How to Keysign:

- Verify Fingerprint
- Verify Identity
- Obtain authentic copy of key
- many traps to be avoided
- Sign and send key
- currently needs MTA
- $O(n)$
- $O(n^2)$

This is a typical keysigning protocol
It is hard to set up
242 registered participants
LET'S DECENTRALISE

Keysigning "parties" suck

ALL THE THINGS!
A party without beer only to obtain fingerprints
Keysigning "Parties" are not fun
people miss them
They don't print anything
Single point of failure
Pros: Very accurate, hard to misread
Cons: Very long
WTF

IS THIS SHIT?
Base16

610CB25237B370E9EB21
08E89CEE1B6B059B598E
Pros: looks familiar to nerds
Cons: Hard to distinguish characters
NOT SURE IF
B IS 8
Verifying fingerprints is hard
Pros: Shorter than other things
Cons: Probably too big of an alphabet
Base58Check
9r9knGannSDvoJyUoGbgyWDUEwWGdx7rUC
Pros: Probably simpler to compare
Cons: Easy to have collisions
Eventually, you've verified the fingerprint and the identity. You try to obtain an authentic copy of the key.
short key ids

evil32.com
Of course you don't use short key ids. Do you..?
Nobody uses short keyids. Except Debian.

More detailed info are present on SecureApt (interesting things starting from How apt uses Release.gpg). Please understand what you are doing first.

These are the commands which you need to run.

```
$ gpg --recv-keys 2D230C5F
$ gpg --export -a 2D230C5F | sudo apt-key add -
$ apt-get update
```
I don't always use short key ids
private key, you need to follow these guidelines when signing peoples keys:

During the Event

1. Keysigning is always done after meeting in person
2. During this meeting you hand each other your OpenPGP key fingerprint and at least one government issued key fingerprints are usually distributed as key fingerprint slips, created by a script such as gpg-key2ps (pack...
3. You check whether the name on the key corresponds with the name on the ID and whether the person in front of him is.

After the Event

You now have the printed public key information from the other participants.

Example key IDs for the other participants will be E4758D1D, **C27659A2**, and 09026E7B. Replace these IDs with the other participants.

1. retrieve the keys:
   1. gpg --recv-keys E4758D1D **C27659A2** 09026E7B
2. sign the keys:
   1. gpg --sign-key E4758D1D
   2. gpg --sign-key **C27659A2**
   3. gpg --sign-key 09026E7B
3. export the keys
   1. gpg --armor --export E4758D1D --output E4758D1D.signed-by.01234567.asc
   2. gpg --armor --export **C27659A2** --output **C27659A2**.signed-by.01234567.asc
   3. gpg --armor --export 09026E7B --output 09026E7B.signed-by.01234567.asc
4. Email the key users (use the email address that was part of the key's user ID) and attach the corresponding signed key to the key server:
   1. gpg --send-keys --keyserver keyserver.ubuntu.com E4758D1D
For the party, you will need these strips and an official photo ID, such as a driver's license or passport.

After the Party

Step 1: Get other people's keys

You now have the printed public key information from the other participants.

Example key IDs for the other participants will be E4758D1D, C27659A2, and 09026E7B. Replace these IDs with the key IDs you received from the other participants.

1. Find the key ID numbers on each printout and get the public keys from the keyservers:

   gpg --recv-keys E4758D1D C27659A2 09026E7B

Step 2: Sign the keys

1. Sign a key:

   gpg --sign-key E4758D1D

i. If a key has multiple user IDs, GPG will ask if you want to sign all of them.
As error message, you haven't configured gpg server.

Try this:
```
gpg --keyserver subkeys.pgp.net --recv-keys 6092693E &
gpg --export --armor 6092693E | sudo apt-key add -
```

Updated

It seems that you can not connect to server:
```
gpg: keyserver timed out
```

Do you have a firewall block port 11371 of hkp service.

You can use port 80 instead of 17371:
```
gpg --keyserver subkeys.pgp.net:80 --recv-keys 6092693E
```

on the Internet
issue1579: GnuPG ignores the fingerprint
Also: v3 keys still accepted
So you use the fingerprint instead of short key ids however, currently shipped gnupg version do not check for the fingerprint of the key to be imported
Let's not use Keyservers

- leaks data
- trivial MITM attacks (issue1579)
- packet forgery
- OCaml... srsly.
http://bugs.gnupg.org/gnupg/issue1579
I Don't Always Target Users

But When I Do, It's Me
Let's define our target users.
It's my mom!
The Gold Standard:
# .caffrc -- vim:ft=perl:
# This file is in perl(1) format - see caff(1) for details.

$CONFIG{'owner'} = 'Username';
#$CONFIG{'email'} = '[user]@[domain]';
#$CONFIG{'reply-to'} = 'foo@bla.org';

# You can get your long keyid from
# gpg --with-colons --list-key <yourkeyid|name|emailaddress..>
#
# If you have a v4 key, it will simply be the last 16 digits of
# your fingerprint.
#
# Example:
# $CONFIG{'keyid'} = [ qw{FEDCBA9876543210} ];
# or, if you have more than one key:
# $CONFIG{'keyid'} = [ qw{0123456789ABCDEF 89ABCDEF76543210} ];
#$CONFIG{'keyid'} = [ qw{0123456789abcdef 89abcdef76543210} ];

# Select this/these keys to sign with
#$CONFIG{'local user'} = [ pw{123456789abcdef Gadafe7543210} ];
# Additionally encrypt messages for these keyids
That's the pinnacle, the gold-standard of contemporary keysigning
LET'S MAKE THEM USE BASE16, OCAML, AND PERL FOR THEIR CRYPTO
AM I THE ONLY ONE AROUND HERE

WHO IS SICK OF FINGERPRINTS?
Fast forward 20 years
We see the fire burning
We need to do something about the situation
Two decades later:
- mobile computing
- WiFi
- QR Codes
Yet, our machines cannot talk to each other
Introducing: GNOME Keysign

Key Fingerprint

F289 F7BA 977D F414 3AE9
FDFB F70A 0290 6C30 1813
Leveraging 2000s technologies

Type fingerprint

F289 F7BA 977D F414 3AE9
FDFB F70A 0290 6C30 1813

Open Image

Step 1: Scan QR Code or type fingerprint and click on 'Download' button
(evolution:19692): gtkhtml-WARNING **: found token with no open
(evolution:19692): Glib-CRITICAL **: Source ID 31815 was not found when attempting to remove it
(evolution:19692): Glib-CRITICAL **: Source ID 31886 was not found when attempting to remove it
(evolution:19692): Glib-CRITICAL **: Source ID 31966 was not found when attempting to remove it
(evolution:19692): Glib-CRITICAL **: Source ID 31991 was not found when attempting to remove it
(evolution:19692): Glib-CRITICAL **: Source ID 31960 was not found when attempting to remove it
(evolution:19692): Glib-CRITICAL **: Source ID 33080 was not found when attempting to remove it
(evolution:19692): Glib-CRITICAL **: Source ID 33462 was not found when attempting to remove it
(evolution:19692): Glib-CRITICAL **: Source ID 33555 was not found when attempting to remove it
(evolution:19692): Glib-CRITICAL **: Source ID 33769 was not found when attempting to remove it
(evolution:19692): Glib-CRITICAL **: Source ID 34560 was not found when attempting to remove it
(evolution:19692): Glib-CRITICAL **: Source ID 34883 was not found when attempting to remove it
(evolution:19692): Glib-CRITICAL **: Source ID 35019 was not found when attempting to remove it
(evolution:19692): Glib-CRITICAL **: Source ID 35619 was not found when attempting to remove it
(evolution:19692): Glib-CRITICAL **: Source ID 36186 was not found when attempting to remove it
(evolution:19692): Glib-CRITICAL **: Source ID 36852 was not found when attempting to remove it

-> geysigning git:(tobi_tmp_email) ->
(evolution:19692): Glib-CRITICAL **: Source ID 36483 was not found when attempting to remove it
GNOME Keysign

- Directly transfers keys
- Sends encrypted email
- No MTA needed
Still waiting for patches
Thank you