Pretty Good Privacy with GnuPG

Steve Revilak

Cabot House

Sep. 24, 2013

Encryption and Signing

Encryption	The purpose is to ensure that a message is readable only by someone possessing a specific key.
Signing	Guarantees that a message was sent by someone with a specific key (and wasn't altered after sending).

(Here I'm using the term "message" in a very generic sense – it could be email, a file, or any arbitrary piece of data).

Leap of faith: You need some level of trust that a particular key belongs to a particular person.

Public vs Private Keys

Keys exist as a pair:

- There's a **public key**. You share this with everyone.
- There's a **private key** (also called a **secret key**). This is a closely guarded secret.

During **encryption**, the sender encrypts the message with the recipient's public key. The recipient uses their private key to decrypt the message.

During **signing**, the sender signs the message with their private key. The signature can be verified by anyone with the corresponding public key.

Goals for this workshop

- Generate a keypair
  - Upload your public key to a keyserver
  - Download my public key.
- Set up your mail program to send and receive signed and encrypted email.
  (Mail program = Mail User Agent, or MUA)
- Send me a signed and encrypted message. (I should be able to decrypt your message, and verify your signature.)
- Let me sent you a signed and encrypted message. (You should be able to decrypt my message and verify my signature.)

Generating a Keypair

We can do these things with GUI tools. I'm including command-line equivalents for reference.

- Generate Key.
  `gpg --gen-key`
  Choose RSA, RSA. Use the longest key possible.
- Upload Key.
  `gpg --send-key KEYID`
- Download my key.
  `gpg --search steve@srevilak.net` OR
  `gpg --recv-key 28C2A300`

Mail Client Basics

Sending:

- You'll use a protocol called SMTP, or Simple Mail Transfer Protocol.

Receiving:

- Two options: IMAP (Internet Mail Access Protocol), or POP (Post Office Protocol)
- IMAP stores all messages on your ESP's mail server. You can move them to local folders, but you have to do this explicitly.
- POP downloads mail from your ESP's mail server. By default, the server copy is deleted; you can also configure your mail client to leave it on the server.
- If you have a lot of mail on the server, the initial synchronization might take a while, especial with POP.

Configuring your MUA (GMail)

GMail:

- Enable IMAP or POP in Gmail's web interface.
- Sending: smtp.gmail.com, port 587, use SSL
- Receiving: imap.gmail.com, port 993, use SSL; OR pop.gmail.com, port 995, use SSL
- `https://support.google.com/mail/troubleshooter/1668960?hl=en&ref_topic=1669040`

Configuring your MUA (Hotmail)

Hotmail:

- Enable POP/IMAP in outlook.com's web interface
- Sending: smtp-mail.outlook.com, port 587, use TLS
- Receiving: imap-mail.outlook.com, Port 993, use SSL; OR pop-mail.outlook.com, port 995, SSL
- `http://windows.microsoft.com/en-us/windows/outlook/send-receive-from-app`

Configuring your MUA (Yahoo)

Yahoo:

- POP is only available for Yahoo Plus Accounts
- Sending: smtp.mail.yahoo.com, port 587, use SSL
- Receving: pop.mail.yahoo.com, port 995, use SSL; OR imap.mail.yahoo.com, port 993, use SSL
- `http://help.yahoo.com/kb/index?page=content&y=PROD_MAIL_ML&locale=en_US&id=SLN4075`

Sending and receiving mail

- We'll take this one step at a time.
- Send me a signed and encrypted message.
- Open your Sent Mail folder. Make sure that you can read the encrypted message you sent!
- I'll respond. Work on downloading, decrypting, and reading my message. Be sure to verify the signature.

Trusting and Signing Keys (1)

How do you verify that a given key belongs to a given person? You check the fingerprint. Here's mine:

```
gpg --fingerprint 28C2A300
...
Key fingerprint = 6F09 15FF 59CE E093 56F4
                  BEEC E772 7C56 28C2 A300
```

If the fingerprint matches, you've got the right key.

Trusting and Signing Keys (2)

Once you trust a key, sign it.

- `gpg --sign-key 28C2A300` OR
  `gpg --lsign-key 28C2A300`

"lsign" is a local signature; it's only visible to you.
To distribute a non-local (aka "exportable") signature:

- Send it to a key server:
  `gpg --send-key 28C2A300`
- Export the key (containing your signature), and send it to the key holder.
  `gpg -a --export 28C2A300 > signed-key.asc`

The key holder will `gpg --import signed-key.asc` to import your signature.

Revocation Certificates

What if (say) your laptop is stolen, and you lose your private key? You revoke it.

- Generate a revocation certificate
  ```
  gpg -a --gen-revoke KEYID > pgp-revoke.asc
  ```

Uploading the revocation certificate “cancels” your key.

Note: you cannot generate a revocation certificate without a private key! Keep the revocation certificate in a safe place.

Backing up your keys

If you lose your private key, you won't be able to decrypt messages. Lost private keys cannot be recovered!

- Backup your private key

  ```
  gpg -a --export-secret-keys KEYID > private-key.asc
  ```

Store a copy of `private-key.asc` in a safe place. For example, keep electronic and printed copies in a safe deposit box.

Wrap Up

- PGP can protect your privacy through encryption.
- PGP can provide non-repudiation through signatures.
- PGP is something that you can (and IMHO, should) use every day.
- My favorite reason for using PGP: because I can!
- GnuPG is a free software implementation of a public standard. Remember: it's hard to backdoor software when the source code is public.

Resources

- GnuPG: `http://gnupg.org/`
- GPG4win: `http://www.gpg4win.org/`
- GPG Tools: `http://gpgtools.org/`
- Riseup.net's Best practices for OpenPGP: `https://we.riseup.net/riseuplabs+paow/openpgp-best-practices`
- Cryptoparty handbook: `https://www.cryptoparty.in/documentation/handbook`
- Surveillance Self-Defense: `https://ssd.eff.org/`