

Technologie sieciowe 2

Autor:

Tymon Tobolski (181037)

Jacek Wieczorek (181043)

Prowadzący:

Dr inż. Arkadiusz Grzybowski

Wydział Elektroniki

III rok

Pn TN 11.15 - 13.00

15 stycznia 2012

1 Cel laboratorium

Celem ćwiczenia jest zapoznanie się z oprogramowaniem GnuPG do szyfrowania i odpisywania dokumentów elektronicznych oraz zarządzaniem kluczami. W ramach ćwiczenia student zapoznaje się z: podstawowymi algorytmami kryptograficznymi, tworzeniem, eksportem i importem kluczy, podpisywaniem e-maili oraz plików, budowaniem sieci zaufania.

2 Generowanie kluczy

Klucz GPG można utworzyć z linii komend wydając polecenie `gpg --gen-key`. Podczas generowania program pyta użytkownika o szereg opcji:

- Typ szyfrowania - RSA, DSA
- Długość klucza - od 1024 do 4096 bitów
- Czas ważności klucza
- Dane właściciela klucza - imię i nazwisko, adres email oraz opcjonalny komentarz

Ostatnim etapem generowania klucza jest dwukrotne podanie hasła zabezpieczającego klucz.

Rysunek 1 pokazuje generowanie klucza z szyfrowaniem RSA, o długości klucza 1024 bitów, ważnego przez 4 dni.

```
2. zsh
% gpg --gen-key
gpg (GnuPG) 1.4.11; Copyright (C) 2010 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.

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) 1024
Requested keysize is 1024 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) 4
Key expires at Thu Jan 19 18:07:59 2012 CET
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: Tymon Tobolski
Email address: 181037@student.pwr.wroc.pl
Comment:
You selected this USER-ID:
  "Tymon Tobolski <181037@student.pwr.wroc.pl>"

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

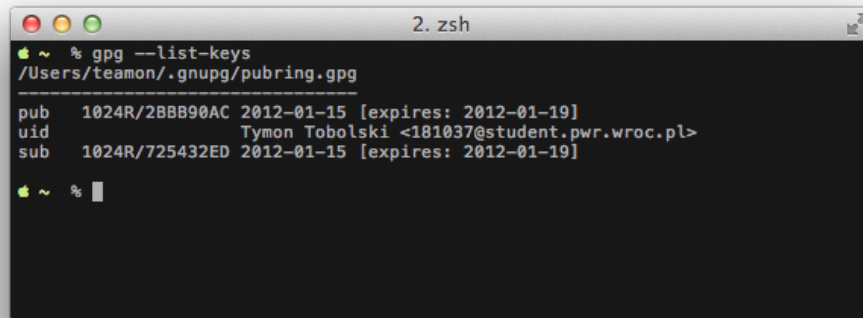
We need to generate a lot of random bytes. It is a good idea to perform
some other action (type on the keyboard, move the mouse, utilize the
disks) during the prime generation; this gives the random number
generator a better chance to gain enough entropy.
+++++
+++++
We need to generate a lot of random 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: key 4B621545 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: 2 signed: 0 trust: 0-, 0q, 0n, 0m, 0f, 2u
gpg: next trustdb check due at 2012-01-19
pub 1024R/4B621545 2012-01-15 [expires: 2012-01-19]
```

Rysunek 1: Generowanie klucza GPG

3 Dodawanie, usuwanie i modyfikacja kluczy

W celu wyświetlenia listy wygenerowanych kluczy można użyć polecenia `gpg --list-keys`.



```
2. zsh
% gpg --list-keys
/Users/teamon/.gnupg/pubring.gpg
-----
pub   1024R/2BB890AC 2012-01-15 [expires: 2012-01-19]
uid           Tymon Tobolski <181037@student.pwr.wroc.pl>
sub   1024R/725432ED 2012-01-15 [expires: 2012-01-19]
% 
```

Rysunek 2: Wyświetlenie listy kluczy

Edycji klucza można dokonać przy pomocy polecenia `gpg --edit-key NAZWA`. Rysunek 3 pokazuje proces edycji klucza o nazwie *Tymon Tobolski <181037@student.pwr.wroc.pl>*. W tym wypadku został zmieniony termin ważności klucza na 2 miesiące przy użyciu komendy `expire`. W celu sprawdzenia wszystkich dostępnych komend można wykorzystać polecenie `help`. Przy zapisie zmian program wymaga podania hasła zabezpieczającego klucz.

```
2. zsh
~ % gpg --edit-key "Tymon Tobolski <181037@student.pwr.wroc.pl>"
gpg (GnuPG) 1.4.11; Copyright (C) 2010 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.

Secret key is available.

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 2012-01-19
pub 1024R/28BB90AC created: 2012-01-15 expires: 2012-01-19 usage: SC
    trust: ultimate validity: ultimate
sub 1024R/725432ED created: 2012-01-15 expires: 2012-01-19 usage: E
[ultimate] (1). Tymon Tobolski <181037@student.pwr.wroc.pl>

gpg> expire
Changing expiration time for the primary key.
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) 2m
Key expires at Thu Mar 15 18:18:46 2012 CET
Is this correct? (y/N) y

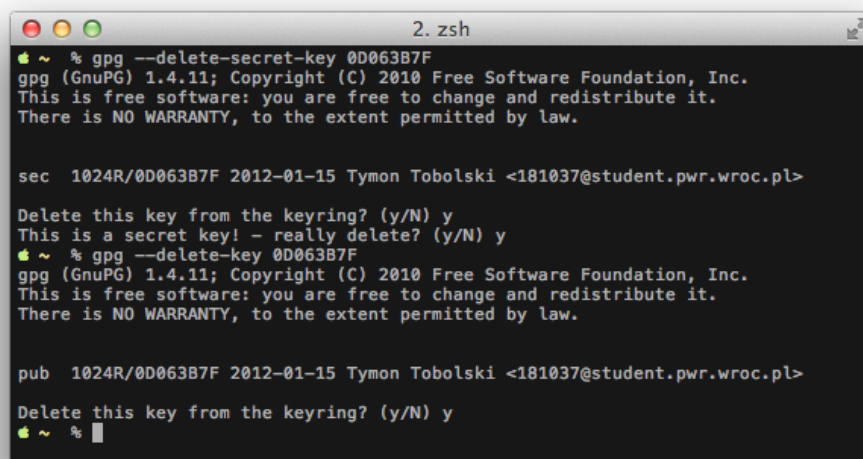
You need a passphrase to unlock the secret key for
user: "Tymon Tobolski <181037@student.pwr.wroc.pl>"
1024-bit RSA key, ID 28BB90AC, created 2012-01-15

pub 1024R/28BB90AC created: 2012-01-15 expires: 2012-03-15 usage: SC
    trust: ultimate validity: ultimate
sub 1024R/725432ED created: 2012-01-15 expires: 2012-01-19 usage: E
[ultimate] (1). Tymon Tobolski <181037@student.pwr.wroc.pl>

gpg> quit
Save changes? (y/N) y
~ %
```

Rysunek 3: Edycja klucza

W celu usunięcia klucza należy najpierw usunąć klucz prywatny przy pomocy polecenia `gpg --delete-secret-key NAZWA`, a następnie usunąć klucz publiczny za pomocą komendy `gpg --delete-key NAZWA`. Samo usunięcie klucza prywatnego nie powoduje skosowania użytkownika. Nie jest możliwe usunięcie tylko klucza publicznego.

A screenshot of a terminal window titled "2. zsh". The terminal shows the execution of two GPG commands to delete a key. The first command is `gpg --delete-secret-key 0D063B7F`, which prompts for confirmation to delete the secret key. The second command is `gpg --delete-key 0D063B7F`, which prompts for confirmation to delete the public key. Both prompts are answered with 'y'. The terminal output includes the GPG version (1.4.11) and a disclaimer about free software and warranty.

```
2. zsh
~ % gpg --delete-secret-key 0D063B7F
gpg (GnuPG) 1.4.11; Copyright (C) 2010 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.

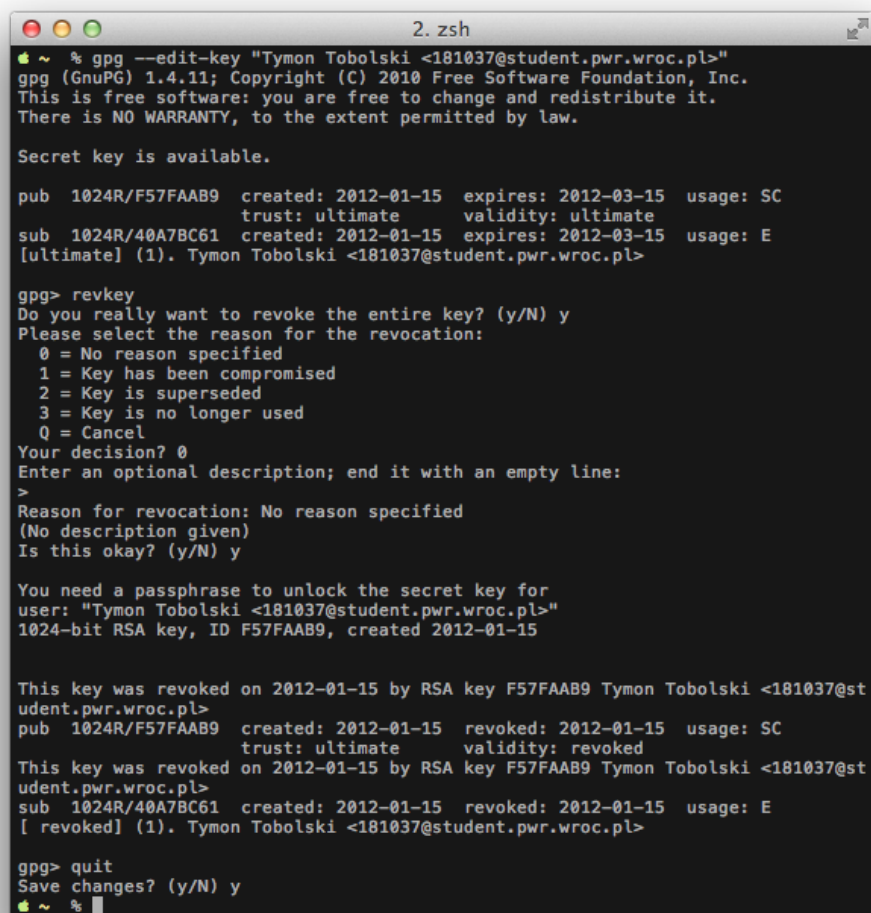
sec 1024R/0D063B7F 2012-01-15 Tymon Tobolski <181037@student.pwr.wroc.pl>
Delete this key from the keyring? (y/N) y
This is a secret key! - really delete? (y/N) y
~ % gpg --delete-key 0D063B7F
gpg (GnuPG) 1.4.11; Copyright (C) 2010 Free Software Foundation, Inc.
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.

pub 1024R/0D063B7F 2012-01-15 Tymon Tobolski <181037@student.pwr.wroc.pl>
Delete this key from the keyring? (y/N) y
~ %
```

Rysunek 4: Usunięcie klucza

4 Unieważnienie klucza

Unieważnienie klucza możliwe jest w trybie edycji za pomocą komendy `revkey`. Rysunek 5 przedstawia przykład unieważnienia klucza za pomocą programu `gpg`. Podczas procesu unieważnienia można opcjonalnie podać przyczynę oraz komentarz. Wymagane jest również podanie hasła zabezpieczającego dany klucz.



```
2. zsh
% gpg --edit-key "Tymon Tobolski <181037@student.pwr.wroc.pl>"
gpg (GnuPG) 1.4.11; Copyright (C) 2010 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.

Secret key is available.

pub 1024R/F57FAAB9 created: 2012-01-15 expires: 2012-03-15 usage: SC
trust: ultimate validity: ultimate
sub 1024R/40A7BC61 created: 2012-01-15 expires: 2012-03-15 usage: E
[ultimate] (1). Tymon Tobolski <181037@student.pwr.wroc.pl>

gpg> revkey
Do you really want to revoke the entire key? (y/N) y
Please select the reason for the revocation:
  0 = No reason specified
  1 = Key has been compromised
  2 = Key is superseded
  3 = Key is no longer used
  0 = Cancel
Your decision? 0
Enter an optional description; end it with an empty line:
>
Reason for revocation: No reason specified
(No description given)
Is this okay? (y/N) y

You need a passphrase to unlock the secret key for
user: "Tymon Tobolski <181037@student.pwr.wroc.pl>"
1024-bit RSA key, ID F57FAAB9, created 2012-01-15

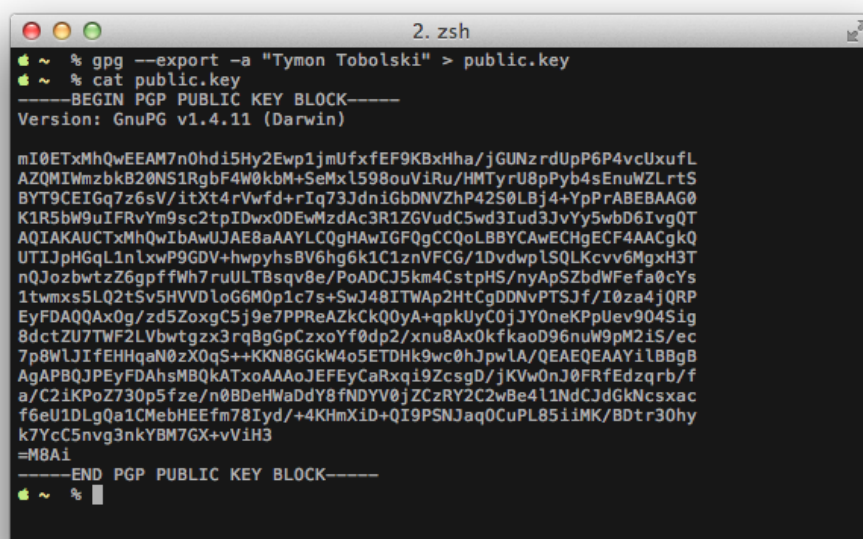
This key was revoked on 2012-01-15 by RSA key F57FAAB9 Tymon Tobolski <181037@student.pwr.wroc.pl>
pub 1024R/F57FAAB9 created: 2012-01-15 revoked: 2012-01-15 usage: SC
trust: ultimate validity: revoked
This key was revoked on 2012-01-15 by RSA key F57FAAB9 Tymon Tobolski <181037@student.pwr.wroc.pl>
sub 1024R/40A7BC61 created: 2012-01-15 revoked: 2012-01-15 usage: E
[revoked] (1). Tymon Tobolski <181037@student.pwr.wroc.pl>

gpg> quit
Save changes? (y/N) y
% 
```

Rysunek 5: Unieważnienie klucza

5 Eksport i import kluczy

Eksport klucza publicznego jak i prywatnego sprowadza się do wywołania 2 komend: `gpg --export -a NAZWA` dla klucza publicznego oraz `gpg --export-secret-key -a` dla klucza prywatnego. Rysunki 6 i 7 pokazują wynik działania tych komend.



```
2. zsh
% gpg --export -a "Tymon Tobolski" > public.key
% cat public.key
-----BEGIN PGP PUBLIC KEY BLOCK-----
Version: GnuPG v1.4.11 (Darwin)

mI0ETxMhQwEEAM7n0hdi5Hy2Ewp1jmUfxfEF9KBxHha/jGUNzrdUpP6P4vcUxufL
AZQMIWmzbk820NS1RgBF4W0kbM+SeMxL598ouViRu/HMTyrU8pPyb4sEnuWZLrtS
BYT9CEIGq7z6sV/itXt4rVwfd+rIq73JdniGbDNVZhP42S0LBj4+YpPrABEBAAG0
K1R5bW9uIFRvYm9sc2tpIDwxODEwMzdAc3R1ZGVudC5wd3Iud3JvYy5wbD6IvgQT
AQIAKAUCxMhQwIbAwUJAE8aAAYLCQgHAWIGF0gCCQoLBBYCAwECHgECF4AACgkQ
UTIJpHGqL1nLxwP9GDV+hwpyhsBV6hg6k1C1znVFCG/1Dvdwpl5QLKcvv6MgxH3T
nQJozbwtzZ6gpffWh7ruULTBsqv8e/PoADCJ5km4CstpHS/nyApSZbdWFefa0cYs
1twmxs5LQ2tSv5HVVDloG6M0p1c7s+SwJ48ITWAp2HtCgDDNvPTSJf/I0za4jQRP
EyFDAQQAxOg/zd5ZoxgC5j9e7PPReAZkCkQ0yA+qpkUyC0jJY0neKPPUev904Sig
8dctZU7TWF2LVbwtgzx3rqBgGpCzxoYf0dp2/xnu8Ax0kfkaoD96nuW9pM2iS/ec
7p8WLJI fEHHqaN0zX0qS++KKN8GGk4o5ETDhk9wc0hJpwLA/QEAEQEAAyilBBgB
AgAPBQJPEyFDaHsMBQkATxoAAAJEFfEYCaRxqi9ZcsgD/jKVw0nJ0FRfEdzqr/f
a/C2iKPoZ730p5fze/n0BDeHWaDdY8fNDYV0jZCzRY2C2wBe4l1NdCJdGkNcsxac
f6eU1DLgQa1CMebHEEfm78Iyd/+4KHmXiD+QI9PSNJaQ0CuPL85iiMK/BDtr30hy
k7YcC5nvg3nkYBM7GX+vViH3
=M8Ai
-----END PGP PUBLIC KEY BLOCK-----
```

Rysunek 6: Eksport klucza publicznego do pliku

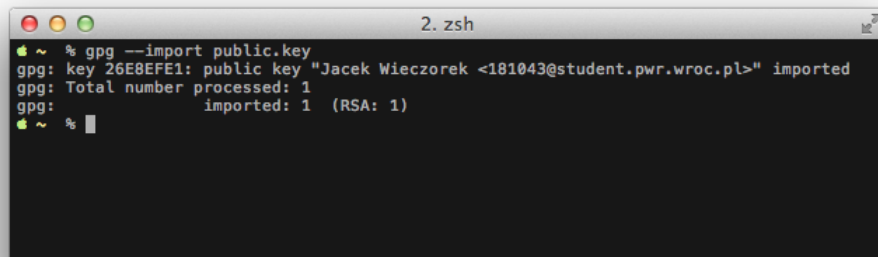

```
2. zsh
% gpg --export-secret-key -a "Tymon Tobolski" > private.key
% cat private.key
-----BEGIN PGP PRIVATE KEY BLOCK-----
Version: GnuPG v1.4.11 (Darwin)

lQH+BE8TIUMBBAD05zoXYuR8thMKdY5LH8XxBfSgcR4Wv4x1Dc63VKT+j+L3FMbn
ywGUDCFps25AdtDUtUYGxeFtJGzPknjMZeFFKLlYkbvxzE8q1PKT8m+LBJ7lmS67
UgWE/QhCBqu8+rFf4rV7eK1cH3fgyKu9yXZ4hmwzVWYT+NktCwY+PmKT6wARAQAB
/gMDAgAvRNJA8/QYNm6KguJswQ0gSM+UTWjL2akwf2fJ83mWQtjMe1eM3iExpkh
AWKAa3VcAW8dHRQ9zecCeTiFLdHSDgbE+1lg3S0CMwldD3pZYrWd20XQkPoiSnIR
1g57KQZXS5yXys/m3mX/MOTZijCACqMtQfWAmMTQdsjWk/jZwJg1u5VIL0tpP2neF
xNL/bLEQSt0TPYZQueFoZMqLejI3+98DUkIiTNPV1M2J15oizYG1zWCMGbaen078
Jovm0oo1x+qGAZ6bghM+4LfQq+0s1WEe4WlhP9k2C8EioTLNV+mnJII8aab4FHW0
RceyrRFA8LN5VF1GnoCxAw/h6Ge09hugScvMs+MBWfnz/maWRgcsIK0Jpo0FQIJK
BYkbhuRMM1aJ2celY1tt8YBfLKK5Z/Sp0SvWckYoE4sRipc4EURwnhf0mGW0rhk
T38ZDthy18CutyDwiIFdkYqYnBZVIyZKb85GZLkIBRjytCtUeW1vb1BUb2JvbHNR
aSA8MTgxMDM3QHNDdWRlbnQucHdyLndyb2MucGw+il4EEwECACgFAK8TIUMCGwMF
CQBPGAGCwkIBwMCBhUIAgKCCwQWAgMBAh4BAheAAAJEFeyCaRxi9Z5ccD/Rg1
focKcobAVeoY0pN0tc51RQhv9Q73cKZUKCynL7+jIMR9050CaM28Lc2eoKX31oe6
71C0wbKr/Hvz6AAwieZJuArLaR0v58gKUmW3VhXn2tHGLNbcJsb0S0NrUr+R1VQ5
aBuJdQdX07PksCePCE1gKdh7QoAwzbz00iX/yNM2nQH9BE8TIUMBBAD6D/N3lmj
GALmP17s89F4BmQKRA7ID6qmRTII6Mlg6d4o+LR6/07hKKDx1y1LTtNYXYtVvC2D
PHeuoGAakLPgHh/R2nb/Ge7wDE6R+RqgP3qe5b2kzaJL95zuxxaUkh8Qcepo3Tnc
6pL74oo3wYaRbijjKRMMeT3BzSEmnCU09AQAQAB/gMDAgAvRNJA8/QYJ9h3/IJ
j4IHcNTdMGz3CX56TYhux7CWvqhybKmsCqFrByiFQ6D45tb/1hcfP0pMJKrR5YtW
WJ5sn3RdCwZYVv+asVhjHv3N+MiiFwUeMGqyQSDfAtk2Hbzh7ug/USIme7C1Bw8
bT/PWA6WE1C8ZXA4vp605tIm7iF8qg+czWWi0Uc4DZXK0jJLzyQDZP6FuEazzie+
ljTQ2Q2Mb9Q/tDOW/rdMXwA0R5svhe7FrgVt1QR6msCXJEE6Dfm6ViiZzockcrJcQ
HaTn97Smpb0S483AsoQzUvivIrBRtL4sSMTY4m9ieS06BDQLE1P7hrFreXDeUxjQ
6rouGxMFVvk4gwk61Rcdvmg2X7/8zx+LuE7QeaPtSfSItexTH7vz+ZJnfv+1KKAnP
cEtXhs2roi6ZIBG9QQR8Dcd7Ar9LGF1yzF0Uo7wCQ2T4Mu6CN8h986s5HzEMY0y2
XG6u4YvcKR9X1UcH6pyIpQYQAQIADwUCTxMhQwIbDAUJAE8aAAKCRBRMgmkaov
WXLIA/4ylcdpydBUXxHc6q2/32vwt0ij6Ge9zqeX83v59AQ3h1mg3WPHzQ2FdI2Q
s0WNgtsAXuJdTXQIXRpDXLMWnH+nLQy4EGtQjHmxxBH5u/CMnf/uCh5l4g/kCPT
0jSqwjgrjy/0YojCvwQ7a9zocp02HAUZ74N55GAT0xl/r1Yh9w==
=ht/A
-----END PGP PRIVATE KEY BLOCK-----
```

Rysunek 7: Eksport klucza prywatnego do pliku

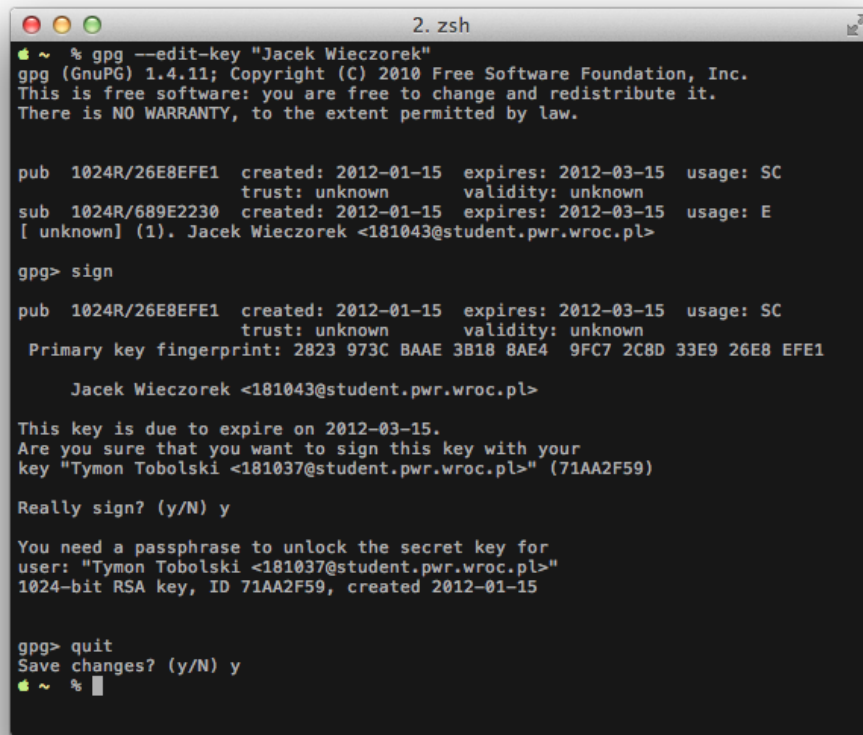
Po wyeksportowaniu klucza publicznego do pliku, można go przesłać innemu użytkownikowi. Może on wtedy zaimportować klucz i podpisać go za pomocą swojego własnego. W celu podpisania klucza przy użyciu programu gpg należy wejść w tryb edycji klucza, a następnie wywołać polecenie **sign**. Następnie należy podać hasło zabezpieczające klucz, którym podpisany będzie klucz. Rysunki 8 i 9 przedstawiają operacje importu oraz podpisywania klucza.

Budowanie sieci zaufania może okazać się czasochłonne, ze względu na konieczność wymiany kluczy między użytkownikami.

A terminal window titled "2. zsh" with standard macOS window controls (red, yellow, green buttons). The terminal shows a command prompt with a green cursor. The user enters the command "gpg --import public.key". The output shows that a public key with ID 26E8EFE1 for "Jacek Wieczorek" was imported successfully. The terminal also shows the total number of keys processed (1) and imported (1), along with the key type (RSA: 1).

```
2. zsh
~ % gpg --import public.key
gpg: key 26E8EFE1: public key "Jacek Wieczorek <181043@student.pwr.wroc.pl>" imported
gpg: Total number processed: 1
gpg:      imported: 1 (RSA: 1)
~ %
```

Rysunek 8: Import klucza z pliku

A screenshot of a terminal window titled '2. zsh'. The user enters the command 'gpg --edit-key "Jacek Wieczorek"'. The terminal displays GPG version 1.4.11 and a disclaimer. It then shows details for a public key (1024R/26E8EFE1) and a subkey (1024R/689E2230). The user enters 'sign'. The terminal shows the primary key fingerprint (2823 973C BAAE 3B18 8AE4 9FC7 2C8D 33E9 26E8 EFE1) and asks for confirmation to sign. The user enters 'y'. The terminal then asks for a passphrase to unlock a secret key for 'Tymon Tobolski'. The user enters a passphrase. The terminal then asks 'Really sign? (y/N)' and the user enters 'y'. Finally, the user enters 'quit' and the terminal asks 'Save changes? (y/N)' where the user enters 'y'.

```
2. zsh
~ % gpg --edit-key "Jacek Wieczorek"
gpg (GnuPG) 1.4.11; Copyright (C) 2010 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.

pub 1024R/26E8EFE1  created: 2012-01-15  expires: 2012-03-15  usage: SC
                        trust: unknown    validity: unknown
sub 1024R/689E2230  created: 2012-01-15  expires: 2012-03-15  usage: E
[ unknown] (1). Jacek Wieczorek <181043@student.pwr.wroc.pl>

gpg> sign

pub 1024R/26E8EFE1  created: 2012-01-15  expires: 2012-03-15  usage: SC
                        trust: unknown    validity: unknown
Primary key fingerprint: 2823 973C BAAE 3B18 8AE4 9FC7 2C8D 33E9 26E8 EFE1

Jacek Wieczorek <181043@student.pwr.wroc.pl>

This key is due to expire on 2012-03-15.
Are you sure that you want to sign this key with your
key "Tymon Tobolski <181037@student.pwr.wroc.pl>" (71AA2F59)

Really sign? (y/N) y

You need a passphrase to unlock the secret key for
user: "Tymon Tobolski <181037@student.pwr.wroc.pl>"
1024-bit RSA key, ID 71AA2F59, created 2012-01-15

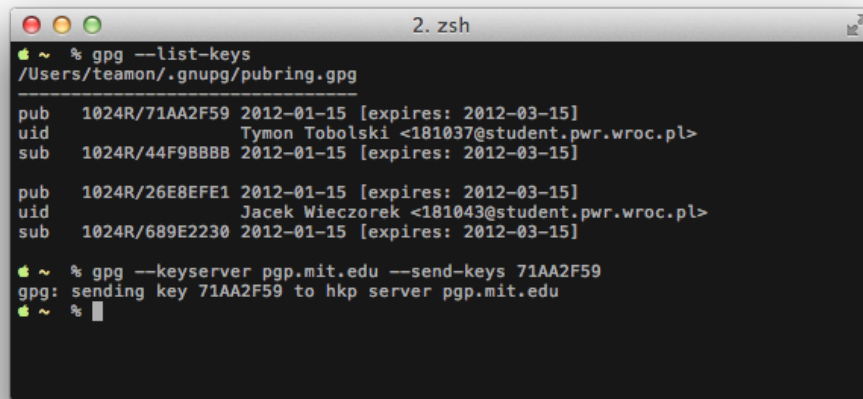
gpg> quit
Save changes? (y/N) y
~ %
```

Rysunek 9: Podpisanie klucza

6 Wykorzystanie serwerów kluczy

Wysłanie klucza na serwer wymaga podania adresu serwera, odbywa się za pomocą komendy `gpg --keyserver SERWER --send-keys ID_KLUCZA`. Rysunek 10 przedstawia proces wysłania klucza na serwer `pgp.mit.edu`.

Serwer kluczy udostępnia możliwość wyszukiwania kluczy na dwa sposoby: za pomocą identyfikatora klucza (`gpg --keyserver SERWER --search-key ID_KLUCZA`) lub poprzez adres email (`gpg --keyserver SERWER --search-key EMAIL`). Rysunek 11 przedstawia obie metody wyszukiwania klucza na serwerze `pgp.mit.edu`.

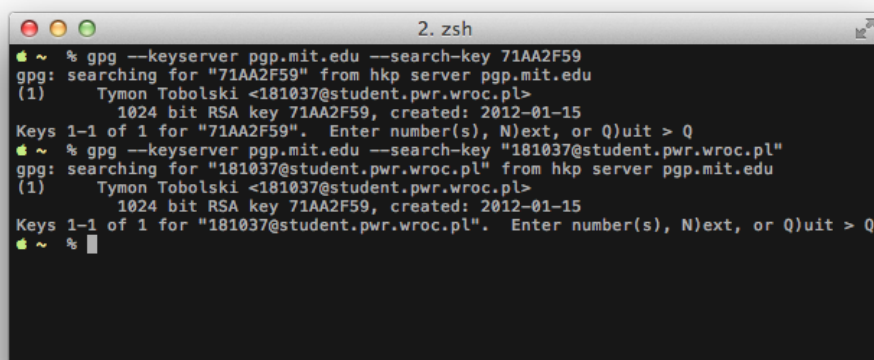


```
2. zsh
% gpg --list-keys
/Users/teamon/.gnupg/pubring.gpg
-----
pub 1024R/71AA2F59 2012-01-15 [expires: 2012-03-15]
uid Tymon Tobolski <181037@student.pwr.wroc.pl>
sub 1024R/44F9B8BB 2012-01-15 [expires: 2012-03-15]

pub 1024R/26E8EFE1 2012-01-15 [expires: 2012-03-15]
uid Jacek Wieczorek <181043@student.pwr.wroc.pl>
sub 1024R/689E2230 2012-01-15 [expires: 2012-03-15]

% gpg --keyserver pgp.mit.edu --send-keys 71AA2F59
gpg: sending key 71AA2F59 to hkp server pgp.mit.edu
% 
```

Rysunek 10: Przesłanie klucza na serwer pgp.mit.edu



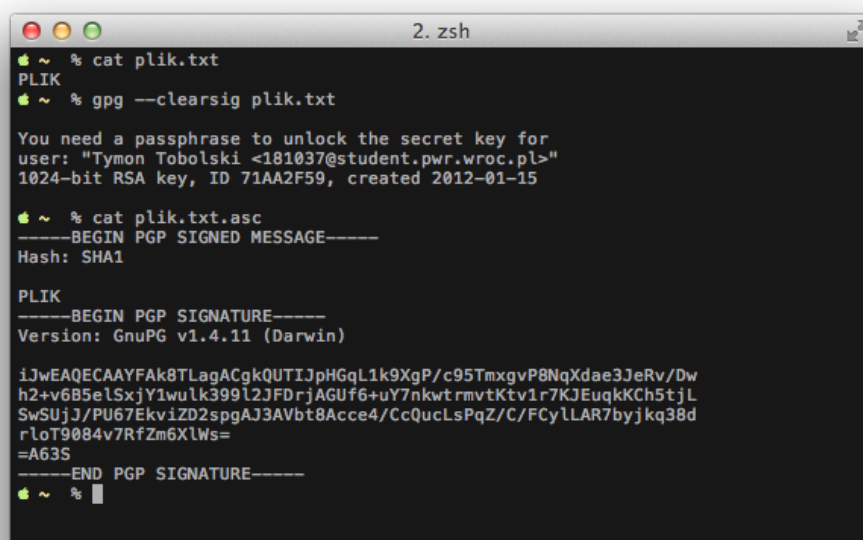
```
2. zsh
% gpg --keyserver pgp.mit.edu --search-key 71AA2F59
gpg: searching for "71AA2F59" from hkp server pgp.mit.edu
(1) Tymon Tobolski <181037@student.pwr.wroc.pl>
1024 bit RSA key 71AA2F59, created: 2012-01-15
Keys 1-1 of 1 for "71AA2F59". Enter number(s), N)ext, or Q)uit > Q
% gpg --keyserver pgp.mit.edu --search-key "181037@student.pwr.wroc.pl"
gpg: searching for "181037@student.pwr.wroc.pl" from hkp server pgp.mit.edu
(1) Tymon Tobolski <181037@student.pwr.wroc.pl>
1024 bit RSA key 71AA2F59, created: 2012-01-15
Keys 1-1 of 1 for "181037@student.pwr.wroc.pl". Enter number(s), N)ext, or Q)uit > Q
% 
```

Rysunek 11: Wyszukiwanie klucza na serwerze pgp.mit.edu

Serwery kluczy znacznie usprawniają wymianę kluczy między użytkownikami. Każdy może zapytać serwer o klucz publiczny podanego adresu email, nie ma potrzeby wysyłania go do każdego użytkownika z osobna.

7 Szyfrowanie, deszyfrowanie, podpisywanie i weryfikacja podpisu plików

Podpisywanie plików odbywa się za pomocą komendy `gpg --clearsign PLIK`. Podpis pliku wymaga podania hasła zabezpieczającego klucz. Po podpisaniu pliku zostaje utworzony plik z podpisem o rozszerzeniu `.asc`. Przykład takiego podpisu prezentuje Rysunek 12.



```
2. zsh
% cat plik.txt
PLIK
% gpg --clearsig plik.txt

You need a passphrase to unlock the secret key for
user: "Tymon Tobolski <181037@student.pwr.wroc.pl>"
1024-bit RSA key, ID 71AA2F59, created 2012-01-15

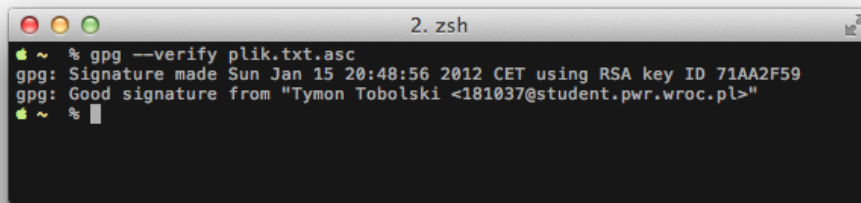
% cat plik.txt.asc
-----BEGIN PGP SIGNED MESSAGE-----
Hash: SHA1

PLIK
-----BEGIN PGP SIGNATURE-----
Version: GnuPG v1.4.11 (Darwin)

iJwEAQECAAYFAk8TLagACgkQUtIJpHGqL1k9XgP/c95TmxgvP8NqXdae3JeRv/Dw
h2+v6B5eISxjY1wulK399l2JFDrjAGUf6+uY7nkwtrmvtKtv1r7KJEUqkKCh5tjL
SwSUjJ/PU67EkviZD2spgAJ3AVbt8Acce4/CcQuLsPqZ/C/FCylLAR7byjkq38d
rloT9084v7RfZm6XlWs=
=A63S
-----END PGP SIGNATURE-----
% 
```

Rysunek 12: Podpisywanie pliku

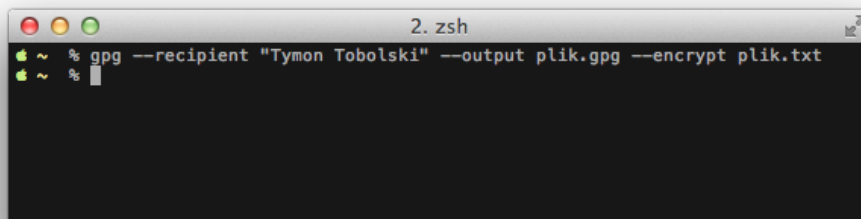
Weryfikacji podpisu można dokonać przy pomocy komendy `gpg --verify PLIK_ASC`. Przykładowa operacja zaprezentowana jest na Rysunku 13.

A terminal window titled "2. zsh" with a dark background. It shows the command `gpg --verify plik.txt.asc` and its output: `gpg: Signature made Sun Jan 15 20:48:56 2012 CET using RSA key ID 71AA2F59` and `gpg: Good signature from "Tymon Tobolski <181037@student.pwr.wroc.pl>"`.

```
2. zsh
% gpg --verify plik.txt.asc
gpg: Signature made Sun Jan 15 20:48:56 2012 CET using RSA key ID 71AA2F59
gpg: Good signature from "Tymon Tobolski <181037@student.pwr.wroc.pl>"
% 
```

Rysunek 13: Weryfikacja podpisu pliku

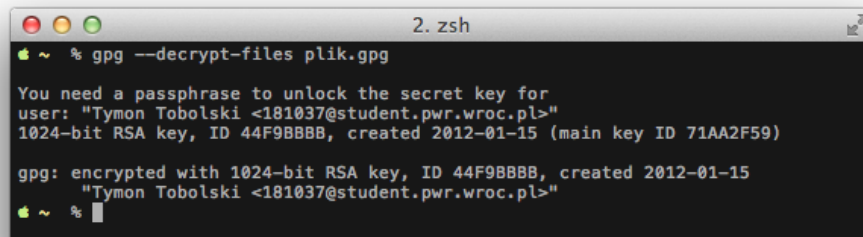
W celu zaszyfrowania wiadomości należy wywołać komendę `gpg --recipient ODBIORCA --output PLIK_GPG --encrypt PLIK`. Po zaszyfrowaniu powstaje plik `.gpg`. Przykład szyfrowania wiadomości znajduje się na Rysunku 14.

A terminal window titled "2. zsh" with a dark background. It shows the command `gpg --recipient "Tymon Tobolski" --output plik.gpg --encrypt plik.txt`.

```
2. zsh
% gpg --recipient "Tymon Tobolski" --output plik.gpg --encrypt plik.txt
% 
```

Rysunek 14: Szyfrowanie wiadomości

Deszyfracja wiadomości sprowadza się do wykonania komendy `gpg --decrypt-files PLIK_GPG`. Ta operacja wymaga podania hasła zabezpieczającego klucz. Przykład użycia obrazuje Rysunek 15.

A terminal window titled "2. zsh" with standard macOS window controls (red, yellow, green buttons) in the top-left corner. The terminal shows a command prompt where the user has entered "gpg --decrypt-files plik.gpg". The output of the command is displayed in white text on a dark background. It starts with a prompt "You need a passphrase to unlock the secret key for" followed by user information: "user: 'Tymon Tobolski <181037@student.pwr.wroc.pl>'". Then it shows key details: "1024-bit RSA key, ID 44F98888, created 2012-01-15 (main key ID 71AA2F59)". Next, it states "gpg: encrypted with 1024-bit RSA key, ID 44F98888, created 2012-01-15" and finally the decrypted content: "'Tymon Tobolski <181037@student.pwr.wroc.pl>'". The prompt returns to the shell as "gpg ~ %".

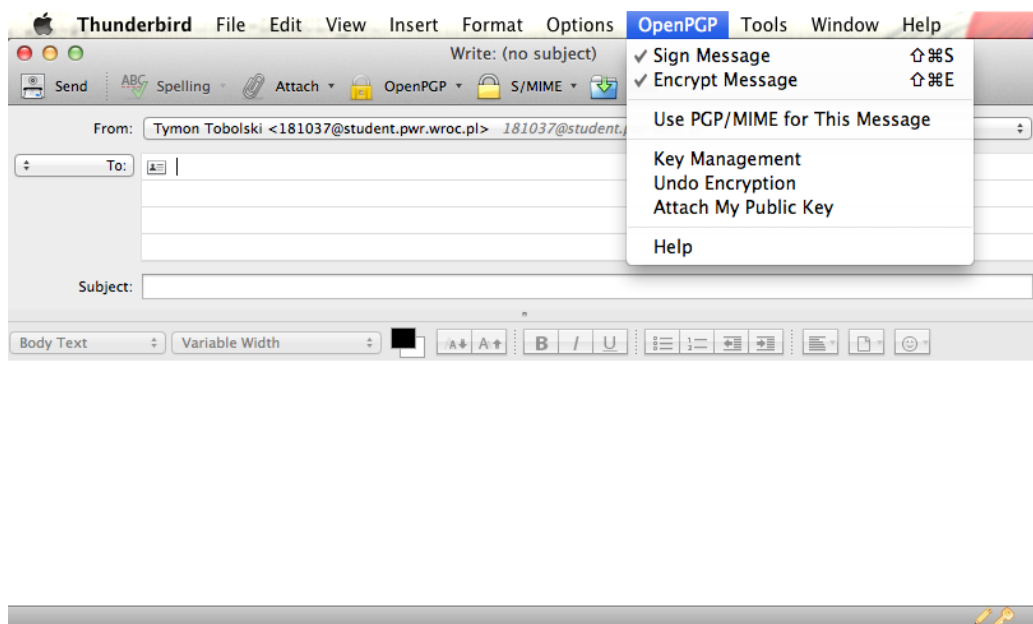
```
2. zsh
gpg ~ % gpg --decrypt-files plik.gpg
You need a passphrase to unlock the secret key for
user: "Tymon Tobolski <181037@student.pwr.wroc.pl>"
1024-bit RSA key, ID 44F98888, created 2012-01-15 (main key ID 71AA2F59)

gpg: encrypted with 1024-bit RSA key, ID 44F98888, created 2012-01-15
"Tymon Tobolski <181037@student.pwr.wroc.pl>"
gpg ~ %
```

Rysunek 15: Szyfrowanie wiadomości

8 Szyfrowanie, deszyfrowanie, podpisywanie i weryfikacja podpisu e-mail

W celu podpisania oraz szyfrowania wiadomości e-mail w programie Mozilla Thunderbird wystarczy wybrać odpowiednią opcję z menu OpenPGP podczas tworzenia nowej wiadomości. Umiejscowienie wspomnianych opcji prezentuje Rysunek 16.



Rysunek 16: Podpisywanie oraz szyfrowanie nowej wiadomości

Program pocztowy Mozilla Thunderbird umożliwia także automatyczną weryfikację tożsamości nadawcy oraz deszyfrację wiadomości przychodzących.

9 Różnice między PGP/inline a PGP/MIME

Domyślną metodą załączania zaszyfrowanej wiadomości jest PGP/inline, w której to informację o użytym kluczu są przechowywane wewnątrz wiadomości. Wadą tej metody są problemy z szyfrowaniem niestandardowych znaków (spoza tablicy ASCII) oraz trudności z szyfrowaniem załączników. Ponadto programy pocztowe, które nie wspierają PGP wyświetlają wiadomości jako niezrozumiały tekst. Metoda PGP/MIME rozwiązuje powyższe problemy przenosząc informacje o kluczu do nagłówka wiadomości.

10 Wnioski

Podpisywanie i szyfrowanie plików oraz wiadomości e-mail pozwala na zapobieganie przedostania się poufnych informacji do osób nieupoważnionych. Nowoczesne programy pocztowe wspierają obsługę OpenPGP, dlatego też generowanie kluczy oraz zabezpieczanie wiadomości nie jest procesem skomplikowanych dla użytkownika.