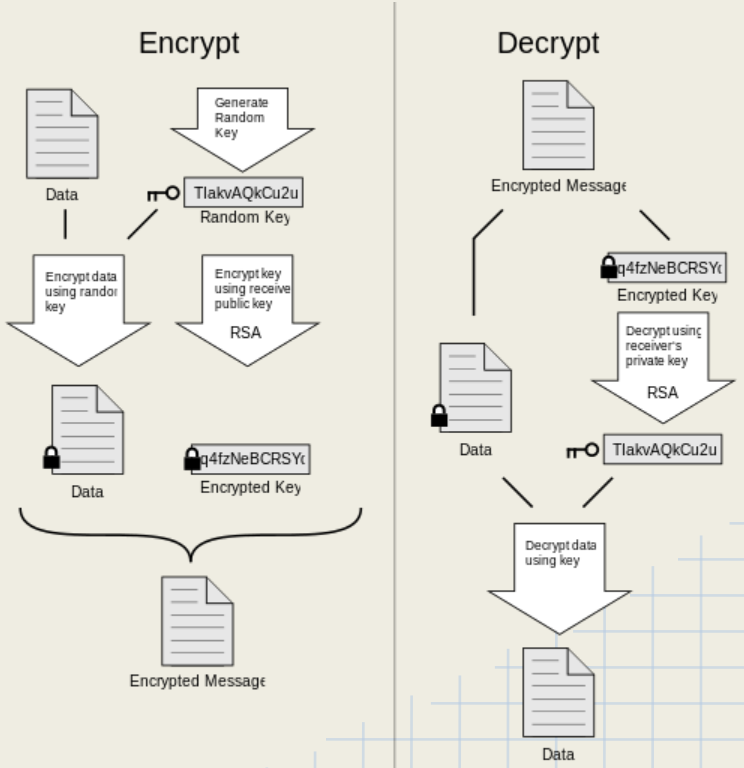
GPG Intro

What is GPG?

- GPG, or GNU Privacy Guard, is a public key cryptography implementation. (Conforms to PGP and RFC 4880, not really just an alternative)
- Best used mostly for email encryption
- Uses **Hybrid Encryption**



Install GPG

- Linux
  - sudo apt-get install gnupg
- Mac (homebrew package manager)
  - brew install gnupg
    - brew install bash-completion
- Windows
  - Install Gpg4win → http://gpg4win.org/download.html

Generate Keys and Revoke Cert

- gpg --gen-key (This will ask you a series of questions)
  - Please select what kind of key you want: **(1) RSA and RSA (default)**
  - What keysize do you want? **4096**
  - Key is valid for? **0**
  - Is this correct? **y**
  - Real name: **your real name here**
  - Email address: **your_email@address.com**
  - Comment: **Optional comment that will be visible in your signature**
  - Change (N)ame, (C)omment, (E)mail or (O)kay/(Q)uit? **O**
  - Enter passphrase: **Enter a secure passphrase here (upper & lowercase, digits, symbols)**

Generate Keys and Revoke Cert

- gpg --list-keys
  - Lists all public keys in your possession
- gpg --list-secret-keys
  - Lists all private keys in your possession
- gpg --send-keys --keyserver pgp.mit.edu key_id
  - Sends key with “key_id” to mit’s pgp server
- gpg --gen-revoke your_email@address.com
  - Generates Revoke cert for key

Identifying and Exporting Keys

- gpg --fingerprint <email | key_id>
- Example:

```
pub   4096R/9C26CCE3 2014-01-30
      Key fingerprint = DE90 4FAC F310 12DF 41E3  4BD8 EF70 3ABE 9C26 CCE3
uid                  Anupam Ghosh <anupam.ghosh93@gmail.com>
sub   4096R/B29AA8D7 2014-01-30
```

- gpg --export --armour email@address.com
- gpg --export-secret-key --armour your_email@address.com
  - VERY DANGEROUS, BE CAREFUL

Get others' public keys

- gpg --import name_of_pub_key_file
  - Imports public key from file.
- gpg --keyserver pgp.mit.edu --search-keys search_parameters
  - Will search keyserver for keys with "search_parameters"
    - Suggested search_parameters is email of recipient.
- Trusting keys
  - gpg --sign-key email@address.com
    -
  - gpg --edit-key email@address.com, trust
    - Can add different levels of trust.

Encryption & Signing

- gpg --armour --encrypt --sign --recipient other_email@address.com file
  - if you want to read what you just encrypted just add your self as a recipient (--recipient your_email@address.com)
- gpg --armour --sign file
  - --output out_file

-- armour?

This returns the ASCII representation of encrypted data rather than raw bytes. (Only ballers can run gpg w/o --armour)

Decryption & Verifying

- gpg --decrypt file
  - This will handle both decrypting and verifying signatures

Workflow for Alice and Bob

Alice

- gpg --gen-key …
- gpg --send-key …
- gpg --import-key bob_public_key OR gpg --keyserver pgp.mit.edu --search_key bob@email.com
- gpg --encrypt --sign bob@email.com
- send email_file
- gpg decrypt email_file

Bob

- gpg --gen-key …
- gpg --send-key …
- gpg --import-key alice_public_key OR gpg --keyserver pgp.mit.edu --search_key alice@email.com
- gpg --encrypt --sign bob@email.com
- send email_file
- gpg decrypt email_file

Misc

- gpg --refresh-keys
- gpg --keyserver pgp.mit.edu --refresh-keys

Sources

- https://www.digitalocean.com/community/tutorials/how-to-use-gpg-to-encrypt-and-sign-messages-on-an-ubuntu-12-04-vps
- http://irtfweb.ifa.hawaii.edu/~lockhart/gpg/gpg-cs.html
- http://www.ted.com/talks/andy_yen_think_your_email_s_private_think_again