# Symmetric Encryption

Miguel Frade – Nuno Rasteiro

Department of Informatics Engineering

School of Technology and Management, Polytechnic Institute of Leiria

November 2019

- A system of Symmetric cipher uses the same key to encrypt and decrypt
- Both sender and receiver need to know the key, agreeing previously the key to use
- The main problem of this system is the exchange of the key between sender and receiver
- It  is easier to intercept the key than use brute force to crack the key
- Other problem is the quantity of keys necessary. If "n" people what to communicate securely, it will be needed "n(n-1)/2" keys

## Symmetric Encryption- Introduction

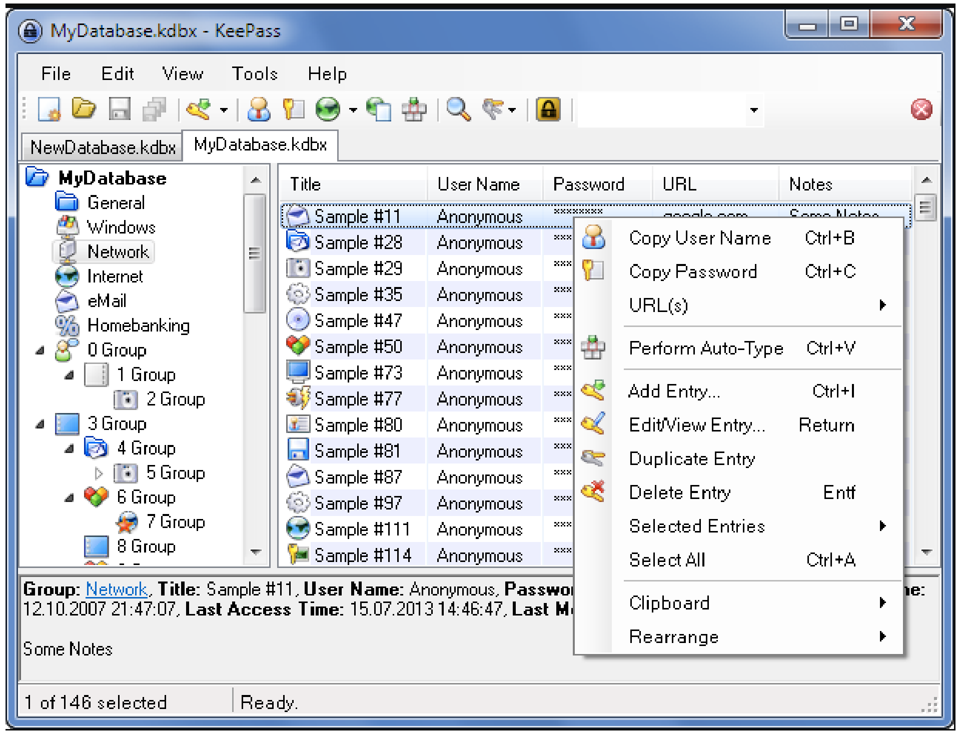
- Tools we will investigate:
    - **Password managers**
    - **GnuPG** – used essentially to exchange emails with asymmetric encryption, however it is also used for Symmetric encryption.
    - **OpenSSL** – it is a tool that implements one of the security protocols widely used, the SSL. It provides and API and a command line interface
    - **Veracrypt** – disk encryption
    - **Steganography applications** – hide in plain sight

## Password Manager

- Password Manager is a program to manage and store passwords
- uses an encrypted database
- can generate passwords with a certain amount of complexity, or you can choose your own
- to access the password database you will need to know the master password

# Password Manager

## Keepass



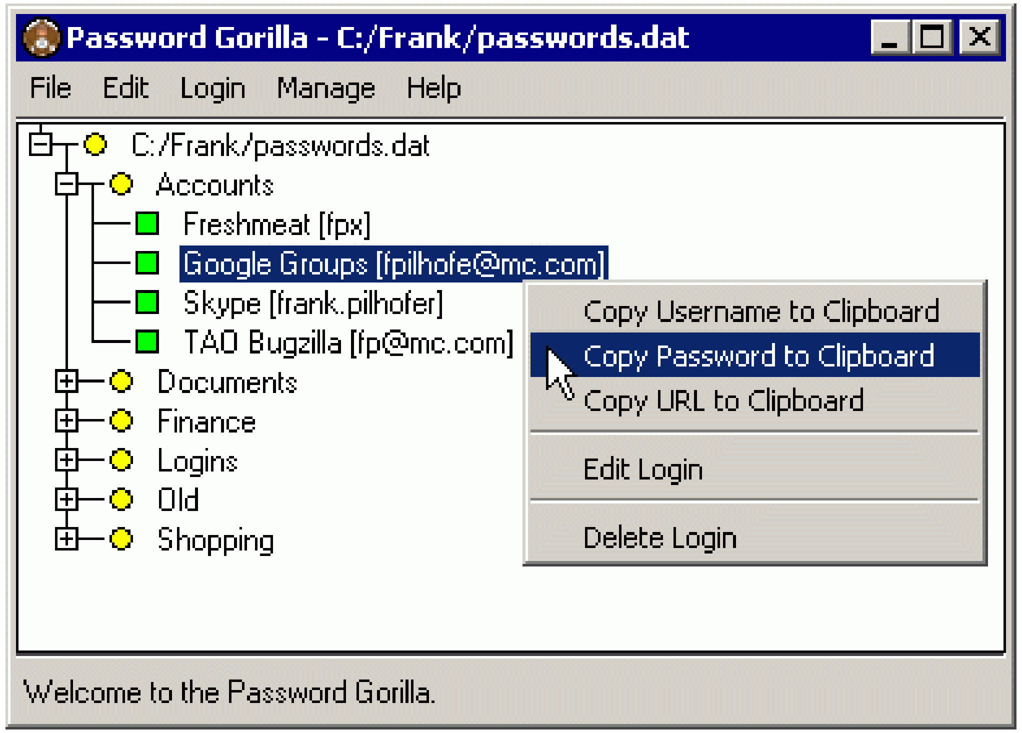
| Title | User Name | Password | URL | Notes |
|---|---|---|---|---|
| Sample #11 | Anonymous | ******** | google.com | Some Notes |
| Sample #28 | Anonymous | *** | | |
| Sample #29 | Anonymous | *** | | |
| Sample #35 | Anonymous | *** | | |
| Sample #47 | Anonymous | *** | | |
| Sample #50 | Anonymous | *** | | |
| Sample #73 | Anonymous | *** | | |
| Sample #77 | Anonymous | *** | | |
| Sample #80 | Anonymous | *** | | |
| Sample #81 | Anonymous | *** | | |
| Sample #87 | Anonymous | *** | | |
| Sample #97 | Anonymous | *** | | |
| Sample #111 | Anonymous | *** | | |
| Sample #114 | Anonymous | *** | | |

**Group:** Network, **Title:** Sample #11, **User Name:** Anonymous, **Passwo**
12.10.2007 21:47:07, **Last Access Time:** 15.07.2013 14:46:47, **Last M**

Some Notes

1 of 146 selected Ready.

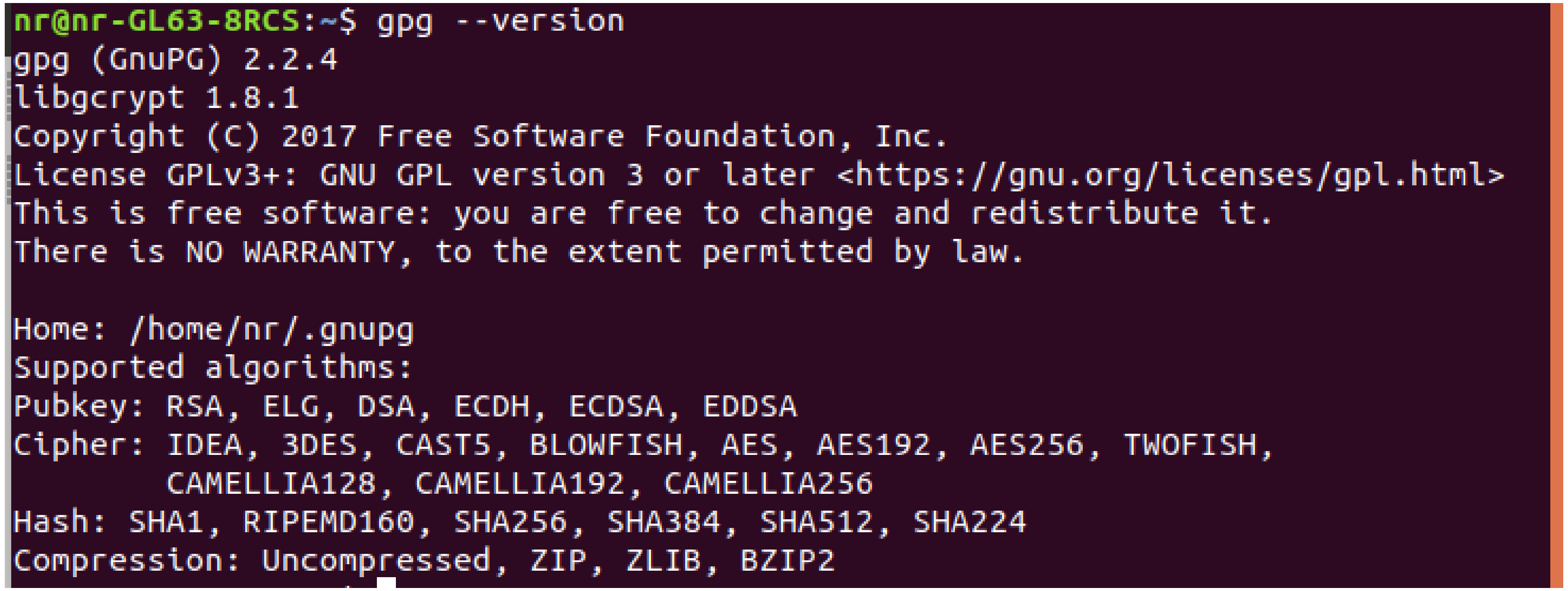
## Gorilla



Welcome to the Password Gorilla.

## GnuPG

- Application that implements the OpenPGP for for exchanging e-mails
- supports several cryptographic algorithms: public key, symmetric and hash
- gpg --version
- man gpg



```
nr@nr-GL63-8RCS:~$ gpg --version
gpg (GnuPG) 2.2.4
libgcrypt 1.8.1
Copyright (C) 2017 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <https://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.

Home: /home/nr/.gnupg
Supported algorithms:
Pubkey: RSA, ELG, DSA, ECDH, ECDSA, EDDSA
Cipher: IDEA, 3DES, CAST5, BLOWFISH, AES, AES192, AES256, TWOFISH,
        CAMELLIA128, CAMELLIA192, CAMELLIA256
Hash: SHA1, RIPEMD160, SHA256, SHA384, SHA512, SHA224
Compression: Uncompressed, ZIP, ZLIB, BZIP2
```

## GnuPG

- gpg –c[a] [--cipher-algo <algorythem>] <File name>
    - Ex
        - gpg -c --cipher-algo 3DES teste.txt
- gpg –o <decrypt file> --decrypt <encrypt file>
    - Ex
        - gpg -o teste_d.txt --decrypt teste.txt.gpg

# OpenSSL

- OpenSSL is a tool that implements one of the most used security protocols: TLS (formerly known as SSL)
- implements several cryptographic algorithms
- lots of control over the encryption:
  - it allows to choose the algorithm, the operation mode: ECB, CBC, CFB or OFB and the initialization vectors
- this flexibility comes with a cost which is the increase of complexity of the command line structure

## OpenSSL

- Syntax
- # Encrypt
- openssl enc -<algoritmo> -in <fich_a_cifrar> [-p] -out <fich_cifrado>
    - Ex: openssl enc -des3 -in teste.txt -p -out teste_c.txt
    - openssl enc -aes256 -in teste.txt -out teste_c.txt

- # Decrypt
- openssl enc -<algoritmo> -d -in <fich_cifrado> -out <fich_decifrado>
    - Ex: openssl enc -aes256 -d -in teste_c.txt -out teste_d.txt

## OpenSSL

- By default the OpenSSL uses a random SALT
  - The first 6 bytes are reserved for this value.
  - This ensures a different encrypted results even if the same message is encrypted with the same key more than once
  - The encryption is always in binary mode
    - On the early days, binary mode wasn’t supported on e-mails
    - So, the encrypted message must be encoded in Base64 (even today to ensure compatability)

## OpenSSL

- #encrypt
- **openssl base64 -in <fich_a_codificar> -out <fich_codificado>**
  - openssl base64 -in teste2.txt -out teste2_c.txt

- # decrypt
- **openssl base64 –d -in <fich_codificado> -out <fich_descodificado>**
  - openssl base64 -d -in teste2_c.txt -out teste2_dea.txt

## OpenSSL

- # Encrypt
- **openssl enc -<algoritmo> -in <fich_a_cifrar> -a -out <fich_cifrado>**
    - openssl enc -base64 teste2.txt -a -out ficheiro_cy.tx
- # Decrypt
- **openssl enc -<algoritmo> -d -in <fich_cifrado> -a -out <fich_decifrado>**
    - openssl enc -base64 -d -in ficheiro_cyf.txt -a -out ficheiro.txt

## Steganography

- Stenography is the art to hide information in another message
- The working principle of steganography is to replace unused or less important bits in normal files (such as images, sound, text, HTML, ...) with bits of information, and hide them this way
- The information hidden can be plain text, encrypted text, or even images, thus creating a disguised communication channel
- Very difficult to analyze the traffic and detecte if communications is taking place with this technique
- It is also used to:
  - create watermarks,
  - digital rights management systems
  - to guarantee the digital content has not been changed

# Steganography

- There are many available applications, for several operating systems
- Steghide
  - has a command line interface
  - Steghide is a steganography program that is able to hide data in various kinds of image- and audio-files. The color- respectively sample-frequencies are not changed thus making the embedding resistant against first-order statistical tests.
- **steghide embed -cf picture.jpg -ef secret.txt**
  - Embed secret data in a cover file thereby creating a stego file
- **steghide extract -sf picture.jpg**
  - Embed secret data in a cover file thereby creating a stego file
- **steghide info picture.jpg**
  - Embed secret data in a cover file thereby creating a stego file

## VeraCrypt

- VeraCrypt is a software for encrypting volumes and containers.
- The data is automatically encrypted and decrypted on the fly (right before it is saved and straight after it is loaded)
- No data can be read without using the password
- VeraCrypt volume works like a normal volume and interact with the OS the same way with the difference it has the encryption and decryption in between

Source:https://www.veracrypt.fr/en/Introduction.html

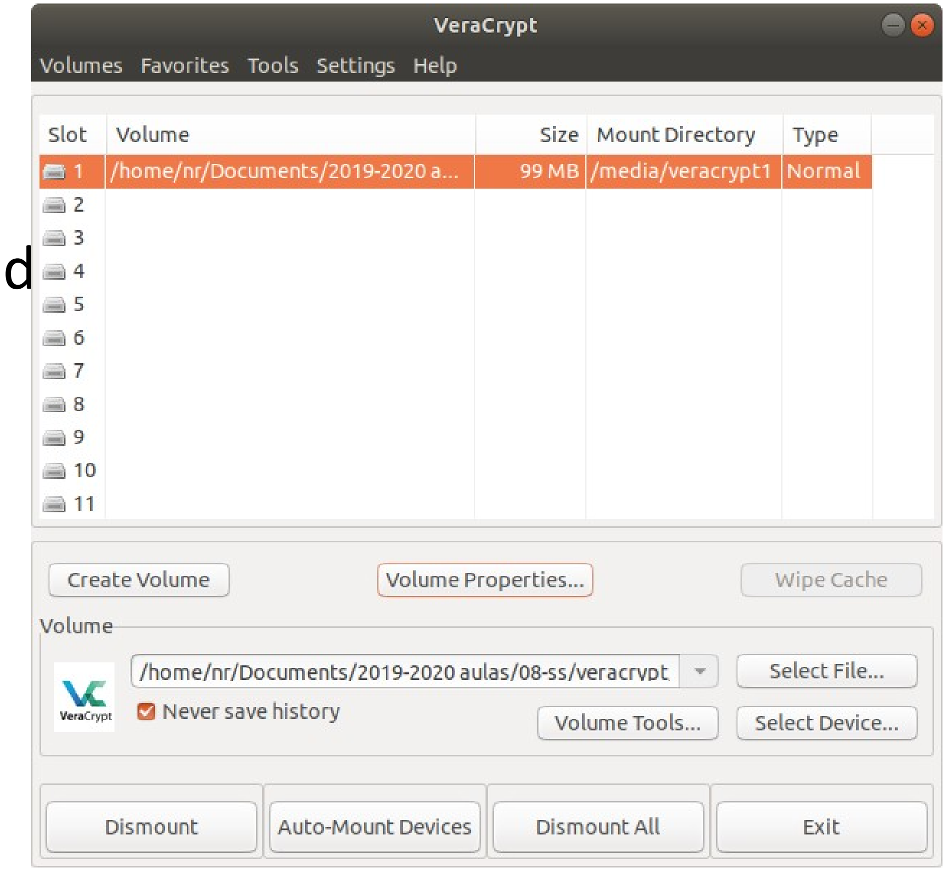
## VeraCrypt

- VeraCrypt does not save any decrypted data to a disk, just uses RAM temporarily.
- When the OS starts, the volume will be mounted, the disk volume will continue to be encrypted

Source:https://www.veracrypt.fr/en/Introduction.html

## VeraCrypt

- Download
  - https://www.veracrypt.fr/en/Downloads.html
  - GUI: veracrypt-1.24-Hotfix1-Ubuntu-18.04-amd64.d
- Installation instructions:
  - sudo apt update;
  - sudo apt upgrade
  - sudo dpkg -i veracrypt-1.24-Hotfix1 -Ubuntu-18.04-amd64.deb



| Slot | Volume | Size | Mount Directory | Type |
|---|---|---|---|---|
| 1 | /home/nr/Documents/2019-2020 a... | 99 MB | /media/veracrypt1 | Normal |
| 2 | | | | |
| 3 | | | | |
| 4 | | | | |
| 5 | | | | |
| 6 | | | | |
| 7 | | | | |
| 8 | | | | |
| 9 | | | | |
| 10 | | | | |
| 11 | | | | |

Volume: /home/nr/Documents/2019-2020 aulas/08-ss/veracrypt

✓ Never save history

## Integrity Functions

- Integrity is an essential security service on which other security services such as authentication or nonrepudiation depend on
- The most common integrity functions are **hash algorithms**
- Integrity checking functions calculate a fixed length value (between 128 and 512 bits depending on implementations) that should always be the same over the same message
- If the message is changed the calculated value is completely different
- These functions are considered cryptographic, although they cannot be used to encrypt or decrypt.

## Integrity Functions

- Most common algorithms:
  - Md5 (128 bits, no longer safe) → md5sum
  - sha1 (160 bits, no longer safe) → sha1sum
  - sha256 (256 bits, todays standard) → sha256sum
  - sha512 (512 bits) → sha512sum
  - whirlpool (512 bits) → whirlpooldeep
  - blake2 (512 bits) → b2sum

## Integrity Functions

- `sha256sum` is used to verify data integrity using the SHA-256 algorithm
- Example:
  - `sha256sum *.txt > SHA256`
  - `cat SHA256`
  - `sha256sum -c SHA256`
    - `-c` Reads the SHA256 sums from the FILE and check them.
- Exercise:
  - Crack this SHA256:
    - `SHA256(%&xxxx)=351b99fb7db4d3cd6f6f8f43fb8c68b1d4a988ed624a8e1e9ad66880cc963469`
    - `xxxx` is a value from `0000` to `9999`
    - Tip: use python or perl