# Securely Share – A Confidential Document Sharing System for Android Devices

## Introduction

As the popularity of mobile communications devices increases, there is a growing tendency to use these as a convenient means of reviewing and revising documents on-the-move. Where these documents are of a confidential nature, particular attention must be paid to the fact that mobile devices are more vulnerable to compromise that traditional desktops, which are usually more extensively protected by the security measures implemented as part of an organization’s internal network.

There are multiple mechanisms for keeping files secure on company servers whilst allowing employees the necessary permissions to work collaboratively with sensitive data as required. As the mobile device culture becomes more prevalent in the workplace, the addition of Mobile Device Management (MDM) applications empowers users to also access corporate data via their mobile devices whilst still allowing IT departments to retain a degree of control over data security.

Thus, it is acknowledged that maintaining the security of confidential documents can be challenging, even with the weight of a corporate IT infrastructure behind it. In this project, we seek to address the issue of allowing groups of users from *different* organizations (i.e. with no shared IT infrastructure) to collaborate securely on confidential documents and furthermore, to access these documents via mobile devices whilst minimizing the risk of exposing sensitive information to a potential attacker.

## Project Aims & Objectives

The primary aim of this project was to implement a scheme for the secure sharing of confidential documents between small (typically < 15) groups of collaborators, subject to the following constraints:

* Groups are self-organizing and represent multiple organizations, hence they cannot draw on the support of any central IT services.
* The documents involved are confidential in nature and hence should be encrypted both in transit and at rest.
* Group members wish to be able to access documents on a mobile device which is running the Android operating system.
* The solution devised should use only well-tested cryptographic techniques and standard libraries and should minimize the amount of trust to be placed in a third-party.

In pursuit of these aims we developed a solution called Securely Share, consisting of a detailed design of the security components of the system and a prototype android application (SecurelyShare) to provide a platform on which to implement and evaluate the various security features. It was acknowledged that, in a live setting, documents would usually originate on a PC rather than on a tablet device and thus the system would also need to a PC-based component. However, within the time constraints of the project it was considered infeasible to develop a fully featured system; our solution is submitted rather as a ‘proof of concept’.

Develop full system design including any set up procedures or desktop applications

* To develop a prototype application for an android tablet

One solution to this is the storing of documents in encrypted format. However, where there is a requirement for collaborative working across more than one organization, this introduces the additional problem of group key control and distribution in order to ensure that only the group members are able to decrypt the documents in question.

### Initial Assumptions/project scope

Scope – digital signatures to protect against Man-in-the-Middle attack

• Shared files are held on a secure web server. Since the web server is not part of the trusted group, it should hold no unencrypted files and no ability to perform decryption.

• Any database used for performing user authentication, etc. is deemed to be secure.

• Any attacker inserting malware onto the mobile device may be able to access files stored on that device but would not be able to access that data whilst held in the device memory.

• The requirement to maintain security of documents at all times is such that the processing overhead associated with key management and encryption/decryption is deemed acceptable.

• For the purposes of this application, it is not necessary to consider performance and battery usage issues associated with the use of the mobile device, as it is deemed that the application will be run relatively infrequently.

In order to protect from active attackers, the initial intention is to ensure that documents are not saved to disk in unencrypted format - an unencrypted document will be held in memory and a simple rich text editor will be implemented within the application to allow reading and editing. It is acknowledged that a fully developed app would need to provide integration with existing 3rd-party document-handling applications. This would require further work and also involve use of secure hardware to ensure that security is maintained at all stages. Currently it is anticipated that such features will be beyond the scope of this project.

It is anticipated that, in a fully-developed system, initial creation of documents would take place in a desktop environment; therefore, implementation of a suitable desktop client is outside the scope of this application.

Original plan to use web server, then basic client/server - final design involved more sophisticated key sharing so that cloud storage can be used

Original aim for data not to be stored anywhere in plaintext - had to compromise on that due to time limitation but still included in security design

### Overview of Report

• Introduction to the scenario and the challenges it presents

• Overview of the project and its objectives

• Presentation of the solution – its theory and practical implementation

statement of the threat model we are trying to secure against

In this introduction we have looked at…

### The Solution

• Dropbox used for all exchanging of files

• Public key cryptography (RSA) for exchanging group encryption keys

• No transmission or storage of plaintext AES Encryption with CBC used for data encryption (currently with 128 bit key)

• All key information held in encrypted KeyStores

No GUI for admin - out of scope

Means of distributing certificates is outside of scope - as tese are public, any means will do.

Talk about decision not to implemenmt passing decrypted data directly to another app without needing to write to external storage

## Background Material

In this chapter we will introduce some key aspects of the android architecture and its built in security features, review some related work in the field of group key sharing that was used to inform our design decisions and look briefly at some of the commercial applications which offer features similar to Securely Share

Presentation of preliminary research findings

### Preliminary research

Mention papers I read

### Things you need to know about security

### Analysis of other available solutions

#### related work

#### commercial products

#### research papers

### Android Architecture and Security Features

### Review of Related Work

### Review of commercially available solutions

See Android Data Encryption Saud Alharbi Sept 2010 which covers issues to do with key generation

Talk about the android apps I reviewed - boxcryptor, Cryptonite, EDS lite

#### Boxcryptor

#### Safemonk

## Project Specification

This is what I ended up trying to design after completing initial research

Initial plan was…

After initial research it became apparent that there was far more to learn about android development than simply java “in the small” and hence a larger part of the development time needed to be allocated to learning the basics of the operating syste,me. Decided to aspire to best practice rather than simply what I could get away with to make it work (wherever possible) although on occasion compromises had to be made in order to achieve a working prototype.

Decisions

I would develop prototype for a single API and test on one device – no backwards compatibility hence no need to use compatibility libraries eg for Action Bar

Decided on threat model

## System Specification

## Analysis and Specification

How you analysed the problem, including user requirements. Give an appropriate specification of the solution (note that you are not expected to produce a requirements specification to industrial standards).

#### Security Requirements

Talk about TPM

Reasons why didn't choose ID based cryptography or password based solution - aim is to use simplest solution that works

Separation between design and what is implemented in prototype - all design elements are needed for security.

Lots of issues at the start with ensuring that it remained a project about crypto - kept coming up with design solutions that simply turned it into an exercise in access control

Planned to uses SSL to encrypt in transit for protection agains replay attacks- use of Dropbox Sync API made this unnecessary as everything was handled by that

Did not implement signing in prototype as largely meaningless with self-signed certificates. Purchase of appropriate certificates for authentication of signatures would be required for a complete solution

#### Android Prototype Requirements

## Solution Design

Design. A high-level account of the structure of your software and how it works. What algorithms does it use? How do these compare with alternatives? What were the main design decisions you took, and their justifications?

Althou prototype allows for encryption of files stored on the device, in practice the very fact that there are files on the device that the user wants to encrypt violates our central tenet that plaintext should never be stored to disc.

Android has inherent protection by sandboxing apps so files in internal storage have extra protection. FOr the purposes of development and testing, files written to app protected external storage so that they can be inspected with a file manager without the need to root the device

decision not to implement threads at this stage

When app is deleted, files should be deleted so keystore would be removed from device

Use of Java program running on PC to develop encryption to begin with

No ability to recover keys if app gets deleted - could re download from Dropbox as long as private key and certificate backed up.

3 parts

* Admin system – basic with no front end. Designed to run on system with Dropbox installed and running so files uploaded to dropbox simply by saving them in the correct location rather than worrying about using Dropbox API within program.
* Android prototype
* PC version in java that just does basic encryption/decryption. Used primarily for intial development and testing of encryption mechanisms. Single username hardcoded for testing. Fully developed application with user interface outside of scope of this project.

## Assumptions & design decisions

nelenkov.blogspot.co.uk - credential storage enhancements in Android 4.3

Assumption that encrypted blobs are probably also created on PC - simple PC version of program developed to address this, although no gui developed

Assumption - no static IP so peer to perr sharing is not possible

Early design considerations

• who generates keys

• how are keys generated

• how are keys distributed

• issue with public key - how would we stop Mallory uploading bogus documents

• issue with where to place trust

• how to manage letting decryption know group - flirted with shared preferences

Issue: if generate [rivate key on device, it is device specific - ability to import would allow same keystores to be used on multiple devices for4 same user

• decision to ignore considerations like battery life and

User authentication and need to block after failed attempts

• large files

• network connectivity

• battery life

• small memory

• multithreading for gui

• where to encrypt

• model to use for file distribution and storage

Means of distributing certificates is outside of scope - as these are public, any means will do.

## Implementation & Testing

Implementation and testing. A detailed account of the implementation and testing of your software. Explain what data structures you used, and how the algorithms were implemented. What implementation decisions did you take, and why? There is no need to list every little function and procedure and explain its working in elaborate detail; use your judgement on what is appropriate to include.

### Implementation

use of .xps, .xeb

use of bundle for passing data between activities

use of interface for passing data back from dialog

Performance problem - introduced buffering

Splash screen and initialization

didn't use onStoreState etc. - didn't worry about restoring exact user position as prototype and system stores GUI stuff

Use of singleton

Keystores moved to external storage for testing and demonstration purposes. In a production app, these should be moved back to internal storage in order to take advantage of the additional protection afforded by android’s inbuilt security mechanisms

Removal of keystores upon 3 successive failed password attempts – al present it just shows a message saying that the keystores have been deleted but doesn’t actually remove them from the device. In a live system this would need to be implemented.

### Testing

Testing - it is ok to say that I tested by inspection

Explain why unit testing is not meaningful

## Implementation & Challenges

• secure data exchange is a non-trivial problem – particularly against active attacker

• mobile devices have inherent security risks which adds additional complication

• use of xml rather than java for managing onClick - why was this done and when is it not applicable

Major issues with keystore, certificates and default providers.

Challenge of absence of built in file manager

### Dropbox issues

• dbx stuff does not implement serialisable or parcelable

• Dropbox synchronization issues - developed everything using App specific access then discovered that this doesn't allow any use of shared folders so had to redesign

Issue: if generate [private key on device, it is device specific - ability to import would allow same keystores to be used on multiple devices for4 same user

Challenge of unavailability of BKS on pcs in school

No access to key tool in android

Include information about algorithms and key lengths

Fragments

For improvement, use custom file extension registered with Dropbox then would only ever see encrypted files

### Admin process

Make sure I write about admin process

Key generation and sharing currently done from PC rather than android device - what do we think of this as an idea?

Folder management and sharing done outside of app - could easily be added

## Project Management

Challenges of Agile methodology when using a new language or API - lot of time spent learning how to implement stuff that ended up not being needed.

Use of Java program running on PC to develop encryption to begin with

e.g. shared preferences - implemented during one of the iterations but ultimately abandoned for a simpler model

keystores may have been handled differently once had to use Bouncy Castle anyway

## Evaluation

### Achievements

• What works well

### Explain how well solution meets objectives -

### What you have learned - why android development was a challenge

major challenge of the fact that android is an operating system not a programming language - event driven programming

### Further work

• From prototype to production - next steps

Write as though you are porviding a basisi for a good cs graduate to continue the work - assume they have already done some android development

### Security Evaluation

Attacks and issues to consider

• anonymity

• forward secrecy

• revokation

• man-in-middle

Delete keys after failed password attempts

Write about how p[rotocols as as important as implementation - need to support this view from academic papers

Talk about why it doesn't matter that encrypted copies of group key are available on dropbox

nelenkov.blogspot.co.uk - credential storage enhancements in Android 4.3

out of bounds channel - side channel attack

Write about issues to do with public key distribution and the need for signing

Talk about decision not to implemenmt passing decrypted data directly to another app without needing to write to external storage

Don't zero out passwords after use

No implementation of digital signatures so vulnerable to man-in-middle

Decision to use same password for keystore and aliases - trade off of added security against temptation for users to use insecure passwords or write them down

Maybe argue why solution is secure here

### Prototype Evaluation

dependent on exactly correct alias for groupid and folder name

• Is designed as a “proof of concept”

• Aspires to use “best practice” within the code

• Uses well-tested cryptographic techniques and standard libraries

• Adheres to the stated security requirements

No ability to change passwords etc added at present

### Evaluation of Personal learning

• zero knowledge starting point

• Android is a whole new operating system not just ‘Java with extra bits’

• Unfamiliar API’s operating in a sub-optimal environment

## Conclusion

## Summary/conclusions

Conclusions. Here you will summarise your achievements and also the deficiencies of your project. You can also say what you would or could have done, if you had had more time or if things had worked out differently. It is important to be completely honest about the deficiencies and inadequacies of your work, such as they are. Part of your aim is to demonstrate your ability to recognise problems that remain

## Stuff to think about

TODO checkout understanding of callback

TODO decide why I didn't use the encrypted folder approach like many of the other apps

secondary aim – to gain an understanding of the basic features of theandroid operating system and some of the techniques involved in developing applications for mobile devices and some of the challenges encountered in an event driven environment

Group needs admin, although any group member can serve in this role. It is also possible to delegate this to an administrator who is not part of the group without giving them access to the group encryption key. However, if the admin was corrupt, the fact that they had access to the private key for signing the encrypted group key would still be a problem.. Useful phrase “a more sophisticated attacker”

Threats:

* Malware on device
* Attacker snooping around external storage but not one with root access
* Lost device with app open (minimal protection) but can unlink from dropbox remotely so would only have a very small window of opportunity to decrypt files currently stored on devicewhilst keystore is unlocked
* Could have had different password for each group
* Could make user re-enter password for each file – trade off between added security in event of lost device and templtation for user to choose a weaker password

The following chapter outlines the salient points from the initial analysis which in turn led to the final design.

Some desin decisions were taken early on but, due to the initial lack of familiarity with the android operating system, much of the design of the prototype evolved as issues came tolight which needed to be addresses.

Client side encryption was used because of the desire to ensure that server had no access to plaintext

Minimise requirement to involve IT department

Client server model was considered but has an IT overhead and , would need to ensure communications are secure. Could use server just to create and distribute keys but requires trusting server and needs user to authenticate to server

Working on the basis that we wanted the simplest viable solution, looked at using existing cloud services that offer an android API

Intended to provide the user with a choice of which online file server to use but having begun the implementation using Dropbox, it quickly became apparent that attempting to design for multiple APIs was not realistic within the available timeframe, hence it was decided to develop the prototype with Dropbox initially as this is widely used and is well supported in both Windows and Linux environments.

Encryption of filenames

Harnesses all the existing security features of Dropbox and android without compromising on fundamental principles regarding trusted third parties