POLITECNICO DI MILANO
SCUOLA DI INGEGNERIA DELL'INFORMAZIONE
CORSO DI LAUREA MAGISTRALE IN INGEGNERIA INFORMATICA

Design and Implementation of Mobile Applications

Watch Dog

Design Document

Responsabile:
Prof. Luciano Baresi

Progetto di:
Claudio Rizzo Matricola n. 800471
Emanuele Uliana Matricola n. 799256

ANNO ACCADEMICO 2012-2013

Contents

1. Project context and purpose

1.1 Context

Watchdog is an application for Android devices developed by Claudio Rizzo and Emanuele Uliana for the "Design and Implementation of Mobile Applications" (DIMA in short) course at "Politecnico di Milano, Corso di studi in Ingegneria Informatica Magistrale". The advisor is professor Luciano Baresi from the Deepse (Dependable, Evolvable, Pervasive Software Engineering) group of the Deib (Dipartimento di Elettronica, Informazione e Bioingegneria).

1.2 Purpose

The application purpose is to provide a way to remote control a mobile phone from another cellphone: this is very useful in case of theft or loss, since it's possible to localize the target phone or to trigger a full volume repeating alarm sound. The application is mainly intended to be used by the same user on two (or more) different mobiles, althought it's not mandatory.

2. Project Planning

2.1 Overview

We had the idea to develop a remote control application after seeing there was not something similar not embedded in a more general application (in fact every anti-malware for mobile devices have a similar functionality, but it comes along with many others features) and because we really like the security/privacy (and cryptography) aspects of computer science. We planned the security/cryptography aspects from the beginning, such that the Confidentiality/Integrity/Availability paradigm is always respected along with the extremely important authentication and non-repudiation aspects.

3. Requirements Analysis

3.1 Actors

As said before, the main actor is usually a single user which uses the application on two different mobile phones, one being the stolen/lost one and the other the rescue one. It's also possible a second actor to exists: a (very) trusted friend may be the one who sends remote commands to the target telephone; since a password authentication scheme is employed, it's not possible to do any abuse in this case.

3.2 Functional Requirements

The application allows the user do to the following tasks:

- Initialization
- Mobile Phone Association
- Password Change
- Unilateral Deassociation
- Remote Localization
- Mobile Phone Mark
- Remote Alarm Triggering
- Local Alarm Off
- Perimeter Selection

3.3 Non Functional Requirements

The remote control of a cellphone is a critical activity and has many security and privacy requirements: the next paragraphs show them briefly: for in-depth explainations see the design section (4.3).

3.3.1 Privacy and Security: problems and Solutions

Sender authentication: While the sender (telephone) authentication plays indeed a key role, it's even more crucial the authentication of the person behind a control message; that's the reason for employing a password based authentication scheme: in the initialization wizard the user is required to insert a password which is going to be needed to send a message to that telephone (the basic assumption is the password is known only by the mobile owner and by some people, possibly no one, he trusts). The password is stored hashed with SHA-256 in the application preferences, along with the hashing salt (a random token) to avoid both time-to-memory attacks (such as rainbow tables) and the equality of two hashes generated from two equal passwords; the salt is sent to another telephone after the process of public keys authentication (See section 4.3.6).
Integrity: The message received must be exactly the one sent: every transmission error or tampering must be detected and cause the abort of the current command session: no retransmission is done.
Authentication: The receiver must have a secure way to understand which telephone the received message comes from.
Non forgeability: Nobody should be able to forge a command message which is both valid and correctly authenticated.
Non Repudation: The sender must not be able to deny he sent a specific message (if he actually did it). Digitally signing every command message can ensure integrity, authentica- tion, non repudiation and a weak defense against non forgeability: symmetric encryption (and in particular AES-256 in GCM mode of operation) is needed for full protection.
Message Confidentiality: No one should be able to detect that and which command is sent to a mobile phone, so the command message is encrypted with the symmetric cipher AES-256 in GCM mode of operation (used for performance reasons and for a supplementary integrity check).
Asymmetric keys management: Asymmetric keys management Digital signatures (and shared secrets computation as we will see) require asymmetric cryptography: in the initialization wizard the application gener- ates and stores in the preferences a key pair based on the elliptic curves; the reasons for this choice are performances and the smaller key length with re- spect to other keys (like RSA and DSA ones) at a fixed level of security. This makes the 140 characters (bytes) Android limit for a single sms no more a problem.
Symmetric key/initialization vector management: AES-256, being a symmetric cipher, encrypts and decrypts a specific message with the same key, and, given the communication channel is not secure, the two parts must agree on the same key in some way; in particular ECDH is used to compute a common secret once and for all, then, when in need to send a message, the sender picks up a random 32 bytes salt, forwards it to the receiver, then both parts use a keyschedule algorithm (PBKDF2 with HMAC-SHA-256) to derive the same key starting from the secret and the salt. Furthermore the GCM mode of operation requires for every message the sender to generate a 12 bytes ran-

dom initialization vector and to send it to the receiver.
Public keys mutual authentication:
While dealing with asymmetric cryptography, the main problem is to bind a public key with a real user to avoid active Man-In-The-Middle (MITM from now on) attacks. Neither a Pulic Key infrastructure (PKI) or a Web Of Trust (WOT) is employed, because they are both potentially insecure for various reasons (in the PKI case the presence of a trusted element, a certification autorithy hierarchy, which may be compromised/untrusted/fake; in the WOT case the presence of a net of trusted elements, the ones who signed a specific public key, which might be fake/bad persons; furthermore a key with no signatures is not automatically a fake one, but there isn't a way to tell), so the application uses a modified version of the Socialist Millionaire Protocol (SMP) to authenticate to each one each other key; this requires the two parts to have a common secret (an answer to a particular question set up on the fly by the users during the SMP), which is easy to achieve, since the two users are likely to be the same person or two people who trust themselves.

3.3.2 Performance

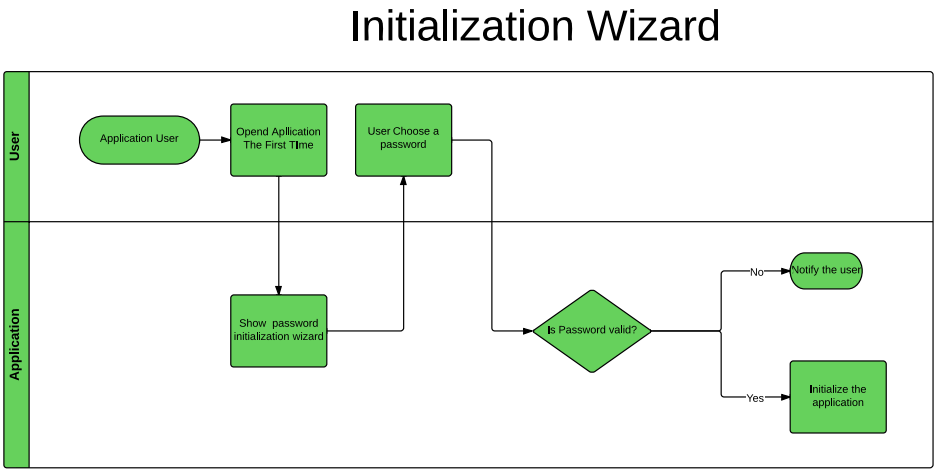
We chose the crypto algorithms with an eye on the performances of the whole system: the key idea is the bottleneck must be the sms and not the computation time required by the encryptions/decryptions; for this reason the command messages are encrypted with a symmetric algorithm and not with RSA or ElGamal (or another asymmetric algorithm), since symmetric cryptography is faster than asymmetric at least by two orders of magnitude (they are very likely to be 3 anyway); however to do ECDH and ECDSA the application needs also an asymmetric key pair, which is generate during the intitial wizard once an for all, so an acceptable overhead. The public keys mutual validation (SMP + ECDH in practice) takes some time, but it's done only one time per association, which means two telephones have to do it only when they associate themselves. Finally the digital signature/verification process are quite fast and so is the key-derivation from the secret and the salt.

4. Use Cases

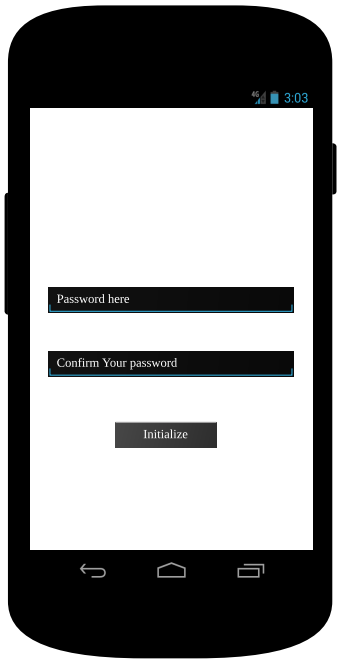
4.1 Initialization

The first time the appliction is opened the user sees a welcome message and his prompted to comlete the initialization wizard which consists in selecting and confirming a password which will be the one needed by other telephones/users to send commands to that mobile. The password scheme forces to select an alphanumeric string (some special characters are also allowed) between 8- 20 characters and with at least a letter and a number. In case of error (bad password/confirmation password different from the base one) the user is notified back and asked to retry. In case of success the application redirects to the main view, which is the "sms localization" one.

4.1.1 Activity Diagram



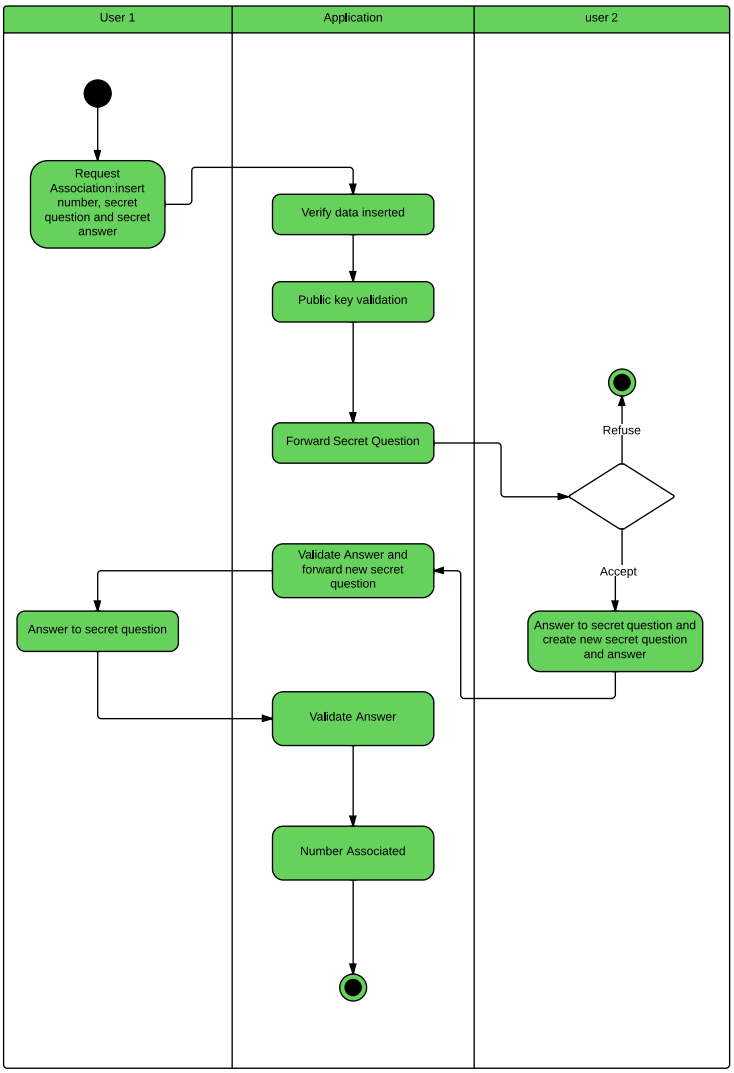
4.1.2 User Interface Design



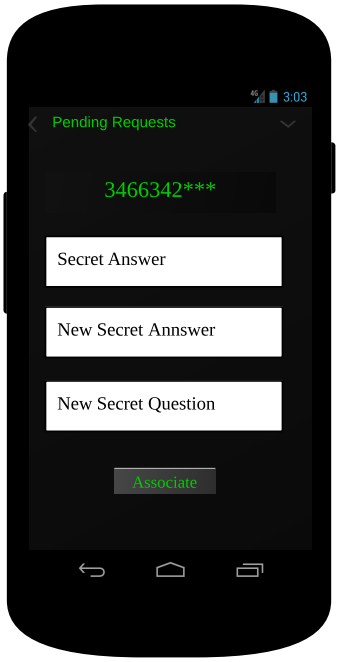
4.2 Mobile Phone Association

For enhanced security it's required two telephones are associated before they can send commands to the other one; the association is bidirectional and can be removed only locally (the other telephone won't know anything, but will obviously fail in sending commands). This totally cuts away the possibility of a Man In The Middle (MITM) both active and passive. A user who wants to associate his telephone with another one first selects a secret question and a secret answer and sends the first to the other mobile, whose user can decide to accept or refuse the request. In the second case the first user is notified back and the association ends with a "failure", otherwise the second user types what he thinks is the secret answer and selects his own secret question/answer. The first user now has to type the answer to that question and, if all goes smooth, the association is done. Every passage is done by sms sending.

4.2.1 Activity Diagram



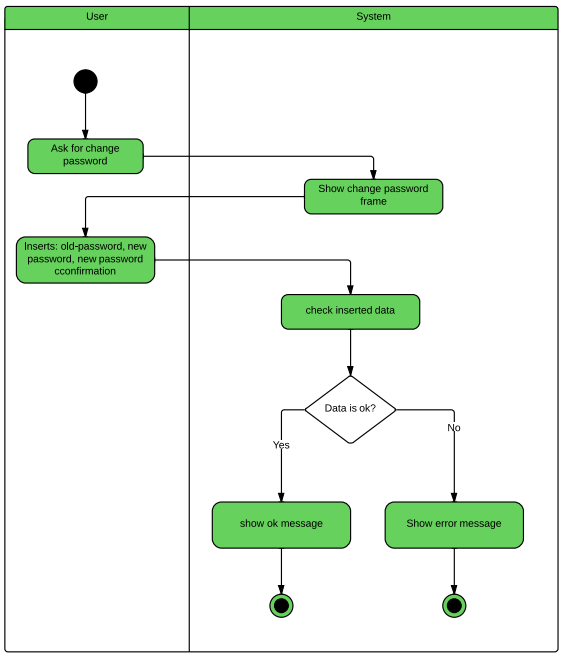
4.2.2 User Interface Design



4.3 Password Change

The user types the old password, the new password and the confirmation new password; the system checks whether the old one is correct and the new ones are coherent and rules compliant (see initialization wizard use case). If not an error message is prompted and nothing is done, otherwise the password is succesfully updated and the user notified back.

4.3.1 Activity Diagram



4.3.2 User Interface Design

4.4 Unilateral Deassociation

4.4.1 Activity Diagram

4.4.2 User Interface Design

4.5 Remote Localization

4.5.1 Activity Diagram

4.5.2 User Interface Design

4.6 Mobile Phone Mark

4.6.1 Activity Diagram

4.6.2 User Interface Design

4.7 Remote Alarm Triggering

4.7.1 Activity Diagram

4.7.2 User Interface Design

4.8 Perimeter Selection

4.8.1 Activity Diagram

4.8.2 User Interface Design