Lab 5: Android Covert Channels

Due: March 21st at 11:59pm

Overview

In this lab, you will work in groups of two to develop a inter-application covert channel. A covert channel is a communication mechanism where the fact of communication (as well as the content) is covert to a casual observer. For example, embedding information within an image is a form of a covert channel. You will attempt to implement two covert channels: one must come from previously published research, but the other can be of your own design.

Deliverables

Your submissions will include at least two distinct Android projects, a sending Application and a receiving application for your covert channels. You should name the sending Application, `CovertSender`, and the receiver, `CovertReceiver`.

Please submit all requisite files for both applications, and each application must have an associate `README`, this will make grading easier. application will run. You must still provide a `README` file. Your `README` file must include, (1) your name, (2) the name of all *relevant* files in your submission, and (3) a short description of the program and your development process. Failure to include a `README` or not including sufficient information in the `README` will adversely affect your grade on this lab.

Project Retrieval and Submission Instructions

To submit this lab, once you've created the Android project in Eclipse, you will share the project over SVN to the following URI

```
https://cs71.cs.swarthmore.edu/svn/cs71/submission/<username>/05
```

Where `<username>` is replaced with your swarthmore username. Click finish, and then follow the on-screen instructions to commit. You may update and commit your project until the deadline. All commits after the deadline will be rolled back to the submission just prior to the deadline. **Only one submission per group (of two!).**

Covert Channels

A covert channel in computer security is a communicate channel that is difficult to detect and prevent. It enables an attacker to covertly communicate information between processes via mechanism that are "out-of-band" of normal communication. For the purposes of this lab, we will consider a covert channel as an *non-overt* communication channel, which implies that it may be detectable. We describe a non-overt channel as a communication method that outside the scope of the pre-described Android inter-process communication methods. For example, using Intents or writing a message directly to shared media (even encrypted) is an overt communication channel; however, strong obfuscations (such as stegonography) or using non-traditional communication means (such as the accelerometer and vibrator) is perfectly acceptable covert channel for this assignment.

Your goal in this lab is to develop two covert channels that will successfully transfer a secret message between two applications running on the same device. The first covert channel will must be from published research, and the second may be of your own design. Be creative!

> **WARNING!** This is a non-trivial assignment that will require an attention to detail, touching on many components of the Android system. Get started early!

Application Setup

You will develop two applications, `CovertSender` and `CovertReciever`. Both applications will have an associated (private) Service that will run even when the primary Activity is in the background. Your applications should allow the user to select which of the covert channels to use, after which, the graphical part of the application may close while the Service associated with the application persists.

`CovertSender` should attempt to transmit a message, described as `secret.txt`, which should be stored in Internal Storage. Here is the message to transmit:

> Slowly, desperately slowly it seemed to us as we watched, the remains of passage debris that encumbered the lower part of the doorway were removed, until at last we had the whole door clear before us. [1]

Internal Storage is private to each Application, unlike External Storage, which is written to the SD Card, so the `CovertReceiver` cannot access the file directly. Once the covert sending routine is done, the `CovertReceiver` will have a copy of `secret.txt` written to External Storage, which implies that `secret.txt` can be ex-filtrated off the device. It is not required, but recommended, that you provide mechanisms for the `CovertReciever` to view the current state of file transfer, e.g., via a TextView. Further, the `CovertReceiver` should parameterized the filename of `secret.txt` with a timestamp, that is, save to a file `secret-<timestamp>.txt` where `<timestamp` is replace with the timestamp at the beginning of the message transfer. This is to ensure that you are not using previous transferred information and performing covert communication in earnest.

If you wish to break your covertly communication applications into multiple applications, a pair for each covert channel, you may do so. However, please provide details in your `README`.

[1] What is the significance of this text to cryptography and history?

Managing Services

Your applications services must be declared private so that other applications cannot access them directly. You may allow your service to be bound; however, only one application may bind to it. Additionally, you may start a thread in your Service, if you want, but that is not necessary.

You'll find that once your service is created, you will want to stop and start it arbitrarily. You can use the Android built-in mechanism for that by navigating to Settings→Applications→Manage Running Services, or you can build a switch into your covert applications. I would suggest the latter, a built-in switch, which will help your debug process. Note that there are multiple goals of this lab, and learning to use Services is one of them. Please do mind your design and implementation of the service.

Soundcomber Covert Channels

You `must` implement at least one of the covert channels listed in Soundcomber paper[2] (Section 4.2.2):

- *Vibration Settings Channel*: The `CovertSender` modulate the vibration settings on and off, which results in a notification to the `CovertReceiver`.
- *Volume Settings Channel*: The `CovertSender` modulates the volume settings up and down, which results in a notification to the `CovertReceiver`.
- *Wake Lock Channel*: The `CovertSender` turns on and off the screen, which can be detected by the `CovertReceiver`.
- *File Lock Channel*: The `CovertSender` locks access to a file which blocks the `CovertReciever` from doing the same, thus communicating information.

For your second covert channel, you may implement one of the above channels, or you may implement a new covert channel of your own design.

Your covert channels may not be high bit rate, that is, it may take some time to communicate the message, which is fine. There is no requirement to communicate the message quickly, just to communicate the message successfully.

Protocol and Encoding Information

You will need to develop a strategy for encoding information as well as signals for indicating the start and end of message transfer. You can encode information using a binary system, i.e., send 0's and 1's, or using more symbols, e.g., Morse code. You can choose to transmit the message with punctuation, spaces, and capitalization, or you could chose to send just the letters of the message, assuming the receiver can reconstruct the core of the message latter. Regardless, **you must describe your encoding strategy in your** `README`.

Grading

Grading in this lab will be based on in-lab demonstrations on Friday March 22. Here is the grading breakdown for working submissions:

[2] `http://www.cs.indiana.edu/~kapadia/papers/soundcomber-ndss11.pdf`

- 8 points: First covert channel
- 1.5 points: Second covet channel
- .5 points: Design and execution

For non working submission, the following grades will be assigned:

- 7/10: Covert channel successfully communicates information but partial message sent
- 6/10: Covert channel is unsuccessful in communicating any information but significant progress made
- 5/10 (and below): Nonfunctional implementations.

Extra Credit

For **1 point** of extra credit: Perform analysis of the bitrate of the covert channels you implemented. Refer to the SoundComber paper for details of this analysis technique, and further, compare your results to the results in SoundComber. Was your performance better or worse?

Competition

There will be a competition in this lab for the most interesting and well designed Covert Chanel. The winner of the competition will receive **1 point** additional extra credit on this submission.