

## Kryptos v1.0

### Main Menu

#### Status

Three submenus:

- 1- Sender: - to perform encryption, signature and hash operations.
- 2- Receiver: - to perform decryption, verification and hash operations.
- 3- Exit: - exit the application

#### Operation:

Sender Operations:

- 1- Encrypt: - to encrypt files using symmetric ciphers.
- 2- Public Key Encrypt: - to encrypt files using asymmetric ciphers.
- 3- Sign: - to sign files.
- 4- Hash Generate: - to hash files.
- 5- Mac Generate: - to generate Message Authentication Code.

Receiver Operations:

- 1- Decrypt: - to decrypt files using symmetric ciphers.
- 2- Public Key Decrypt: - to decrypt files using asymmetric ciphers.
- 3- Verify: - to verify signed files.
- 4- Hash Verify: - to verify hash files.
- 5- Mac Verify: - to verify Message Authentication Code.

**NOTE:** in the case of asymmetric decryption and verification operations, the user will not be allowed to generate keys. The keys for these operations can only be generated during Encryption and/Or signature operations.

#### Algorithm:

- 1- Symmetric algorithms: in order to encrypt files using one of the symmetric key ciphers the user must be a [Sender] and the [Operation] must be [Encrypt]. To decrypt the user must be a [Receiver] and the [Operation] must be [Decrypt]. The program supports the following ciphers:
  - DES
  - IDEA
  - MARS
  - RC5
  - RC6
  - Rijndael
  - Serpent
  - Triple DES
  - TwoFish
- 2- Asymmetric algorithms: in order to encrypt files using an asymmetric key cipher the user must be a [Sender] and the Operation must be [Public Key Encrypt]. To decrypt the user must be a [Receiver] and the [Operation] must be [Public Key Decrypt]. The program supports the following asymmetric key ciphers:
  - RSA: The program supports OAEP-MGF1 padding, PKCSv1.5 padding and raw (no padding) RSA.
- 3- Signature/verification algorithms: in order to sign a file the user must be [Sender] and the [Operation] must be [Sign]. To verify the user must be [Receiver] and the [Operation] must be [Verify]. The program supports the following signature/verification algorithms:
  - DSA

- ECDSA: the user specifies the type of ECDSA, EC2N or ECP.
- RSA
- 4- Hash algorithms: the user generates Hash if in the [Sender] mode and verifies Hash in the [Receiver] mode, the Operation is either [Hash Generate] or [Hash Verify]. The program supports the following hash algorithms:
  - HAVAL
  - MD2
  - MD5
  - PanamaHash
  - RIPEMD160
  - SHA-1
  - SHA-256
  - SHA-384
  - SHA-512
  - Tiger
- 5- MAC Algorithms: currently the program supports the following MAC algorithms
  - MD5MAC
  - HMAC

#### Algorithm Parameters:

Before using an algorithm the user must explicitly specify the algorithm parameters.

- 1- Symmetric key cipher parameters: the user specifies the cipher's block size, key size and number of rounds. In some cases one of the parameters may be fixed depending on the ciphers implementation.
- 2- Asymmetric key cipher parameters:
  - a. RSA: (all padding schemes) the user specifies the following parameters
    - i. length of the modulus: -
      1. OAEP-MGF1: minimum size of the modulus of RSA keys 337 bits, maximum size 4096 bits.
      2. PKCSv1.5: minimum size of the modulus of RSA keys 89 bits, maximum size 4096 bits.
      3. RAW: minimum size of the modulus of RSA keys 32 bits, maximum size 4096 bits.
    - ii. The public exponent: the user has 3 choices:
      1. 3:  $e = 3$ .
      2. 17:  $e = 17$
      3.  $2^{16}+1$ :  $e = 65537$
      4. random: here the user specifies the public exponent's number of bits, and the program generates e randomly. The number of bits must be between 3 and the size of the modulus.
- 3- Signature parameters:
  - a. DSA: the size of the modulus must be between 192 and 1024. The default is 1024.
  - b. ECDSA: the user chooses the curve from a list of NIST recommended curves.
  - c. RSA: same as Asymmetric Key cipher.
- 4- Verification parameters: same as signature parameters except for RSA, there is no need to specify any parameters, the algorithm parameters are extracted from the public key.
- 5- Hash parameters: no algorithm parameters needed.
- 6- MAC parameters: no algorithm parameters needed, the output length of the current implemented algorithm is the default length (16 bytes for both MD5MAC and HMAC).

## Mode:

### 1- Symmetric key ciphers:

IV generation: the IV size depends on the cipher's block size. The user can save the generated IV for latter use.

IV loading: the user will not be allowed to load an IV with size different than the underlying cipher block size. Supported modes:

- a. ECB: default
- b. CBC
- c. CBC\_CTS.
- d. CFB: the user is allowed to choose the feedback size.
- e. CTR: feedback size fixed to the cipher block size.
- f. OFB: feedback size fixed to the cipher block size.
- g. OCB

### 2- Asymmetric key ciphers:

IV Generation: The IV generated will always be of the same size as the modulus.

IV loading: the user is allowed to load an IV as long as its size is greater than the cipher's modulus size. Supported modes:

- a. ECB
- b. CBC

## Key

1- Symmetric key ciphers: the user is allowed to generate, save or load cipher keys through this menu. In the case of DES, the user is allowed to generate weak and semi-weak keys.

2- Asymmetric key ciphers: the user is allowed to generate, save or load RSA keys.

- a. Key generation: according to user choices of the modulus and public key exponent, the program generates both RSA public and private keys. If the user decides to save the key pair, he will be asked to choose a name for the private key, both keys will be saved under the same name with different extensions, <name>.public for public key and <name>.private for the private key.
- b. Key load: the user is allowed to load saved RSA public key during encryption and saved RSA private key during decryption.

### 3- Sign:

- a. DSA
  - i. System Parameters: if the user is a [Sender], he is allowed to generate, save and load system parameters which are used to generate DSA keys.
  - ii. Key generation: the user is allowed to generate DSA keys if he is a [Sender]. The key is generated from loaded system parameters. If the user did not load or generate system parameters, the program will generate both the system parameters and DSA keys.
- b. EDSA:
  - i. Key generation: if the user is a [Sender] (signer) he must choose the recommended curve before generating the ECDSA keys. If the user is a [Receiver] (verifier), he is only allowed to load ECDSA key.
- c. RSA: same as RSA cipher.

4- Verify: if verifying, the user is only allowed to load public keys for verification.

5- MAC Generate/Verify: the user is allowed to generate, save or load a key.

## Input/Output:

1- Symmetric and Asymmetric cipher encryption: the user chooses the input file and the output (ciphertext) file.

- 2- Symmetric and Asymmetric cipher decryption: the user chooses the ciphertext file and the output (decrypted message) file.
- 3- Signing: the user chooses the file to be signed and the output file (signature file).
- 4- Verifying: the user chooses the message file and the signature file to be verified.
- 5- Hash Generate: the user chooses the message file to be hashed and the output file.
- 6- Hash Verify: the user chooses the message file and the hash file to be verified.
- 7- MAC Generate: the user chooses the message file and the output file.
- 8- MAC Verify: the user chooses the message file and the MAC file to be verified.