Smartphone as a security token - NFChat

Network and Computer Security

Alameda, Group 26

83448
Dorin Gujuman

83475
Hugo Guerreiro

92255
Nuno Silva

1 Problem

Chat services have existed in some form or another for decades. In general, it is a process by which users on a computer network can quickly communicate with one another using text, images or sound rather than using email. Each user has a piece of software which communicates with a common server that connects the chat sessions.

Many people overshare their own private information on websites such as Facebook, Twitter, Snapchat and Instagram. What happens is that because of social media and the ease of connecting with new people your contact list may become so large that new problems emerge like impersonators, which might commit fraud or defamation. Another problem are chat-bots or spam accounts, that send you unwanted messages.

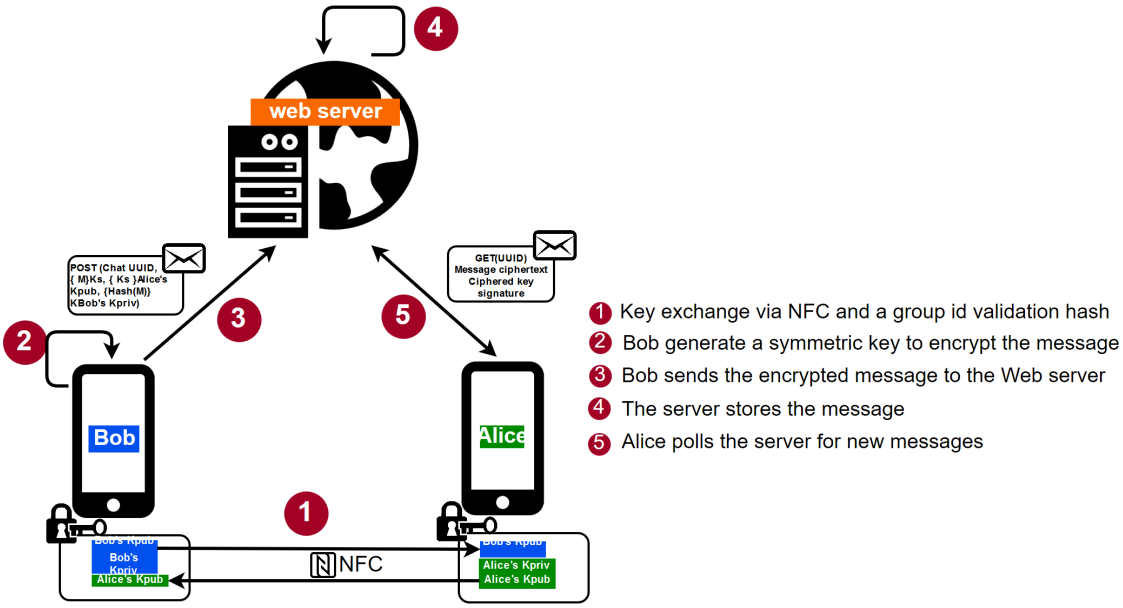
In conclusion, there is a need for a personal chat where you can be confident in the security and identity of the participants.

2 Requirements

The Electronic Frontier Foundation lists some traits that contribute to the security of instant messengers:

- Having communications encrypted in transit between all the links in the communication path.
- Having communications encrypted with keys the provider does not have access to (end-to-end encryption).
- Making it possible for users to independently verify their correspondent's identity e.g. by comparing key fingerprints.
- Having past communications secure if the encryption keys are stolen (forward secrecy).
- Having the software's security designs well-documented.

3 Proposed solution



NFChat aims to solve the described problems by offering a chat service that leverages the proximity to your closest friends and their smartphones.

You start a conversation by touching two phones enabling the users to share completely private and secure messages.

When the user installs the app it generates a personal pair of asymmetric keys (RSA). The expected flow of the app is as described in the diagram. First, the two users initiate a chat conversation by sharing their public keys (by NFC) and agreeing on a chat identification (UUID). When someone sends a new message to the conversation, the sender generates an symmetric key (AES key) with which he encrypts the message plaintext. The sender then uses the public key provided by the other member of the chat to encrypt the new symmetric key and appends the ciphered symmetric key to the message. The sender then sends the ciphered message and the chat id to the server. The receiver will poll the server for new messages and be able to decipher them.

In a first version, NFChat will only allow two users to start a new conversation (i.e to share their public keys) by using the NFC technology. We will then allow the users to send each other messages that flow through a Web Server to, in the future, take advantage of the security provided by the SSL/TSL protocols. Finally, the advanced version improves the chat by allowing the creation of groups. In this version we will also include a verified certificate for the server connections and digital signatures for the messages to guarantee non-repudiation and their integrity.

4 Tool references

- Type of keys: Symmetric (AES) and Asymmetric (RSA)
- Integrated Development Environment : Android Studio and SDK Tools
- Android Beam to share data by NFC
- Java SE Security
- Certificates and Web Servers (HTTPS, Let's Encrypt, Django/Flask)

5 Work plan

Work Plan			
Dates	Dorin	Hugo	Nuno
From Oct. 30th to Nov. 5th	NFC Communication	Android App Base	Generate keys in Android
From Nov. 6th to Nov. 12th	Key Sharing App	Key Sharing App	Key Sharing App
From Nov. 13th. to Nov.19th	Web-server Base	Develop App UI	Chat interface
From Nov. 20th to Nov. 26th	Sending messages through server	Sending messages through server	Sending messages through server
From Nov. 27th to Nov. Dec. 3rd	Group chat	Group chat	Group chat
From Dec. 4th to Dec. 12th	Finishing touches and debugging	Finishing touches and debugging	Finishing touches and debugging