sftp root@10.0.2.10
root@10.0.2.10's password: root
Connected to 10.0.2.10.


sftp> cd /tmp/pycore.54569/cmpA.conf


sftp> mget sign.txt*
Fetching /home/ada/sign.txt to sign.txt
/home/ada/sign.txt          100%   25     0.0KB/s   00:00
Fetching /home/ada/sign.txt.asc to sign.txt.asc
/home/ada/sign.txt.asc      100%  473     0.5KB/s   00:00


sftp> exit
```

Confirm signature.

```
alan@cmpB~$ gpg --verify sign.txt.asc sign.txt
gpg: Signature made Wed 09 Mar 2016 19:20:15 GMT using RSA key ID 3AAB4367
gpg:  Good  signature  from  "Ada  Lovelace  (Ada  Lovelace  March  key)
<alovelace@cedat.mak.ac.ug>"
```

This page is intentionally blank