A look into the Mobile Messaging Black Box

33rd Chaos Commmunication Congress #33c3

Roland Schilling @NerdingByDoing
Frieder Steinmetz @twillnix

December 8, 2016

Hamburg University of Technology
Security in Distributed Applications

Introduction

Messaging - An Analogy

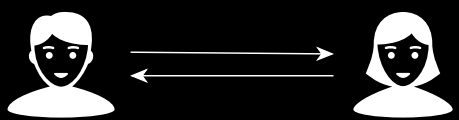
- You're at a party
- Friend approaches you and needs to tell you something in private
- What do you expect when you say private?
- You enter a separate room, you trust the location
- What does a separate room offer you?
  - Confidentiality
  - Authenticity
  - Integrity
  - Forward Secrecy
  - Future Secrecy
  - Plausible Deniability

A Private Room

You are now alone in a closed room with your Friend

- Both of you have absolute Confidentiality that you are alone
- Nobody can overhear your talk
- Your exchange is completely private

We call this confidentiality



In Sight of Each Other

The room you're in is small enough that you can always see each other

- You know that the words you speak are received just as you spoke them
- There is no way either of you hears something other than the other says

We call this integrity

You Know Each Other

Since you're long-time friends, you're absolutely sure, whom you're talking to

- Nobody can impersonate your friend or you, without the other noticing
- You're talking directly, without a phone or webcam in between

We call this authenticity

It's a One-Time Talk

Suppose somebody was present in the room with you

- They could overhear your conversation
- They would only learn the contents of this particular conversation
- They would not learn anything about past conversations you had

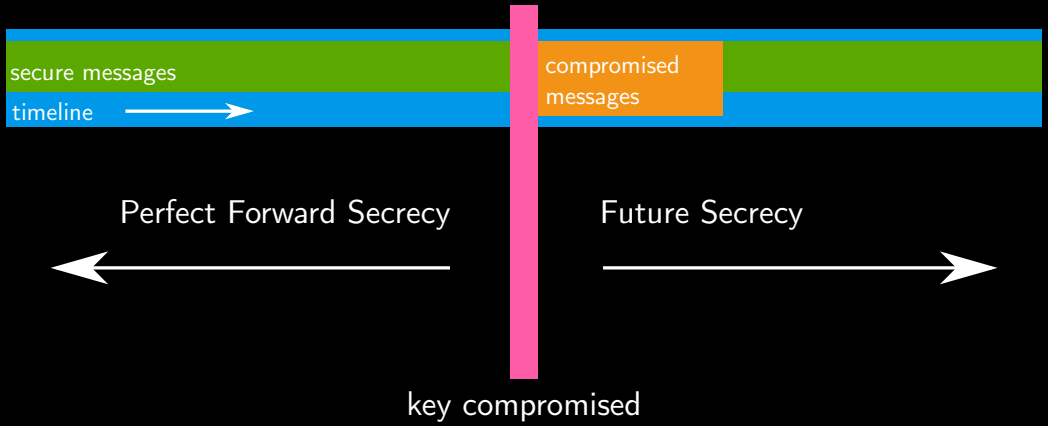
We call this forward secrecy

→ They would also not learn anything about future conversations you might have

We call this future secrecy

It’s a One-Time Talk

Perfect Forward- and Future Secrecy



It’s a One-Time Talk Between Only You Two

There are no witnesses in the room

- Either of you can later deny to other having made any statement
- Neither of you can prove to other that any of you have made a particular statement

We call this deniability

Enter Messaging Services

Messaging

Messaging

Parties involved

- Depending on the scenario, at least one additional party is involved, the messaging provider
- push message provider might also be involved
- In federated scenarios, multiple relaying parties may be involved

⇒ Messaging solutions should be designed so as to minimized the data these parties may obtain

Problems:

- How to find your peers? (address book leakage)
- How to relay messages to the recipient? (participation leakage)
- Relay always knows the exact time a message was sent
- Relay always knows the size of transmitted messages (padding helps)

Traditional Messaging

What if this conversation had taken place via SMS?

Your providers

- would know the contents of your exchange: no confidentiality
- could change the contents of your exchange: no integrity
- could reroute your messages and impersonate either of you: no authentication
- would know all messages you ever exchanged: no forward Secrecy
- would know all messages exchanged in the future: no future secrecy
- could store all messages and use them as proof of the exchange: no deniability

Messaging translates badly to our offline communication expectation

Confidentiality

Deniability

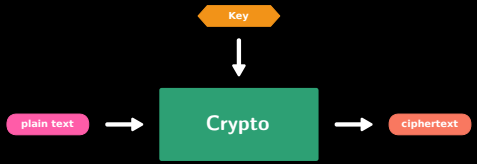
From:
either of us
or nobody

To:
both of us
or nobody

The Shortest Introduction to Encryption You Will Ever Get

Symmetric Encryption:

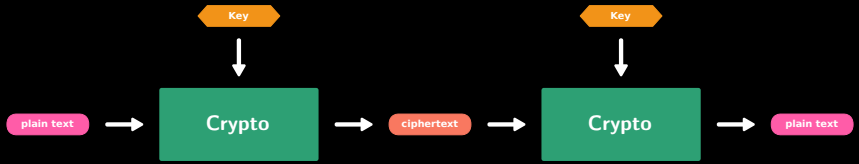
→ Encryption and decryption with the same key



The Shortest Introduction to Encryption You Will Ever Get

Symmetric Encryption:

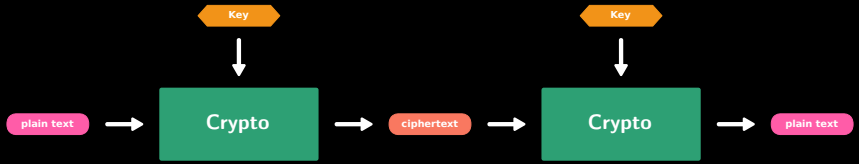
→ Encryption and decryption with the same key



The Shortest Introduction to Encryption You Will Ever Get

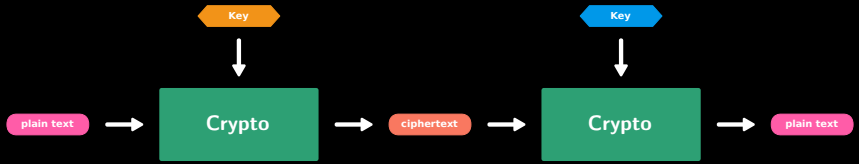
Symmetric Encryption:

→ Encryption and decryption with the same key



Asymmetric Encryption:

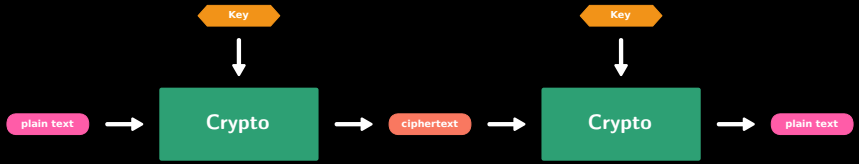
→ Encryption and decryption with different keys



The Shortest Introduction to Encryption You Will Ever Get

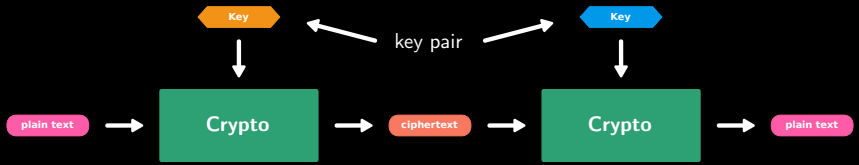
Symmetric Encryption:

→ Encryption and decryption with the same key



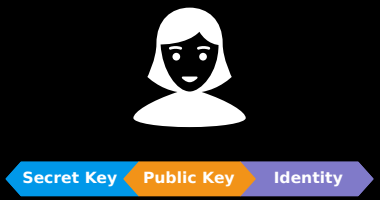
Asymmetric Encryption:

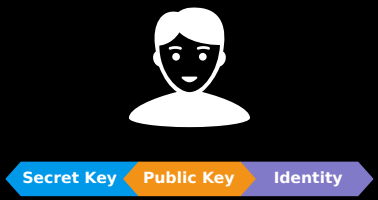
→ Encryption and decryption with different keys



Public-Key Cryptography – In a Nutshell

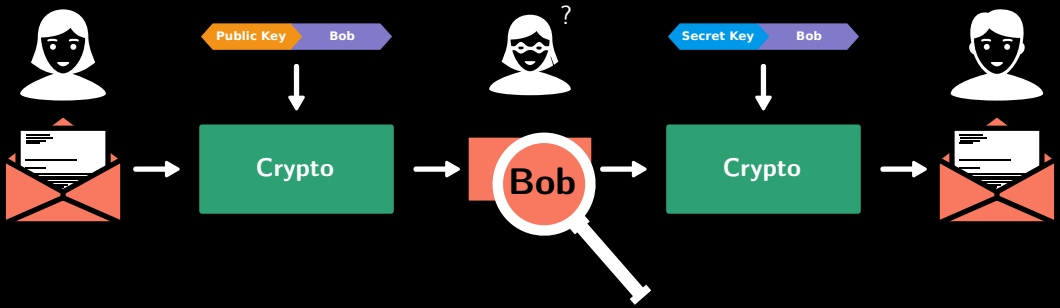
Public-Key Cryptography – In a Nutshell





- Both parties publish their identities and public keys
- Any message can be encrypted with anyone's public key and only be decrypted with its corresponding secret key

Public-Key Cryptography – In a Nutshell



- Both parties publish their identities and public keys
- Any message can be encrypted with anyone's public key and only be decrypted with its corresponding secret key

Key Management

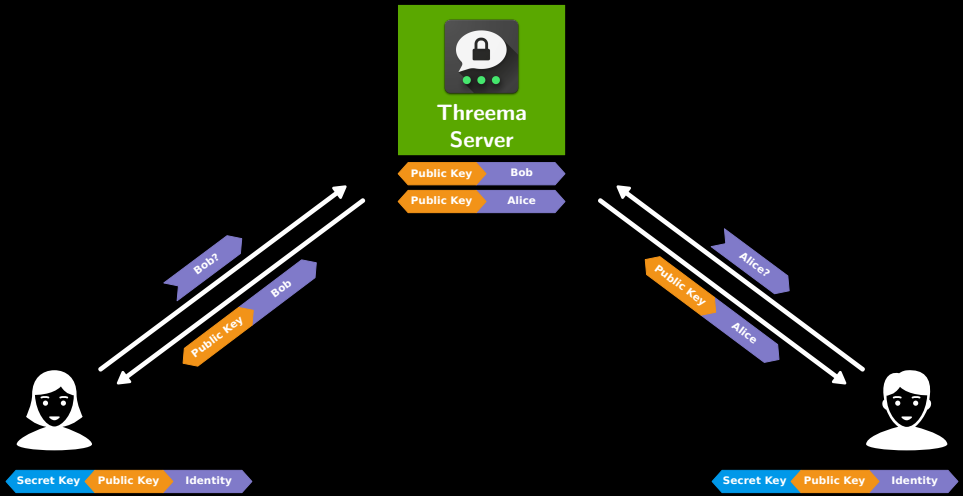
Cryptography is rarely, if ever, the solution to a security problem. Cryptography is a translation mechanism, usually converting a communications security problem into a key management problem.

—Dieter Gollmann

Key Management

- identity keys stored on devices, what if stolen
- keys for back-end communication layer hard-coded, almost impossible to replace
- how to deal with key compromise? (answer: key rotation)

Key Management



Case Study - Signal

Signal - Notes

- Server-side cached short-term keys (prekeys) fetched by sender
- Pairwise long-lived symmetric secret key between participants
- Multiple messages without answer → perform KDF on chaining key
- ECDH: Curve25519, AES-CTR (no padding), AES-CBC (PKCS7 padding)
- HMAC-SHA256 for integrity
- Future secrecy only if private keys are not leaked (duh!). Since private keys go into new shared keys during ratcheting, the attacker lacks material to compromise next key after obtaining current one.
- Since shared keys are only deleted after messages are received, there is a window in which keys could be compromised before reception.
- Deniability is always a theoretical claim as long as a transmission server has the ability to log messages, their senders and recipients.

Reverse-Engineering Threema

Enter Threema

Threema

- Gained popularity in Germany after Facebook purchased WhatsApp
- All promise – no proof; first openly contemplating to OSS the code, later backing away from that statement
- Interest in its inner workings

Quick Shoutouts

- Jan Ahrens for releasing his findings about the handshake before we did
- OpenMittsu for releasing the first working OSS client

Reverse-Engineering – What to look for?

- Test for common pitfalls in implementation
  - Handling of TLS
  - Handling of keys and nonces
  - NaCl implementation errors
  - Uncommon data leaks
  - Bugs
  - …?
- Find out how protocol is designed
  1. Understand handshakes
  2. Understand protocol
  3. decipher messages

Positive side-note: Threema had released a security white paper early on

Threema - Notes and Open Questions

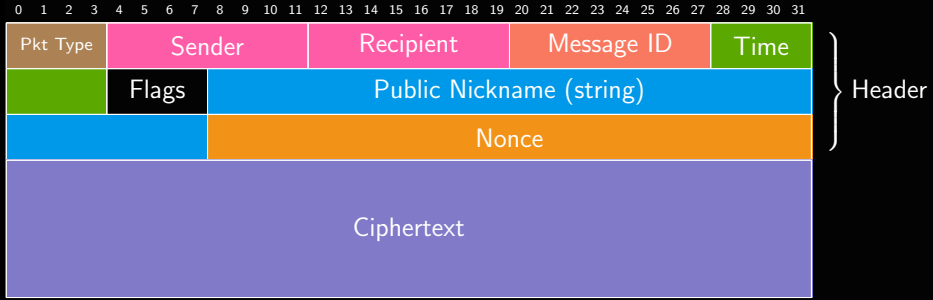
Notes:

- PFS only on transport layer (attacker sniffing packets from the outside will not learn contents after private key acquisition)

Q:

- How often is the handshake performed?

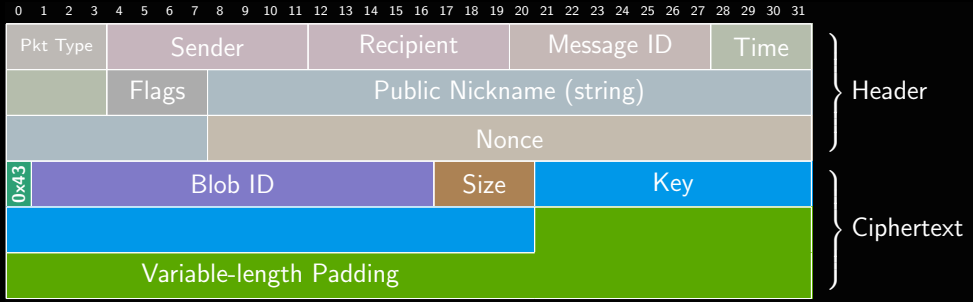
Threema Packet Format



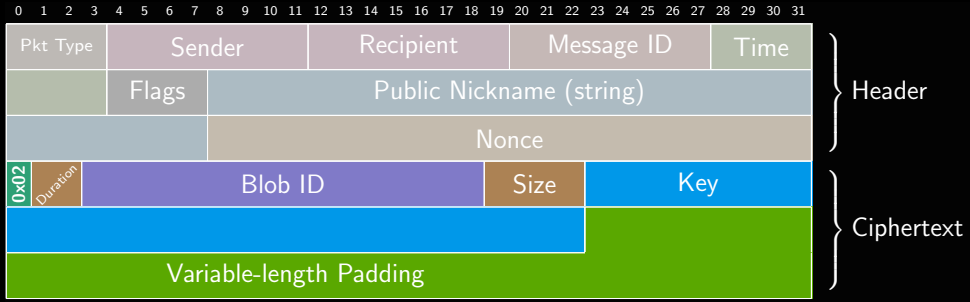
Threema: Special Messages

- Polls
- Images with Caption
  - Case of caption leak found
- Audio Messages
  - Leak Android version
  - Possible StageFright vector

Threema Image Messages



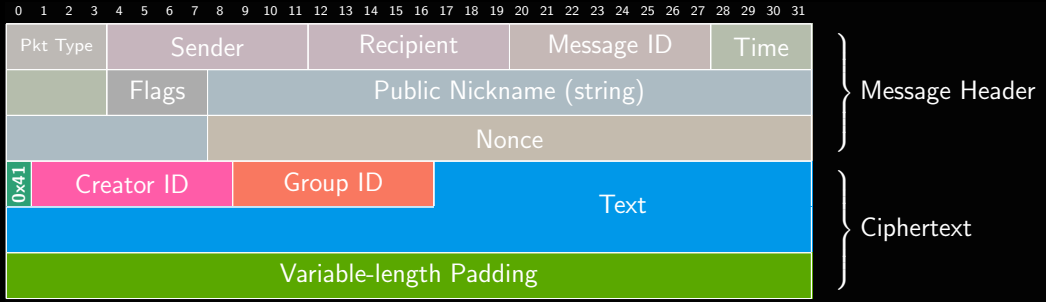
Threema Audio Messages



Threema Audio Messages – Notes

- Audio length is lost on forwarding

Group Messages



Group Messages

- Group IDs aren’t unique. They are created locally and only work together with the group creator’s ID

Thank You!

Roland Schilling
schilling@tuhh.de
@NerdingByDoing

Frieder Steinmetz
frieder.steinmetz@tuhh.de
@twillnix

Beamer Theme: Metropolis by Matthias volgelsang
Color Theme: Owl by Ross Chirchley
Icons: The BIG collection by Sergey Demushkin
Foundation Icon Fonts 3 by ZURB

Thanks to Jan Ahrens and Philipp Berger – their work has made ours somewhat easier
Thanks to Maximilian Köstler for his initial work on Threema