Lec 3: Cryptographic Privacy

CS492E: Introduction to Software Security

Sang Kil Cha

Cryptographic Privacy

Secure Computing with Cryptography

Cryptography is used everywhere in everyday computing:

- Secure wifi protocols: WPA2/WPA3.
- Secure web protocols: TLS.
- Secure booting: TPM.
- etc.

But most people do not know how to use it properly for their ***privacy***.

Motivation

Q: Company X asks you to send your ***private*** information, e.g., social security number, bank account information, etc. though email. What do you do?

1. Send in plaintext hoping that no one will be interested to see it.
2. Make a password-protected ZIP file and send it.
3. Else?

Password-protected ZIP File

Hi Alice,

I made a password-protected ZIP file.
You should be able extract it with the password “1234”.

Thanks,
Bob

(secret.zip 📎 attached.)

Email

- Extremely vulnerable to forging attacks.
- Popular protocols include: SMTP, POP, IMAP, etc.

SMTP (Simple Mail Transfer Protocol)

```
S: 220 smtp.example.com ESMTP Postfix
C: HELO relay.example.org
S: 250 Hello relay.example.org, I am glad to meet you
C: MAIL FROM:<bob@example.org>
S: 250 Ok
C: RCPT TO:<alice@example.com>
S: 250 Ok
C: RCPT TO:<theboss@example.com>
S: 250 Ok
C: DATA
S: 354 End data with <CR><LF>.<CR><LF>
...
```

Privacy Concerns

- Can you trust email servers?
- Can you trust email service providers?
- Can you trust your ISP?
- Can you trust the government?

Pretty Good Privacy (PGP)

PGP is a commercial tool that we use for cryptographic privacy. Typically, we use ***GPG*** (GNU Privacy Guard), which is a free version of PGP.

Both PGP and GPG implement the ***OpenPGP*** standard.

```
$ gpg --version
gpg (GnuPG) 2.3.3
libgcrypt 1.9.4
Copyright (C) 2021 g10 Code GmbH
...
```

PGP is Widely Used!

- Linux kernel releases with PGP signatures.
- APT packages are signed with PGP keys.
- Git and GitHub support PGP key signing.
- ...

Generating a PGP Key

```
$ gpg --gen-key
```

Read `https://docs.github.com/en/authentication/managing-commit-signature-verification/generating-a-new-gpg-key` for further information.

PGP Keyring

You maintain a list of keys in a keyring, and those keys include your public key, your private key, and others' public keys. If you installed PGP/GPG on your machine, it starts with an empty keyring: it is your own business to add/maintain a valid set of keys.

```
$ gpg --list-keys          # List keys
$ gpg --list-sigs          # List keys with signatures
```

Importing/Exporting Key(s)

```
$ gpg --import <keyfile>
$ gpg --export --armor <key ID or identity>
```

Which Public Key Can You Trust?

- You should always verify the public key that you received indeed belongs to the sender: you should check the fingerprint of the key[1].
- Fingerprint is a short identifier (in hex).
- It is a good practice to put your PGP fingerprint in your namecard.

[1] `$ gpg --fingerprint`

Problem

Suppose you received a message signed with Charlie's key. However, your keyring doesn't have Charlie's key.

- In a strict scenario, you should not trust this message because you don't know Charlie.
- However, it could be ***cumbersome*** to manually validate keys everytime you communicate with a new person.

How do we safely extend the trust boundary?

Web of Trust

- Key idea: let's trust keys that are signed by someone that I trust.
- By signing a key, you can assign a certain ***level*** of trust to the key (owner).

Signing (Validating) Keys

After checking the fingerprint of a key with its owner, you can validate the key by signing it.

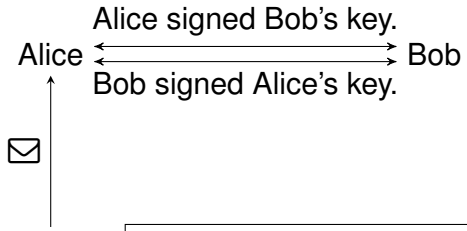
```
$ gpg --edit-key <key ID>
gpg> fpr
gpg> sign

$ gpg --sign-key <key ID>
```

PGP Trust Levels

- Ultimate = this is only applicable to your own key.
- Full = I trust the owner as much as myself.
- Marginal = The owner understands the implication of key signing and properly validates keys before signing them.
- None = I do not trust the owner.
- Unknown = I am not sure about the owner.

Example Scenario



Alice has two public keys of Bob:

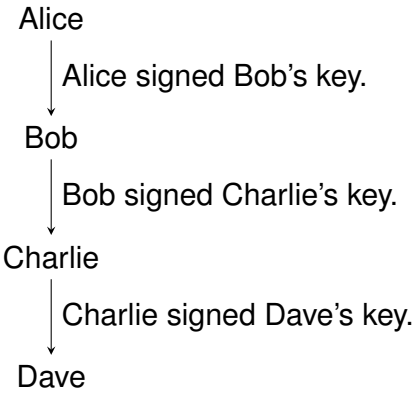
1. One key is signed by Alice (and Bob himself).
2. Another key is only self-signed.

PGP Validation Algorithm

A key is valid if one of the conditions met.

- The key has been signed by fully trusted person.
- The key has been signed by three marginally trusted keys (can be configured with `--marginals-needed`).
- The path length of trusted keys from the key back to my own key is five steps or less (can be configured with `--max-cert-depth`).

Example Web of Trust



Can Alice trust Dave's key?

Size of Web of Trust Matters

We need many people to fully enjoy the power of Web of Trust. Since there's no central server, the usefulness of PGP really depends on how many people are within your web of trust.

Key Signing Party

- Meet people in person and obtain key IDs as well as fingerprints.
- You should confirm their identities upon receiving the information.
- After the party, download and sign each of the keys that you have confirmed.

[2]

[2] https://xkcd.com/364/

PGP Key Server

You can upload/download keys through PGP key servers. By sending a signed key, your trust information will be automatically appended to the keys in the key server.

```
$ gpg --keyserver <keyserver> --send-keys <key ID>
$ gpg --keyserver <keyserver> --recv-keys <key ID>
```

* Example key server: `keyserver.ubuntu.com`

What Do You Do When Your Key Expires?

1. Create a new key.
2. Extend the key expiry .

What Do You Do When Your Key Expires?

1. Create a new key.
2. Extend the key expiry (preferred way).

In-Class PGP Key Signing Party

1. Make a team of two or three people.
2. Each team member should share their PGP keys and sign each other's keys, while making sure to check their identities.
3. You also sign professor's key and send it to `keyserver.ubuntu.com`.
4. Make sure all the team members completed all the tasks above.
5. One of the team members (only one member) sends a signed and encrypted email to professor (`sangkilc@kaist.ac.kr`). The content of the email should be the output of `gpg --list-sigs`.
6. After professor's confirmation, you are all set. Wait for other teams.

HW1 is Out

Due by tomorrow. It is a ***warm-up*** exercise.

Question?