Criptext

Criptext's end-to-end encryption system

Technical white paper

August 8, 2018

Contents

Introduction

This white paper provides a technical explanation of Criptext's end-to-end encryption system. Please visit Criptext's website at www.criptext.com/security for more information.

The Criptext Email platform allows people to confidently send emails and attachments.

Our encryption technology was designed using the open source Signal Protocol library and is designed to prevent third parties, including Criptext, from having access to your data. What's more, even if encryption keys from a user's device are ever physically compromised, they cannot be used to go back in time to decrypt previously transmitted emails.

This document gives an overview of the inner workings of Criptext

Terms

Public Key Types

- Identity Key Pair - A long-term Curve25519 key pair, generated at install time.
- Signed Pre Key - A medium-term Curve25519 key pair, generated at install time, signed by the Identity Key, and rotated on a periodic timed basis.
- One-Time Pre Keys - A queue of Curve25519 key pairs for one time use, generated at install time, and replenished as needed.

Session Key Types

- Root key - A 32-byte value that is used to create Chain Keys.
- Chain key - A 32-byte value that is used to create Email Keys.
- Email Key - An 80-byte value that is used to encrypt email contents. 32 bytes are used for an AES-256 key, 32 bytes for a HMAC-SHA256 key, and 16 bytes for an IV.

Client Registration

At registration time, a Criptext client transmits its public Identity Key, public Signed Pre Key (with its signature), and a batch of public One-Time Pre Keys to the server. The Criptext server stores these public keys associated with the user's device identifier. At no time does the Criptext server have access to any of the client's private keys.

Initiating Session Setup

To communicate with another Criptext user, a Criptext client first needs to establish an encrypted session. Once the session is established, clients do not need to rebuild a new session with each other, new sessions will be established when new devices are linked to the user account.

To establish a session:

- The initiating client ("initiator") requests the public Identity Key, public Signed Pre Key, and a single public One-Time Pre Key for the recipient.
- The server returns the requested public key values. A One-Time Pre Key is only used once, so it is removed from server storage after being requested. If the recipient's latest batch of One-Time Pre Keys has been consumed and the recipient has not replenished them, no One-Time Pre Key will be returned.
- The initiator saves the recipient's Identity Key as Irecipient, the Signed Pre Key as Srecipient, and the One-Time Pre Key as Orecipient.

- The initiator generates an ephemeral Curve25519 key pair, Einitiator.
- The initiator loads its own Identity Key as Iinitiator.
- The initiator calculates a master secret as master_secret = ECDH(Iinitiator, Srecipient) || ECDH(Einitiator, Irecipient) || ECDH(Einitiator, Srecipient) || ECDH(Einitiator, Orecipient)
  If there is no One Time Pre Key, the nal ECDH is omitted.
- The initiator uses HKDF to create a Root Key and Chain Keys from the master_secret.

Receiving Session Setup

After building a long-running encryption session, the initiator can immediately start sending emails to the recipient. Until the recipient responds, the initiator includes the information (in the header of all emails sent) that the recipient requires to build a corresponding session. This includes the initiator's Einitiator and Iinitiator.

When the recipient receives a message that includes session setup information:

- The recipient calculates the corresponding master_secret using its own private keys and the public keys advertised in the header of the incoming message.
- The recipient deletes the One-Time Pre Key used by the initiator.
- The initiator uses HKDF to derive a corresponding Root Key and Chain Keys from the master_secret.

Sending Emails

Clients will request from the server the necessary keys of each recipient to establish a session.

Internal

Once a session has been established, clients send emails that are protected with a Email Key using AES256 in CBC mode for encryption and HMAC-SHA256 for authentication.

The Email Key changes for each email transmitted, and is ephemeral, such that the Email Key used to encrypt an email cannot be reconstructed from the session state after an email has been transmitted or received.

The Email Key is derived from a sender's Chain Key that "ratchets" forward with every email sent. Additionally, a new ECDH agreement is performed with each email roundtrip to create a new Chain Key. This provides forward secrecy through the combination of both an immediate "hash ratchet" and a round trip "DH ratchet".

External

Clients will prompt the user for a password and create a pseudo-identity with all the required keys to establish an encrypted session for the external emails. This way, the clients will encrypt the emails with the Signal protocol, and the key bundle for these pseudo-identities will be encrypted using the password provided by the user. Both the encrypted email and the encrypted key bundle will be uploaded to the server, where they will stay up to 10 days after which they will be deleted.

The emails that are sent to these external users, will contain a link to Criptext where they'll be prompted for the previously established password. Once the correct password is entered, the client will be able to decrypt the key bundle, and request the necessary keys from the sender to establish an encrypted session and decrypt the email's content.

Sending Attachments

Attachments of any type are also end-to-end encrypted:

- The criptext user ("sender") sending an email, generates an ephemeral 32 byte AES256 key.
- The sender encrypts the attachment with the AES256 key in CBC mode with a random IV.
- The sender uploads the encrypted attachment to the Criptext servers.
- The sender transmits the encrypted email to the recipient that contains the encryption key and the pointer to the file in the Criptext servers.
- The recipient decrypts the email, retrieves the encrypted file from our servers and decrypts the file.

Linking new Devices

When linking new devices, based on the user's email, the server will send a notification to all of the previously verified devices, prompting the user to Approve or Deny the new device

When a device is denied, it will be added to a blacklist for that user, and subsequent requests will be ignored by the server.

When a new device is approved, the following occurs:

- The new device will generate a temporal key bundle and upload it to the server.
- The device which approved it, will establish an encrypted session with the new device, and start sending differential backups of the existing mail-box.
- The new device will receive each differential backup, decrypt it and store the emails locally.

This process will continue in the background until there are no more differential backups.

Verifying Keys

Criptext users additionally have the option to verify the keys of the other users with whom they are communicating so that they are able to confirm that an unauthorized third party (or Criptext) has not initiated a man-in-the-middle attack. This can be done by scanning a QR code, or by comparing a 60-digit number.

The QR code contains:

- A version.
- The user identifier for both parties.
- The full 32-byte public Identity Key for both parties.

When either user scans the other's QR code, the keys are compared to ensure that what is in the QR code matches the Identity Key as retrieved from the server.

The 60-digit number is computed by concatenating the two 30-digit numeric fingerprints for each user's Identity Key. To calculate a 30-digit numeric fingerprint:

- Iteratively SHA-512 hash the public Identity Key and user identifier 5200 times.
- Take the first 30 bytes of the final hash output.
- Split the 30-byte result into six 5-byte chunks.
- Convert each 5-byte chunk into 5 digits by interpreting each 5-byte chunk as a big-endian unsigned integer and reducing it modulo 100000.
- Concatenate the six groups of five digits into thirty digis.

This will be not be available in the initial release of Criptext, but will be added in a subsequent release.

Conclusion

Emails sent using Criptext are protected with an end-to-end encryption protocol so that third parties and Criptext cannot read them and so that the emails can only be decrypted by the recipient. All emails and attachments sent by Criptext are protected by end-to-end encryption.

Criptext servers do not have access to the private keys of Criptext users, and Criptext users have the option to verify keys in order to ensure the integrity of their communication.

The Signal Protocol library used by Criptext is open source and available here: https://github.com/whispersystems/libsignal-protocol-java/