**End-to-End Encryption (E2EE)**

**This is where the data is encrypted by the sender and only the receiver can decrypt it**

**E2EE technology**

* PGP
* SMIME
* OTR
* ZRTR
* SSL/TLS

The idea of E2EE and zero knowledge systems is that you cannot reveal the details of the communication to your adversary, even if coerced, or even if they wanted to

E2EE messaging apps

* Signal, chat secure, crypto cat

E2ee offers protection for data in transit, but not for when it’s been received

Another security method will be needed to secure data client side

**Steganography**

**The idea of hiding data in another set of data e.g., a picture file**

* There could be a message in the binary of the photo when decoded
* The file being used to hide the data is called the **Carrier**

Steganography is less obvious than encryption

Encryption isn’t hidden at all, its obvious when something has been encrypted however **steganography is the art of hiding data in plain sight**

**OpenPuff** (Steganography on windows)

Plausible Deniability

* this application also allows for a **decoy to be created in case of coercion** etc.
* type in the decoy passwords and the decoy carrier will be opened instead, containing the decoy message

When finding a file to use as a carrier, don’t just download any photo from the internet

This could be easily found, downloaded, and compared to your carrier and differences between the two files could easily be identified

1. Resize, crop and compress any photo downloaded from the internet to change it from what is accessible online
2. Use an original photo

Attackers don’t usually try to brute force encryption, they’ll often go for the low hanging fruit

* It is much easier to install a keylogger or watch over your shoulder than to brute force an encrypted password for your hard disk

**Security is a weak link phenomenon**

* Security is only as strong as its weakest link

Attackers to the Defenders: “Today you were lucky, but remember we only have to be lucky once, you have to be lucky always”

An attacker only has to be lucky once and they shoot for the weak spot first

* Find your weakest spots, secure those and then tighten parameters
* It’s okay to tune an engine, but the engine has to be running correctly first
* Risk-based approach

Absolutely no point implementing security features to be used on an outdated browser without patches or on an account with a weak password