A look into the Mobile Messaging Black Box

33rd Chaos Commmunication Congress #33c3

Roland Schilling   @NerdingByDoing
Frieder Steinmetz   @twillnix

November 21, 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

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 else that the other says

We call this integrity

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

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

$\rightarrow$ They would also not learn anything about future conversations you might have

We call this future secrecy

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

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

Case Study - Signal

- Server-side cached short-term keys (prekeys) fetched by sender
- Pairwise long-lived symmetric secret key between participants
- Multiple messages without answer $\rightarrow$ 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

Threema

- Gained polularity 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
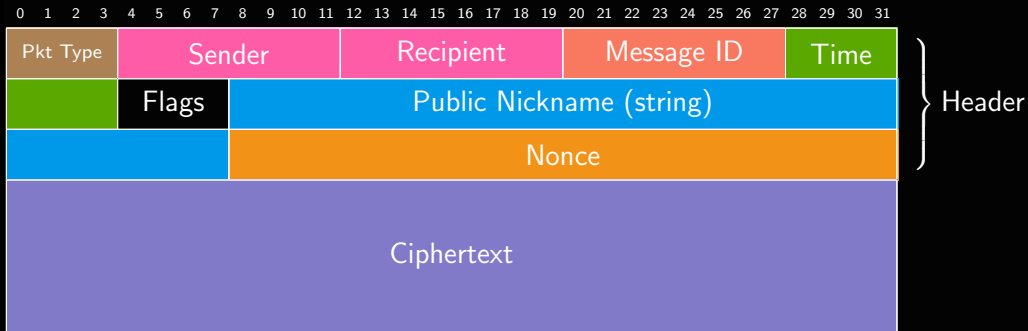- OpenMittsu for releasing the first working OSS client

Reverse-Engineering – What to look for?

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

Positive sidenote: Threema had released a security whie paper early on
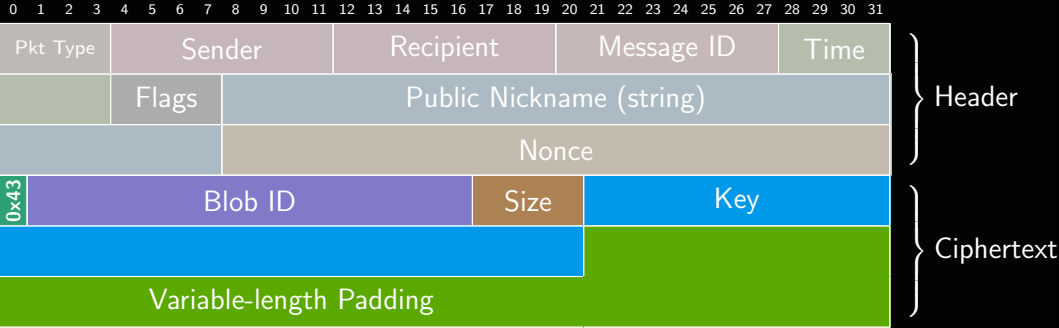
Message Packet (Threema Protocol Layer)

| 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 |

| Pkt Type | Sender | Recipient | Message ID | Time |

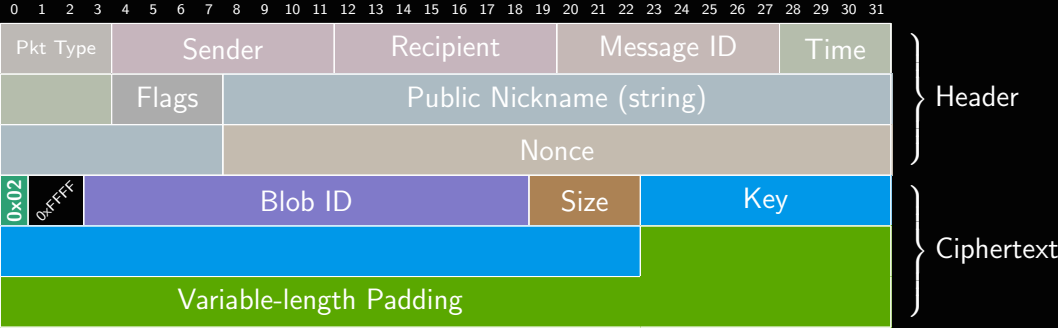| | Flags | Public Nickname (string) |

| | Nonce |

| Ciphertext |

Header

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

Image Message

| 0 1 2 3 | 4 5 6 7 8 9 10 11 | 12 13 14 15 16 17 18 19 | 20 21 22 23 24 25 26 27 | 28 29 30 31 |

Pkt Type	Sender	Recipient	Message ID	Time	⎫
Flags	Public Nickname (string)				⎬ Header
	Nonce				⎭
0x43	Blob ID	Size	Key		⎫
					⎬ Ciphertext
Variable-length Padding					⎭

Audio Message

Group Message Packet