LAB: PGP Key Signing Party

We will use the command-line since that is going to be consistent across all platforms we installed. The GUI will work for most sections except choosing our key server below. Our local keyserver does not yet implement all HTTP methods required by the spec.

We are using software tools from the FOSSDEM folk. These can be freely downloaded from https://github.com/FOSDEM/keysigning and then modified for your environment/key signing party.

Prerequisites:

1. You should have created a key from the previous PGP exercises or already have one you use. If not look at the PGP labs or run `gpg --gen-key`
2. Make sure you have a revocation certificate stored safely away.

   `gpg --gen-revoke IDENTITY`

uploading your key

1. List your keys to figure out your key-id. Note that these days most new software will not list the short key-ids because collisions have occurred. Also look at the last 16 characters to see the long id format (or use the `--keyid-format long option`). Note that you can always use email addresses as the key-id but it's possible for people to have multiple addresses.

```
reikan:~ preso$ gpg --list-keys
/Users/preso/.gnupg/pubring.kbx
------------------------------
pub   rsa4096 2019-02-18 [SC] [expires: 2020-02-18]
      DA9C65C302271B148A9813CB6037C625516A4AA8
uid           [ultimate] Randy Bush <randy@psg.com>
sub   rsa4096 2019-02-18 [E] [expires: 2020-02-18]

reikan:~ preso$ gpg --list-keys --keyid-format long
/Users/preso/.gnupg/pubring.kbx
------------------------------
pub   rsa4096/6037C625516A4AA8 2019-02-18 [SC] [expires: 2020-02-18]
      DA9C65C302271B148A9813CB6037C625516A4AA8
uid                 [ultimate] Randy Bush <randy@psg.com>
sub   rsa4096/EACBFED525F8BD95 2019-02-18 [E] [expires: 2020-02-18]
```

2. Keep note of your fingerprint especially incase your keys are listed with the short-id format

```
reikan:~ preso$ gpg --fingerprint DA9C65C302271B148A9813CB6037C625516A4AA8
pub   rsa4096 2019-02-18 [SC] [expires: 2020-02-18]
      DA9C 65C3 0227 1B14 8A98  13CB 6037 C625 516A 4AA8
uid           [ultimate] Randy Bush <randy@psg.com>
sub   rsa4096 2019-02-18 [E] [expires: 2020-02-18]
```

3. We have created `ksp.workshop.` you can upload your keys here.

```
$ gpg --send-keys --keyserver ksp.workshop DA9C65C302271B148A9813CB6037C625516A4A
gpg: sending key 6037C625516A4AA8 to hkp://ksp.workshop
```

4. Make sure you can see your key in the keyserver by visiting http://ksp.workshop/keys in your browser. Note that ordinarily this directory would not be exposed by the webserver.

5. Once all done, the instructors will generate a text file that will be available at http://ksp.workshop/

6. Next we shall get up one at a time (with our government ID) and tell the class:

- Our name
- The ID (long format)
- The fingerprint (use the NATO aphabet if you can)

7. Some people may ask to see the government ID before signing and then they'll sign.

```
$ gpg --sign-key IDENTITY
```

8. Next each one of us uploads the new signature to the local keyserver and optionally to the default keyserver configured in your gpg installation.

```
$ gpg --send-keys --keyserver ksp.workshop IDENTITY

# optional
$ gpg --send-keys IDENTITY
```

9. At the end of the day we can download the copy of all our signatures for *our* key from the keyserver. To do this, simply download the key from http://ksp.workshop/keys , save it as a text file and import it

```
$ curl -O http://ksp.workshop/6037C625516A4AA8 # may not be available on windows
$ gpg --import 6037C625516A4AA8 # use the name of the text file you saved
```