Kryptophone X
Security Features

1. Index of Content

2. Description

This document describes the security features of Kryptophone X.

We want to be very, very clear, KRYPTOPHONE X gives you **REAL end-to-end encryption** and here we are going to explain how such difficult work is properly done.

We have read about multiple applications claiming end-to-end encryption to discover later , it was not real end-to-end and the provider’s servers were storing everything as clear data. (**Whatsapp, Zoom and Encrochat** just for example).

Some new companies claim to have invented new encryption algorithms (that no one has verified). An encryption algorithm takes years to become really reliable and considered as valid option to secure communication and data, so? Marketing departments like to tell stories… ☺

Our company has been established in 2010, anyway we do not invent new encryption algorithms; we leave that mission to few elected teams of mathematical geniuses.

Kryptotel deploys the existing well known and tested algorithms in a smart way that give YOU and us peace-of-mind.

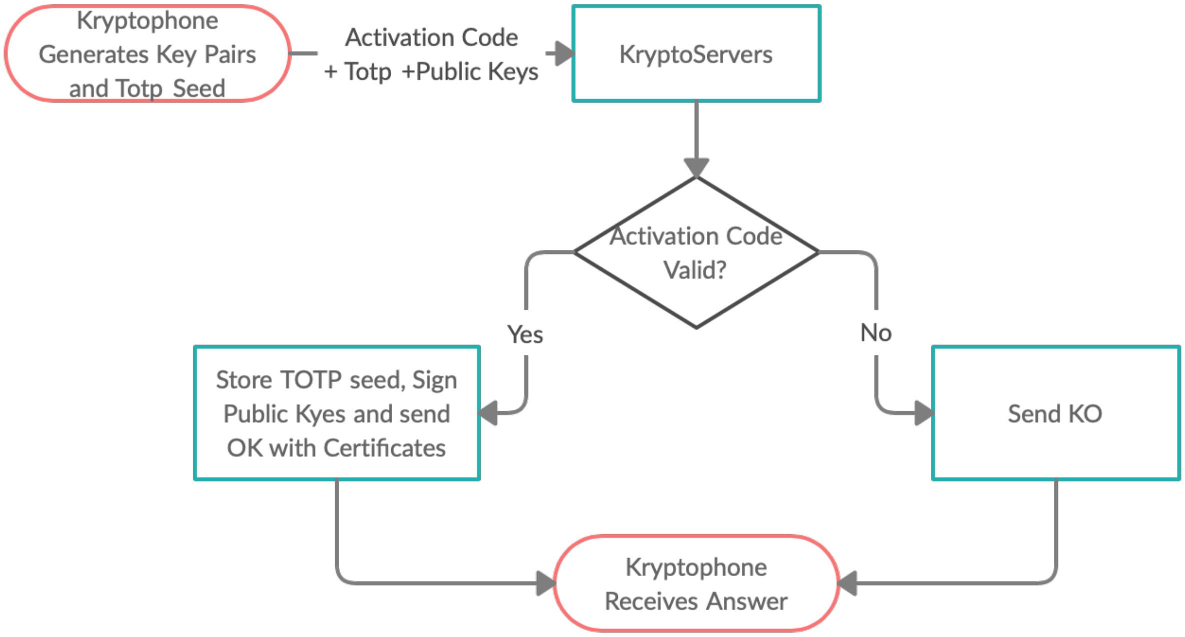
3. Keys Generation

At the first start of your Kryptophone, you will be asked to insert the activation code that you have received by our secure channel.
Your Kryptophone, will generate:
1) Private Keys and Public Keys (Elliptic Curve Sec521r1, one key pair for encryption and one key pair for authentication)
2) Time One Time Password Seed (used for second factor authentication)

The following data will be forwarded to the Krypto Servers:
1) Public Keys
2) Totp Seed
3) Activation Code
All encrypted and signed with the authentication private key. (We will explain later how it's done the encryption and signing).

If the data is valid, the Registration Server will generate and store 2 certificates for the Kryptophone, basically the certificates are the public keys signed from the internal Certification Authority.

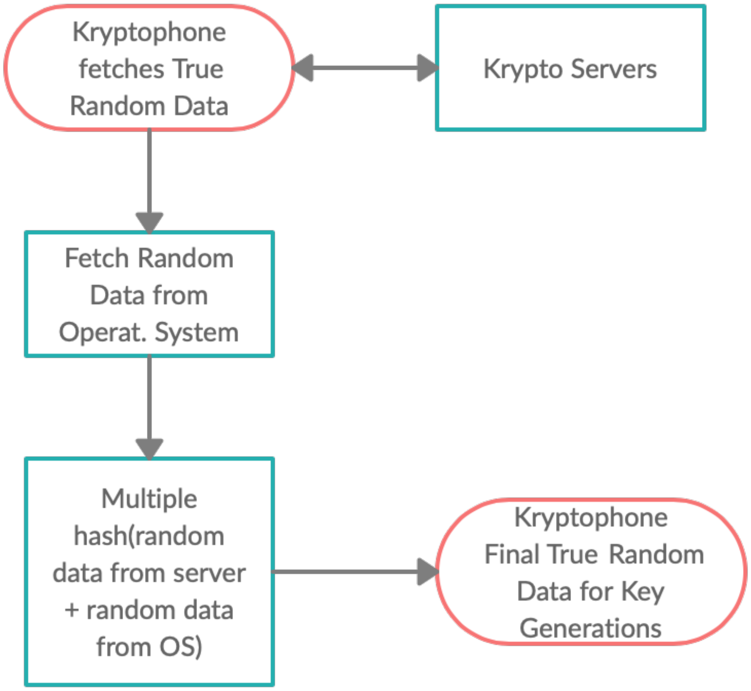
Here a simple data flow:


Kryptophone Generates Key Pairs and Totp Seed
Activation Code + Totp +Public Keys
KryptoServers
Activation Code Valid?
Yes
No
Store TOTP seed, Sign Public Kyes and send OK with Certificates
Send KO
Kryptophone Receives Answer

4. True Random Data

To generate proper key pairs, the mobile device needs a reliable source of unpredictable random data.
The operating system (IOS) gives a very reliable source of random data, anyway your Kryptophone increases the randomness applying up to 8 million hashes (SHA3) to the random data obtained from the operating system and additional random data supplied from the servers that have True Random data generators on board.

Here a design how the random data is collected:


Kryptophone fetches True Random Data
Krypto Servers
Fetch Random Data from Operat. System
Multiple hash(random data from server + random data from OS)
Kryptophone Final True Random Data for Key Generations

5. Encryption of Private Keys

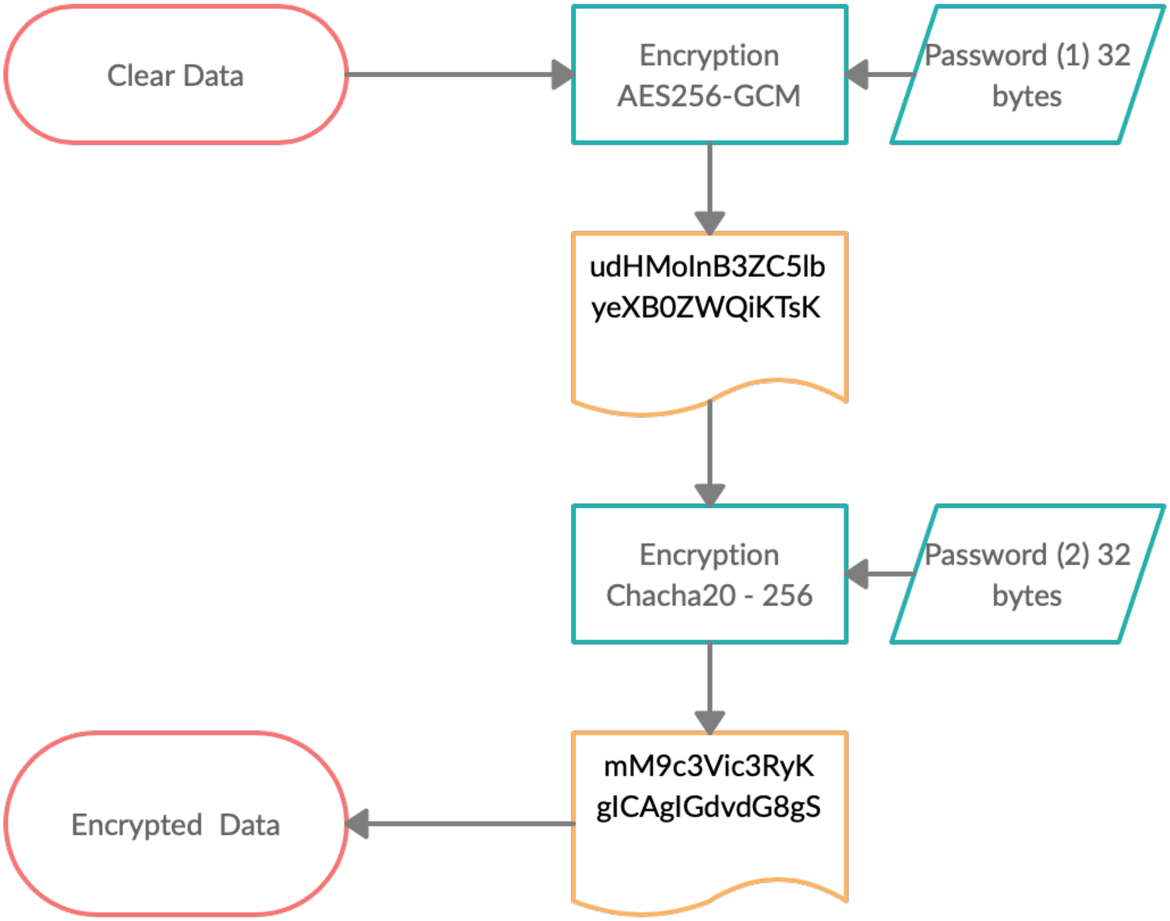
For security reasons, we do not store the keys pair and the Totp seed on the local storage as clear data.
Kryptophone will ask you to setup a “Master Password” (minimum 8 chars).
Such password will be requested every time the app starts.

The Kryptophone uses your Master Password to expand it to a stronger password of 64 bytes (512 bits) to encrypt your keys pairs and Totp Seed.

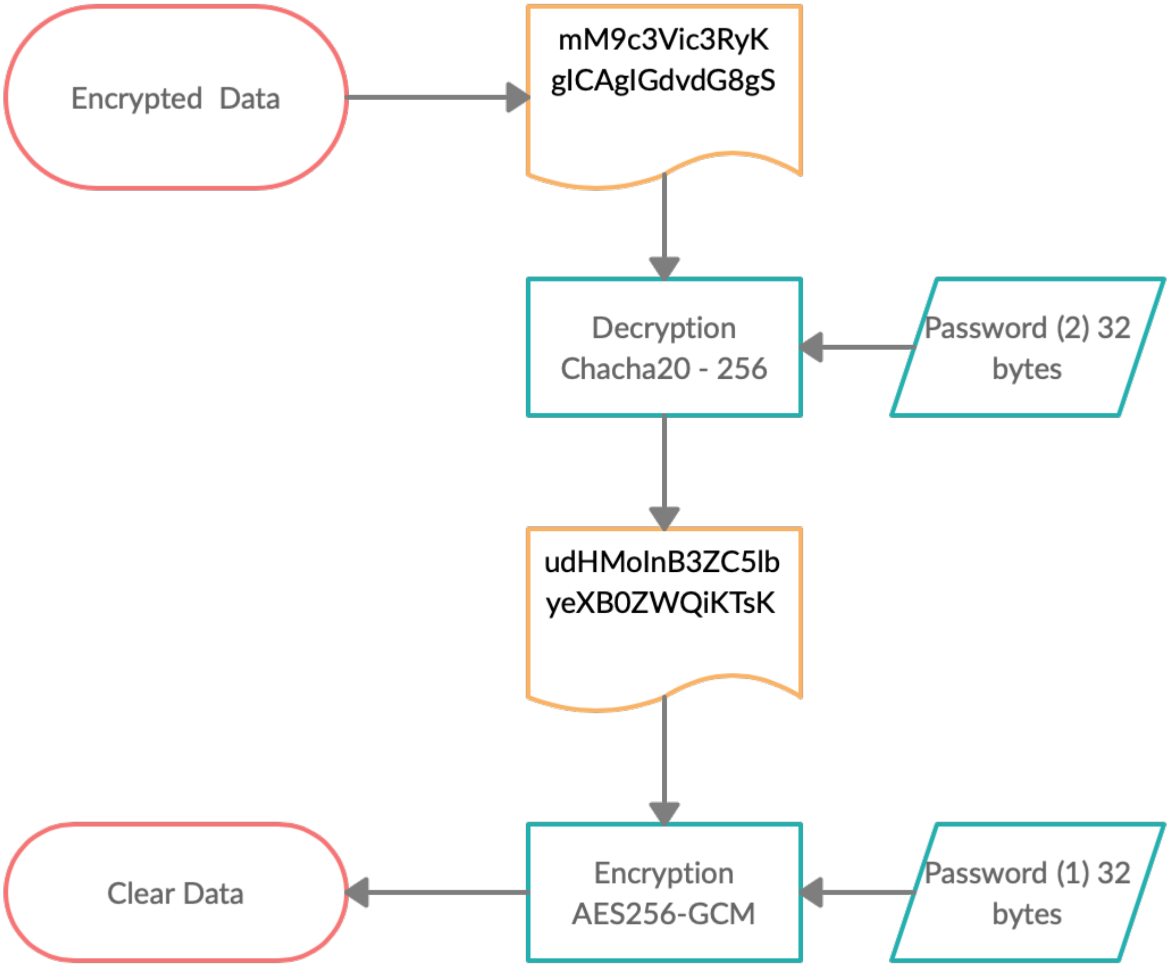
The encryption is applied with 2 layers by:
1) AES – 256 bits with GCM
2) Chacha20 - 256 bits

Be careful, if you forget your master password, your Kryptophone is completely gone, no more usable, no way to recover the access.

Here the design how the double layer encryption works:



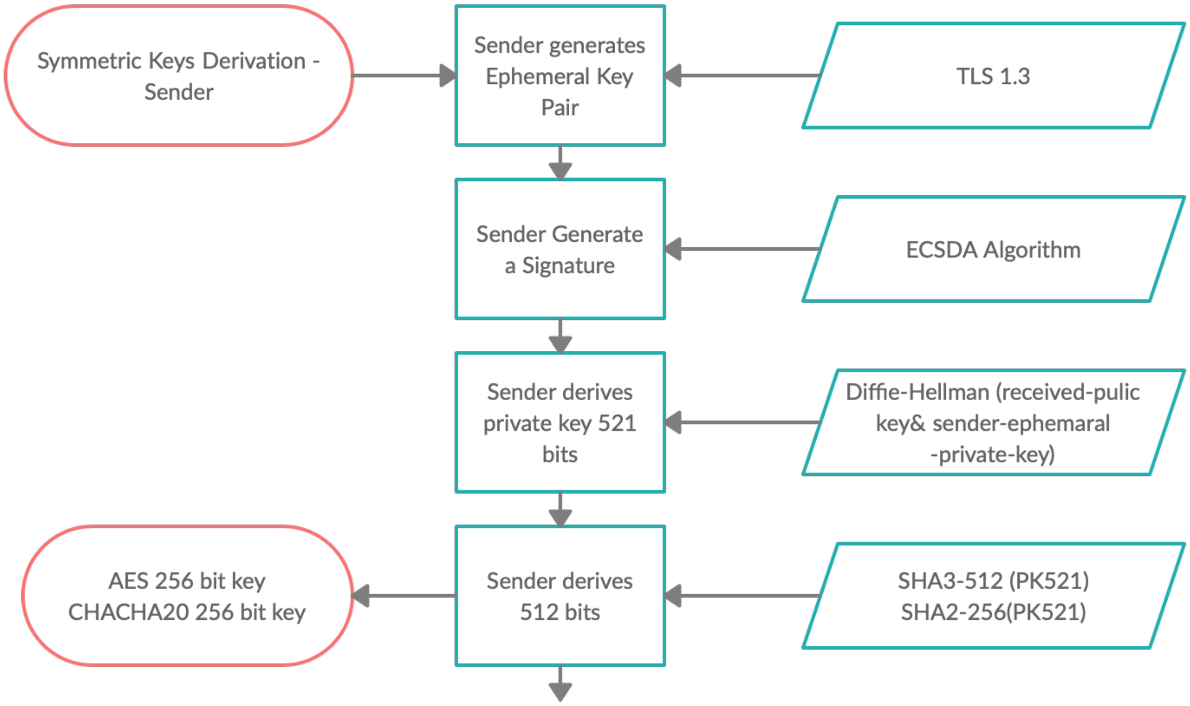
And here the decryption process:



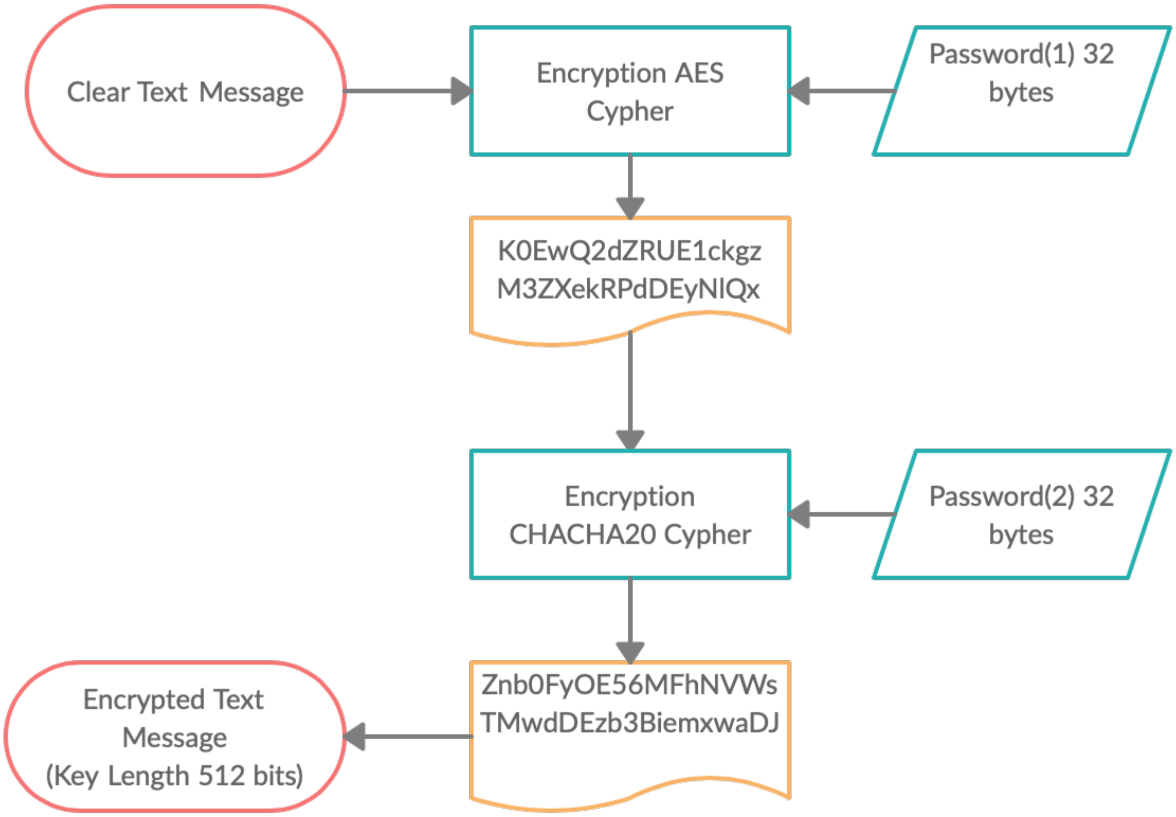
6. Message Sending

To send a message, your Kryptophone need a shared session key with the recipient with a length of 512 bits (256 bits for AES and 256 bits for Chacha20).Here the asymmetric encryption plays his proper role.
The Kryptophone will get the public key from a signed certificate of the recipient (from cache or server request over TLS), will generate an ephemeral keys pair by Elliptic Curve SEC521R1 and will derive a session key using Diffie-Hellman Algorithm.

Here the design to illustrate this **phase 1 (session key agreement):**


Symmetric Keys Derivation - Sender
Sender generates Ephemeral Key Pair
TLS 1.3
Sender Generate a Signature
ECSDA Algorithm
Sender derives private key 521 bits
Diffie-Hellman (received-pulic key& sender-ephemaral -private-key)
AES 256 bit key CHACHA20 256 bit key
Sender derives 512 bits
SHA3-512 (PK521) SHA2-256(PK521)

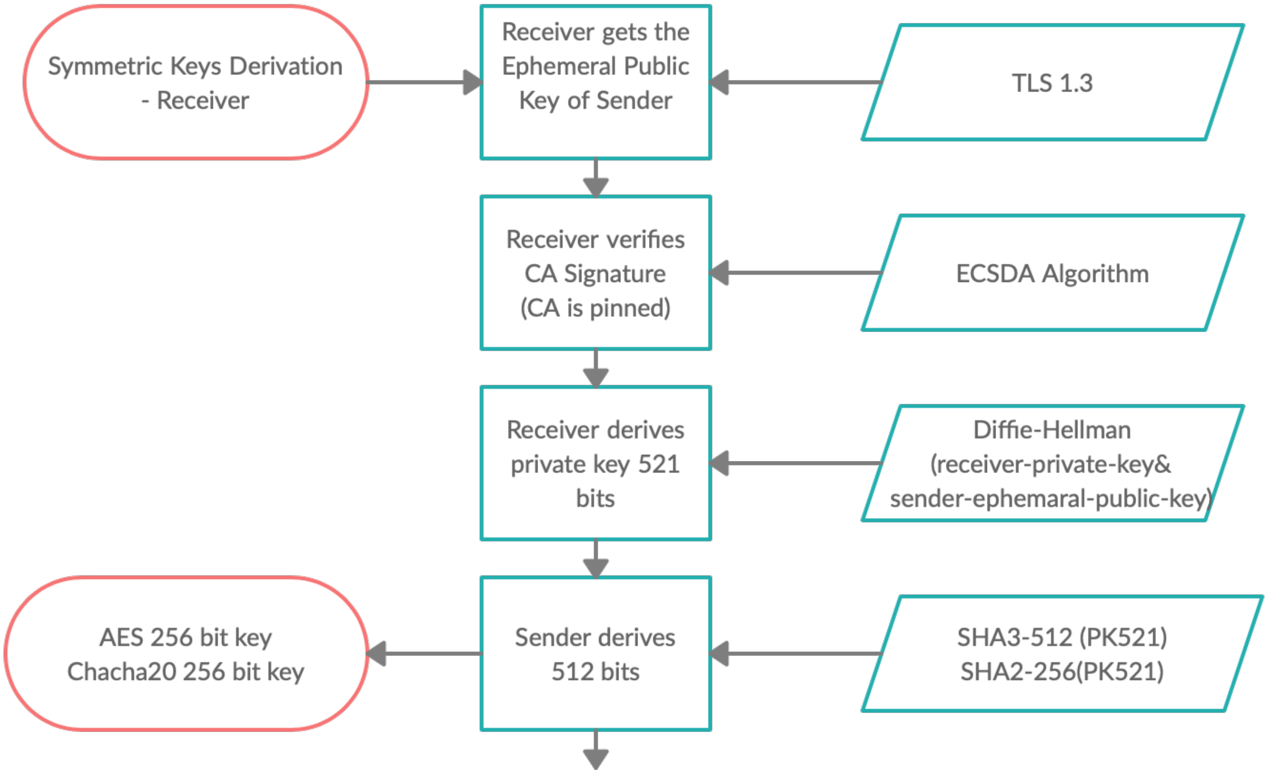
Once the sender has the session keys above the message body can be encrypted using symmetric algorithms, AES 256 bits and Chacha20 256 bits.
Here the **phase 2 (encryption process):**


Clear Text Message
Encryption AES Cypher
Password(1) 32 bytes
K0EwQ2dZRUE1ckgz M3ZXekRPdDEyNlQx
Encryption CHACHA20 Cypher
Password(2) 32 bytes
Znb0FyOE56MFhNVWs TMwdDEzb3BiemxwaDJ
Encrypted Text Message (Key Length 512 bits)

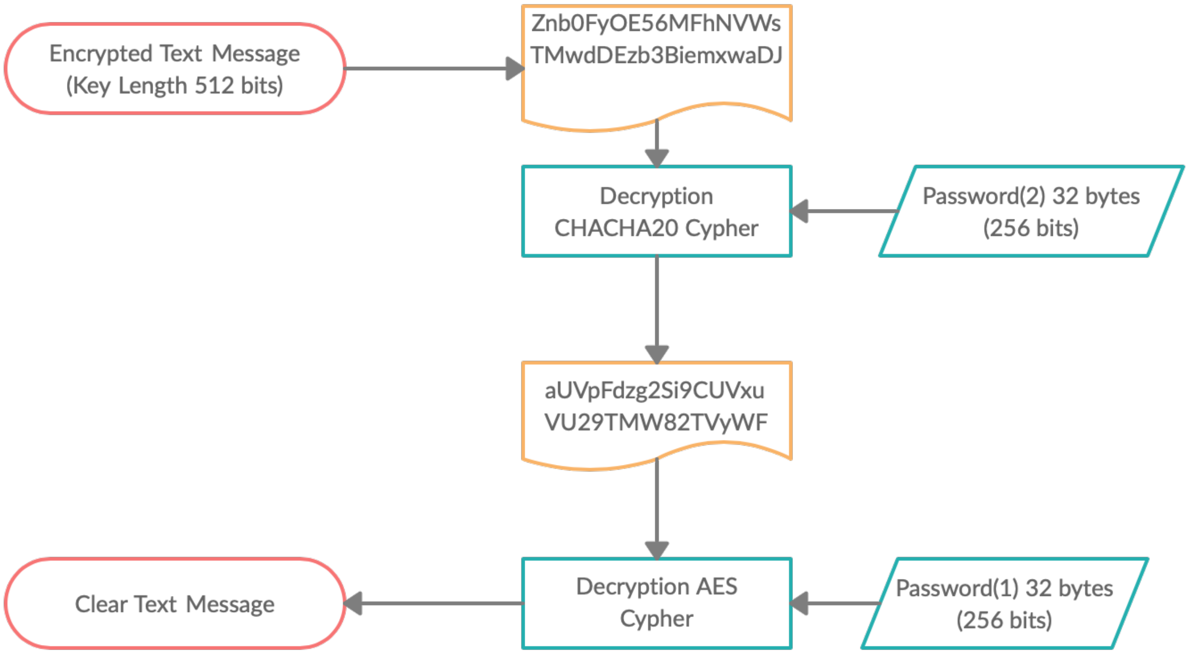
The encryption body is packaged with the ephemeral public key generated for the message and the whole block is signed with the authentication private key of the sender.

7. Message Receiving

The Kryptophone receiver make the same process to obtain a shared session key with the sender. It gets the ephemeral public key of the sender, the one received with the message and apply Diffie-Hellman algorithm to obtain the same 512 bits session key. Here the design to illustrate this **phase 1 (session key agreement):**

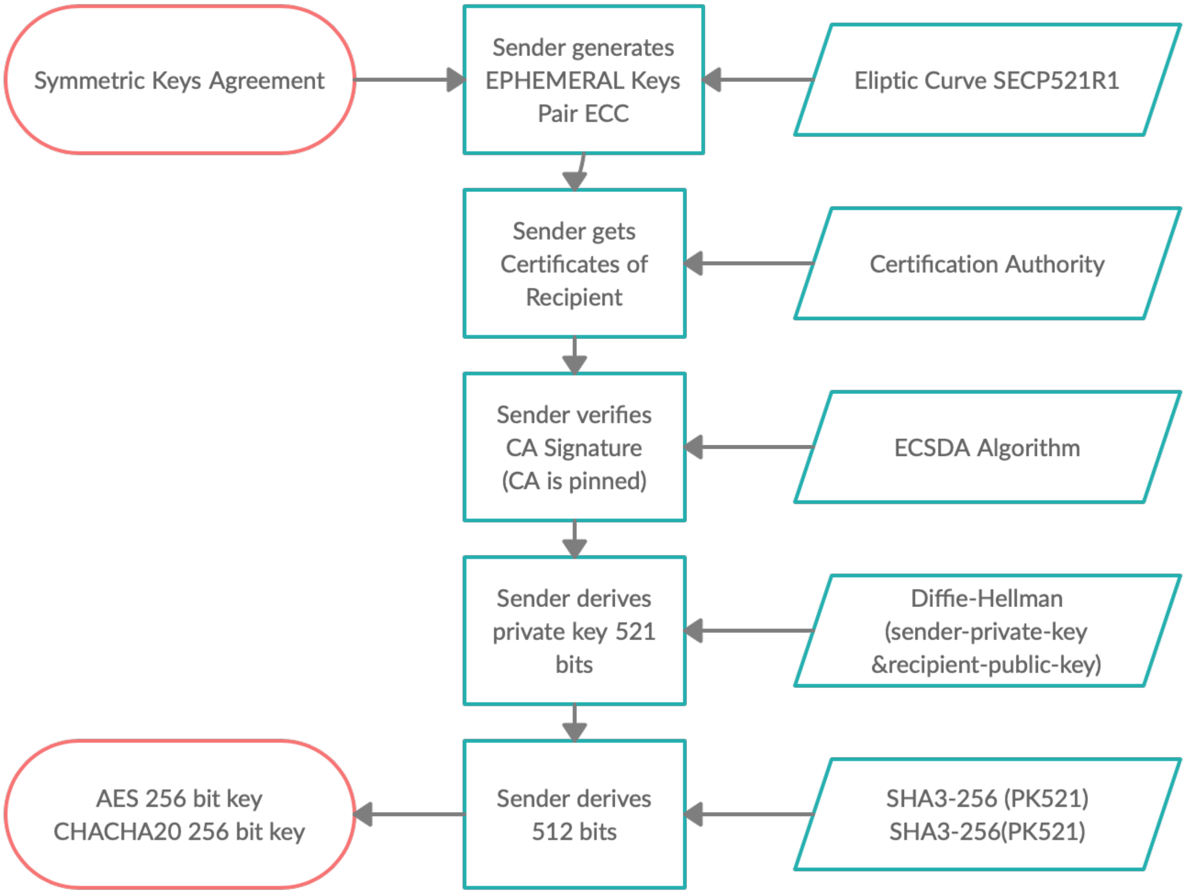

Symmetric Keys Derivation - Receiver
Receiver gets the Ephemeral Public Key of Sender
TLS 1.3
Receiver verifies CA Signature (CA is pinned)
ECSDA Algorithm
Receiver derives private key 521 bits
Diffie-Hellman (receiver-private-key& sender-ephemaral-public-key)
AES 256 bit key Chacha20 256 bit key
Sender derives 512 bits
SHA3-512 (PK521) SHA2-256(PK521)

Once the sender has the session keys above the message body can be decrypted using symmetric algorithms, AES 256 bits and Chacha20 256 bits.
Here the **phase 2 (decryption process):**

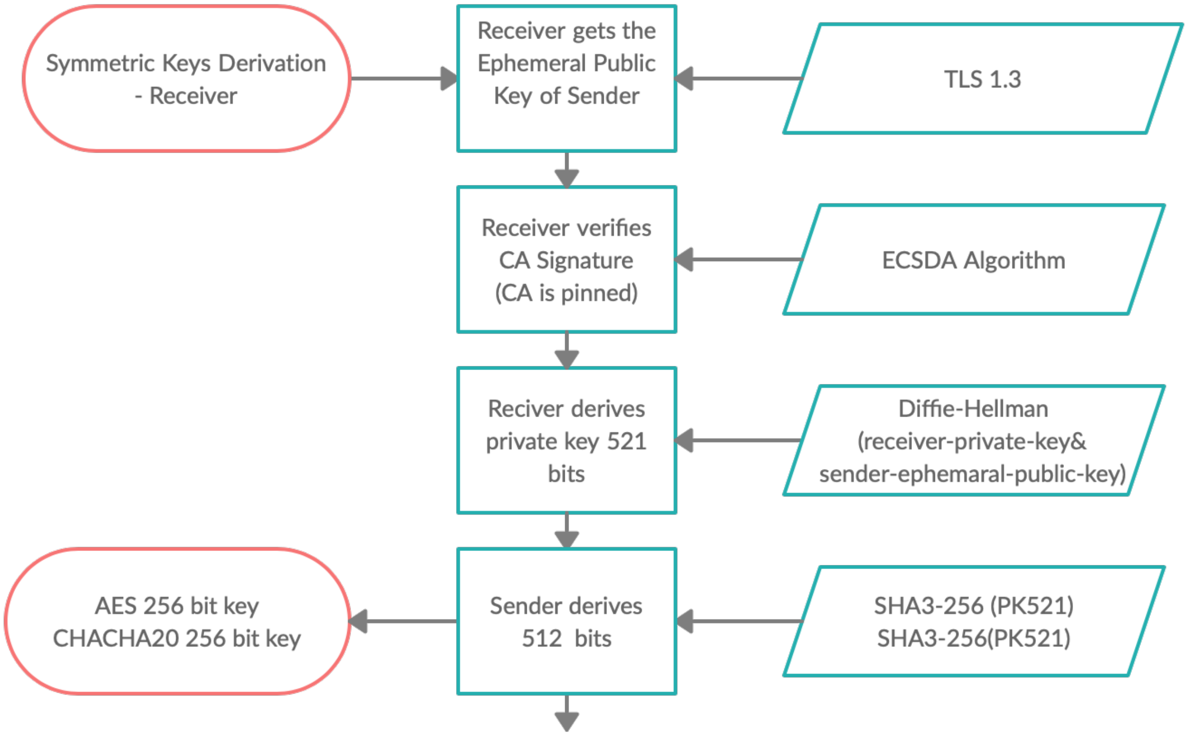


8. Voice/Video Call

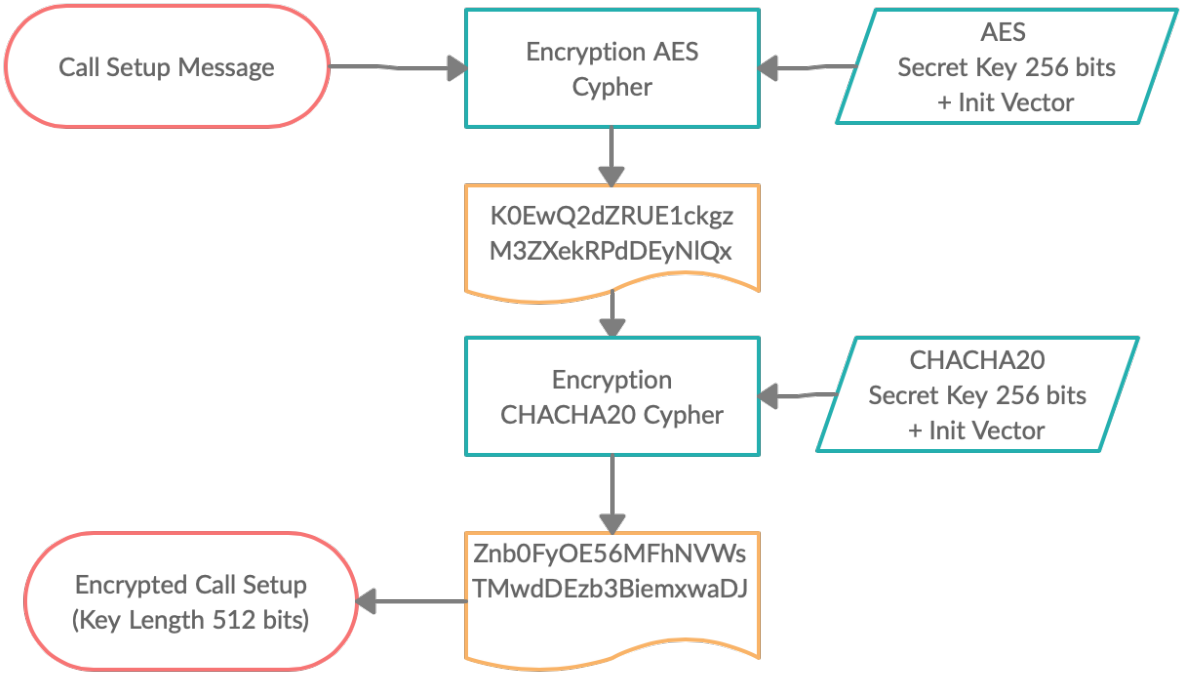
The voice/video calls work in the same way for phase 1 (Session Key Agreement), here the data flow for the “Caller” or “Sender”:


Symmetric Keys Agreement
Sender generates EPHEMERAL Keys Pair ECC
Eliptic Curve SECP521R1
Sender gets Certificates of Recipient
Certification Authority
Sender verifies CA Signature (CA is pinned)
ECSDA Algorithm
Sender derives private key 521 bits
Diffie-Hellman (sender-private-key &recipient-public-key)
AES 256 bit key
CHACHA20 256 bit key
Sender derives 512 bits
SHA3-256 (PK521)
SHA3-256(PK521)

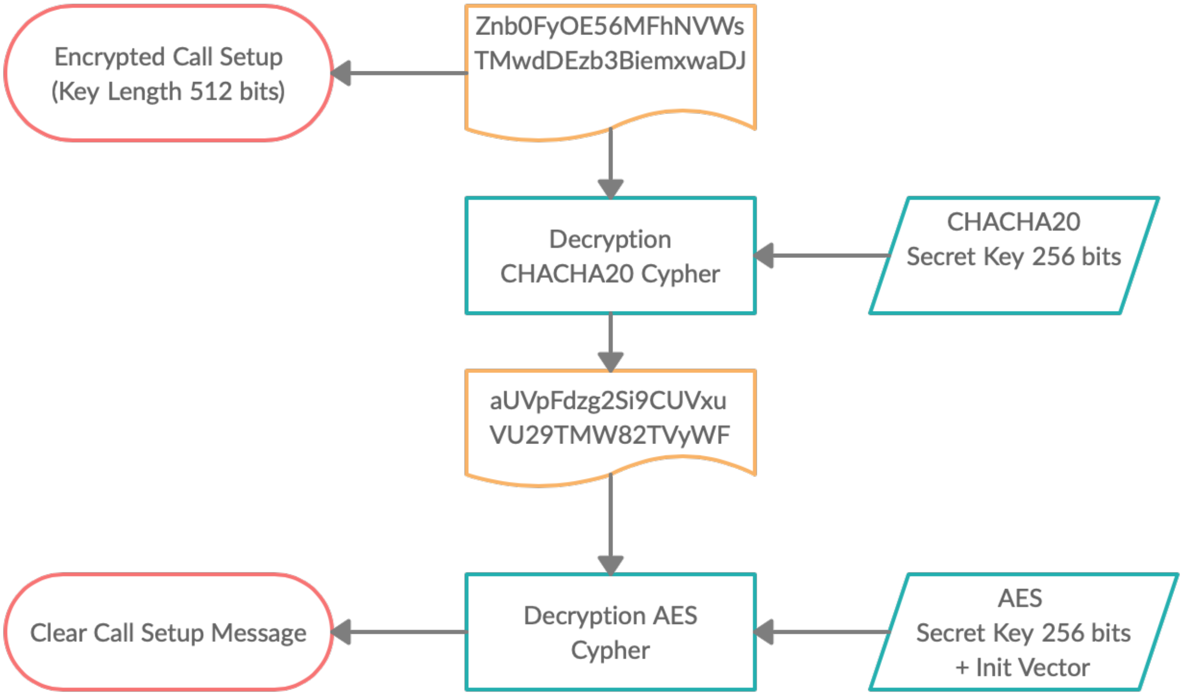
And here the data flow for the “Called” or receiver:


Symmetric Keys Derivation - Receiver
Receiver gets the Ephemeral Public Key of Sender
TLS 1.3
Receiver verifies CA Signature (CA is pinned)
ECSDA Algorithm
Reciver derives private key 521 bits
Diffie-Hellman (receiver-private-key& sender-ephemaral-public-key)
AES 256 bit key CHACHA20 256 bit key
Sender derives 512 bits
SHA3-256 (PK521) SHA3-256(PK521)

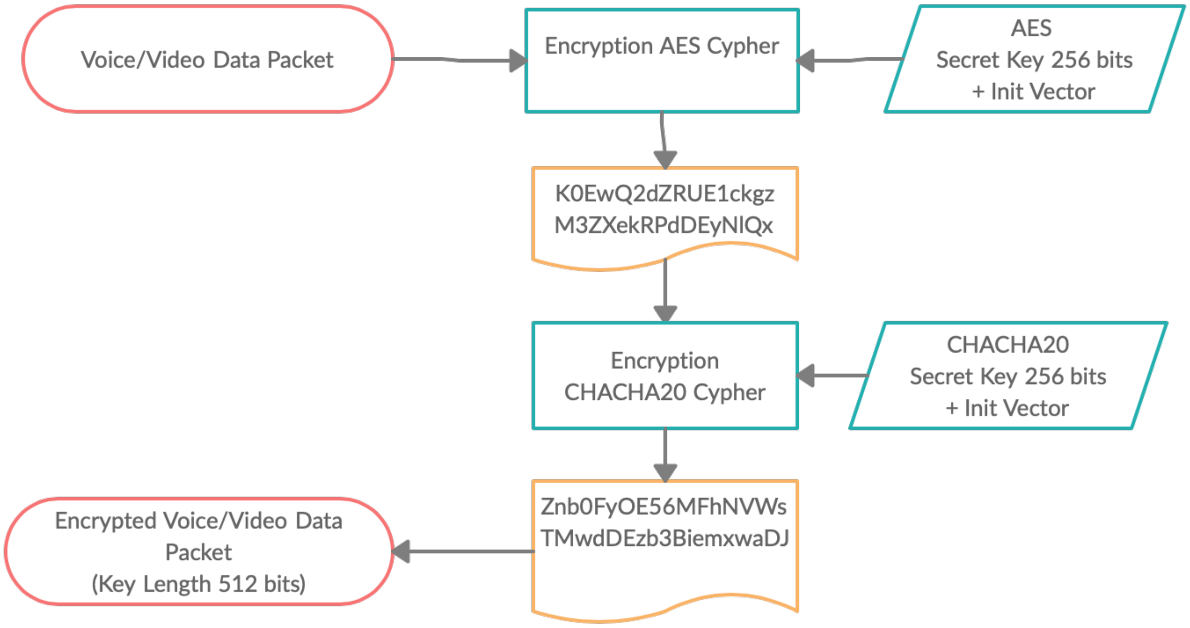
Once the key agreement is done, the phase 2 Call Setup starts as follow:


Call Setup Message
Encryption AES Cypher
AES Secret Key 256 bits + Init Vector
K0EwQ2dZRUE1ckgz M3ZXekRPdDEyNlQx
Encryption CHACHA20 Cypher
CHACHA20 Secret Key 256 bits + Init Vector
Encrypted Call Setup (Key Length 512 bits)
Znb0FyOE56MFhNVWs TMwdDEzb3BiemxwaDJ

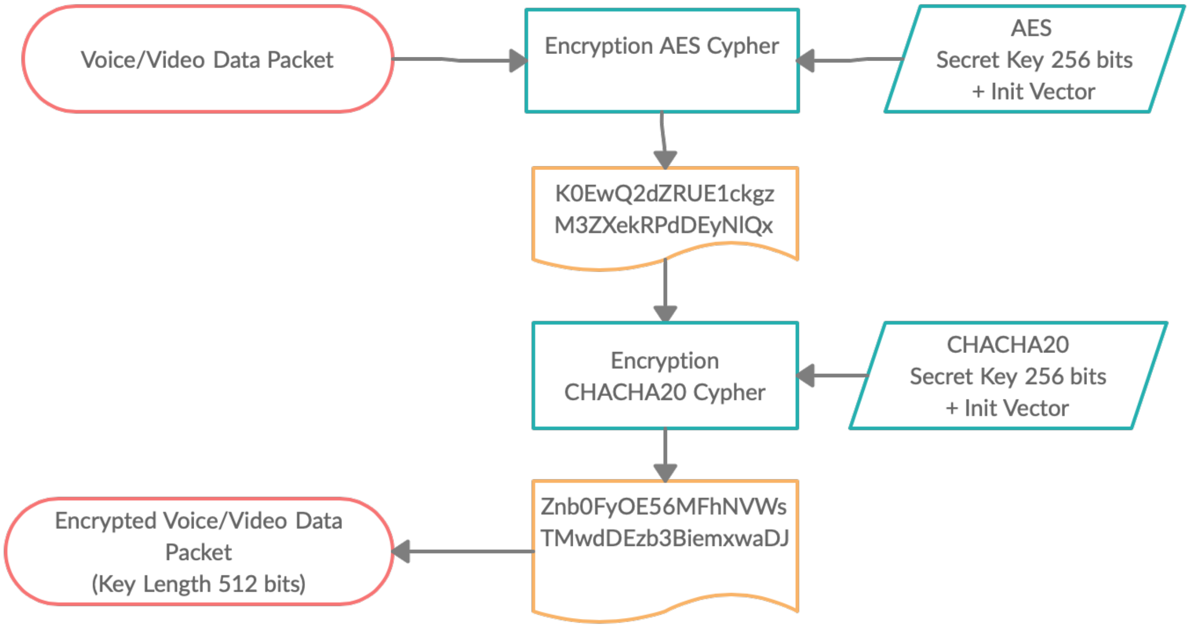
And here is the call setup for the receiver:



If the called/recipient answer the call, **phase 3 (voice/video packets flow) starts as follows:**



And here the data flow for receiving channel (in effect each party send and received packets at the same time):



9. Encrypted Data-At-Rest

The application does NOT use any local database. All the messages are encrypted with the private keys of the Kryptophone, and they are stored in the Krypto Servers. They are decrypted and kept in RAM when app shows them to you.
The Krypto Servers do not have the private keys to decrypt the messages that have been generated and kept in your Kryptophone. If you lose the phone you can use the “Wipe Link” supplied, to remove all the encrypted data from the Krypto Server and block the Kryptophone from any further usage.

Files sent/received are stored in the internal cache of the app and at the Krypto Servers. Each file is stored in an encrypted format with 2 layers of encryption (CHACHA20 and AES). The key required of 512 bits the Init Vectors and GCM tags are stored inside the body of the message (RAM) where each file is attached.
During the execution of the app, the files are decrypted inside the application local cache folder to be shown to the user.

Once the app goes in background and when it’s closed, the non-encrypted files are removed. iOS operating system keeps the whole storage fully encrypted.

10. Microphone protection

The application gets exclusive access to the microphone when it’s required the usage like in an audio or video call. The exclusive access is managed from the operating system.

11. Camera protection

The application works on exclusive access of the camera when the app is using it. The exclusive access is managed from the operating system

12. Secure Delete (RAM)

Variables in the crypto library developed for the solution, are wiped before the release.

13. Anti DNS Poisoning

Kryptophone does not use DNS to avoid re-routing, we use static ip addresses.

14. Anti TLS-Session Hijacking

Kryptophone make a proper pinning to public key of the servers, it does not trust and Certification Authority outside the internal one.

15. Vpn Integrated (optional activation)

Kryptophone has an integrated Vpn module that can keep hidden your ip address and even let work the app where could be blocked. When activated, it adds a further layer of encryption to your communication. You can connect the Vpn with a single tap in any moment.