David Stoner

Secure Texting - Software Implementation of a Secure SMS Application for Android Using PGP

Introduction and Motivation

Short Message Service (SMS) is open to several vulnerabilities. SMS messages are sent in cleartext and are vulnerable to attack (both passive and active) at several different points between sender and receiver. SMS messages are transmitted wirelessly and are stored on servers for unknown periods of time until they are retrieved by the recipient. Both sitting on servers and during transmission, SMS messages are vulnerable to attack that compromises authenticity and confidentiality. Passive attacks include intercepting messages either over the air or while located in servers. Active attacks include SIM cloning and theft.

A user of SMS messaging is at the mercy of the technology. However, services can be provided to give users both confidentiality and verify the authenticity of their received messages. There are some applications currently on the market that provide portions of these services but generally lack some of the best practices commonly taught. For instance, several applications (for instance, SMS Encryption Droid) rely solely on a secret key encryption algorithm. While effective, the more often the secret key is used the more ciphertext is available which increases the likelihood of a successful attack. Changing the secret key requires either transmitting the new secret key using alternative methods, using the old key to encrypt the new or meeting in person which is not always possible or desirable. Another, more secure application, TextSecure, uses public/private key pairs to set up communications and then relying on a server (operated by the provider of text secure) to provide a secret key for communication. This level of communication, however, does not provide non-repudiation services to the participants.

This project will create an application for the Android operating system that will provide confidential text messaging via pretty-good-privacy. Additionally, the application will allow users to digitally sign their messages so both sender and receiver are certain who they are communicating with. The application will use the mobile devices contacts list to track certificates and levels of trust. The application stands out from other applications because it provides open source implementations of strong algorithms and eliminates reliance on third party servers.

The implementation will differ from traditional PGP slightly. In typical PGP, a new secret key is generated for every message. Because SMS messaging usually involves several transmissions back and forth, the secret key delivered with the initial message will be reused for all messaging within a 30 minute time frame.

Users will be able to trade public keys in a variety of methods. In person, users can use Bump (a third party application) to trade public keys. Users will also be given options to send public keys over email or copy paste the keys from other sources. Because some sources may be less secure than others, the user will be able to assign levels of trust beyond the traditional three levels available in PGP.

Language, compiler, development and execution platform

This software implementation will be coded in Java using the Eclipse compiler with the Android SDK. The target platform is Android mobile devices. Testing will occur on a Galaxy S4 running Android 4.2.2. There are no plans to other mobile operating systems.

Additional libraries

This implementation will utilize the Android SDK for general interfaces such as sending text messages, user input, user interface generation, message compression and editing/saving contact information (public keys, key chain implementation, interfaces for sharing public keys). Java libraries from the GNU Crypto project will be used to implement the RSA algorithms (for establishing communications and passing the session key), SHA1 or SHA-256 for signing (undetermined based on message size limitations) and AES for session encryption.

Detailed Assumptions

The application will need to meet the maximum memory usage limitations of Android applications which is 24 MB.

SMS messages are limited to 160 characters per message. Goal will be to limit the size of messages to limit the number of SMS messages sent/received.

Detailed specification of the input and output of the program

SMS Message Sending:

The application will require the following inputs from a user who desires to send an SMS message:

- 160 or less alphanumeric characters (message)
- Password (used to decrypt the locally stored private key)
- Destination telephone number

The application will also require the following information (stored on the device):

- Destination's public key

The application will have the following outputs which will be stored locally:

- Secret session key randomly generated by the host device - this will be stored for a short period of time encrypted using a password
- Hash of message

The application will have the following outputs which will be transmitted to the recipient:

- Signature of message (private key encryption of hash)
- Ciphertext (message encrypted with secret session key)
- Encrypted session key (encrypted with destination public key)

SMS Message Receiving:

The application will require the following inputs from a user who receives an SMS message:

- Password (used to decrypt the locally stored private key for decryption)

The application will also require the following information (stored locally):

- Sender's public key (stored locally or received)

The application will have the following outputs:

- Secret session key (decrypted) - to be stored locally and secured using the user's password
- Plain text message
- Indication of verification (verification based on computed hash of message and decrypted signature of sender)

Keychain:

The application will require the following inputs from a user who adds a trusted user:

- Name
- Telephone number
- Public key
- Level of trust

The application will have the following outputs when a trusted user is added:

- Signed public key (using private key)

The application will require the following inputs when adding :new users

- Certificate signed by known user

The application will have the following output:

- Level of trust based on certificates signing the public key
- Level of validity

References to the detailed descriptions of the implemented functions

Details of the implemented functions within GNU Crypto can be found within gnu-crypto-2.0.1.zip available at http://www.gnu.org/software/gnu-crypto/#downloading. Details for AES and RSA functions for encrypting and decrypting can be found in gnu-crypto-2.0.1 -> docs -> gnu-crypto.info. Details for SHA-1 hashing can be found in gnu-crypto-2.0.1 -> docs -> gnu-crypto.info-2.

List and initial analysis of similar programs reported earlier in the literature or on the web

- TextSecure – SMS/MMS application that uses Off-the-Record Messaging with a combination of secret key encryption and private/public key encryption.
- SSE – Universal Encryption App – provides secret key SMS messaging that uses a variety of open source encryption methods.
- SMS Encrypt – provides secret-key SMS messaging using an unknown algorithm.
- PGP SMS – Performs public/private key messaging.

Procedures for testing the functionality and performance of the program

The majority of testing will occur on emulated devices using the Android SDK. This testing will include:

1. Message encryption – The following procedure verifies messages and keys are encrypted correctly on the device.
    a. Perform encryption of session key using public key.
    b. Print out encrypted session key for verification.
    c. Decrypt session key using destination private key (known) using CrypTool 2
    d. Perform encryption of message using session key.
    e. Print out encrypted message for verification.
    f. Decrypt message using session key in CrypTool 2.
    g. Hash message.
    h. Verify hash using CrypTool 2.
    i. Sign hash.
    j. Verify hash using CrypTool 2.
2. Keychain testing – The following procedure verifies the web of trust of public keys
    a. Start multiple emulated devices.
    b. Begin by adding devices to the keychain as trusted entities (second tier)
    c. Add additional devices as trusted entities to the second tier (makes up the third tier)
    d. Add devices that can no longer be trusted by some entities but not all
    e. Verify starting device gives the appropriate level of trust to each device it knows (either directly or indirectly).

Additional real hardware testing of message sending and receiving will occur with two Android based devices.

Plan of experiments to be performed using the program

Experiments will coincide with functionality testing outlined in the previous section. This functionality testing will include sending of text messages from real mobile devices as well as using emulators to simulate a network of devices with additional users receiving varying levels of trust.

Time schedule

- Progress Report 1: Oct. 15-17 (25%)
    - Completion of basic infrastructure (application created to send a text message)
    - AES encryption interfaces implemented (application able to send encrypted message using AES only).
- Progress Report 2: Nov. 5-7 (50%)
    - RSA interfaces implemented (application signs message, encrypts session key with public key, etc).
    - Storage of public keys in contacts.
- Progress Report 3: Nov. 19-21 (75%)
    - Complete trust algorithms for PGP key-chain.
    - Complete sharing methods of public keys (email, Bump, etc).

A list of possible areas, where the specification can change depending on the progress of the project

- Allow user to select session encryption (expand from AES-128 to AES-192, AES-256, Twofish, DES, Triple DES).
- Allow user to select hashing functions (expand from SHA-1 to SHA-256).
- MMS encryption

Tentative table of contents of your final report

1. Overview of SMS
2. Overview of SMS security risks
3. Overview of PGP and Web-of-Trust
4. Current SMS encryption applications
5. Details of how PGP is implemented.
6. Details of how web of trust is implemented.

List of literature

A. Medani, A. Gani, O. Zakaria et al, "Review of mobile short message service security issues and techniques towards the solution" in *Scientific Research and Essays*, vol. 6, no. 6, pp. 1147-1156, Mar. 2011.

B. Schneier, *Applied Cryptography*, 2nd ed. New York: Wiley, 1996

C. Paar and J. Pelzl, *Understanding Cryptography*, New York: Springer, 2010

A. Consalves. (1012, Sep. 07). *Moble malware shifting to SMS fraud* [Online]. Available: http://www.csoonline.com/article/715700/mobile-malware-shifting-to-sms-fraud

Network Associaites, Inc. (1999). *An Introduction to Cryptography (ver. 6.5.1)* [Online]. Available ftp://ftp.pgpi.org/pub/pgp/6.5/docs/english/IntroToCrypto.pdf

Open PGP Message Format, RFC-4880, 2007