KeyBone Documentation

Release 0.3.2

The World

Nov 05, 2016

Contents

1 The command-line client **3**
- 1.1 Installation . . . 3
- 1.2 Initialize your GPG key home . . . 3
- 1.3 Create a GPG key pair . . . 3
- 1.4 Importing a key . . . 4
- 1.5 Delete a key . . . 4
- 1.6 Listing keys . . . 4
- 1.7 Sign data . . . 5
- 1.8 Verify Signatures . . . 6
- 1.9 Encrypt data to a known recipient . . . 6
- 1.10 Decrypt data if you have a private key in the keyring . . . 7
- 1.11 Export a private key . . . 8
- 1.12 Export a public key . . . 8
- 1.13 Backup your keyring for emergencies . . . 9
- 1.14 Wipe your keyring . . . 9
- 1.15 Recover from a backup . . . 10

2 Internals Reference **11**

3 Indices and tables **13**

4 Indices and tables **15**

Python Module Index **17**

Contents:

CHAPTER 1

The command-line client

1.1 Installation

```
$ pip install keybone
```

1.2 Initialize your GPG key home

initializes a new keybone key home, which includes generating a fernet key to encrypt the key metadata

```
usage: keybone quickstart [-h] [-f] [--home HOME] conf_path
```

Positional arguments:

conf_path	the path to the config file

Options:

-f=False, --force=False	flag to force even if destination exists
--home=keys	the path to the key home. Defaults to a "keys" folder relative to the config file path

```
$ keybone quickstart ~/.keybone.yml --home=~/.keybone
-----------------------------------------------------------
fernet_key: d68R6KPnWx6Izv10y8lbJ7mDvw_0OP0c6arlT4d1RHA=
key_home: ~/.keybone
-----------------------------------------------------------
you might want do add the following line to your ~/.bashrc
export KEYBONE_CONFIG_PATH='/home/sn0wden/.keybone.yml'
```

1.3 Create a GPG key pair

generate a key if it does not exist already.

```
usage: keybone create [-h] [--secret <passphrase>] [-n] <name> <email>
```

Positional arguments:

name	the user name

email the user email

Options:

--secret a passphrase. If not provided it will be asked with getpass

-n=False, --no-secret=False create without password

```
$ keybone create "Edward Snowden" "snowden@protonmail.com" --secret="freed0m"
WARNING changing mode of /home/sn0wden/.keybone to 0700
INFO Generated: 3B9D624364FF560192B4136F209C5ABEBE490B74
```

1.4 Importing a key

imports a key

```
usage: keybone import [-h] <key>
```

Positional arguments:

key as string, with linebreaks

```
$ keybone import "$(curl 'http://pgp.surfnet.nl:11371/pks/lookup?op=get&
↪search=0x00411886')"
INFO imported: ABAF11C65A2970B130ABE3C479BE3E4300411886
INFO imported: 0F6A146532D869AEE438F74B6211AA3B00411886
```

1.5 Delete a key

delete a key if it does not exist already.

```
usage: keybone delete [-h] [-f] <recipient>
```

Positional arguments:

recipient the recipient to encrypt the data to

Options:

-f=False, --force=False delete without confirmation

```
$ keybone delete ABAF11C65A2970B130ABE3C479BE3E4300411886
deleting Linus Torvalds torvalds@linux-foundation.org 79BE3E4300411886␣
↪ABAF11C65A2970B130ABE3C479BE3E4300411886
ok
```

1.6 Listing keys

lists existing keys

```
usage: keybone list [-h] [--email] [--private] [--public]
```

Options:

--email=False only print the email of the keys

--private=False show only private keys

--public=False show only public keys

```
$ keybone list
+------------------+------------------------------------------+-----------------------
 ↪--------+---------+--------+
|      keyid       |               fingerprint                |             email     ␣
 ↪        | private | public |
+------------------+------------------------------------------+-----------------------
 ↪--------+---------+--------+
| 209C5ABEBE490B74 | 3B9D624364FF560192B4136F209C5ABEBE490B74 |     ␣
 ↪snowden@protonmail.com    |   yes   |  yes   |
| 79BE3E4300411886 | ABAF11C65A2970B130ABE3C479BE3E4300411886 | torvalds@linux-
 ↪foundation.org |    no   |  yes   |
| 6211AA3B00411886 | 0F6A146532D869AEE438F74B6211AA3B00411886 | torvalds@linux-
 ↪foundation.org |    no   |  yes   |
+------------------+------------------------------------------+-----------------------
 ↪--------+---------+--------+
```

1.7 Sign data

signs the given data for the given keyid

```
usage: keybone sign [-h] [--secret <passphrase>] [--no-secret]
                    <recipient> <data>
```

Positional arguments:

recipient any identification for the key: fingerprint, id or email

data the content to be signed

Options:

--secret a passphrase if necesssary

--no-secret=False set this flag if the destination key doesn not have a secret

```
$ keybone sign 3B9D624364FF560192B4136F209C5ABEBE490B74 'This is really mine'
-----BEGIN PGP SIGNED MESSAGE-----
Hash: SHA1

This is really mine
-----BEGIN PGP SIGNATURE-----
Version: GnuPG v1

iQEcBAEBAgAGBQJX5hVdAAoJECCcWr6+SQt0OX8IAJDGkswhqTGjqmhl9oh0wB7o
8JjhzM8c59mrQkw2uVycUQP8SvRSDsbKh7oKeQmruDszWvbOZOahqtn6w4lZf9og
tOdkrt1aERC4iD2Z+W87kQRrbqEQTh0QovsKe40rzRMkk0PftBX1Mh7zmTx9sP84
2XUkHyOkNa932Zh2pmyJTMQbQfBaL5B9AnmdgxJCwJ1GsteauGffNHLlMpv90Yf5
qgeZLfd2Kyswfe14JMRCRM6o1krS23lCuoqZM6aqeuWLlDOnBNHwPuVTvGo5xtvk
Q16NTZzw5hMnntIV9CFO3Ss8GpVOBv1RupTsj5mpFOnE3EBX0z7kcbXcxxBIrYI=
=pxA0
-----END PGP SIGNATURE-----
```

1.8 Verify Signatures

verifies the given data for the given keyid

```
usage: keybone verify [-h] <signed-data>
```

Positional arguments:

signed_data the signed data to be verified

```
$ keybone verify '-----BEGIN PGP SIGNED MESSAGE-----
Hash: SHA1

This is really mine
-----BEGIN PGP SIGNATURE-----
Version: GnuPG v1

iQEcBAEBAgAGBQJX5hVdAAoJECCcWr6+SQt0OX8IAJDGkswhqTGjqmhl9oh0wB7o
8JjhzM8c59mrQkw2uVycUQP8SvRSDsbKh7oKeQmruDszWvbOZOahqtn6w4lZf9og
tOdkrt1aERC4iD2Z+W87kQRrbqEQTh0QovsKe40rzRMkk0PftBX1Mh7zmTx9sP84
2XUkHyOkNa932Zh2pmyJTMQbQfBaL5B9AnmdgxJCwJ1GsteauGffNHLlMpv90Yf5
qgeZLfd2Kyswfe14JMRCRM6o1krS23lCuoqZM6aqeuWLlDOnBNHwPuVTvGo5xtvk
Q16NTZzw5hMnntIV9CFO3Ss8GpVOBv1RupTsj5mpFOnE3EBX0z7kcbXcxxBIrYI=
=pxA0
-----END PGP SIGNATURE-----'
signature valid: TRUST_UNDEFINED
```

1.9 Encrypt data to a known recipient

encrypts the data to a known recipient

```
usage: keybone encrypt [-h] [--sign-from <sign-from-recipient>]
                       <recipient> <plaintext>
```

Positional arguments:

recipient the recipient to encrypt the data to

plaintext the content to be encrypted

Options:

--sign-from sign from this recipient

```
$ keybone encrypt ABAF11C65A2970B130ABE3C479BE3E4300411886 'Hey Torvalds,
it seems like someone exploited short-id collisions on PGP keys
and could be pushing malware to the kernel mainstream

Check this out:

 Search Result of 0x00411886: https://pgp.mit.edu/pks/lookup?search=0x00411886&
↪op=index
 Fake Linus Torvalds: 0F6A 1465 32D8 69AE E438  F74B 6211 AA3B [0041 1886]
 Real Linus Torvalds: ABAF 11C6 5A29 70B1 30AB  E3C4 79BE 3E43 [0041 1886]

Cheers,
Sn0wden'
```

```
-----BEGIN PGP MESSAGE-----
Version: GnuPG v1

hQEMA4i86A8BL1TKAQf/cII0rdg02b/uaSuMlOd5om1H6LhlcBSnUsO6b3O9eom4
+rjOcn634opAo5L1YgtlmI0Nh9nflQWhFW8kj0Do6oy4NC4jar92YrlVsB/PGbdU
xHY0YhXKcqJn0xPRB/FWRK+eup2fwcQnJRKHkT9t2cIZu1kre19NiNAd5pciWv3D
TAIliMAoloUwwz7ZNH08aWEWTxUSeIY2EzOo9UigZon0FD5GKGUi8dGXhxh90M8V
bur5ETmRnih7PR1IUF6GdvnnvcliDU5YiqgVtNx61oe/8wKVYflEfar04GO5kYfH
ISWur5FhtDyov8Q8pacKhlyrPJ9MFZRfJJxfgzsUddLBDwFiQ5rmCQdb8Ya6uXbz
g+bYORMltOUbfBxZRgLQLqFeKvXJ4MpOclWiDasUqt+QcD89Ow9vAjRcHJPAC3ha
4trESQya7tq7BaVMeaAfUSa5JY8aMraBUorX6Oh+l34UATUxJszfJ/+qGKOyv/cY
mr/307O2Zp7By13nyYKfuzDZXaKKyYlm2YydZW4ZHB/2FXhB6o2CZ46B1xfjiT1e
8gEPQo0YWNINybbIMDV4v5hamqcbPo8OuP6Jy3w/tACWf0YC6nRmKyBwtnDY3R4T
Rp8WsM015WuNFxpilEEl4D7mqrHpMO48BgxcHgIbK1lfY2HMSzJ3yRI7DHG0lFXl
1lIFpvD5C1tVUFw2/yYAPNohTxBEWXhYyK5Q6iZnl6e1/h4ErlrQ5DMWGnzgksQP
SqqSRsRsO8sbQ2tgQsIeO96Wsl6cAlG13NxmDHQgHkIeAM6JlMysCo/fW3fBcCBP
Hdguj5KUi+58FIrH4H2CvF55XDyj0LioEqzFGF801i9TeKOiLdMrHWXsWBnoiaZw
0M6eJYeVokdhLghvecjwR1oGVHv5GwCi7TfZQuSDuMLtPWnGhiqTplLtEfsv/Z/u
xUj3Y33RILIdQcW4pt8dQfQ=
=ZXQp
-----END PGP MESSAGE-----
```

1.10 Decrypt data if you have a private key in the keyring

decrypts the data to a known recipient

```
usage: keybone decrypt [-h] [--secret <passphrase>] [-n] <ciphertext>
```

Positional arguments:

ciphertext	the content to be decrypted

Options:

--secret	a passphrase
-n=False, --no-secret=False	create without password

```
keybone decrypt --secret='freed0m' '-----BEGIN PGP MESSAGE-----
  Version: GnuPG v1

  hQEMAyCcWr6+SQt0AQf/SIiYdqvXSDyeY1sNO8wiTGKf+c43BR4zyzULNjWrlbkt
  jic2z2wsYbnKZvRusLo6U9GxlTXTahTdXPwf0FnuUyH5RR4tU6Q71KoyzHAQQM8i
  xEGRTxoJqdCZTbF5s6wL0Nyyv7JlJkHlI1l8BLQ8igmIUPNeBAkRiknRkenPvcmk
  1r2jisdpwPS5OVzKWUAUuv/Z8MkQvR7bzDxdHDqkT6bM+LoyyrZyvy+xXvmAAWfO
  N8Q8sxRia8yEu9Z0zaSsG7cZxHOdF9oIksMFcnq94FzveiAl1/c8CJ53PhgaAi4W
  hzL27zU+rlPuzy9F7AtMoFidCicT0ui3FrI1eSaTTNJSAXqoPD1FbRVof25X/FfR
  LmVBKOdO6bmPicrZKuGFw9IYqho8GL1N3fK6aWdiJOPdJTb4z7cYNd4yiGRLwanF
  1v1zidzdz4pIYQuUb4KEtIo8tg==
  =M4aT
  -----END PGP MESSAGE-----'
Wow,
this really works!
```

1.11 Export a private key

prints the private key of the given email

```
usage: keybone private [-h] <recipient>
```

Positional arguments:

recipient any identification for the key: fingerprint, id or email

```
$ keybone private 3B9D624364FF560192B4136F209C5ABEBE490B74
-----BEGIN PGP PRIVATE KEY BLOCK-----
Version: GnuPG v1

lQO+BFfl2CwBCADhQwj6dDbUe0eTVpy/goQib+02g9D+J7BX9+Q3LqQ4z74fmsBq
FdEoc8DA9fCYxNIMqd+oLl90m6Ur6OVwXj7RgtvXyyGJLLEDtJCspR1a/aFRFdLu
+9qYAkXdJO5PMrBvyLbriGcLYfT6hQibEh9W+DlqsfVJycSPOAsLxRCJLFiDx0BI
4hOPpYLi/jvYutpOlbJmzL53wNCgPT0m+0qKq4uJYsoE7qZdU2jyzYl7mZKrdCgh
McV8li1L1MsGP765Q2iRDyy1rDaYTs8DlKY2LruLz3EU0EXPOlaayaOMJTmjAjil
bJ4BoUYn1E/LHhGIMZutiwV9SwxX1g0vn4oRABEBAAH+AwMCV+RcQeOA3Udganyl
Fux1Qbc8vS+PmyPg7cMU/TFTD3Ne4XldSYrnO5Gb/6ByHljgHdGVKxEfjiQZ4We+
PsU+XYLoUJbvouKyDk3jMK/i2/bD9hByFRxK3Q0e89JgBn2nkMecI666z/tqSPvj
rb+U4F9xCMNpbcvTKegWlP8vG66sA1/Lqj8YmHv28O7JFk2U65msGnfqytSOxwzD
j+/7F3uQF/lCF7MJiOPn3Yj2N8EcdimCKNbRKR8Xycjha1qU7I6vns6j3GPIRWrx
//SOdGbvY/hzJJ2JjQ51NhxucBD13Zt0K27qdIlKSFUAW6WNRr3EQKGXHteRTZ1H
gOcIjgQ4QYVBPsrziG3gxF8FuF0xnAaK/bZmk/Kd3Tg1wRRhxntFybWY1QWx6cMf
q+e8YoR63cvLuCU04l9YfIynBlQFVJttRg3D6A7YCRwlN3cW7CV6DOHZ+IkgHvD/
p1QrGw13+NngLeSAYkt/etQ6txupWh6E6JGwxD7kw36Ek3/uY9LN7R/JlnY9IK52
9k3XiO5OJWLE3mxT9A0HWFgifgO5Ts+DXqTTOiPDlvO5gnydArYvWERZf8THscLB
39VKvvbKiIkE/Jz3gEUAniBsHeESQd6EB5IMrIcUHeunlrdokBSDXqqqTH6hbOKt
ZN/3lT3s/o9QWA8tJO9o/1XUD092/Ub33GFGcUsepZpjs84nRB/n/FcM46aUTDsl
h/cODa/4USZl1szOeckmZJs22+xuCaf5oc8Tao0UCA6zRSWx6dQ15x1meAeEntZN
tqwh0GiEGW6o/fWYClg1MUVhgxYCVGjk76k4dTxMQ7T7Kpmtg9TFE+qZq7ckwzlm
teKu0orowTZ5/f9sIjr1Xukplc5m+6LClICpKP6/rwHl/lFEzWE/BtLoSb9zPYCq
zbT2RWR3YXJkIFNub3dkZW4gKGdBQUFBQUJYNWRnc1NBNnI2ZjZIZlRZR0ZOZXNw
aDRVMDlxX2FjdTdEdmtkTmpzc3gtR1k5bk81a0dQVm95NFp2cG05OXFVcmNxdWd0
amRZWjdUTHN1LVc2SkY5ZzVOVnBOQlQ0dFRRS21FV2RUWlAxNDNzS1IyemRqU21i
ZndmbFpiV2xwS2R5UTJxOEE2SmxhR0djSVM1Vl9NVENlRlJmRVdRZEx5VlJGbHhW
b1lrOE5mcXRfVks5dEJ0T2ZfNVkwLVIyVHJ6ajBwXykgPHNub3dkZW5AcHJvdG9u
bWFpbC5jb20+iQE4BBMBAgAiBQJX5dgsAhsvBgsJCAcDAgYVCAIJCgsEFgIDAQIe
AQIXgAAKCRAgnFq+vkkLdEAICADTqNUIm1k8jyizAucPiTDZuHjwVae9ze7mWmEf
84Lz1wWifwXPYQiqtTz6nWvJ+cKd8joJw2gCdH1tsFn95x1flTXPscVKcPypxwGX
od8snMfEXqiHE0AKotSz1vWwwpdx9+tSFG+hZqMDZ4MIyh6bV5Aeg2hR6ib0EfsM
3UQVkzYW18IrnN83GajJb7al1xSGPfsEgAJGDd3lXtiHUmjWg32jfkYTBRajGna/
R8qMLfRagg7iKJSJJkiIEYHL4YdSKDS7Y/PXPIK+7EQ+jHM1MLBrgGjNl49nsqYL
wP4lrPC6Xq9f6aEXI7h394jzIc+XsDKng6OcqELcROT8DtvD
=fE1g
-----END PGP PRIVATE KEY BLOCK-----
```

1.12 Export a public key

prints the public key of the given email

```
usage: keybone public [-h] <recipient>
```

Positional arguments:

recipient any identification for the key: fingerprint, id or email

```
$ keybone public 3B9D624364FF560192B4136F209C5ABEBE490B74
-----BEGIN PGP PUBLIC KEY BLOCK-----
Version: GnuPG v1

mQENBFfl2CwBCADhQwj6dDbUe0eTVpy/goQib+02g9D+J7BX9+Q3LqQ4z74fmsBq
FdEoc8DA9fCYxNIMqd+oLl90m6Ur6OVwXj7RgtvXyyGJLLEDtJCspR1a/aFRFdLu
+9qYAkXdJO5PMrBvyLbriGcLYfT6hQibEh9W+DlqsfVJycSPOAsLxRCJLFiDx0BI
4hOPpYLi/jvYutpOlbJmzL53wNCgPT0m+0qKq4uJYsoE7qZdU2jyzYl7mZKrdCgh
McV8li1L1MsGP765Q2iRDyy1rDaYTs8DlKY2LruLz3EU0EXPOlaayaOMJTmjAjil
bJ4BoUYn1E/LHhGIMZutiwV9SwxX1g0vn4oRABEBAAG09kVkd2FyZCBTbm93ZGVu
IChnQUFBQUFCWDVkZ3NTQTZyNmY2SGZUWUdGTmVzcGg0VTA5cV9hY3U3RHZrZE5q
c3N4LUdZOW5PNWtHUFZveTRadnBtOTlxVXJjcXVndGpkWVo3VExzdS1XNkpGOWc1
TlZwTkJUNHRUUUttRVdkVFpQMTQzc0tSMnpkalNtYmZ3ZmxaYldscEtkeVEycThB
NkpsYUdHY0lTNVZfTVRDZUZSZkVXUWRMeVZSRmx4Vm9ZazhOZnF0X1ZLOXRCdE9m
XzVZMC1SMlRyemowcF8pIDxzbm93ZGVuQHByb3Rvbm1haWwuY29tPokBOAQTAQIA
IgUCV+XYLAIbLwYLCQgHAwIGFQgCCQoLBBYCAwECHgECF4AACgkQIJxavr5JC3RA
CAgA06jVCJtZPI8oswLnD4kw2bh48FWnvc3u5lphH/OC89cFon8Fz2EIqrU8+p1r
yfnCnfI6CcNoAnR9bbBZ/ecdX5U1z7HFSnD8qccBl6HfLJzHxF6ohxNACqLUs9b1
sMKXcffrUhRvoWajA2eDCMoem1eQHoNoUeom9BH7DN1EFZM2FtfCK5zfNxmoyW+2
pdcUhj37BIACRg3d5V7Yh1Jo1oN9o35GEwUWoxp2v0fKjC30WoIO4iiUiSZIiBGB
y+GHUig0u2Pz1zyCvuxEPoxzNTCwa4BozZePZ7KmC8D+Jazwul6vX+mhFyO4d/eI
8yHPl7Ayp4OjnKhC3ETk/A7bww==
=B5zN
-----END PGP PUBLIC KEY BLOCK-----
```

1.13 Backup your keyring for emergencies

With a single your whole keybone environment will be exported to a single plaintext blob that can be easily recovered later on.

Keep your backup in a VERY VERY VERY safe place, as it has the encryption key for your keyring as well.

generates base64 encoded backup of the whole keyring, ready to be imported again at any time

```
usage: keybone backup [-h] [-p PATH]
```

Options:

-p, --path destination path of the encrypted tarball

```
$ keybone backup > emergency-backup.keybone
INFO Compressing keyring
INFO Compressing keyring/pubring.gpg
INFO Compressing keyring/pubring.gpg~
INFO Compressing keyring/random_seed
INFO Compressing keyring/secring.gpg
INFO Compressing keyring/trustdb.gpg
INFO Compressing keybone.yml
```

1.14 Wipe your keyring

In case of emergency you might need to backup your keyring and then wipe it from its original location.

generates base64 encoded backup of the whole keyring, ready to be imported again at any time

```
usage: keybone backup [-h] [-p PATH]
```

Options:

-p, --path destination path of the encrypted tarball

```
$ keybone wipe --no-backup --force
WARNING deleting: /home/sn0wden/.keybone.yml
WARNING deleting: /home/sn0wden/keys/pubring.gpg
WARNING deleting: /home/sn0wden/keys/pubring.gpg~
WARNING deleting: /home/sn0wden/keys/secring.gpg
WARNING deleting: /home/sn0wden/keys/trustdb.gpg
```

1.15 Recover from a backup

This command assumes that your keyring was destroyed, or else you need to pass the `--force` option to overwrite any existing files.

recovers a keybone GPG setup from an encrypted tarball

```
usage: keybone recover [-h] [-f] path
```

Positional arguments:

path the path to an encrypted tarball containing a new keychain

Options:

-f=False, --force=False flag to force even if destination exists

```
$ keybone recover emergency-backup.keybone
WARNING replacing config file: /home/sn0wden/.keybone.yml
WARNING replacing existing key home: /home/sn0wden/.keybone
INFO setting mode 0700 on directory /home/sn0wden/.keybone
INFO writing keyring file /home/sn0wden/.keybone/pubring.gpg
INFO setting mode 0600 on /home/sn0wden/.keybone/pubring.gpg
INFO writing keyring file /home/sn0wden/.keybone/random_seed
INFO setting mode 0600 on /home/sn0wden/.keybone/random_seed
INFO writing keyring file /home/sn0wden/.keybone/pubring.gpg~
INFO setting mode 0600 on /home/sn0wden/.keybone/pubring.gpg~
INFO writing keyring file /home/sn0wden/.keybone/secring.gpg
INFO setting mode 0600 on /home/sn0wden/.keybone/secring.gpg
INFO writing keyring file /home/sn0wden/.keybone/trustdb.gpg
INFO setting mode 0600 on /home/sn0wden/.keybone/trustdb.gpg
```

CHAPTER 2

Internals Reference

CHAPTER 3

Indices and tables

- genindex
- modindex
- search

CHAPTER 4

Indices and tables

- genindex
- modindex
- search

k

keybone.util, 11

K

keybone.util (module), 11