BACHELOROPPGAVE:

THEIA, the User Interaction Archiver

FORFATTERE:

Brage Celius
Jannis Schaefer
Eirik Vestreng Solberg

DATO:

15.05.2015

Sammendrag av Bacheloroppgaven

Tittel:	**THEIA, the User Interaction Archiver**
Dato:	15.05.2015
Deltakere:	Brage Celius Jannis Schaefer Eirik Vestreng Solberg
Veiledere:	Mariusz Nowostawski
Oppdragsgiver:	Biometrics Lab NISLab
Kontaktperson:	Soumik Mondal, soumik.mondal@hig.no
Nøkkelord:	Android, Sikkerhet, biometri, Informasjonssikkerhet
Antall sider:	94
Antall vedlegg:	
Tilgjengelighet:	Åpen

Sammendrag: Denne bacheloroppgaven drøfter mulighetene for å logge brukerinteraksjon med Android enheter beregnet på privatmarkedet. Vi viser at logging av berøringsdata ikke er mulig uten modifikasjoner av Android operativsystemet, og fremstiller en implementasjon som leser fra `/dev/input/eventX` filene, tyder dataene som finnes der og logger de til en database.

Summary of Graduate Project

Title:	**THEIA, the User Interaction Archiver**
Date:	15.05.2015
Participants:	Brage Celius Jannis Schaefer Eirik Vestreng Solberg
Supervisor:	Mariusz Nowostawski
Employer:	Biometrics Lab NISLab
Contact Person:	Soumik Mondal, soumik.mondal@hig.no
Keywords:	Android, Security, Biometrics, Information Security
Pages:	94
Attachments:	
Availability:	Open

Abstract:

In this thesis, we explore the logging of user interaction on Android devices targeted at the consumer market. We show that logging touch interaction is not possible without modifications of the Android operating system, and propose a sample implementation which reads the `/dev/input/eventX` files, decodes the data given there and logs it to a database.

Abstract

In this thesis, we explore the logging of user interaction on Android devices targeted as the typical consumer uses on a daily basis, e.g. excluding development devices. We explain how touch events are propagated in the Android operating system and determine in which stages of the chain events could be intercepted. We investigate various possibilities such as using an overlay that runs in the foreground to log data, having Android's AccessibilityService API send us AccessibilityEvents and undocumented standard API calls. As a result, we show that logging touch interaction is not possible on current Android versions without modifications of the operating system due to security restrictions. We show that giving applications access to touch events compromises Andorid's security model and that many of the methods to gain this access are penetrations of the security model themselves. There are several different modifications that can be utilized in order to log touch interaction. We propose an application which uses administrator rights to interface with the touch screen device itself and obtains and logs touch interaction data. We also interact with and log other sensors in order to provide associated accelerometer and gyroscope data. In addition we developed a companion application which filters and exports the logged data into different formats. In the future, this thesis will be used for continuous authentication and biometric research on the Android platform.

Preface

We would like to thank our supervisor Mariusz Nowostawski for his continued support throughout all stages of the project and our employers, Soumik Mondal and Patrick Bours, who made this project possible in the first place by providing equipment and user feedback during the implementation phase. We would also really like to give a shoutout to Nikolay Elenkov, the author of Android Security Internals [1]. Without the time we saved by reading his book, we would most likely not been able to finish the research phase with any substantial results.

Contents

List of Figures

List of Tables

1 Introduction

1.1 Employer

Our employer for this project is the Norwegian Biometrics Laboratory which is a part of NISlab [2] at Gjøvik University College [3]. The Norwegian Biometrics Laboratory conducts research in several biometric fields, some of which are behavioural biometrics and continuous authentication [4] conducted by Patrick Bours and Soumik Mondal. We will be working closely with Patrick and Soumik to provide a tool to be used in their research.

1.2 Background

In today's world there is a lot of attention around security and how to protect your own sensitive information. No matter if its cryptography or plain physical security it all normally boils down to two of the three factors of authentication: Something you know like a password or something you have like a key or keycard.

The third factor is something you are. Normally most strong authentication methods implement two-way authentication using the first two factors. This method of authentication is not all to reliable in the way that passwords can be forgotten and keys can be lost. With biometric authentication the hassle with remembering a password or keycard can be avoided since the authentication uses what you are, which one seldom forgets or leaves at home.

The Norwegian Biometrics Laboratory are conducting research on biometric authentication. They have previously done experiments regarding user interaction on a computer running Windows OS using a tool that was developed in a previous bachelor project called BeLT [5] Now they wish to expand their research field by looking into continuous authentication on Android devices. This project can be seen as the first step in behavioral biometric security in Android, by creating a proof of concept which captures the required information.

1.2.1 THEIA, the User Interaction Team

Our group consists of three students of the Bachelor of Science in Information Security programme of GUC [6]. None of us has had any noteworthy experiences programming on the Android platform, but we have developed java applications before. One of our members also has had written a simple application for the windows phone platform.

Our basic programming courses were held in C++ [7, 8], therefore does the challenge in reading native code for Android consist of gaining an overview over the relevant code base. Unfortunately none of us has had any experience with code bases as big as the Android operating system. Other courses of our programme which were highly relevant when creating the application were software engineering [9], operating systems [10], datamodelling and database systems [11] and software development [12].

1.3 Task description

The task this thesis is based on was to create an application for Android hand held phones which captures the natural behaviour of a human interacting with the device. The software should be non-intrusive, and in addition to be able to capture and store the information also be able to retrieve and represent it in a high level overview.

There were three main parts specified:

Key interaction Key related events and their timing information on a millisecond level need to be captured.

Swipe interaction Direction, distance, acceleration and pressure of touchscreen related events (hereafter: *touch events*) need to be captured.

Additional Hooking into other applications and capturing some related information for our employers next level of work.

After some discussion with our employer the parts of task were adapted and specified further (see Appendix D, EMP-15-006 and EMP-15-007). This resulted in goals, limitations and requirements explored in Appendix A as well as a new division of the task:

Research and experiments Interception of data should be acquired without the need to modify the host operating system (see Appendix A). Try to intercept the touch events without such modifications, or show why this is not possible without modifying the system.

Implementation of an application Implement an application that intercepts touch events and stores them into a database.

Export of application gathered data The data should be exported in a format specified with the employers (who will be using this data). A CSV file should be generated, whose columns will be determined in cooperation with the employers.

Additional Other types of events, such as sensor information should be gathered also. Key presses on the on screen keyboard should be intercepted as well.

1.4 Project Goals, Limitations and Requirements

1.4.1 Effect Goals

For our employer this project is expected to accomplish:

- Strengthen the Biometrics Laboratory's ability to do research on biometrics and alternative authentication methods.
- Provide a means to research continuous authentication on Android devices.

For the group members we expect to accomplish:

- Gain a deeper understanding of the Android operating system, especially regarding touch/sensor input and interrupt handling and how this is relayed to applications.
- Gain experience in developing applications for Android devices.

1.4.2 Result Goals

The desired results of this project consists of:

- An application for Android OS which has the functionality of logging user interaction with the touch screen.
- A bachelor thesis describing the projects execution, decisions and academic challenges related to it and the resulting application.

1.4.3 Limitations

It is possible for an application of this kind to collect additional biometric information from various other sources like sound, accelerometer, etc. Due to the relative short development time, the size of the project group and how unclear it is whether this application is possible to create or not given the limitations we have, this project focuses on capturing interaction with the touch screen. Other features may be implemented after our primary goals are completed to our employers satisfaction.

1.4.4 Requirements

The main goal of the project would be simple to attain if modification of the operating system were feasible. Normally the operating system handles those events and determines which applications should receive the information contained. Because of efficiency and security concerns, applications are prevented from accessing the raw information. If the operating system is run in debug mode or an application gains sufficient rights these measures can be circumvented, it is although necessary to modify or exploit weaknesses to achieve those privileges. Since there are some legal implications of distributing applications with such features, and test participants may object to such modifications of own devices, the application should to be able to log behaviour with normal privileges if possible. The project has to be finished within the due date of 15th of May and the project will be considered finished at this point regardless of future application maintenance. The application must be able to generate log-files in plain text and/or CSV format, so that output can easily be integrated with pre-existing systems.

1.5 Equipment specifications

For this thesis, our employer supplied a Motorola Nexus 6 phone. All our development was targeted at this particular device. If not explicitly mentioned, we have only tested the experiments on this single hardware configuration and results may vary. This however is unlikely due to the Android compatibility project defining how a device should react to various inputs and which capabilities the hardware should have [13].

The version of Android used in our experiments is `android-5.0.1_r1 build LRX22C` (API level 21), as this was the version that was pre-installed on the phone and Android `5.1` (API level 22) was first released after we started researching. The source code of the Android platform was of this version, and the source code of the kernel from the `msm` project (we checked out head version `eec2459384835d85318caddbd8245876afc1933b`). If not mentioned otherwise, we refer to code from this branch of Android.

2 Research

Android is an operating system originally designed for mobile phones, but has grown to be implemented for many device types such as tablets, televisions, watches and cars. Android is based on the Linux kernel and it's source code is publicly released, which makes it possible to build one's own variant completely from source.

Operating systems in general are some of the most sophisticated and complex systems designed by man. Android is no exception, with the source of the kernel we inspected weighting in at about 1.5GB in more than $45,000$ files. The somewhat comparable Linux kernel spans just above 15 million lines of code in more than $37,000$ files [14]. The rest of the Android project which does not include the kernel is about 35GB in more than $75,000$ *folders* (more than $500,000$ files).

These metrics visualize the difficulty of gaining an overview over the workings of the Android operating system. Doing an exhaustive analysis is not feasible unless substantial manpower is afforded. For those reasons, we have to limit our research both in depth, in account to what degree we trace function calls, and in width, representing the number of of modules to inspect. We focus on modules and functions with an apparent effect on touch events and their distribution and the distribution chain of such events.

2.1 Overview over Android architecture

As illustrated in Figure 1, the Android operating system is based on a Linux kernel. The security model is therefore quite similar to *nix systems. Each running application has its own associated UID and is isolated from other processes using file permissions [1, chapter 1, "Android's security model" Section]. Access to system files is protected by requiring the rights of `root` (UID 0), `system` (UID 1000), whitelisted system processes or any of the protected GID's (defined in the *android_filesystem_config.h* header) [1, chapter 2, "Permissions and Process Attributes" Section]. Starting with Android 4.3, Android implements SELinux as an additional mandatory access control for all processes, disregarding the processes' UIDs [16]. The first implementations used permissive mode, but from Android 5.0 onwards restrictive mode is the default configuration.

Applications can be either programmed in java and executed in the Dalvik VM or be compiled as a native binary. They mainly interact with the application framework which in turn may expose interfaces to lower system functions and hardware devices. Access to such functions normally requires declaring the accompanying permission [1, chapter 2, "Permissions" Section].

Most of Android's system applications and the application framework itself are written in java and run in the Dalvik VM [1, chapter 1, Figure 1-1], and do therefore adhere to the same security limitations as user installed applications.

2.1.1 Touch event propagation

Most sensors are available to any application should the user choose to grant the required permissions, therefore we took a closer look at touch events (aka motion events) which are by design only visible to the affected application. The process which detects and dis-

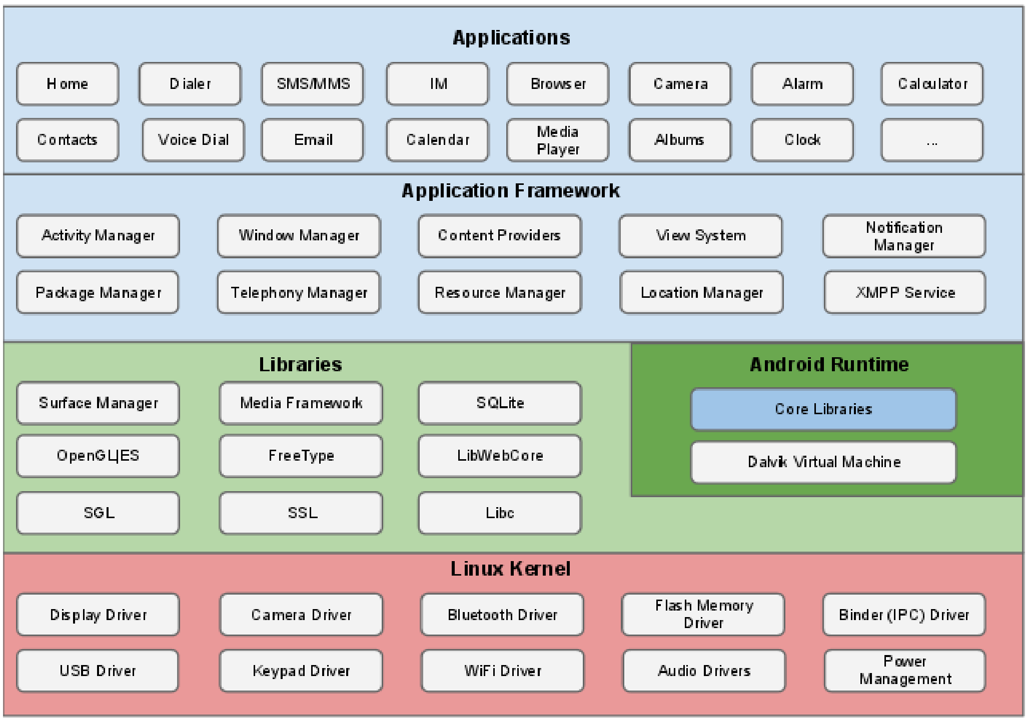


Figure 1: The Android architecture (source: [15])

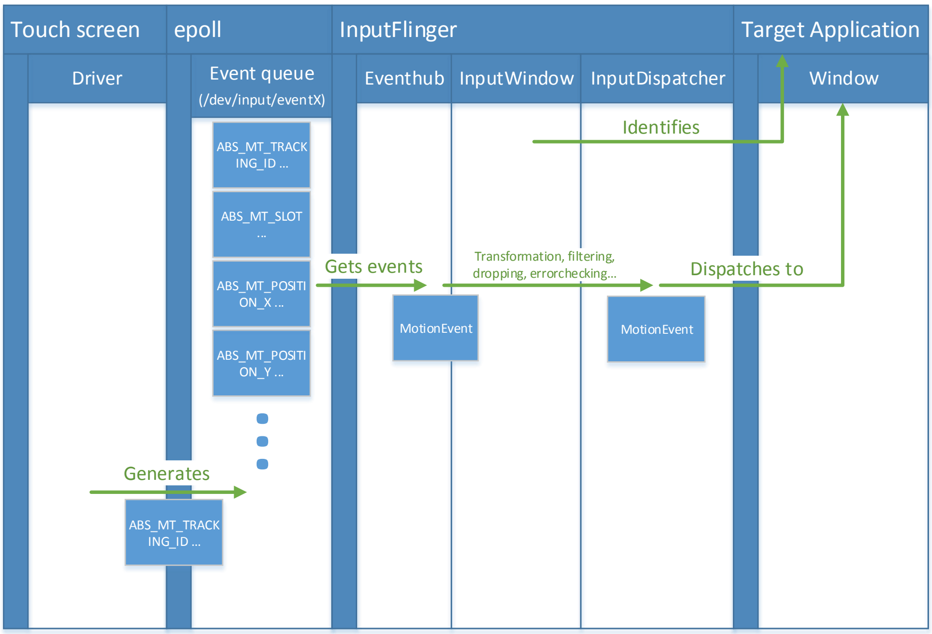


Figure 2: Motion event propagation chain simplification

```
shell@shamu:/dev/input $ ls -l
ls -l
crw-rw---- root     input     13,  64 1970-01-12 16:37 event0
```

Figure 3: Access rights for /dev/input/event0

patches those events is illustrated in Figure 2 and explored in the following paragraphs:

The touch screen driver does, as most other hardware drivers, use `epoll` to dispatch its events to the input services. On Android the file descriptors `epoll` registers for this purpose are the `eventX` files (where 'X' is a number assigned to the component/socket) in the `/dev/input` folder. These files are normally only accessible with the UID or GID of `root`(see Figure 3).

The `inputflinger` service, which is part of the application framework, interacts with `epoll` to retrieve the registered events. The `getEvents`-function [17] does the work of reading the events, and the code which sends them to the affected windows is found in the `InputDispatcher` [18] and `InputWindow` [19] files.

It is noteworthy that code in the `inputDispatcher` checks the registered windows for some special flags that can be set by the `WindowManager`. Some of those flags will result in the window receiving copies of selected events [18]. Setting the right combination of flags is however impossible from Android `4.0.3` onwards, since these flags will be modified silently (see Capturing touch data with an overlay).

When the window receives an event, it will handle it as illustrated in Figure 4.

2.2 Research, Proof of Concepts and Experiments

We started our research by trying to identify the event propagation chain described in Section 2.1.1. To aid our efforts we downloaded a copy of the source code for both the Android operating system and the kernel running on our device. We then identified several stages of the chain in which interception is possible. For each of those we tried to identify methods to acquire touch events, and tried to infer whether those are implementable on a consumer device without modifications of the operating system or exploiting weaknesses. The following Sections detail those methods.

2.2.1 Screen Overlay

We discovered that it is possible to create an overlay that runs in the foreground on top of every other application. This immediately caught our attention as something that we might be able to use so we wanted to investigate deeper. Our first thoughts were that an overlay running on top of every other application should, based on the way Android OS handles touch events(see Figure 2), be the target application to receive the touch events. With this hypothesis as the root we started researching this possibility.

Capturing touch data with an overlay

During research we came across that Android has been prone to so called "tapjacking" attacks. Tapjacking involves showing something to the user and having them act based on what they see, for example click a button that says to start the game. However, what actually happens is that the click is received by an underlying view that can do whatever

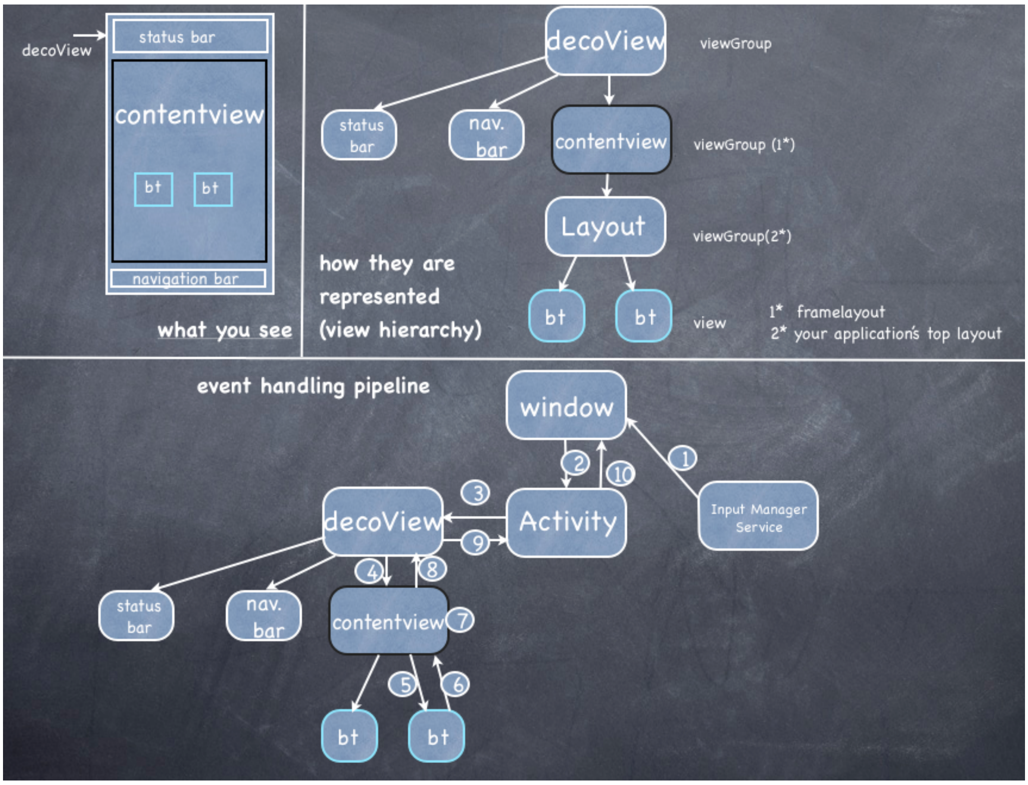


Figure 4: Application window event handling chain (source: [20])

it wants with that click. In this way can users be tricked into doing actions they did not intend, for example downloading malicious software [21].

On Android devices, tapjacking has been accomplished using toasts or screen overlays. Depending on the flags defined for such elements, touch events can be passed through to the view laying behind. For example if you create a screen overlay and specify the type TYPE_SYSTEM_OVERLAY [22], the InputDispatcher [18] will not select the screen overlay as the target view and all touch events will pass through to the underlying view. Also if you specify the flags FLAG_WATCH_OUTSIDE_TOUCH and FLAG_NOT_TOUCH_MODAL the InputDispatcher [18] will duplicate the touch event and dispatch a MotionEvent.ACTION_OUTSIDE to the overlay view. This introduces some security concerns as a screen overlay could be used to log touch data without the user noticing.

However after researching a bit more we found out that according to the Android documentation you will not receive the full touch gesture in the MotionEvent.ACTION_OUTSIDE, only the first touch event [23]. As such it is not usable for our purposes as we would not get the entire gesture. Also this lessens the security concern a bit as you would only get relevant data for taps and not entire gestures.

Also, we found a post on stackoverflow.com [24] that claims that Google changed it so that screen overlays of TYPE_SYSTEM_OVERLAY will no longer receive any touch events after Android 4.x. We checked the source code of several Android versions and determined that as of Android 4.0.3 [25] Google changed the way overlay view works so that you can no longer get touch events using FLAG_WATCH_OUTSIDE_TOUCH on a screen overlay view of TYPE_SYSTEM_OVERLAY or TYPE_SECURE_SYSTEM_OVERLAY.

More specifically they adjusted a method in PhoneWindowManager called adjustWindowParamsLw. In Android version 4.0.3 and higher, if you specify TYPE_SYSTEM_OVERLAY or TYPE_SECURE_SYSTEM_OVERLAY the flags FLAG_NOT_TOUCHABLE and FLAG_NOT_FOCUSABLE will be automatically added and the FLAG_WATCH_OUTSIDE_TOUCH removed by the OS.

With these changes, screen overlays can no longer be used to intercept touch events. From a security stand-point this is a good thing as screen overlays can no longer be used to intercept touch events and pass them on at the same time. As a result, tapjacking attacks are no longer possible to achieve using screen overlays. However this also means that this method is not usable in our application.

Another possibility is to define the overlay with TYPE_SYSTEM_ALERT, this way you will become the target application and at such receive all the touch events. However doing so will result in consuming the touch events so that no underlying view will receive them.

Injecting events

Although it is not possible to let the events pass through the overlay view after we get them, what if we could capture all the events with an overlay view and afterwards inject them into the correct window? Then we would be able to create a screen overlay of the type TYPE_SYSTEM_ALERT and intercept all gestures and pass them on afterwards.

We did some research on this topic and it turns out it is possible to inject events on Android using Instrumentation [26], however you will need the system permission INJECT_EVENTS which is only grant-able to applications signed with the system key(see Appendix: F).

Another way to inject touch events is to write them directly to the Linux event files located in /dev/input/eventX, where X is a number representing the input device, as the touch driver does(see Touch event propagation). This way you avoid the Android permission issues, however this method requires root access as the event files by default has permission set to 660(read and write for owner and group only)(see Figure 3). [27]

These two methods are as far as we know the two most usable ways of injecting events to other applications in Android. However since neither of these methods are accessible without having root or knowing the system signing key we can not use either for our project.

The fact that it is not possible to inject events on a standard Android device lessens security concerns as those injections would create many possibilities to cause unintended behaviour. If one was able to modify or generate fake touch events there are many ways to alter the program flow of the targeted application. For example could you use these possibilities to generate clicks on advertisements in your own application or if the user has the payment password remembered on Google Play conduct payments, thus generating more income. Also, you could possibly click buttons and links without the user's consent forcing them to download applications or malware as you please. You could even generate a bunch of random touch events at a fast pace, thereby disabling the use of the phone. A potentially even worse security implication could be the ability to compromise the permission granting request, modifying the user's input and accepting requests at will.

Conclusion

We have determined that it is possible to capture gestures using an overlay view. However in doing so you will consume the gestures and they will not be passed on to the underlying view. There is also no way of passing on gestures to a view in another application without having permissions only grant-able to system applications or applications with root access. This implies that the phone cannot be used for anything else as long as the overlay view is open. Although this is good security practise, it poses a problem for applications with a legitimate need to register touch events not meant for them. As our application is to be used in biometrics research the user needs to be able to interact with the phone while touch data is being logged. Because of these factors, an overlay is not usable for our purposes.

2.2.2 AccessibilityService

Another idea we had was based on an API offered by Android to support usability for users that require additional or different information in order to use the device, for example users with visual or hearing disabilities. To do this, Android provides developers with the possibility to create an `AccessibilityService` [28]. `AccessibilityServices` can register to receive a callback whenever an `AccessibilityEvent` of the specified type has been fired. `AccessibilityEvents` [29] are sent by the system upon notable events in the user interface, for example if the focus has changed or a button has been clicked. Then the `AccessibilityService` can act upon the `AccessibilityEvent` and provide some feedback to the user as it sees fit.

Examining the contents of accessibility events

After determining that this could be a possibility we wanted to get a look at what information we could get through `AccessibilityEvents`. Therefore we conducted an experiment creating our own `AccessibilityService` (source code can be found in Appendix C.1).

In the experiment we tried to catch all events possible and we also enabled touch exploration which allows us to get some gestures. While running the experimental application we received `AccessibilityEvents` corresponding to events we created on the device (see Android documentation [29] for a complete list of `AccessibilityEvents`). For example we received information about the current focus and when an application is closed/started. Due to touch exploration being enabled we also received information about gestures and touch events. On each touch or gesture we performed we received AccessibilityEvents to go with them, and it soon became clear that they follow a certain pattern based on whether gesture detection recognizes the gesture or not.

If the gesture is recognized it will follow the pattern described in Figure 5 and the gesture is identified with a type. For example GESTURE_SWIPE_UP, representing an upwards swipe. Should the gesture not be recognized, another pattern will be followed, described in Figure 6 and `AccessibilityEvents` of TYPE_VIEW_HOVER_ENTER and TYPE_VIEW_HOVER_EXIT will be given instead of the gesture identification. These `AccessibilityEvents` contain information on when the touch gesture has entered and exited the focus of a view, for example a `TextView` or a `Button`.

Examining the contents of each `AccessibilityEvent` further we discovered that there is no data regarding the x and y position throughout the gesture. According to the Android documentation on `AccessibilityEvents` [29], the only x and y coordinate

TYPE_TOUCH_INTERACTION_START The user has touched the screen.

TYPE_GESTURE_DETECTION_START Starting gesture detection.

onGesture Result of gesture detection, refer to gIdToString() in Appendix: C.1 for a complete list of possible results.

TYPE_GESTURE_DETECTION_END Ending gesture detection.

TYPE_TOUCH_INTERACTION_END The user stopped touching the screen.

Figure 5: AccessibilityEvent pattern with gesture detection

TYPE_TOUCH_INTERACTION_START The user has touched the screen.

TYPE_TOUCH_EXPLORATION_GESTURE_START Starting touch exploration gesture.

TYPE_VIEW_HOVER_ENTER The gesture enters a focus (e.g. overlaps with a view).

TYPE_VIEW_HOVER_EXIT The gesture exits a focus.

TYPE_GESTURE_DETECTION_END Ending gesture detection.

TYPE_TOUCH_INTERACTION_END The user stopped touching the screen.

Figure 6: AccessibilityEvent pattern without gesture detection

onAccessibilityEvent:	[type] TYPE_TOUCH_INTERACTION_START
onAccessibilityEvent:	[type] TYPE_TOUCH_EXPLORATION_GESTURE_START
onAccessibilityEvent:	[type] TYPE_VIEW_HOVER_ENTER [class] android.widget.ListView
onHoverEvent:	[scrollX] -1 [scrollY] -1
onAccessibilityEvent:	[type] TYPE_VIEW_HOVER_ENTER [class] android.widget.TextView
onHoverEvent:	[scrollX] -1 [scrollY] -1
onAccessibilityEvent:	[type] TYPE_VIEW_HOVER_EXIT [class] android.widget.TextView
onHoverEvent:	[scrollX] -1 [scrollY] -1
onAccessibilityEvent:	[type] TYPE_VIEW_HOVER_EXIT [class] android.widget.ListView
onHoverEvent:	[scrollX] -1 [scrollY] -1
onAccessibilityEvent:	[type] TYPE_TOUCH_EXPLORATION_GESTURE_END
onAccessibilityEvent:	[type] TYPE_TOUCH_INTERACTION_END
onAccessibilityEvent:	[type] TYPE_TOUCH_INTERACTION_START
onAccessibilityEvent:	[type] TYPE_GESTURE_DETECTION_START
onGesture:	[type] GESTURE_SWIPE_RIGHT
onAccessibilityEvent:	[type] TYPE_GESTURE_DETECTION_END
onAccessibilityEvent:	[type] TYPE_TOUCH_INTERACTION_END
onAccessibilityEvent:	[type] TYPE_TOUCH_INTERACTION_START
onAccessibilityEvent:	[type] TYPE_GESTURE_DETECTION_START
onGesture:	[type] GESTURE_SWIPE_UP_AND_RIGHT
onAccessibilityEvent:	[type] TYPE_GESTURE_DETECTION_END
onAccessibilityEvent:	[type] TYPE_TOUCH_INTERACTION_END

Figure 7: Sample output(excerpt) from AccessibilityService test

that appears on any `AccessibilityEvent` are on events of type TYPE_VIEW_HOVER_ENTER and TYPE_VIEW_HOVER_EXIT that can be accessed through `getScrollX()` and `getScrollY()`. These however are offsets solely relevant to scrolling views. Therefore they are not useful in determining the absolute screen coordinate, and for views which are not scrolling ones they were all -1, as seen in Figure 7.

Conclusion

Using an `AccessibilityService` we were able to get some useful information and even determine the type of gesture performed. However our goal is to get precise touch data containing x and y coordinates throughout the gesture. Even if `AccessibilityEvents` of type TYPE_VIEW_HOVER_ENTER and TYPE_VIEW_HOVER_EXIT did contain x and y coordinates they would only relate to the scrolling offset of the related view. As such, the touch information we are able to collect through an `AccessibilityService` will not be sufficiently detailed to be used in biometrics research.

2.2.3 Dropped methods

There are some stages in the event propagation in which the interception requires access which exceeds the permissions granted to normal applications. From Figure 2 one can infer that there are four stages in the event propagation chain which could be targeted to obtain touch events:

1. The touch screen driver
2. The event queue managed by `epoll`
3. The `inputFlinger` service and other related parts of the Android application framework
4. The other applications

The touch screen driver is, as the event queue and the application framework, protected from modification by normal applications. Changes to any of these are outside of the scope of our research (see Section 1.3), but could be implemented if one has access to the source code of the device and the required platform keys.

Modifying the driver seems to be the most risky and difficult approach, but the greatest disadvantage of this method is that the driver may vary between different hardware implementations. In theory the driver could be made to e.g. write all events to an additional log file which is readable by an application or the user.

Since the event queue managed by `epoll` is assigned a file descriptor in the file system, the events could be read as if the queue was a file. There is also the `getevent`-command, which simplifies parts of this process. Unfortunately are these files not readable by all applications.

The `inputFlinger` service and related application framework entities could also be modified to log events to accessible locations or dispatch copies to another application. These modifications would result in as precise information logged as the intended application receives and patches should be portable between many configurations. But these require access to either the platform key and the source code of the Android version installed or a device with an unlocked boot loader and a pre-patched system image.

Lastly is the modification of other applications not really a realistic option, since applications are isolated from each other and the target application is determined and accessed by a service in the application framework. This implies it would require changing *all* other applications to intercept all touch events. This is, beside being extremely impractical, not always possible since not all source code is public.

2.3 Conclusion of Research

As touch events travel up in the Android architecture, access to them is denied by the security model of the operating system. While the event still is somewhere between the kernel and the application framework, non-system applications have no means of reading its contents. At these stages, the only way of accessing such an event would have to be either provided by the Android system itself, or by a modified or added system application.

We looked at different methods of interacting with the application framework and did not identify any that would allow access to touch events targeted at other applications in Android versions exceeding `4.0.3`. Neither accessibility services can obtain touch data of any substantial detail, nor is there any attainable permission in the application framework which would allow so.

3 Development

As we determined that it is not possible to log global touch events without modifications to the operating system we had to choose a suitable method to achieve this. There are several ways in which you can modify the Android operating system in order to get access to the touch events. For example you could modify a part of the kernel that handles the touch event to make them available or send them to your application. Also you could simply root the device in order to gain administrator rights. That way you would gain read and write access to the `/dev/input/eventX` files (see Figure 3) and you would be able to read the touch event directly as they are being queued for propagation.

We believe that rooting the device and reading from `/dev/input/eventX` would be the best solution because it is less intrusive than making modifications to the Android operating system. In addition, this solution makes installing the application a much simpler feat than if we had to replace a part of the operating system on installation.

In order to be able to implement this solution we had to root the phone. The way we achieved this was to first unlock the boot loader by restarting the phone into the boot loader using the `fastboot oem unlock` command. Then we installed the teamWin recovery solution[30] which has an installation option for SuperSu, which we in turn used to gain root privileges on the device.

The Implementation phase lasted for about a month and within this time we wrote the Android and Desktop applications which together consists of roughly 2900 lines of code. However it is safe to say that we also have thrashed at least 1000 lines of code during this process as well. Javadoc for the source code is available at the Theia webpage [31].

3.1 Specifications

The original specifications implemented everything as a standalone Android application which stored the all data in plain on the device (see Appendix A). After some discussion we decided that the usage of a binary storage format would likely give better performance during logging. After some feedback from our supervisor we decided to implement the storage in the form of an SQLite Database. This solved a problem wherein several sensors wrote to the same file and potentially conflicting with each other. As an additional bonus the database is in a format that is widely supported and not exclusively bound to our source code. For more information about how the database is structured see Figure 15 and Section 3.2.1.

Since we chose to use a binary format and our employers wanted CSV file output, the need for a converter application arose. Because desktop computers are several orders of magnitude faster performing such conversion processes than mobile phones, which have more limited cpu and memory resources, we choose to develop the converter for desktop computers. For portability reasons we chose to implement it in Java.

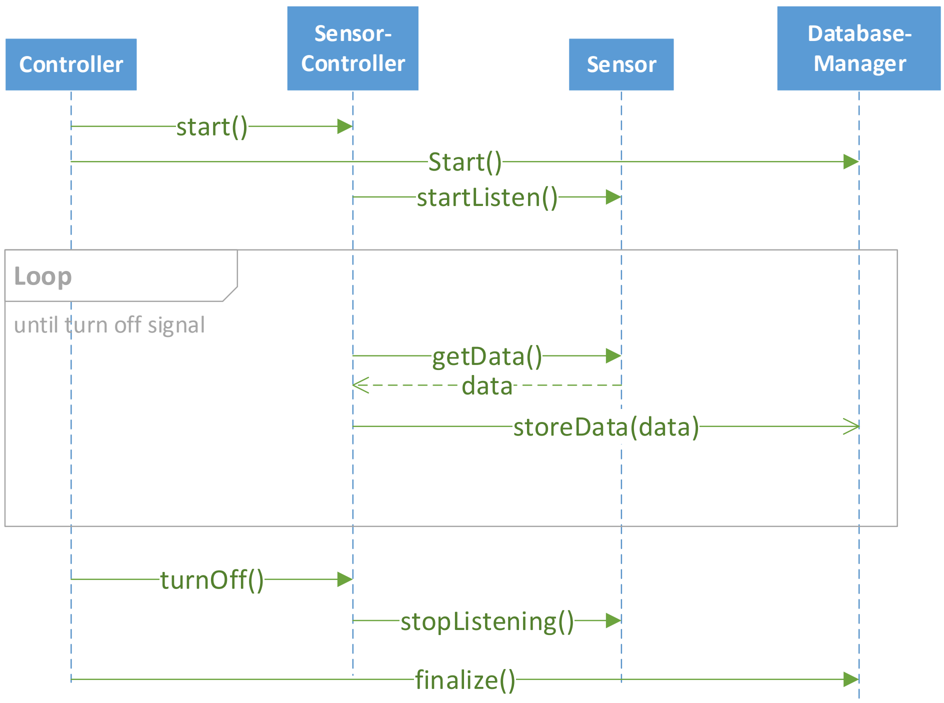


Figure 8: App process view and dataflow in the Android application

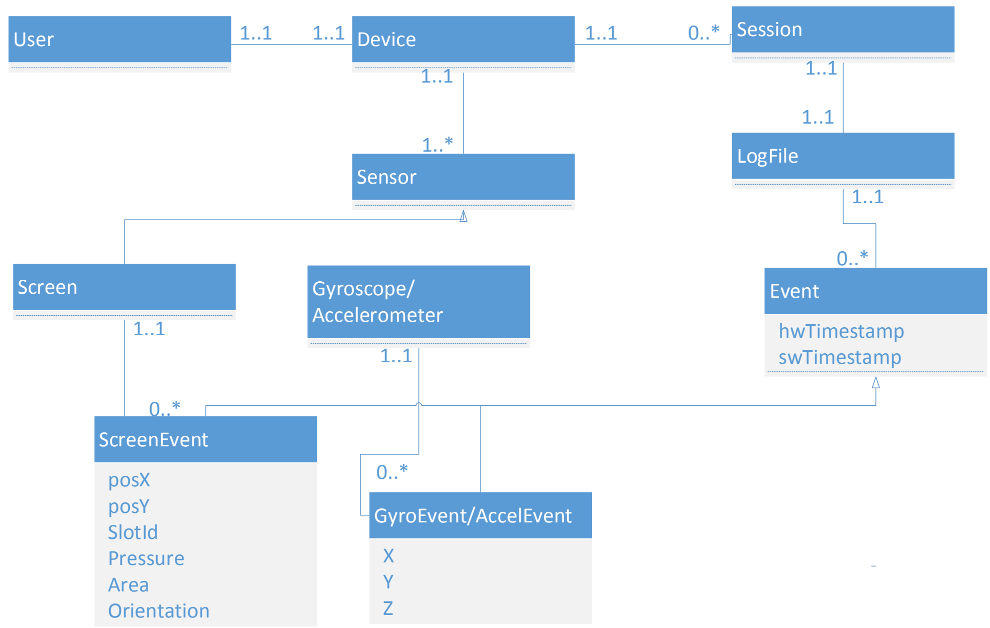


Figure 9: The current domain-model of the Android application

3.2 Android Application

3.2.1 Program architecture and Design

The android application is designed with modularity and extensibility in mind. Therefore it is implemented with a controller which manages two sensor controllers, one for the touch screen and one for all the commonly available sensors. We have decided to only implement the gyroscope and accelerometer sensors as these were requested by our employer and seemed the most reasonable to implement. This changed our domain-model from the one in the application specifications (see Appendix A) to the model represented in Figure 9. Both accelerometer and the gyroscope share the same attributes and data points while touch screen events have other ones.

As seen in Figure 8 all functions are started from the `ServiceLauncher` class, which is the control interface the user can use to set experiment meta data and settings for each specific experiment. Sensors are set to sample as often as possible which results in a much more granulated dataset. The main reason for this is that, it is easy to sample down data sets but counter-productive to up-sample them. Data can always be discarded later, but non-existent data cannot be created out of thin air.

ServiceLauncher

The `ServiceLauncher` is the main activity of the application and the first one the user interacts with when he starts the application. Here, the user is presented with a menu (see Figure 10). From this menu the settings menu is accessible (see Figure 11), in which they can set meta data and choose which sensors should be logged in the session they are about to start. After the settings are set the user can start the logging session by pressing the start button which will start three services: `ServiceController`, `TouchService` (if

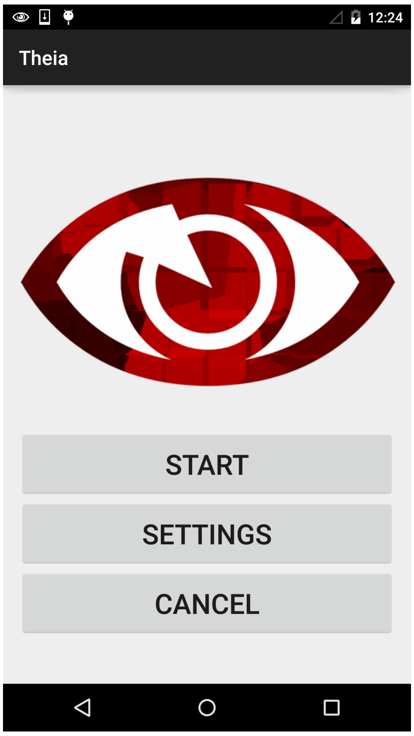


Figure 10: User Interface of application

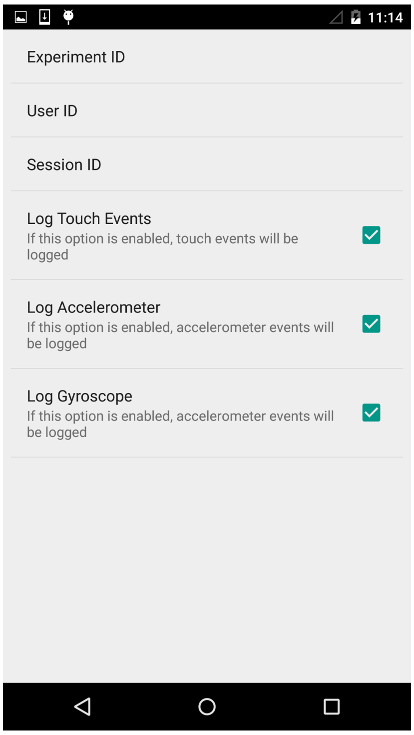


Figure 11: Settings menu of application

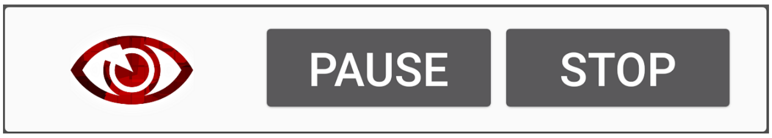


Figure 12: Notification when logging is in progress

Figure 13: Notification when logging is paused

touch data is logged) and SensorService (if any other sensors are logged).

ServiceController

Upon creation, the `ServiceController` will fire a notification that can not be removed until the service is stopped. It will also register a `BroadcastReceiver` to listen for actions from the notification. The notification consists of a pause or resume button, depending on the state of the notification (see Figure 12 and 13), as well as a stop button to stop the logging process. Whenever the `ServiceController` receives a broadcast from the notification, it is responsible for determining what command has been issued and act accordingly. For example if the user presses the pause button in the notification, a broadcast will be sent to the `ServiceController` with a "pause" tag. The `ServiceController` is then responsible for updating and conveying the message to `TouchService` and `SensorService` using a broadcast. In addition, it will update the notification layout if needed. `ServiceController` is also responsible for writing the data base to file upon logging stop.

TouchService

When the `TouchService` service is created it will open a shell, list the files located in `/dev/input/eventX` and determine which one(s) corresponds to touch screens. This is necessary because the file name belonging to the screen may vary from one device to another, and some devices might even support more than one touch screen [32, chapter 12]. For every file that corresponds with a touch screen, it will spawn a thread that opens a shell and executes the `su getevent -lt /dev/input/eventX` command where "eventX" corresponds to the name of the current file. This command will output a continuous stream of lines as new lines are added to the file (see Figure 14). It will then continuously read the output, decode the information(see the "Format of /dev/input/eventX" section), transform it into a `TouchEvent` (see the Utility and Helper Classes section) and append it to an array of `TouchEvents`.

Every set period of time (by default 1 second) the array of `TouchEvents` will be flushed to the database using the DatabaseManager. Upon creation, `TouchService` will also register a `BroadcastReceiver` to listen for broadcasts from `ServiceController` and act accordingly.

Programming this class provided us with some challenges. Amongst other things we had some trouble with `BufferedReader`'s `readLine()` blocking the thread when we were trying to pause the application. The reason was because `readLine()` was waiting for an entire new line to be available from the output before reading. Because the touch driver does not add a line separator until just before writing the next touch event to file, the `readLine()` call to blocked. If it was waiting for a new touch event as we tried to pause the logging process like this, we had no way of terminating the thread.

In order to be able to read from the file in a non-blocking manner, we implemented a `java.io.InputStreamReader` with a self developed, non-blocking version of `readLine()`. This function reads the line character by character, but does not block if no more characters are available from the stream.

Format of /dev/input/eventX

The `/dev/input/eventX` file corresponding to the touch screen has a particular format that is defined in the linux kernel documentation [33]. For a sample snippet of the output format, refer to Figure 14. In this format, every new line is started with a timestamp surrounded by square brackets. This timestamp is an arbitrary value that can vary from device to device, however the value corresponds to an offset from a given time, for example time since last boot. Following the time stamp, the type of event will be specified as EV_ and a code defining the event type. For our case, the relevant codes are "ABS" which indicates a touch event and "SYN" which indicates the end of an event. After the type of event has been specified, the type of information will follow. For events of type EV_SYN, they will always be followed by SYN_REPORT and 00000000 signalling the end of a touch event in our case. For events of type EV_ABS there exists a wider range of possible information and values (all values are given in hexadecimal):

ABS_MT_TRACKING_ID is followed by a value that identifies a touch gesture provided by one finger. In the case that there is more than one finger on the screen, each will be identified by it's ID.

ABS_MT_SLOT identifies the finger the touch event belongs to. This information is only present if there are more than one contact point with the screen, and will be indexed starting from 0 from the first contact point.

ABS_MT_POSITION_X is the x position of the touch event.

ABS_MT_POSITION_Y is the y position of the touch event.

ABS_MT_TOUCH_MAJOR represents the length of the largest axis on the contact surface.

ABS_MT_TOUCH_MINOR represents the length of the smallest axis on the contact surface. Not all devices, including our own, support this feature, in which case only ABS_MT_TOUCH_MAJOR will be given.

ABS_MT_PRESSURE represents the pressure of the current touch gesture.

ABS_MT_ORIENTATION represents the orientation of the touch, as in which direction the finger is pointing. This value is defined arbitrarily. On our device it ranges from 0-255 in clockwise direction.

There exist additional types of information other than these that may be returned if the device supports them. However the listed information types are the most relevant ones as well as the ones we rely on in our application. For a complete list of available types see the kernel documentation [33].

```
[ 27241.278961] EV_ABS ABS_MT_TRACKING_ID 00000885
[ 27241.278961] EV_ABS ABS_MT_POSITION_X 00000377
[ 27241.278961] EV_ABS ABS_MT_POSITION_Y 00000521
[ 27241.278961] EV_ABS ABS_MT_PRESSURE 00000039
[ 27241.278961] EV_ABS ABS_MT_TOUCH_MAJOR 00000006
[ 27241.278961] EV_SYN SYN_REPORT 00000000
[ 27241.299783] EV_ABS ABS_MT_POSITION_X 00000359
[ 27241.299783] EV_ABS ABS_MT_POSITION_Y 00000509
[ 27241.299783] EV_ABS ABS_MT_TOUCH_MAJOR 00000005
[ 27241.299783] EV_SYN SYN_REPORT 00000000
[ 27241.306799] EV_ABS ABS_MT_POSITION_X 00000347
[ 27241.306799] EV_ABS ABS_MT_POSITION_Y 000004fc
[ 27241.306799] EV_ABS ABS_MT_TOUCH_MAJOR 00000006
[ 27241.306799] EV_SYN SYN_REPORT 00000000
[ 27241.313814] EV_ABS ABS_MT_POSITION_X 00000333
[ 27241.313814] EV_ABS ABS_MT_POSITION_Y 000004ee
[ 27241.313814] EV_SYN SYN_REPORT 00000000
[ 27241.320822] EV_ABS ABS_MT_POSITION_X 00000320
[ 27241.320822] EV_ABS ABS_MT_POSITION_Y 000004e1
[ 27241.320822] EV_ABS ABS_MT_TOUCH_MAJOR 00000007
[ 27241.320822] EV_SYN SYN_REPORT 00000000
[ 27241.327830] EV_ABS ABS_MT_POSITION_X 0000030c
[ 27241.327830] EV_ABS ABS_MT_POSITION_Y 000004d6
[ 27241.327830] EV_ABS ABS_MT_TOUCH_MAJOR 00000004
[ 27241.327830] EV_SYN SYN_REPORT 00000000
[ 27241.335656] EV_ABS ABS_MT_POSITION_X 000002f9
[ 27241.335656] EV_ABS ABS_MT_POSITION_Y 000004cb
[ 27241.335656] EV_ABS ABS_MT_TOUCH_MAJOR 00000005
[ 27241.335656] EV_SYN SYN_REPORT 00000000
[ 27241.342811] EV_ABS ABS_MT_POSITION_X 000002e6
[ 27241.342811] EV_ABS ABS_MT_POSITION_Y 000004c0
[ 27241.342811] EV_ABS ABS_MT_TOUCH_MAJOR 00000006
[ 27241.342811] EV_SYN SYN_REPORT 00000000
[ 27241.349802] EV_ABS ABS_MT_POSITION_X 000002d7
[ 27241.349802] EV_ABS ABS_MT_POSITION_Y 000004b8
[ 27241.349802] EV_ABS ABS_MT_TOUCH_MAJOR 00000007
[ 27241.349802] EV_SYN SYN_REPORT 00000000
[ 27241.356838] EV_ABS ABS_MT_POSITION_X 000002c7
[ 27241.356838] EV_ABS ABS_MT_POSITION_Y 000004b0
[ 27241.356838] EV_SYN SYN_REPORT 00000000
[ 27241.363916] EV_ABS ABS_MT_POSITION_X 000002b9
[ 27241.363916] EV_ABS ABS_MT_POSITION_Y 000004a9
[ 27241.363916] EV_ABS ABS_MT_TOUCH_MAJOR 00000006
[ 27241.363916] EV_SYN SYN_REPORT 00000000
[ 27241.370999] EV_ABS ABS_MT_POSITION_X 000002aa
[ 27241.370999] EV_ABS ABS_MT_POSITION_Y 000004a3
[ 27241.370999] EV_SYN SYN_REPORT 00000000
[ 27241.377960] EV_ABS ABS_MT_POSITION_X 0000029b
[ 27241.377960] EV_ABS ABS_MT_POSITION_Y 0000049e
[ 27241.377960] EV_ABS ABS_MT_TOUCH_MAJOR 00000007
[ 27241.377960] EV_SYN SYN_REPORT 00000000
[ 27241.413518] EV_ABS ABS_MT_TRACKING_ID ffffffff
[ 27241.413518] EV_SYN SYN_REPORT 00000000
```

Figure 14: Sample output from su getevent -lt /dev/input/eventX

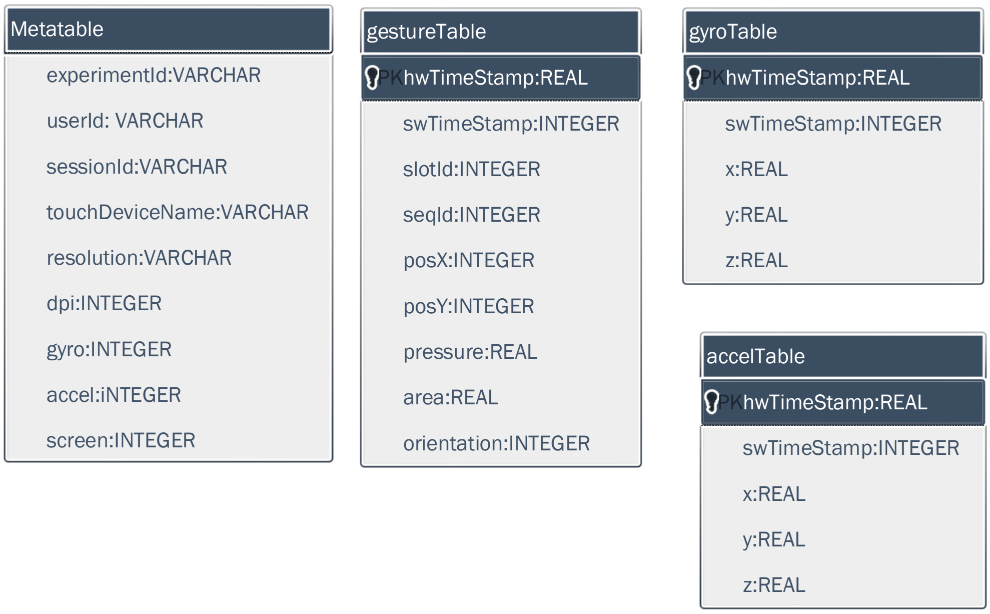


Figure 15: The architecture of the database stored on the phone

SensorService

Upon creation the `SensorService` service starts listening to the sensors that the user specified in the settings screen. It also registers a `BroadcastReceiver` to listen for Broadcasts from the `ServiceController` and act accordingly. Whenever a sensor registers an event, this service is notified, collects the data from the sensor and inserts it into the database using the DatabaseManager.

DatabaseManager

The `DatabaseManager`'s task is to instantiate the database and afterwards pushing all events sent to it to the corresponding tables. Each type of event has its own table with different attributes. The `metaTable` will at all times only have one row with data as its purpose is to hold, as the name hints, the metadata of a session: experiement id, user id, session id, which sensors were active during the experiment and so on.

All events are inserted into different tables with corresponding columns in each table. two columns worth noticing in particular are `hwTimeStamp` and `swTimeStamp`. The difference between these being that `hwTimeStamps` are created by the hardware system call for the sensors while `swTimeStamps` are created by us in the application. The reason for having to implement another timestamp was that the hardware generated timestamp does not necessarily use the same offset and format as the other timestamps. Said offsets and formats are not mentioned in the Android documentation and are therefore purely up to the manufacturers to decide.

`gyroTable` and `accelTable` have the same fields, but since this can change we have chosen to log them in different tables to minimize time spent rewriting. Coincidently this also makes it easier to implement more sensors to log as gyroscope and accelerometer are only a subset of the recommended sensors for Android[13].

Utility and Helper Classes

TouchEvent holds data on a touch event such as x and y coordinates, timestamp and other relevant data.

Settings contains the settings provided by the user or saved since the last session.

SettingsActivity displays the settings list and is responsible for updating the settingsfile as settings are changed.

MetaContainer holds the meta data for the current session.

Resource Files

activity_service_launcer.xml contains the layout for the ServiceLauncher class.

notification_pause.xml contains the layout for the notification in it's default state.

notification_resume.xml contains the layout for the notification in it's paused state.

settings.xml contains the contents used to populate the settings list.

3.3 Desktop Application

We evaluated several designs and architectures for the desktop application. The first iteration had a static view of the database in several predefined columns and provided a button for CSV export. Data was populated to the fitting column. This we refined further by dynamically adjusting the amount and names of the columns in regards to the database contents.

When we considered different output formats (different CSV definitions as well as different file types) and played with the thought of implementing filters and such, we came to use the pipeline architecture which is currently implemented.

We made the desktop application (dubbed "Raw Converter") using the JAVAFX-library to create the graphical user interface. Thanks to the pipeline architecture, the application is widely extensible. This is facilitated by the `DbProcessor` interface. Using this interface, we implemented two algorithms: The CSV exporter which exports the database to a comma separated file and a simple down-sampling algorithm to remove rows which differentiate too little from the previous row (see Section 3.3.3).

3.3.1 Program architecture and Design

The applications main data structure is located in the `Controller` class and consists of two observable lists which each correspond to a `ListView` container which in turn creates the GUI element presented to the user. The first list represents all avilable modules. A module may be chosen from the first list to be added to the second one. By filling up the second list with these modules, the algorithms and their order of execution is set. All of the algorithms implement the `dbProcessor` interface which assures the availability of functions for representing the class in the list, parametrizing it (if necessary) and executing the algorithm on the database before passing the results down the pipeline. This makes it possible to export several files with different levels of filtering and focus areas, as well as performing multiple passes with the same algorithm.

As a precaution the raw database which is loaded into the application is never manipulated itself, but a internal copy is created. Therefore, the original raw data file will never

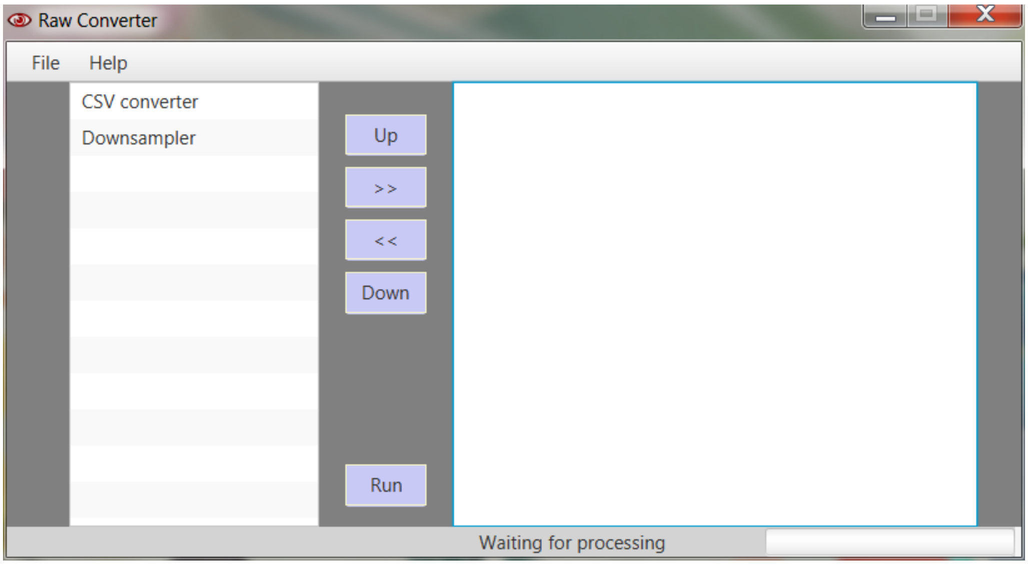


Figure 16: A screen shot demonstrating normal usage of the program

be changed as a result of using the converter for filtering and exporting. This ensures the integrity of raw data by hindering accidental overwrites.

We Implemented a helper class to ease the implementation of configurable algorithms as a `DbProcessor`. This class provides access to the metadata of the loaded database like the names of the contained tables, which columns each table has and the type of the columns in a given table.

3.3.2 User Interface

The user interface is designed with the goal of providing user-friendly access to all implemented algorithms and an intuitive way to extract data correctly in both simple and more complex cases, such as exporting to 4 separate files with different granulation for comparison of biometric indicators. This is done by simply placing the desired algorithms in the correct order (see Figure 16). Note that exporter algorithms do not alter the data, they only read it and pass it on.

To inform the user about conversion progress, there is also built in a progress bar that updates itself after each pipeline algorithm is finished. In future it would be a nice upgrade that it also tries to gauge the progress of individual modules as well, but because of time constraints and the level complexity this change has, this was not attempted.

Currently click and drag functionalities are not implemented. This is a feature that may be added in the future, but the value it would add was not great enough to warrant looking further into it.

3.3.3 Implemented algorithms

CSV exporter

The “CSV exporter” algorithm creates an ASCII file and pushes the entire database as comma separated values into it. Table 1 shows an excerpt of such a generated `.csv` file.

The first column represents the sequence number, which is an unique id of the event. This is simply a counter which increases throughout the file.

The next line shows what kind of data is to be expected over the next lines. If the

Seq	Type	Coordinates	Timestamp	GestureId	Slot	Pressure	Area	Orientation
764	T	581_1573	23714399046374	750	-1	85	9	-1
765	T	598_1627	23714415150853	750	-1	85	7	-1
766	T	608_1681	23714430945280	750	-1	85	-1	-1
767	T	613_1741	23714442498509	750	-1	85	6	-1
768	A	-1.0558319_ 8.389282_ 6.3039246	23714443499186	750				
769	T	610_1806	23714454383874	750	-1	62	-1	-1
770	T	599_1876	23714464296634	750	-1	62	5	-1
771	T	582_1953	23714475740436	750	-1	62	4	-1
772	T	556_2040	23714485814394	750	-1	37	3	-1
773	A	0.09815979_ 5.7317047_ 9.138657	23714496151790	750				
774	G	0.72224426_ - 0.0021362305 - 0.034088135	23714503546686	750				
775	T	510_1799	23714506207519	0	-1	42	-1	-1
776	T	510_1799	23714522982676	775	-1	63	5	-1
777	T	514_1776	23714530073769	775	-1	63	-1	-1
778	T	508_1763	23714537110072	775	-1	63	-1	-1
779	T	497_1745	23714544181478	775	-1	63	4	15
780	A	-0.8930359_ 7.3597717_ 6.519409	23714548806530	775				
781	T	486_1726	23714556894342	775	-1	63	-1	0
782	T	474_1702	23714565963457	775	-1	63	5	-1
783	T	464_1681	23714575406999	775	-1	63	-1	-1
784	T	453_1655	23714581758926	775	-1	63	6	-1
785	T	444_1630	23714590520801	775	-1	63	5	-1

Table 1: Excerpt from a file generated by exporting a SQLite database using the Raw Converter and `CSV exporter` (formatted for readability)

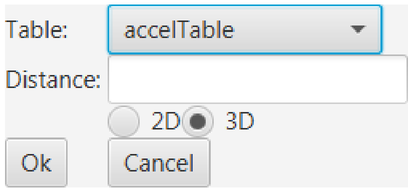


Figure 17: Screen shot taken after loading a database into the application and opening the extended menu of the downsampler

`Type` is *A* the following is accelerometer data and *G* stands for gyroscope data. Both of these are followed by coordinates of the X, Y and Z axis representing the movements of the device and are represented in the X_Y_Z format. The next column is a time stamp in nanoseconds. The last column in use for both accelerometer and gyroscope is the `GestureId` which represents which gesture the row is a part of and refers to the sequence number of the first touch event in the gesture.

T in the `Type` column represents a touch event in a gesture. The difference for touch rows is the coordinate column which here only have X_Y as there is no Z axis on the screen. Time stamps are the same as both gyroscope and accelerometer. `GestureId` differs in that it will be 0 to indicate the start of a new gesture. The `Slot` column represents how many and which fingers are a part of the gesture. The value is zero indexed, but will remain -1 if only one finger is present in gesture. `Pressure` measures how hard the given fingers is pressing while `area` represents how large area the finger is touching is. `Orientation` corresponds to the rotation of the touch area ellipsis, and has the value -1 if undefined or unknown.

Down sampler

The Down sampler objective is removing unnecessary data rows which slow down the analysis of the processed data and/or represent unwanted noise. The way this is done is to calculate the distance between the coordinates between two rows, and, if the distance between is too short, removes the latter row from the table.

This algorithm uses the `getTableNames()` function in the `DbMetaDataWrapper` class which returns all the table names in a `List<String>`. This is put in a `ObservableList<String>` for illustration see Figure 17. There are some minor tweaks to be done there given that it shows all the tables in the database including database-schema, which is really unnecessary considering what the algorithms are doing. The following formula is used to calculate distance between the points:

$$\mathrm{if}(\sqrt{(row_0.x - row_1.x)^2 + (row_0.y - row_1.y)^2 + (row_0.z - row_1.z)^2} < d) : remove\ row_1$$

. Here row_0 is the current row, row_1 is the next row and d is the distance set by the user.

There are some disadvantages to this filtering algorithm which are mainly caused by SQLite not implementing an updatable resultset. This results in that to remove the correct rows from the pipeline database, a separate query has to be ran in the database instead of simply editing the results. Subsequently this leads to worse performance.

An improvement which we could look into is that this algorithm disregards differences in timestamps between rows, meaning that it may filter out more rows than intended. A scenario could be that filtering of touch events is requested, and two subsequent events with a low distance in between are compared even though as their distance in time is great. The long interval between them indicates that this is not a case of noise, but the algorithm will remove the second event anyway.

This algorithm uses the `getTableNames()` function in the `DbMetaDataWrapper` class which returns all the table names in a `List<String>`. This is put in a `ObservableList<String>` for illustration see figure 17. The return value does also list the table which contains the database schema, which may not be of use to the requesting function.

3.3.4 Further Improvements

The current application does not use optimization features, such as running conversions concurrently (when possible) and take advantage of multi-processor systems. If performance problems are encountered this should be implemented.

As of now the utility of the currently implemented algorithms is minimal due to time limitations. Modules that converted the SQLITE database into for example MYSQL, ACCESSDB or ORACLE for other more advanced/different data presentations could make it more useful and make the process of importing the data into for example MATLAB or SPSS easier and more streamlined.

4 Conclusion

4.1 Results

Android has several mechanisms which isolate user installed applications. In these thesis we have tried to intercept events targeted at other applications which our application should have been sandboxed from. These touch events are generated by the touch screen driver, then transformed by the application framework, which in turn identifies the target application and dispatches the event. We have shown that Android's security mechanisms prevent any application that is not integrated into the system image (or uses a modification of the system image) from obtaining touch events at any of these stages.

We have further shown that accessibility services do only grant access to a very limited range of events. These events are stripped of any of the related touch information and do not bypass the event propagation chain. Also we have shown that as of Android `4.0.3` there is no longer any way to capture and simultaneously pass through events using screen overlays.

Furthermore have we demonstrated that injection of events into the event propagation chain is not possible without the INJECT_EVENTS permission or write access to the driver's event buffer. Therefore it is not possible to consume events to so programmatically dispatch them anew.

During development we have firstly created an application for the Android platform which is capable of injecting touch and sensor events. We have here chosen to implement the least intrusive method to intercept touch events by tapping directly into the drivers event buffer and processing the data ourselves. These events are logged into a local database.

Secondly we have developed a desktop application which can open these databases and perform transformations and filtering. It also can export the events to CSV files. We have pnly implemented a limited amount of modules, but the possibilities for developing additional modules are virtually endless. Usage of external applications, libraries and additional algorithms is implementable using interfaces to the existing code base.

4.2 Reflections on Results

Understanding a project as huge as the Android operating system is an enormous task. Even though we only needed to look at some parts of Android, were a mere two months an exceedingly short amount of time to gain a sufficient overview. It is said that it takes a developer around a year before they can make meaningful contributions to the Linux kernel. The code base we had to inspect is several orders of magnitude greater than that. We were surprised over the low amount of literature on the subject and the mostly lacklustre documentation. We imagine that the inner workings of Android are passed down from one generation of developers to the following at the Google headquarters. Alternatively it just takes a horrid amount of time to understand the majority of it.

We found it extremely disheartening when our first attempt at attaining Android's source code failed due to insufficient hard drive space remaining. Motivation took an even greater hit when we realized that the source of the kernel was in another castle. We soon abandoned the idea of doing searches and inquiries by hand, and motivation took another hit when even scripted searches took hours to complete without returning the expected files. We later found out that a lot of files and functions changed names in later Android versions, which is not really helpful if sources refer to the old file or function name.

Tracing the touch events through the system was one of the biggest code comprehension challenges we have ever faced. Just two of the most relevant files in the `inputFlinger` service clock in at just above $11,000$ lines. In the `inputDispatcher.cpp` file alone we had to trace more than 20 lengthy function calls. This whole process took a sizeable chunk of our research time. Seeing one after another of our proposed methods for interception and injection of touch events fail may be a victory for the security of the Android platform, but to us it feels more like a defeat.

A great surprise was the small amount of touch data obtainable using accessibility services. We believe that more detailed access to this data may aid development of accessibility services targeted at other disabilities than only those related to sight. We understand the security concerns of allowing an application to unlimitedly access live touch data. Reading it may reveal sensitive information that was not intended to be shared with outside application such as passwords, and may also be used for surveillance purposes. Write access to the touch events poses an even greater risk, as we explored in section 2.2.1.

Still we would argue that these concerns are outweighed by the possible gains to functionality it provides and may even increase security for users that currently have to root their device to use such functionality. Furthermore are accessibility services never silently allowed but require an active approval additionally to the approval of permissions from the user before any access is granted. This request is made in such a way that it is assured that it is impossible to programmatically impersonate an user. We believe that the warning message which would be displayed would be adequate for the user to make the decision, given that it conveys the possible impact on security.

We would in retrospect have liked to spend more time the implementation phase. The Android and the desktop application have had more functionality and wd could have performed more testing. As it stands the solution is in a usable state, but as we expressed in the previous section there is room for further development.

This project has been an insightful experience and has taught in how the Android operating system is implemented. Our previous knowledge of operating systems and basic comprehension of the workings of LINUX helped us a lot in trying to learn how events are propagated through the `/dev/input/eventX` files. We also did gain a deeper understanding of how LINUX uses said event and device files during this project.

We are pleasantly surprised by the speed in which we were able to write our applications. We did honestly not expect to spend two weeks less on development than estimated. It was fun to see the code base grow at a steady pace and it was a huge boost on morale to finally see some more tangible results of our efforts.

4.3 Future development and research

Additional algorithms could be added to the RAW CONVERTER in order to be more versatile, to better fit the needs of researchers. If many algorithms were to be implemented and included into the application, it would soon fit a wide range of research purposes and become a great utility for processing databases.

We programmed the application in a modular way so that logging additional sensors or adding formats to convert would be easy to implement in the desktop application. This will be useful when the need for additional sensor data arises.

The user interface could be made more intuitive. For example should the two main lists should get descriptions as to what their data represents.

Practical upgrades to the Android application would include adding additional sensors and a capability of recognizing the currently active window for each touch event, this could be used to research how the user's use patterns diverge using different applications. Further could recognition of the on-screen keyboard and logging of the entered data be added also. For privacy and security improvements this would include automatic suspension of logging should a password field or similar be detected.

To remove the need for manual transfer of database files after each logging session, remote logging capability or the automatic transfer of database files to remote servers could be added. This would of course require a risk analysis to aid protection of the potential sensitive nature of the log data.

Experiments on the sensor's reliability and precision should be conducted to evaluate if there are any bottlenecks in the event propagation chain. This would help evaluate if the precision and correctness of the touch data applications receive. Further does it need to established whether our application actually intercepts 100% of the events under all circumstances and that for example no events are dropped if the device is under great processing loads.

We did not have the time to test this application on more than one device, the Motorola Nexus 6. On this phone the application did not seem to have any impact on the overall responsiveness of the device and the user experience. This may not hold true on every device and it is therefore something that should be tested thoroughly. The format that the touch screen driver uses to describes its events may also vary from phone to phone, therefore our application is not guaranteed to be compatible with all other devices.

If the application is modified to run entirely in the background and be able to recognize the user, it may be used as a tool for continuous authentication. The data needed we already collect. The only difference would be how it is processed: If the data is for example input into an ANN which increments and decrements a so called "trust value", the program could evaluate the trustworthiness of an user. If this value reaches below a certain threshold the phone could lock itself. The application could then also wipe the device data, if the application is a device administrator.

According to recent studies most people have quite different behavioural patterns interacting with touch screens especially taking speed of swipe, area of touch and pressure into consideration [35]. This makes the implementation of the aforementioned ANN easier and maybe also decreases the needed sampling rate during normal runtime. This is an interesting topic for further research.

Should methods of acquiring touch data without modification of the system become available, our program could be rewritten to be deployable on any Android device. This would create the possibility for long-term usage experiments, where participants could continue using their private device as usual.

Other research that may be explored could result in a more detailed and complete view over the propagation chain than we were able to attain the relatively short time we had available. Differences between driver implementations and the format they use to describe their events may also be documented, which in turn will aid making applications such as ours compatible to a wider range of hardware.

5 How we worked as a team

5.1 Methods and tools

We decided on meeting on Mondays through Wednesdays from 0815 until around 1600 and set 20 hours a week as the mean workload. To reach the 20 hours each of us had to fill the remaining hours as we saw fit ourselves.

Work was distributed among us by identifying tasks together and putting them on TRELLO, an online scrum development board. Whenever one of us finished a task, the correlating entry on the board was marked as completed and a new task selected from the unfinished ones. Prioritisation was discussed, but not strictly enforced.

Notes of experiments, research progress and so on were kept online in google documents to be concurrently edited and reviewed by all members. Reports, the meeting logs and all source code was placed in a `.git` repository on `bitbucket.org`, which is free of charge to students at Gjøvik University College. Additionally this repository included a simple bug tracker, which we utilised as well. To ease the usage of the repository we used the SOURCETREE software, which is developed and distributed by the same vendor as our repository.

LaTeXsource code was generated and edited using TeXSTUDIO and JABREF. The document template was provided by GUC and we used it without modifications.

The Android application was created using ANDROID STUDIO as the IDE and the desktop application using INTELLIJ and ECLIPSE.

5.2 Project Progress

The original project plan started with a 3-4 weeks research period. Afterwards there was a development phase where we expected to use Scrum over three sprints of two weeks each. The release schedule was one after each sprint to acquire feedback from our employers to then prioritize and possibly implement eventual improvements within the next sprint. After this we planned to use 2 weeks exclusively for finishing the report and thesis writing. Afterwards there were 10 days for preparation of the presentation. In this scenario we had a months worth a buffer to counter any unexpected events that may have delayed our progress (see Appendix E).

Before starting the research phase we had to specify what elements of the Android source code we were looking for, and we had to create a basic outline of the application based on the description given to us by our employer. It took us two weeks of work on the specifications to agree with the employers on the created outline.

As we progressed into the research phase it became quite clear that this phase would last a lot longer than anticipated. After about three weeks of looking through developer documentation we concluded that the project goal of acquiring touch data is not possible with the given limitations. But it is difficult to prove the non-existence of *any* possibility, therefore we needed to add some weeks to be able to make an argument strong enough for our claim.

The research phase prolonged itself until 08.04.2015(see Appendix: D) when we finally felt that our collected evidence is strong enough to conclude that it is impossible to get the gestures without using root privileges. We then began the process of rooting the phone and writing our application with root privileges. At this point we had already gone beyond the planned finish of the project and were utilizing the previously allocated buffer.

Because of the relative low number of group members and our good experiences working together previously, we moved away from pure Scrum and took a more eXTREME PROGRAMMING inspired approach with one weeks sprints and continuously dynamically prioritized tasks. This proved to be a quite effective way of implementing the system and we finished the implantation within four sprints, instead of the 6 weeks originally estimated. All of the promised features were implemented and we even found some time to make some minor additions, for example the down sampling algorithm in the Raw Converter.

Bibliography

[1] Elenkov, N. 2014. *Android Security Internals: An In-Depth Guide to Android's Security Architecture*. No Starch Press, San Francisco, CA, USA, 1st edition.

[2] NISlab. Nislab webpage. https://www.nislab.no/.

[3] Gjøvik University College. April 2015. Gjøvik university college. http://english.hig.no/about.

[4] NISlab. Biometrics laboratory description on nislab's webpage. http://www.nislab.no/biometrics_lab.

[5] Robin Stenvi, Magnus Øverbø, L. J. 2013. Belt bachelor thesis. http://brage.bibsys.no/xmlui/handle/11250/143069.

[6] Gjøvik University College. 2012. Gjøvik university college - bachelor of science in information security. http://english.hig.no/course_catalogue/student_handbook/2012_2013/studies_20012_2013/faculty_of_computer_science_and_media_technology/bachelor_of_science_in_information_security.

[7] Gjøvik University College. 2012. Gjøvik university college - fundamental programming. http://english.hig.no/course_catalogue/student_handbook/2012_2013/courses/avdeling_for_informatikk_og_medieteknikk/imt1031_dl_fundamental_programming.

[8] Gjøvik University College. 2012. Gjøvik university college - object-oriented programming. http://english.hig.no/course_catalogue/student_handbook/2012_2013/courses/avdeling_for_informatikk_og_medieteknikk/imt1082_object_oriented_programming.

[9] Gjøvik University College. 2013. Gjøvik university college - software engineering. http://english.hig.no/course_catalogue/student_handbook/2013_2014/courses/avdeling_for_informatikk_og_medieteknikk/imt2243_software_engineering.

[10] Gjøvik University College. 2013. Gjøvik university college - operating systems. http://english.hig.no/content/view/full/27966/language/eng-US.

[11] Gjøvik University College. 2013. Gjøvik university college - data modelling and database systems. http://english.hig.no/content/view/full/28102/language/eng-US.

[12] Gjøvik University College. 2014. Gjøvik university college - software development. http://english.hig.no/content/view/full/31045/language/eng-US.

[13] Google Inc. January 2015. Android 5.0 compatibility definition. http://static.googleusercontent.com/media/source.android.com/de//compatibility/android-cdd.pdf.

[14] The Linux Foundation. March 2012. Linux kernel development: How fast it is going, who is doing it, what they are doing, and who is sponsoring it. http://go.linuxfoundation.org/who-writes-linux-2012.

[15] Android Open Source Project. Feburary 2015. Security. http://source.android.com/devices/tech/security/index.html.

[16] Android Open Source Project. Feburary 2015. Security-enhanced linux in android. http://source.android.com/devices/tech/security/index.html.

[17] Android Open Source Project. Eventhub.cpp. Location in the Android 5.0.1 branch: frameworks/services/native/inputflinger.

[18] Android Open Source Project. Inputdispatcher.cpp. Location in the Android 5.0.1 branch: frameworks/services/native/inputflinger.

[19] Android Open Source Project. Inputwindow.cpp. Location in the Android 5.0.1 branch: frameworks/services/native/inputflinger.

[20] Chen, B. March 2014. Android ui internal : Pipeline of view's touch event handling. http://pierrchen.blogspot.com/2014/03/pipeline-of-android-touch-event-handling.html.

[21] Lookout. December 2010. Android touch-event hijacking. https://blog.lookout.com/blog/2010/12/09/android-touch-event-hijacking/.

[22] Google Inc. 2015. Android documentation on windowmanager.layoutparams. https://developer.android.com/reference/android/view/WindowManager.LayoutParams.html.

[23] Google Inc. 2015. Android documentation on motionevent. https://developer.android.com/reference/android/view/MotionEvent.html.

[24] Lu, S. February 12. Stackoverflow response. http://stackoverflow.com/a/9462091/4272283.

[25] Google Inc. 2015. Android 4.0.3 source code for phonewindowmanager.java. https://github.com/android/platform_frameworks_base/blob/master/policy/src/com/android/internal/policy/impl/PhoneWindowManager.java.

[26] Google Inc. 2015. Android documentation on instrumentation. https://developer.android.com/reference/android/app/Instrumentation.html.

[27] Motisan, R. April 2012. Programmatically injecting events on android. http://www.pocketmagic.net/injecting-events-programatically-on-android/.

[28] Google Inc. 2015. Android documentation on accessibilityservice. https://developer.android.com/reference/android/accessibilityservice/AccessibilityService.html.

[29] Google Inc. 2015. Android documentation on accessibilityevent. https://developer.android.com/reference/android/view/accessibility/AccessibilityEvent.html.

[30] TeamWIn. About. URL: https://twrp.me/about/.

[31] Team, T. May 2015. Javadoc. http://hovedprosjekter.hig.no/v2015/imt/is/theia/javadoc.html.

[32] Levin, J. 2014. *Android Internals - Volume I: A Confectioner's Cookbook*. Jonathan Levin. URL: https://books.google.no/books?id=onhDnwEACAAJ.

[33] Henrik Rydberg, T. L. F. Multi-touch protocol. https://www.kernel.org/doc/Documentation/input/multi-touch-protocol.txt.

[34] Lu, S. February 2012. Stackoverflow response to question about overlays. http://stackoverflow.com/a/9462091/4272283.

[35] Frank, M., Biedert, R., Ma, E., Martinovic, I., & Song, D. Jan 2013. Touchalytics: On the applicability of touchscreen input as a behavioral biometric for continuous authentication. *Information Forensics and Security, IEEE Transactions on*, 8(1), 136–148. doi:10.1109/TIFS.2012.2225048.

A App Specifications (excerpt)

2 Use case Specification

Use case Start Gathering

Initiator User

Purpose Start the process of gathering user interaction data. The information gathering process will run until stopped by the user.

Pre-conditions No gathering session is currently running.

Post-conditions A process should be running on the device that continuously gathers user interaction data. A file to log data is created.

Description The user initiates the use case by using the GUI to start gathering user interaction data. This use case initiates the interaction gathering use case.

Event Flow

Actor Action	*System Response*
1. The user starts the interaction gathering using the GUI.	
	2. The device creates a file locally in a predefined format and a name conforming to the time of gathering.
	3. The device starts the interaction gathering process.

Alternate Event Flow

1.1 The interaction gathering process is already running.

alt. The user is prompted whether the current session should be stopped.

Use case Stop Gathering

Initiator User

Purpose Stop the process of gathering user interaction data.

Pre-conditions A gathering session must be running.

Post-conditions The gathering session must be stopped and a current log file stored on the device.

Description The user initiates the use case by using the GUI to stop gathering user interaction data.

Event Flow

Actor Action	*System Response*
1. The user stops the interaction gathering using the GUI.	
	2. The device stops gathering user interaction data.
	3. The device ensures that all gathered events are written to the logfile.

Use case Interaction Gathering

Initiator The "Start Gathering" use case.

Purpose Log interaction data of a session to file.

Pre-conditions The "Start Gathering" use case must be started.

Post-conditions A file should be present on the device containing interaction data of the current session.

Description This use case is initiated by the "Start Gathering" use case. It will continuously log the interaction data locally on the device in a predefined format until stopped by the "Stop Gathering" use case.

Event Flow

Actor Action	*System Response*
1. The user interacts with the device.	
	2. The device gathers the input and appends it to the log file.
	3. Loop back to 1.

Alternate Event Flow

2.1 There is no space available on the device to log data.

alt. A warning is displayed and the “Stop gathering” use case invoked.

3.1 The user stops the interaction gathering.

alt. Invoke “Stop gathering” use case.

3.2 The user pauses the interaction gathering.

alt. Invoke “Pause gathering” use case.

Use case Pause Gathering

Initiator User

Purpose Pause the interaction gathering.

Pre-conditions A gathering session must be running.

Post-conditions The gathering session should be paused.

Description This use case is initiated by the user. It is intended to enable the user to pause the gathering session when needed, e.g. when entering sensitive information. The session will remain paused until started by the user.

Event Flow

Actor Action	*System Response*
1. The user hits the pause button.	
	2. The gathering session is temporarily stopped.
3. The user hits the pause button.	
	4. The gathering session is resumed.

Alternate Event Flow

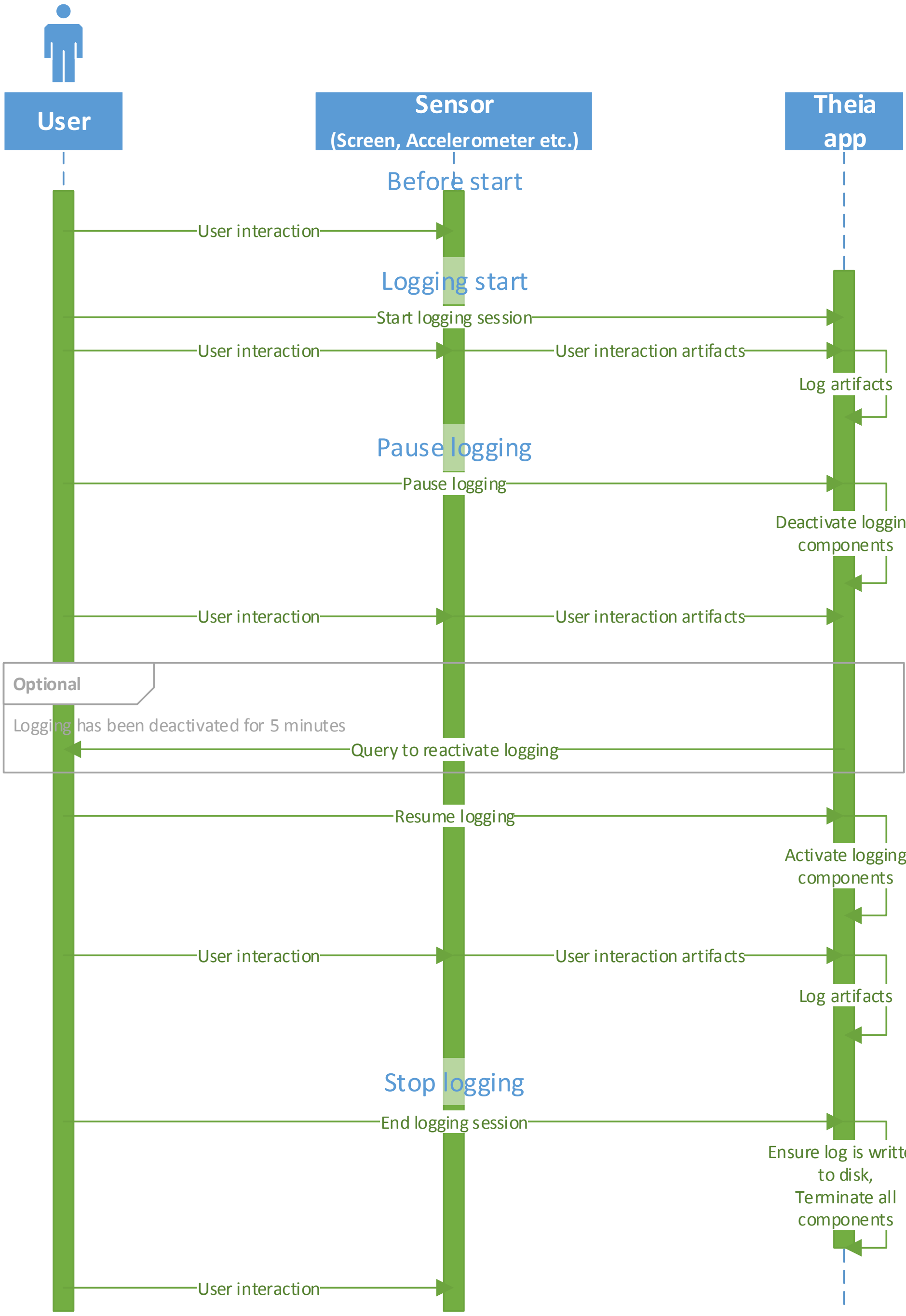
3.1 The pause has lasted 5 minutes.

alt. A reminder is displayed, prompting the user to resume.

3.2 The user hits the stop button.

alt. Invoke “Stop gathering” use case.

Sequence diagram



3 Operational Requirements

Security

- Passwords should be censored in the log file.

Reliability

- The application must be able to log 100% of the targeted interaction data. Failing to do so would result in inaccurate data which would be inadequate for further research.

Efficiency

- Interaction data should be logged in such a way that it does not affect the performance of the device to a degree noticeable by the user.
- The application should be able to log screen interaction at a minimum rate of 1 coordinate every $16ms$.

Usability

- The application must run in the background and should not visually interfere with the user experience.
- The user should easily be able to pause the interaction gathering when needed.
- The user should easily be able to infer whether a gathering session is currently running.

4 Domain Model

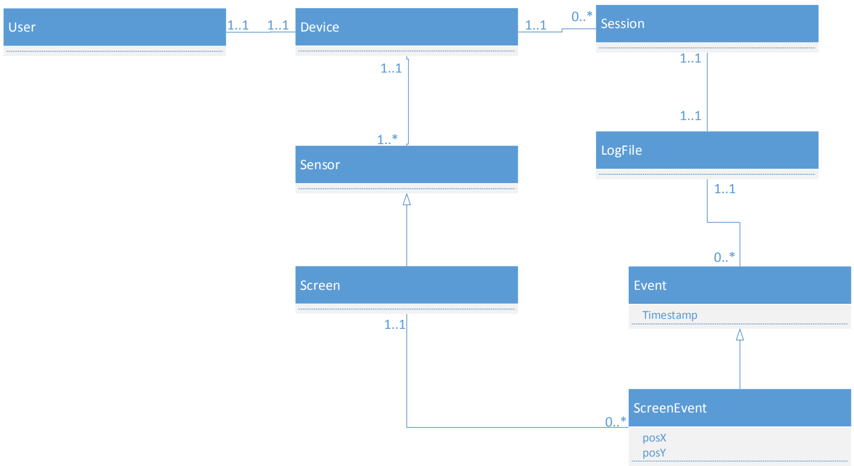


Figure 1: the Domain model of the application

Since this is an application we are not interested in other than the device in question and therefore we do not take into account that a user can have several devices.

As seen in the figure each device have 1 or more sensors, but as the application preliminary only has the task of logging the screen movements this is the only sensor specified. The screen generate screen events which is a subclass of Event which is logged in a logfile.

Each logfile belongs to one and only one session which starts as the screen is turned on and stops when the screen is turned off.

B Group contract

Due to all our group members being Norwegian, the following contract is written in Norwegian. A translation is available upon request.

Gruppekontrakt

Fyaous

20. januar 2015

Denne kontrakten gjelder samarbeidet innen bacheloroppgaven. Undertegnende er medlemmer av gruppen "Fyaous". Undertegnende forplikter seg til å følge disse reglene, som har blitt vedtatt i fellesskapet:

1. Målsettningen
   (a) Gruppementaliteten skal følge mottoet: "*En for alle, og alle for en!*".
   (b) Gruppen har satt seg som mål å oppnå en slik kvalitet på prosjektet at det tilsvarer karakteren "*A*" og er en verdig kandidat til Eureka-prisen.
2. Medlemmene kan ha disse rollene/vervene:
   (a) *Gruppeleder*: Har lederansvar for gruppen, som også omfatter det organisatoriske.
   (b) *Oppmøteansvarlig*: Har funksjoner som er relatert til oppmøte til avtalte møter o.l. som omfatter, men ikke er begrenset til:
      i. Ansvar for at alle gruppemedlemmene møter til avtalt tid
      ii. Oprettelse av kontakt med uteblivende medlemmer for klarering og sikring at gruppen kan utføre arbeidet på en effektiv måte
      iii. Rett til muntlig advarsel ved hendelser relatert til oppmøte
   (c) *Vara*: Ved uteblivelse til et gruppemedlem med en spesifikk rolle tar den evt. vara over alle funksjonene til dette medlemmet inntil det ikke er lengre uteblivende. Det kan være flere vara, men bare en per rolle.
   (d) *Møteleder*: Fungerer som ordstyrer under møter, samt at møtelederen passer på gruppenes tidsbruk og har et generelt lederansvar. Dette vervet går på rundgang mellom alle medlemmene.
   (e) *Referent*: Har ansvaret for å lage referatet til en møte, slik at det kan bli brukt til å peke til avgjørelser og annet på et senere tidspunkt. Dette vervet går på rundgang mellom alle medlemmene.
3. Tekniske hjelpemidler:

(a) Følgende programvarer og/eller tjenester er vedtatt brukt i prosjektarbeidet:

   i. *Git*

      A. Git skal brukes til lagring av filer relatert til prosjektarbeidet, spesielt dersom filen(e) skal brukes av flere gruppemedlemmer.

      B. Alle gruppemedlemmene skal ha tilgang på gruppenes innhold lagret på Git.

      C. Alle gruppemedlemmene forplikter seg til å sørge for at det ligger den mest aktuelle versjonen av en fil på Git (gruppenes fellesmappe) dersom et annet medlem kan trenge det. Ansvaret for det har den enkelte som jobber med en fil.

   ii. *Powerpoint*

      A. Powerpoint skal brukes til å lage gruppenes presentasjoner

      B. Dersom mulig benyttes gruppens powerpoint-mal.

   iii. LaTeX (LyX)

      A. Oppgaver, innleveringer o.l. skal skrives og genereres av LaTeX, der det er alternativt mulig å benytte seg av LyX.

      B. Hig's LaTeX-mal for bachelor oppgaver skal benyttes.

   iv. *Gantskjema* skal benyttes som fremdriftsplan.

   v. Utviklingen av programvare skal være testbasert.

(b) Ved bruk av digitale hjelpemidler hersker det krav om sikkerhetskopiering:

   i. Gruppemedlemmene har ansvar for dokumenter de jobber med, med spesiell vekt på ansvaret for at dokumentenes integritet og tilgjengelighet er garantert.

   ii. Ved hendelse som skader integriteten og tilgjengeligheten er medlemmet som hadde ansvaret forpliktet til å rette opp feilen raskest mulig. En slik rettelse skal ikke gå utover det andre arbeidet, noe som innebærer at en slik rettelse ikke regnes som en tildelt oppgave. Er det umulig å gjennomføre rettelsen skal gruppelederen straks kontakteres slik at gruppen kan opprettholde størst mulig effektivitet.

4. Oppmøte:

   (a) Beskjed om møte skal være gitt minst 24 timer før.

   (b) Alle gruppemedlemmene møter til avtalt tid.

      i. Skulle en bli forsinket eller utebli er en forpliktet til å melde til oppmøteansvarlig om forholdene så fort som det er kjent at slikt inntreffer. Uteblivelsen/forsinkelsen skal loggføres.

A. Ved forsinket oppmøte uten gyldig grunn forpliktes det til innkjøp og distribusjon av snacks til de andre gruppemedlemmene ved et påfølgende møte, eller alternativt til et annet tidspunkt som avtales med hele gruppen.

B. Gjentatt forsinkelse og/eller uteblivelse kan medføre straff:

- *Muntlig advarsel* dersom oppmøte/forsinkelse ikke skjer med god nok grunn, det virker som det er mangelfulle holdninger og det har skjedd minst to ganger. Dersom fullstendig uteblivelse skjer uten grunn kan det også gis muntlig advarsel. (Utdeles av oppmøteansvarlig)
- *Skriftlig advarsel* dersom gruppen vedtar at medlemmets gjentatte uteblivelse/forsinkelse går utover gruppens produktivitet i betydelig grad.
- *Fradragelse av verv* dersom gruppen vedtar at medlemmets gjentatte uteblivelse/forsinkelse går utover vervets funksjon.

ii. Ved uteblivelse med gyldig grunn skal, om nødvendig, oppgavetildelingen tilpasses for å opprettholde gruppenes produktivitet.

iii. Ved uteblivelse/forsinkelse med gyldig grunn skal også grunnen loggføres.

5. Kommunikasjon mellom gruppemedlemmene:

(a) Informasjon til alle gruppemedlemmene skal gis på møter, epost og/eller telefon (sms)

(b) Gruppemedlemmene forplikter seg til å sjekke eposten sin minst én gang daglig.

6. Arbeidsbetingelser:

(a) Alle gruppemedlemmene forventes å gi prosjektet en viss prioritet over annet.

i. Fellesmøte er mandag til onsdag 08.15 til 16.00, med mindre annet er avtalt.

(b) Det skal herske effektiv jobbing når gruppen er samlet, videre hersker det et generelt krav om deltagelse.

(c) Frister skal settes slik at det er mulig å gjennomføre oppgaven(e) i en tilstrekkelig kvalitet, og det er den enkeltes ansvar å gjennomføre tildelte oppgaver innen fristen med en tilstrekkelig kvalitet.

(d) Det skal herske klarhet om oppgavetildelingen til en hver tid, og gruppeansvarlig skal ha en oversikt om hvem som holder på med hva.

(e) Alle skal tildeles oppgaver der arbeidsmengden skal være så likt som mulig, i den grad at gruppen kan jobbe effektivt.

   (f) Medlemmene loggfører eget, selvstendigt arbeid samt arbeidstid i et eget dokument som lagres på Git.
   (g) Ting som trengs å tas opp for hele gruppen skal ikke tas opp i pausen.
   (h) Dersom det kreves av arbeidsgiver behandles informasjon som gruppen behandler konfidensielt.
   (i) Det forventes at alle medlemmene følger vanlig folkeskikk til enhver tid.

7. Rutiner ved avgjørelser:
   (a) Avgjørelser tas dersom det hersker uenighet om noe relatert til gruppearbeidet og/eller det trenges klarhet om noe.
   (b) Avgjørelser skal tas av alle gruppemedlemmene som denne avgjørelsen betreffer.
   (c) Avgjørelser utsettes ikke når noen ikke er til stedet.
      - Ved kritiske avgjørelser kontakterer gruppelederen de uteblivende gruppemedlemmene, som da får muligheten til å gi sin stemme via gruppelederen. Gruppelederen forpliktes til å ikke forfalske stemmen. Er det umulig å få kontakt innen rimelig tid tas avgjørelsen uten å ta hensyn til medlemmene som ikke kunne kontakteres.

8. Beskjeder og advarsler ved brudd:
   (a) Advarsler som utdeles skal loggføres på gruppens Git, men ikke beskjeder.
   (b) Muntlige beskjeder og advarsler
      i. Det skal gis en muntlig advarsel dersom et medlem hindrer gruppens arbeid, leverer ikke arbeid til fristen eller leverer meget dårlig arbeid som ikke tilsier den loggførte tidsbruken.
      ii. Ved forsinkelse på mer enn 10 minutter som det ikke har blitt varslet om, gis det en muntlig beskjed som skal loggføres.
   (c) Skriftlige beskjeder og advarsler
      i. Det skal gis en skriftlig advarsel når den 3. muntlige advarselen har blitt gitt.
   (d) Nominasjon til eksklusjon av medlemmer:
      i. Eksklusjon skal bare skje dersom medlemmet hindrer gruppen i å oppnå en rimelig effektivitet, og det skal følges retningslinjene i studiets emnebeskrivelse (tilgjengelig:
).
      ii. Nominasjon til eksklusjon skal forangåes en gruppeavgjørelse der alle andre medlemmene er involvert, og vurderingen innledes når den 5. advarselen har blitt gitt.

9. Disse reglene anerkjennes av alle gruppemedlemmene og skal hverken endres, legges til eller trekkes fra uten en bekreftende flertallsavgjørelse der alle medlemmene har blitt inkludert. Videre er de ikke uttømmende og fraskriver ikke ansvaret om å velge en hensiktsmessig oppførsel og arbeidsmetode, og ha hensiktsmessige holdninger.

Signatur gruppemedlemmene:

Brage Celius

Jannis Schaefer

Eirik V. Solberg

C Source code examples

C.1 AccessibilityService

```
package com.theia.servicetest;

import android.accessibilityservice.AccessibilityService;
import android.util.Log;
import android.view.KeyEvent;
import android.view.MotionEvent;
import android.view.accessibility.AccessibilityEvent;

import java.util.ArrayList;

/**
 * Created by Brage on 11-Mar-15.
 */
public class accessibilityService extends AccessibilityService {



    @Override
    public void onCreate() {
        Log.v("THEIA", "Service Created");
    }

    @Override
    protected boolean onGesture(int gestureId) {
        Log.v("THEIA", String.format("onGesture: [type] %s", gIdToString(
             gestureId)));
        return true;
    }

    @Override
    protected boolean onKeyEvent(KeyEvent event) {
        Log.v("THEIA", String.format("onKeyEvent: [characters] %s [keyCode] %s",
              event.getCharacters(), event.getKeyCode()));
        return false;
    }




    @Override
    public void onAccessibilityEvent(AccessibilityEvent event) {

       //Evaluate source
       Log.v("THEIA", String.format(
               "onAccessibilityEvent: [type] %s [class] %s [package] %s [time] %s
                    ",
               idToText(event), event.getClassName(), event.getPackageName(),
               event.getEventTime()));

    }

    @Override
    public void onInterrupt() {
        Log.v("THEIA", "INTERRUPTED");
    }

    @Override
```

```
protected void onServiceConnected() {
    super.onServiceConnected();
    Log.v("THEIA", "AccessibilityService allowed");
}

/**
 * Converts the ID's returned by AccessibilityEvent.getEventType() into
     strings
 * @author Brage Celius
 * @param event
 * @return
 */
private String idToText(AccessibilityEvent event) {
    switch (event.getEventType()) {
        case AccessibilityEvent.TYPE_TOUCH_EXPLORATION_GESTURE_START:
            return "TYPE_TOUCH_EXPLORATION_GESTURE_START";
        case AccessibilityEvent.TYPE_TOUCH_EXPLORATION_GESTURE_END:
            return "TYPE_TOUCH_EXPLORATION_GESTURE_END";
        case AccessibilityEvent.TYPE_TOUCH_INTERACTION_START:
            return "TYPE_TOUCH_INTERACTION_START";
        case AccessibilityEvent.TYPE_TOUCH_INTERACTION_END:
            return "TYPE_TOUCH_INTERACTION_END";
        case AccessibilityEvent.TYPE_GESTURE_DETECTION_START:
            return "TYPE_GESTURE_DETECTION_START";
        case AccessibilityEvent.TYPE_GESTURE_DETECTION_END:
            return "TYPE_GESTURE_DETECTION_END";
        case AccessibilityEvent.TYPE_VIEW_HOVER_ENTER:
            return "TYPE_VIEW_HOVER_ENTER";
        case AccessibilityEvent.TYPE_VIEW_HOVER_EXIT:
            return "TYPE_VIEW_HOVER_EXIT";
        case AccessibilityEvent.TYPE_VIEW_SCROLLED:
            return "TYPE_VIEW_SCROLLED";
        case AccessibilityEvent.TYPE_VIEW_CLICKED:
            return "TYPE_VIEW_CLICKED";
        case AccessibilityEvent.TYPE_VIEW_LONG_CLICKED:
            return "TYPE_VIEW_LONG_CLICKED";
        case AccessibilityEvent.TYPE_VIEW_FOCUSED:
            return "TYPE_VIEW_FOCUSED";
        case AccessibilityEvent.TYPE_VIEW_SELECTED:
            return "TYPE_VIEW_SELECTED";
        case AccessibilityEvent.TYPE_VIEW_ACCESSIBILITY_FOCUSED:
            return "TYPE_VIEW_ACCESSIBILITY_FOCUSED";
        case AccessibilityEvent.TYPE_VIEW_ACCESSIBILITY_FOCUS_CLEARED:
            return "TYPE_VIEW_ACCESSIBILITY_FOCUS_CLEARED";
        case AccessibilityEvent.TYPE_WINDOW_STATE_CHANGED:
            return "TYPE_WINDOW_STATE_CHANGED";
        case AccessibilityEvent.TYPE_NOTIFICATION_STATE_CHANGED:
            return "TYPE_NOTIFICATION_STATE_CHANGED";
        case AccessibilityEvent.TYPE_ANNOUNCEMENT:
            return "TYPE_ANNOUNCEMENT";
        case AccessibilityEvent.TYPE_WINDOWS_CHANGED:
            return "TYPE_WINDOWS_CHANGED";
        case AccessibilityEvent.TYPE_WINDOW_CONTENT_CHANGED:
            return "TYPE_WINDOW_CONTENT_CHANGED";
        case AccessibilityEvent.TYPE_VIEW_TEXT_CHANGED:
            return "TYPE_VIEW_TEXT_CHANGED";
        case AccessibilityEvent.TYPE_VIEW_TEXT_SELECTION_CHANGED:
            return "TYPE_VIEW_TEXT_SELECTION_CHANGED";
        case AccessibilityEvent.
            TYPE_VIEW_TEXT_TRAVERSED_AT_MOVEMENT_GRANULARITY:
            return "TYPE_VIEW_TEXT_TRAVERSED_AT_MOVEMENT_GRANULARITY";
    }
    return "Unknown";
}


private String gIdToString(int gID) {
```

```
        switch(gID) {
            case 1: return "GESTURE_SWIPE_UP";
            case 2: return "GESTURE_SWIPE_DOWN";
            case 3: return "GESTURE_SWIPE_LEFT";
            case 4: return "GESTURE_SWIPE_RIGHT";
            case 5: return "GESTURE_SWIPE_LEFT_AND_RIGHT";
            case 6: return "GESTURE_SWIPE_RIGHT_AND_LEFT";
            case 7: return "GESTURE_SWIPE_UP_AND_DOWN";
            case 8: return "GESTURE_SWIPE_DOWN_AND_UP";
            case 9: return "GESTURE_SWIPE_LEFT_AND_UP";
            case 10: return "GESTURE_SWIPE_LEFT_AND_DOWN";
            case 11: return "GESTURE_SWIPE_RIGHT_AND_UP";
            case 12: return "GESTURE_SWIPE_RIGHT_AND_DOWN";
            case 13: return "GESTURE_SWIPE_UP_AND_LEFT";
            case 14: return "GESTURE_SWIPE_UP_AND_RIGHT";
            case 15: return "GESTURE_SWIPE_DOWN_AND_LEFT";
            case 16: return "GESTURE_SWIPE_DOWN_AND_RIGHT";
        }
        return "UNKNOWN";
    }

}
```

Configuration file

```
<?xml version="1.0" encoding="utf-8"?>
<accessibility-service xmlns:android="http://schemas.android.com/apk/res/android"
 android:description="@string/accessibility_service_description"
 android:packageNames="@null"
 android:accessibilityEventTypes="typeAllMask"
 android:accessibilityFlags="flagRequestTouchExplorationMode|flagIncludeNotImportantViews"
 android:accessibilityFeedbackType="feedbackAllMask"
 android:notificationTimeout="100"
 android:canRetrieveWindowContent="true"
 android:canRequestTouchExplorationMode="true"
 android:canRequestFilterKeyEvents="true"
 android:settingsActivity="com.theia.servicetest.settingsActivity"
/>
```

C.2 Other source code

Complete source code and other experiments are available upon request.

D Meeting log

Meetings

Time		Attendees	Agenda	Summary	
Date	**Time**			**Log**	**Descisions**
2015-01-13	14.00 - 1500	Jannis, Brage, Eirik, Soumik, Mariusz	Emp-15-001: Who is our Advisor Emp-15-002: Project name Emp-15-003: Project Agreement Emp-15-004: Meeting Schedule Emp-15-005: Storage of Data and source code Emp-15-006: Software requirements Emp-15-007: Eventual	Soumik & Mariusz have agreed to be our supervisors The name "Theia" has been approved. The agreement form needs to be filled out and signed by us, then handed in to Soumik Meetings with employers will be bi-weekly, Soumik makes a proposal when he has contacted Patrick. Some members of our group are unavailable on thursdays 09-12 and fridays 09-14. Meetings with Mariusz will probably be on mondays. Storage on Bitbucket has been approved. - No rooting/modification of android operating system - Filetype: plaintext, csv (better not XML), name: DATE-Timeoflogstart - The program needs to create file on stop - UI: mostly the program runs in the background, possible to chose type, see status, stop/start - Capture: Touchscreen. + swipes: XY+timestamp(+pressure if possible) + High sampling rate (polling as often as possible, 16ms in example) -> good performance is required Questions for research and the bachelors thesis which we should answer: - First some research how others are intercepting data - How do those who create a translucent app pass events down to intended target? - Permissions in this context? (Which does an app have and which can be gained, relevant changes in api-versions) - How to substitute windowmanager? - Why can't you root the phone? Normally you'd just root and provide a phone to research participants	
19.01.2015	09:00 – 14:45	Jannis, Brage	1. Create meeting log 2. Plan future work 3. Continue work on the project plan	1. Created meeting log 2. Project plan has to be finished, afterwards we need to start with research 3. Work on the homepage (Brage) 3. Widen overview over project requirements, planning (Jannis)	- Group leader: Jannis
20.01.2015	08:30 - 15.00 (Eirik absent 10.15 - 12.30)	Jannis, Brage, Eirik	1. Prosject agreement 2. Group contract 3. Meeting shedule with employer (Emp) and supervisors (Sup) 4. Continue projectplan	1. Eirik got the project agreement and delivers it to Soumik tomorrow (if possible) 2. We reviewed the group contract, readied it for asking for feedback & evt signing 3. Send e-mail to Soumik and Patrick 4. Work on the homepage (Brage) 4. Preparing agenda for next Emp- and Sup-meeting (Jannis)	- Tommorows meeting starts first at 09:00 - Section 8 and discussion of section 4 postoned to after have gotten feedback from Mariusz
21.01.2015	09:00 - ??	Jannis, Brage, Eirik	1. Levere Prosjektavtale 2. Avtale møtetidspunkter med oppdragsgiver / veileder 3. Fortsette på prosjektplanen	1. Prosjektavtalen er signert av oppdragsgiver, skal leveres til Hilde Bakke (A228) 2. Avtale med oppdragsgiver ferdigstilt, sendt epost til Mariusz	- Møtetidspunkter oppdragsgiver: Annenhver tirsdag 12:00, første 27.01.2015
2015-01-25	15:00 - 19:00	Eirik	Work on the project plan		
2015-01-26	08:15 - 15:00	Jannis, Brage, Eirik	1. Continue work on project plan	1. Project plan touch-up (Jannis) 1. Start project kravspec (Brage) 1. Research & comparision of bug tracking tools (Eirik)	1. Decided to take 20 min breaks every 2 hours
2015-01-27	08:15- 1700 (Eirik absent 08:15 - 12:00)	Jannis, Brage, Eirik	1. Prepare for emp-meeting 2. Continue Kravspec 3. Meeting (emp) 4. Risk analysis	1. Prepared 2. Preparations did take too long for us to do this before the meeting 3. See under 4. finished assessment, still have to write about mitigation.	- Mariusz will notify us of a meeting slot for deciding the meeting shedule onwards
2015-01-27	12.00 - 13.00	Jannis, Brage, Eirik, Soumik, Patrick	Emp-15-008: Access to our trello board Emp-15-009: Licensing of project, eventual disclosement agreement Emp-15-010: Copyright of work/application Emp-15-011: Review of project plan Emp-15-012: "Loggbok" (work and meeting logging) Emp-15-013: Project name "acronym" proposal	Employers: Not necessary. Private repository, code licensed research only? See above Report should be written as we og along (which was our intention) Alright Archiver (grabber), team for our developement team	

		Emp-15-014: Eventual	Ask Magnus Øverbø (BeLT) about how they did bugtracking, solution where emp sends a mail which we put in issue tracker on bitbucket may be adequate Pausebutton: Reminder to unpause after 5? Min BeLT choices: Password fields, mouse movement storage reducing technique… Timing issue: Streaming (file output) should not affect capture timing precision Do more than asked for --> boosts grade	
2015-01-28 0815 - 15:00	Jannis, Brage, Eirik	1. Finish risiks and deliver projectplan 2. Continue kravspek	1. Finished and delivered project plan.	- Plan going forwards: 1. Kravspec of application focusing on application and interface between modules 2. Research phase - Disposition of thesis: 1. Section/chapter on following user input from the interrupt that is generated until passed on to the target application. 2. Section/chapter on identifying possible hooks and reason for why those are valid possiblilties or not with regards to timing issues.
2015-02-02 0845 - 1430	Jannis, Brage, Eirik	1. Meeting preparation for Sup-15-001 - 007 2. Continue kravspec	1. Updated agenda 2. Wrote about choice of bugtracking tool (Eirik) 2. Continued kravspek (Jannis, Brage, Eirik)	- Tardiness will from now on be logged
2015-02-03 0815 - 1600	Jannis, Brage, Eirik			
2015-02-03 14.30 – 15:05	Jannis, Brage, Eirik, Mariusz	Sup-15-001: Review of groupcontract, discussion of section 8 (rules for exclusion of members)	Sort things internally, if it doesnt work, go to supervisor, then go to responsible for course, if nothing works a person might get a different grade . (x warnings and then supervisor).	
		Sup-15-002: Project plan Sup-15-003: "Loggbok" (work and meeting logging)	OK	
		Sup-15-004: Access to our trello board and repository for supervisors	nowostawski@gmail.com, bitbucket= hig mail.	
		Sup-15-005: Discussion of how we plan to set up the thesis, ruff disposition	How events works and are handled on android...-> deeper, what can we do? (Basically like we had intended)	
		Sup-15-006: Android programming: Java vs native	Java will suffice for the project, using native would slow us down considerably and if we hit performance issues we rewrite that bit only. Mariusz has done tests on this. Use proper memory management and thread management and it should be alright. Profiling at the end to check where performance is slow	- Use java for the project, if performance enchancements are needed we can rewrite only the bits required.
		Sup-15-007: Eventual	- Create Google drive for resources, add mariusz with access so we can share information. - Mariusz suggests wiki while editing and latex when finished. - Several modes for how much data you collect? - Estimate how much space log files will take, and how long time to fill up space etc. and include in report. - Evaluate file types, store as binary on device and convert to filetypes on upload?(to save space on device) - Mariusz suggests a rooted and an unrooted phone, tell mariusz when we have decided on the phone we need from patrick/soumik. - Mariusz suggests doing goolge developer examplesto learn, he give us some resources either by email or in our repository/drive. See vogella(person) for examples and explanations to learn. See background tasks and asynchronous / notification handling especially. - Do specification/architecture first. As many resources as possible will make the thesis better. - Handle timestamps for different timezones. (Store gmt+offset) Relative to usage and absolute time of start/tracking. Start of gathering not so important, time of events relative to start time important. - Use Issue tracker on bitbucket and assign Mariusz to issues and he will check it out.	- Create google drive to share resources and add mariusz.
2015-02-04 0815 - 1600	Jannis, Brage, Eirik	1. Forberedelse til at Jannis blir borte neste uke	- Møterom skal bookes på starten av hvert møte for 2 uker frem i tid og føres inn i dette dokumentet slik at det er oversiktlig og greit å vite hvor vi skal være.	
2015-02-09 0800 – 1600 A269				
2015-02-10 0800 – 1200 A269	Eirik, Brage	1. Prepare for emp- og sup-meeting.		
2015-02-10 1200 – 1245	Patrick, Soumik, Erik, Brage	Emp-15-015: Timing issues Emp-15-016: Progress Emp-15-017: Contact information on webpage	Will depend on hardware/sampling rate Put email on webpage for both soumik and patrick(hig)	
2015-02-10 1430 – 1500	Mariusz, Erik, Brage	Sup-15-008: Timing issues	- Prepare a prototype and test what sampling rates will suffice.	

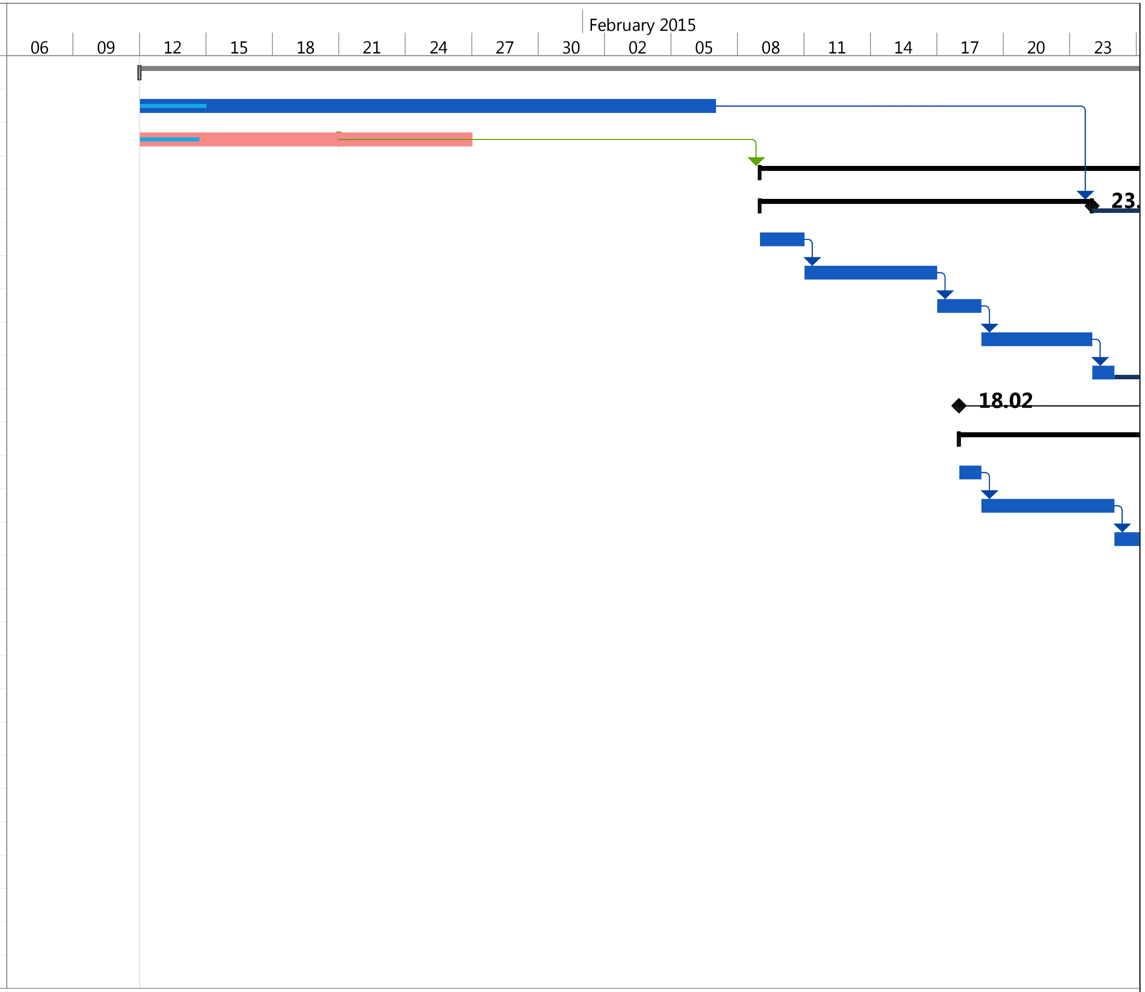
			Sup-15-009: Review architecture Sup-15-010: Contact information on webpage Sup-15-011: Eventual	Put mail on webpage. mariusz.nowostaowski@hig.no - Acellerometer/gyroscope would be great for biometry(recognise shaking osv). Mariusz has done this before. - Gyroscope is heavier to run than accellerometer, but then again provides more accurate data. - Two architectures to choose from, continuous processing(Data is always collected and processed at runtime, 1 thread) or sample processing(where you collect data for x seconds and then pause to process it, 2 threads). - Continuous processing may require lower resolution of data collection or prioritizing. - Prepare a prototype and test what sampling rates - We can use Android's sensor manager and register a callback for each sensor we want to use. Frequency is also defineable. - Capturing gestures unrooted may require native programming. - Nexus 5 will be slightly better than Nexus 4, but Nexus 4 would work as well. - Make test snippets for capturing data and include in thesis. - You can swipe to type on some android	
2015-02-11	0800 – 1600 A268	Brage, Eirik	1. Look at selecting a phone 2. Continue working on architecture		
2015-02-16	0845 - 1600 K204	Brage, Jannis	1. Review and continue Architecture, Kravspec 2. Prepare for tommorrow's sup-meeting		
2015-02-17	0815 – 1430 (Eirik 0815 - 1000) K204	Jannis, Brage, Eirik	1. Review and continue Architecture, Kravspec 2. Prepare for tommorrow's sup-meeting		
2015-02-17	1430-1500	Jannis, Brage, Mariusz	Sup-15-012: Interface programming: Sup-15-013: Eventual	Don't spend time now if need further research Binary file (fast saving) -> convert to json/csv… Conversion may take a lot of time requiring redesign… faster on computer than phone!	
2015-02-18	0815 – 1530 A268	Jannis, Brage	1. Continue work on architecture, implement changes due to yesterdays meeting		
2015-02-23	0815 - 1400 A266	Jannis, Brage, Eirik	1. Update Eirik on last meetings' progress 2. Start research phase		
2015-02-24	0815 - 1600 (Eirik 1300 - 1600)	Jannis, Brage, Eirik	1. Cotinue research 2. Send system requirements to Soumik	2. Forgot about TODO sections in requirements, finish and send tomorrow	
2015-02-24	1100 - 1145	Jannis, Brage, Soumik, Patrick	Emp-15-018: Options menue lockable in config? Emp-15-019: Eventual external (desktop-) application for conversion of logfiles to one of the required formats Emp-15-020: Eventual	Low priority, Our app is meant to be used in supervised/semisupervised experiments Ok Not ordered the phone yet, may be tablet (depends on size)?	
2015-02-24	1430 – 1500	Mariusz, Brage, Jannis, Eirik	Sup-15-014: Finding sources for work Sup-15-015: Eventual	- No sources found on what we want to know, have - Tablet / Phone not ordered yet	
2015-02-25	0815 - 1600 A158	Jannis, Eirik, Brage	1. Finish TODO markers 2. Send system requirements to Soumik 3. Aquire access to android source code base (read) 4. Start looking at permissions	1. Done 2. Done 3. Download took alot more space and time than we had thought and failed, retry asp	- Pulled branch android-5.0.1_r1 build LRX22C from android source code.
2015-03-02	0815 - 1600 K204	Jannis, Eirik, Brage	1. Start inspecting Sourcecode (getting an overwiew)	1. Identifiy relevant chapters in "Android Security Internals" 2. Read Chapter 1-3 in "Android Security Internals" 2. Clone sourcecode base for the 5.0.1_r1 branch (fix failed previous attempt)	Those chapters turned out to be very insightful and helpful in gaining an understandment of the architecture of android
2015-03-03	0815 - 1600 (Eirik 1200 - 1600) K204	Jannis, Eirik, Brage	1. Cont. Reading relevant chapters		- Since there are no available group rooms, tomorrows meeting will be held via Skype
2015-03-03	1430 – 1500	Mariusz, Brage, Jannis, Eirik	Sup-15-016: Update on reserach status, "Android Security Internals" Sup-15-017: Generating javadoc Sup-15-018: Summary of methods we consider	 - Mariusz will take a look, we continue trying - Cpu event registers: Most likely not that useful, low level events like cpu - Install from ADB: Same permission system, same permissions as user (but check it) May be able to attain screen data on attached debugger but that is not that useful	

		Sup-15-019: Which proof of concepts / experiments should we focus on / perform Sup-15-020: Eventual	Everything out of scope - Drop Unsure - Discuss with Mariusz (Up to a day: Just do it)	
2015-03-04 0815 - 1600 Skype	Brage, Jannis, Eirik	1. Cont. Reading relevant chapters		
2015-03-09 0815 - 1530 A061	Brage, Jannis	1. Finish Reading relevant chapters	1. Finished all chapters (Assume Eirik to finish the last chapter when he returns)	
2015-03-10 0815 - 1600 A270	Brage, Jannis, Eirik (Eirik 1200 - 1600)	1. Inspect identified source files		
2015-03-10 1200 - 1300	Patrick, Soumik, Brage, Jannis, Eirik	Emp-15-021: Device administrators Emp-15-022: Accessibility services and restrictions Emp-15-023: May be of interest for Soumiks research http://dl.acm.org/citation.cfm?doid=2037373.2037395 ("[We] show that touch positions are systematically skewed") Emp-15-024: Eventual	If time, build a layer that implements functionality for continuous authentication Upon action, measure and send to comparison value -> increase/decrease trust. When trust level is low enough, lock phone(etc). Make a dummy comparison algorithm to try. Gestures have noise. Doesn't matter for our application (copmarision function needs to address those issues). There will be changes, BELT did data representation really good, all columns had the same meaning for different event types. Maybe later more meetings (for example show progress every other day to receive feedback). Put in report: Discuss possibility of encrypting logs. If time, look at efficiency and building a device driver.	Hide phone in Jannis' secret stash.
2015-03-10 1430 - 1500	Mariusz, Brage, Jannis, Eirik	Sup-15-021: Device managers Sup-15-022: Accessibility services and restrictions Sup-15-023: Prioritasation of different approaches for proof of concept builds Sup-15-024: Build environment - dedicated machine borrowed from it departement? Sup-15-025: Eventual	need to look more into that (not sure) Only accessibility services as a viable lead, should make a proof of concept to next week. No need, this will most likely be more pain than gain Shall the app log keyboard touches or should it be suspended, if not no need to worry about password if not this may be a concern. Should the app know which other app open?	
2015-03-11 0815 - 1600 A268, A269	Brage, Eirik			
2015-03-16 0815 - 1600 K204	Brage, Jannis, Eirik	1. Try to get a background accessibility service to run 1. Cont. Looking at sourcecode	1. Service is running, but not registering any events 1. Found touch exploration, have to take a closer look on the example in the source	
2015-03-17 0815 - 1600 K204	Brage, Jannis	1. Cont. Accessibility service, intercept touch events	1. Events are captured, need now to look at contents	
2015-03-18 0815 - 1600 A162	Brage, Jannis	1. Inspect contents of captured events, determine wether useful information is contained or not		
2015-04-07 0815 - 1600 K204	Brage, Eirik (Jannis 1330 - 1600)	1. Experiment with overlay 2. Start research to find methods using root		
2015-04-07 1200 - 1300	Brage, Eirik, Patrick, Soumik		look into area in touch? for tak i største størrelsen av ellipsen, men ikke den minste (driveren har ikke implementert den fulle API want pressure as well higher sampling rate in accellorometer perhaps store the experiment id user id and session id (look into solutions for this) all settings will be in the settingsFile (also id î) SessionEvent(Id (PK ?),timeStamp (PK ?), posX, posY, Pressure, Area, GyroX, GyroY, GyroZ, AccelX, AccelY, AccelZ, ForeGroundApp, isPassword, ScrKeyboardActive) Session MetaData(Experiment ID, UserID, SessionId, ScreenCapabilities(size)) export csv (and maybe merging a complete database)	
2015-04-07 1430 - 1500	Brage, Jannis, Eirik, Mariusz	Sup-15-026: Eventual		
2015-04-08 0815 - 1600	Brage, Jannis, Eirik	1. Try file method using /dev/input/eventX 1. Design and implementation of the database		
2015-04-13 0815 - 1600 K210	Brage, Jannis, Eirik	1. Cont. File method, automatic detection of right file & filtering of event data		
2015-04-14 0815 - 1600	Brage, Jannis, Eirik (Eirik 1200 - 1600)	1. Finish filtering event data 2. Start controller / Design of main application		

2015-04-14 1430 - 1500	Brage, Jannis, Eirik, Mariusz	Sup-15-27: Presentation of progress	- Storage should be raw data (or as close as possible) - postprocessing should be done later - Two options (csv): 1. One row, increasing index that counts gestures 2. Two csv files: One for touch + time, one extra (which finger, gesture, time) Sampling rate should be as high as possible, do downsampling later (upsampling impossible)	
		Sup-15-28: Two applications or just exit touch service when su fails?	Service terminate	
		Sup-15-29: Communication between Service and Controller	Simplicity: DB access syncronised by default, create function that is called from the service	
		Sup-15-30: Eventual		
2015-04-15 0815 - 1550	Brage, Jannis, Eirik	1. Continue Database 1. Finish filtering of touch and dispatching to database handler		Todays meeting will end earlier than usual due to a company presentation strting at 1600
2015-04-29 0815 - 1600	Brage, Jannis, Eirik	Fixed Issue with unimplemented function in jdbc:sqlite		
2015-04-30 1115 - 1200	Brage, Jannis, Eirik	1. Notifications: necessary to use broadcastreciever or easier option? 2. How to update notification view 3. Should update deployment view etc. 4. Updating project plan?	1. No other way than broadcastreciever 2. rebuild with id	
2015-05-04 0815 - 1800	Brage, Jannis, Eirik	1. source Code open or Not? 1.a Licensing?		
2015-05-05 1100 - 1145	Patrick, Soumik, Brage, Jannis, Eirik	Emp-15-026: Presentation of progress	Still need to do some cleanup on the userinterface, must fix logging bug	
		Emp-15-027: Eventual	We may continue developing and participating in research if interested	
2015-05-06 0815 - 1800	Brage, Jannis, Eirik	Thesis writing		
2015-05-11 0815 - 1800	Brage, Jannis, Eirik	Thesis writing		
2015-05-12 0815 - 1800	Brage, Jannis, Eirik	Thesis writing		
2015-05-12 1430 - 1500	Brage, Jannis, Eirik, Mariusz	Sup-15-031: Eventual	Send in draft by tomorrow morning, Mariusz will give feedback in the course of the day	
2015-05-13 0815 - 1800	Brage, Jannis, Eirik	Thesis writing		

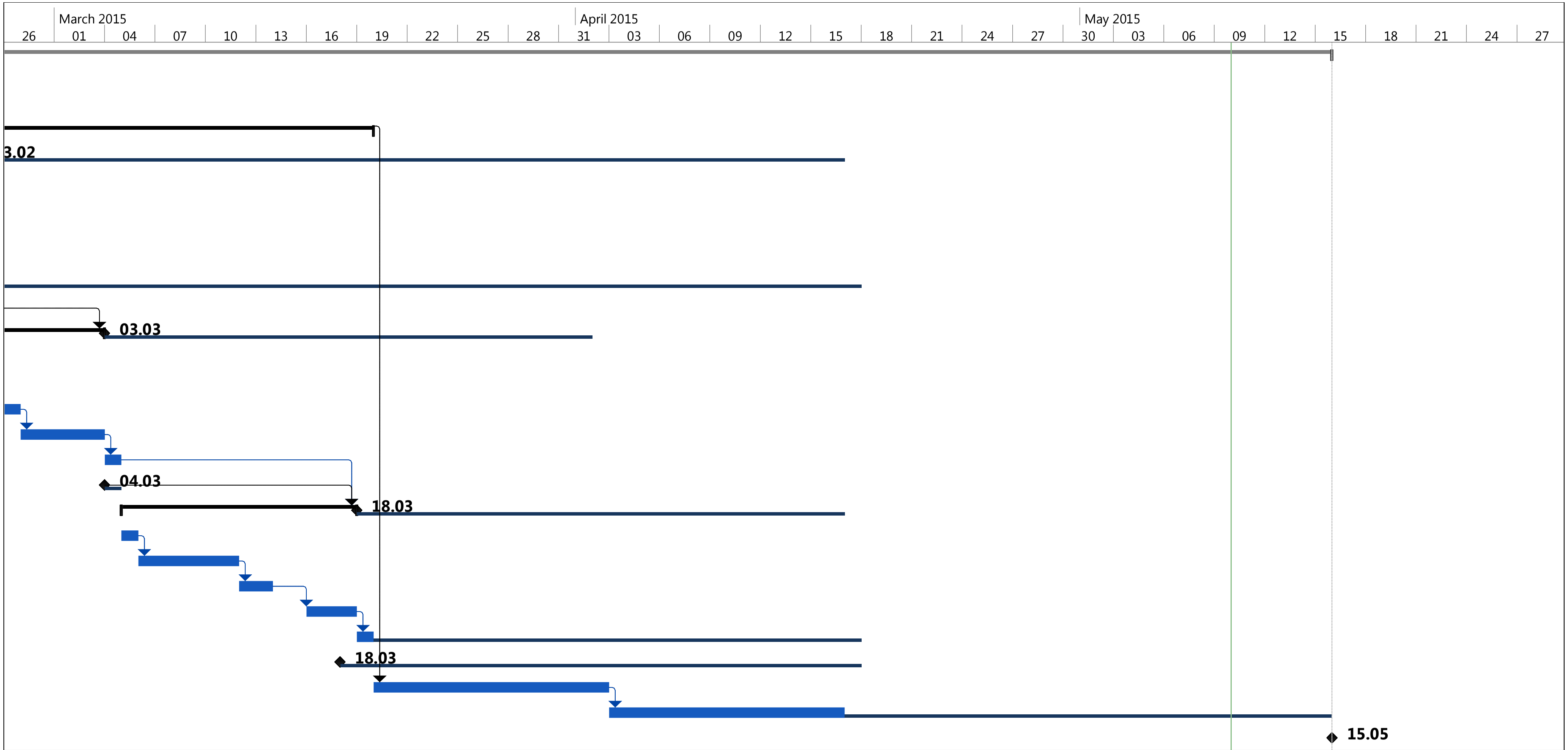
E Gantt of the project plan

ID		Task Mode	Task Name	Duration	Start	Finish
0			**Software Development**	**90 days**	**Mon**	**Fri 15.05.15**
1			Background research	4 wks	Mon 12.01.15	Fri 06.02.15
2			Software Requierments	11 days	Mon 12.01.15	Mon 26.01.15
3			**App development**	**29 days**	**Mon 09.02.15**	**Thu 19.03.15**
4			**1st Iteration**	**11 days**	**Mon 09.02.15**	**Mon 23.02.15**
5			Modeling	2 days	Mon 09.02.15	Tue 10.02.15
6			Programming	4 days	Wed 11.02.15	Mon 16.02.15
7			Debug	2 days	Tue 17.02.15	Wed 18.02.15
8			Testing	3 days	Thu 19.02.15	Mon 23.02.15
9			Review	1 day	Tue 24.02.15	Tue 24.02.15
10			Delivery Demo	0 days	Wed 18.02.15	Wed 18.02.15
11			**2nd Iteration**	**10 days**	**Wed 18.02.15**	**Tue 03.03.15**
12			Remodeling	1 day	Wed 18.02.15	Wed 18.02.15
13			Programming	4 days	Thu 19.02.15	Tue 24.02.15
14			Debug	2 days	Wed 25.02.15	Thu 26.02.15
15			Testing	3 days	Fri 27.02.15	Tue 03.03.15
16			Review	1 day	Wed 04.03.15	Wed 04.03.15
17			2nd Delivery	0 days	Wed 04.03.15	Wed 04.03.15
18			**3rd Iteration**	**10 days**	**Thu 05.03.15**	**Wed 18.03.15**
19			Remodeling	1 day	Thu 05.03.15	Thu 05.03.15
20			Programming	4 days	Fri 06.03.15	Wed 11.03.15
21			Debug	2 days	Thu 12.03.15	Fri 13.03.15
22			Testing	3 days	Mon 16.03.15	Wed 18.03.15
23			Review	1 day	Thu 19.03.15	Thu 19.03.15
24			3rd Delivery	0 days	Wed 18.03.15	Wed 18.03.15
25			Thesis Writing	2 wks	Fri 20.03.15	Thu 02.04.15
26			Presentation	10 days	Fri 03.04.15	Thu 16.04.15
27			Delivery	0 days	Fri 15.05.15	Fri 15.05.15



Project: Software Development
Date: Sun 10.05.15

Task	Inactive Milestone	Start-only	Critical Split
Split	Inactive Summary	Finish-only	Progress
Milestone	Manual Task	External Tasks	Manual Progress
Summary	Duration-only	External Milestone	Slack
Project Summary	Manual Summary Rollup	Deadline	
Inactive Task	Manual Summary	Critical	



Project: Software Development
Date: Sun 10.05.15

Task	Inactive Milestone	Start-only	Critical Split
Split	Inactive Summary	Finish-only	Progress
Milestone	Manual Task	External Tasks	Manual Progress
Summary	Duration-only	External Milestone	Slack
Project Summary	Manual Summary Rollup	Deadline	
Inactive Task	Manual Summary	Critical	

F Permission List Nexus 6 Device

```
shell@shamu:/ $ pm list permissions -f

All Permissions:


+ permission:com.google.android.apps.fitness.permission.C2D_MESSAGE
  package:com.google.android.apps.fitness
  label:null
  description:null
  protectionLevel:signature
+ permission:android.permission.ACCESS_INPUT_FLINGER
  package:android
  label:access InputFlinger
  description:Allows the app to use InputFlinger low-level features.
  protectionLevel:signature
+ permission:com.android.permission.CONNMO_SETTINGS
  package:com.android.sdm.plugins.connmo
  label:null
  description:null
  protectionLevel:signature|system
+ permission:com.android.permission.READ_OMADM_SETTINGS
  package:com.android.omadm.service
  label:null
  description:null
  protectionLevel:signature|system
+ permission:android.permission.BIND_TEXT_SERVICE
  package:android
  label:bind to a text service
  description:Allows the holder to bind to the top-level interface of a text
      service (e.g. SpellCheckerService). Should never be needed for normal
      applications.
  protectionLevel:signature
```

```
+ permission:com.android.gallery3d.filtershow.permission.WRITE
  package:com.google.android.apps.plus
  label:null
  description:null
  protectionLevel:signature
+ permission:com.android.vending.TOS_ACKED
  package:com.android.vending
  label:null
  description:null
  protectionLevel:signature|system
+ permission:android.permission.MANAGE_MEDIA_PROJECTION
  package:android
  label:Manage media projection sessions
  description:Allows an application to manage media projection sessions. These
      sessions can provide applications with the ability to capture display and
       audio contents. Should never be needed by normal apps.
  protectionLevel:signature
+ permission:android.permission.BIND_DREAM_SERVICE
  package:android
  label:bind to a dream service
  description:Allows the holder to bind to the top-level interface of a dream
      service. Should never be needed for normal apps.
  protectionLevel:signature
+ permission:com.google.android.apps.enterprise.dmagent.permission.C2D_MESSAGE
  package:com.google.android.apps.enterprise.dmagent
  label:null
  description:null
  protectionLevel:signature
+ permission:android.permission.ACCESS_BLUETOOTH_SHARE
  package:com.android.bluetooth
  label:Access download manager.
  description:Allows the application to access the Bluetooth Share manager and
      to use it to transfer files.
  protectionLevel:signature
+ permission:android.permission.SEND_DOWNLOAD_COMPLETED_INTENTS
  package:com.android.providers.downloads
```

```
  label:Send download notifications.

  description:Allows the app to send notifications about completed downloads.
      Malicious apps can use this to confuse other apps that download files.

  protectionLevel:signature

+ permission:android.permission.MODIFY_AUDIO_ROUTING

  package:android

  label:Audio Routing

  description:Allows the app to directly control audio routing and override
      audio policy decisions.

  protectionLevel:signature|system

+ permission:com.google.android.youtube.permission.C2D_MESSAGE

  package:com.google.android.youtube

  label:null

  description:null

  protectionLevel:signature

+ permission:com.google.android.apps.now.OPT_IN_WIZARD

  package:com.google.android.googlequicksearchbox

  label:null

  description:null

  protectionLevel:signature|system

+ permission:android.permission.ACCESS_KEYGUARD_SECURE_STORAGE

  package:android

  label:Access keyguard secure storage

  description:Allows an application to access keyguard secure storage.

  protectionLevel:signature

+ permission:android.permission.FILTER_EVENTS

  package:android

  label:filter events

  description:Allows an application to register an input filter which filters
      the stream of all user events before they are dispatched. Malicious app
      may control the system UI without user intervention.

  protectionLevel:signature

+ permission:com.google.android.gms.permission.C2D_MESSAGE

  package:com.google.android.gms

  label:null
```

```
  description:null
  protectionLevel:signature
+ permission:com.google.android.email.permission.ACCESS_PROVIDER
  package:com.google.android.email
  label:Access email provider data
  description:Allows the app to access your email database, including received
      messages, sent messages, usernames and passwords.
  protectionLevel:signature
+ permission:com.google.android.apps.plus.permission.C2D_MESSAGE
  package:com.google.android.apps.plus
  label:null
  description:null
  protectionLevel:signature
+ permission:android.permission.CAPTURE_TV_INPUT
  package:android
  label:null
  description:null
  protectionLevel:signature|system
+ permission:android.permission.MODIFY_NETWORK_ACCOUNTING
  package:android
  label:modify network usage accounting
  description:Allows the app to modify how network usage is accounted against
      apps. Not for use by normal apps.
  protectionLevel:signature|system
+ permission:android.permission.SET_POINTER_SPEED
  package:android
  label:change pointer speed
  description:Allows the app to change the mouse or touch pad pointer speed at
      any time. Should never be needed for normal apps.
  protectionLevel:signature
+ permission:android.permission.TV_INPUT_HARDWARE
  package:android
  label:null
  description:null
  protectionLevel:signature|system
```

```
+ permission:android.permission.CALL_PRIVILEGED

  package:android

  label:directly call any phone numbers

  description:Allows the app to call any phone number, including emergency
      numbers, without your intervention. Malicious apps may place unnecessary
      and illegal calls to emergency services.

  protectionLevel:signature|system

+ permission:android.permission.BRICK

  package:android

  label:permanently disable phone

  description:Allows the app to permanently disable the entire phone. This is
      very dangerous.

  protectionLevel:signature

+ permission:com.google.android.apps.maps.permission.PREFETCH

  package:com.google.android.apps.maps

  label:null

  description:null

  protectionLevel:signature

+ permission:android.permission.BIND_DEVICE_ADMIN

  package:android

  label:interact with device admin

  description:Allows the holder to send intents to a device administrator.
      Should never be needed for normal apps.

  protectionLevel:signature

+ permission:com.google.android.portable.permission.READ

  package:com.google.earth

  label:Read Maps Engine Portable Provider

  description:Allows third party applications to read the Maps Engine provider.

  protectionLevel:normal

+ permission:com.google.android.apps.cloudprint.permission.C2D_MESSAGE

  package:com.google.android.apps.cloudprint

  label:null

  description:null

  protectionLevel:signature

+ permission:android.permission.PERFORM_CDMA_PROVISIONING

  package:android
```

```
  label:directly start CDMA phone setup

  description:Allows the app to start CDMA provisioning. Malicious apps may
      unnecessarily start CDMA provisioning.

  protectionLevel:signature|system

+ permission:com.android.chrome.PRERENDER_URL

  package:com.android.chrome

  label:null

  description:null

  protectionLevel:normal

+ permission:android.permission.DELETE_CACHE_FILES

  package:android

  label:delete other apps' caches

  description:Allows the app to delete cache files.

  protectionLevel:signature|system

+ permission:com.motorola.audiomonitor.permission.SETTINGS

  package:com.motorola.triggerenroll

  label:null

  description:null

  protectionLevel:signature|system

+ permission:com.google.android.gsf.permission.C2D_MESSAGE

  package:com.google.android.gsf

  label:null

  description:null

  protectionLevel:signature

+ permission:android.permission.START_PRINT_SERVICE_CONFIG_ACTIVITY

  package:com.android.printspooler

  label:start print service configuration activities

  description:Allows the holder to start the configuration activities of a print
       service. Should never be needed for normal apps.

  protectionLevel:signature

+ permission:com.google.android.providers.settings.permission.WRITE_GSETTINGS

  package:com.google.android.gsf

  label:Modify Google settings

  description:Allows this app to modify Google settings.
```

```
  protectionLevel:signature

+ permission:android.permission.CAPTURE_AUDIO_HOTWORD

  package:android

  label:Hotword detection

  description:Allows the app to capture audio for Hotword detection. The capture
       can happen in the background but does not prevent other audio capture (e
      .g. Camcorder).

  protectionLevel:signature|system

+ permission:android.permission.WRITE_GSERVICES

  package:android

  label:modify the Google services map

  description:Allows the app to modify the Google services map. Not for use by
      normal apps.

  protectionLevel:signature|system

+ permission:com.google.android.googleapps.permission.GOOGLE_AUTH.goanna_mobile

  package:com.google.android.gsf

  label:Google Tasks

  description:null

  protectionLevel:normal

+ permission:com.google.android.ears.permission.READ

  package:com.google.android.ears

  label:Permission to read Sound Search matches

  description:null

  protectionLevel:signature

+ permission:android.permission.CLEAR_APP_USER_DATA

  package:android

  label:delete other apps' data

  description:Allows the app to clear user data.

  protectionLevel:signature

+ permission:android.permission.CONTROL_LOCATION_UPDATES

  package:android

  label:control location update notifications

  description:Allows the app to enable/disable location update notifications
      from the radio. Not for use by normal apps.

  protectionLevel:signature|system

+ permission:android.permission.MANAGE_APP_TOKENS
```

```
  package:android

  label:manage app tokens

  description:Allows the app to create and manage their own tokens, bypassing
      their normal Z-ordering. Should never be needed for normal apps.

  protectionLevel:signature

+ permission:android.permission.FREEZE_SCREEN

  package:android

  label:freeze screen

  description:Allows the application to temporarily freeze the screen for a full
      -screen transition.

  protectionLevel:signature

+ permission:android.permission.READ_INSTALL_SESSIONS

  package:android

  label:Read install sessions

  description:Allows an application to read install sessions. This allows it to
      see details about active package installations.

  protectionLevel:normal

+ permission:android.permission.USER_ACTIVITY

  package:android

  label:reset display timeout

  description:Allows the app to reset the display timeout.

  protectionLevel:signature|system

+ permission:com.google.android.onetimeinitializer.permission.
    ONE_TIME_INITIALIZED

  package:com.google.android.onetimeinitializer

  label:null

  description:null

  protectionLevel:signature

+ permission:com.google.android.googleapps.permission.GOOGLE_AUTH.geowiki

  package:com.google.android.gsf

  label:Google Map maker

  description:null

  protectionLevel:normal

+ permission:android.permission.INJECT_EVENTS

  package:android
```

```
  label:press keys and control buttons

  description:Allows the app to deliver its own input events (key presses, etc.)
      to other apps. Malicious apps may use this to take over the phone.

  protectionLevel:signature

+ permission:com.android.permission.WRITE_OMADM_SETTINGS

  package:com.android.omadm.service

  label:null

  description:null

  protectionLevel:signature|system

+ permission:android.permission.UPDATE_APP_OPS_STATS

  package:android

  label:modify app ops statistics

  description:Allows the app to modify collected component usage statistics. Not
      for use by normal apps.

  protectionLevel:signature|system

+ permission:android.permission.READ_NETWORK_USAGE_HISTORY

  package:android

  label:read historical network usage

  description:Allows the app to read historical network usage for specific
     networks and apps.

  protectionLevel:signature|system

+ permission:com.google.googlenav.friend.permission.OPT_IN

  package:com.google.android.apps.maps

  label:null

  description:null

  protectionLevel:signature

+ permission:com.google.android.apps.walletnfcrel.permission.C2D_MESSAGE

  package:com.google.android.apps.walletnfcrel

  label:null

  description:null

  protectionLevel:signature

+ permission:android.permission.BIND_PRINT_SERVICE

  package:android

  label:bind to a print service

  description:Allows the holder to bind to the top-level interface of a print
     service. Should never be needed for normal apps.
```

```
  protectionLevel:signature
+ permission:com.google.android.providers.settings.permission.READ_GSETTINGS
  package:com.google.android.gsf
  label:Read Google settings
  description:Allows this app to read Google settings.
  protectionLevel:signature
+ permission:android.permission.BACKUP
  package:android
  label:control system back up and restore
  description:Allows the app to control the system's backup and restore
      mechanism. Not for use by normal apps.
  protectionLevel:signature|system
+ permission:com.android.vending.INTENT_VENDING_ONLY
  package:com.google.android.gsf
  label:Send broadcasts to Android Market.
  description:Can send broadcasts to Android Market requesting app installation
      and removal.
  protectionLevel:signature
+ permission:com.google.android.gallery3d.permission.GALLERY_PROVIDER
  package:com.google.android.apps.plus
  label:null
  description:null
  protectionLevel:signature
+ permission:com.google.android.googlequicksearchbox.LAUNCH_WITH_RECORDED_AUDIO
  package:com.google.android.googlequicksearchbox
  label:Launch voice with recorded audio
  description:Launch voice with recorded audio
  protectionLevel:signature|system
+ permission:com.google.android.partnersetup.permission.ACCESS_PROVIDER
  package:com.google.android.partnersetup
  label:null
  description:null
  protectionLevel:signature
+ permission:android.permission.BIND_VOICE_INTERACTION
```

```
  package:android

  label:bind to a voice interactor

  description:Allows the holder to bind to the top-level interface of a voice
      interaction service. Should never be needed for normal apps.

  protectionLevel:signature

+ permission:com.google.android.googlequicksearchbox.permission.PAUSE_HOTWORD

  package:com.google.android.googlequicksearchbox

  label:null

  description:null

  protectionLevel:signature|system

+ permission:android.permission.ACCESS_CHECKIN_PROPERTIES

  package:android

  label:access check-in properties

  description:Allows the app read/write access to properties uploaded by the
      check-in service. Not for use by normal apps.

  protectionLevel:signature|system

+ permission:android.permission.PROCESS_CALLLOG_INFO

  package:com.android.server.telecom

  label:Register to handle the broadcasted call type/duration information

  description:null

  protectionLevel:signature|system

+ permission:com.android.vending.setup.PLAY_SETUP_SERVICE

  package:com.android.vending

  label:null

  description:null

  protectionLevel:signature|system

+ permission:com.google.android.hangouts.START_HANGOUT

  package:com.google.android.talk

  label:null

  description:null

  protectionLevel:signature

+ permission:android.permission.ACCESS_DOWNLOAD_MANAGER_ADVANCED

  package:com.android.providers.downloads

  label:Advanced download manager functions.
```

```
  description:Allows the app to access the download manager's advanced functions
      . Malicious apps can use this to disrupt downloads and access private
      information.

  protectionLevel:signature|system

+ permission:com.google.android.apps.wallet.permission.WALLET_INTERNAL

  package:com.google.android.apps.walletnfcrel

  label:Wallet Application

  description:Access Wallet-internal data.

  protectionLevel:signature

+ permission:com.google.android.ears.permission.WRITE

  package:com.google.android.ears

  label:Permission to write Sound Search matches

  description:null

  protectionLevel:signature

+ permission:android.permission.MANAGE_DEVICE_ADMINS

  package:android

  label:add or remove a device admin

  description:Allows the holder to add or remove active device administrators.
      Should never be needed for normal apps.

  protectionLevel:signature|system

+ permission:android.permission.NFC_HANDOVER_STATUS

  package:android

  label:Receive Android Beam transfer status

  description:Allows this application to receive information about current
      Android Beam transfers

  protectionLevel:signature|system

+ permission:android.permission.CONTROL_WIFI_DISPLAY

  package:android

  label:control Wi-Fi displays

  description:Allows the app to control low-level features of Wi-Fi displays.

  protectionLevel:signature

+ permission:android.permission.MANAGE_CA_CERTIFICATES

  package:android

  label:manage trusted credentials

  description:Allows the app to install and uninstall CA certificates as trusted
       credentials.
```

```
  protectionLevel:signature|system

+ permission:com.google.android.gsf.permission.CONNECTION

  package:com.google.android.gsf

  label:null

  description:null

  protectionLevel:signature

+ permission:android.permission.UPDATE_DEVICE_STATS

  package:android

  label:modify battery statistics

  description:Allows the app to modify collected battery statistics. Not for use
       by normal apps.

  protectionLevel:signature|system

+ permission:android.server.checkin.CHECKIN.permission.C2D_MESSAGE

  package:com.google.android.gsf

  label:null

  description:null

  protectionLevel:signature

+ permission:com.google.android.apps.enterprise.dmagent.permission.
    AutoRegisterPermission

  package:com.google.android.apps.enterprise.dmagent

  label:null

  description:null

  protectionLevel:signature|system

+ permission:android.permission.READ_FRAME_BUFFER

  package:android

  label:read frame buffer

  description:Allows the app to read the content of the frame buffer.

  protectionLevel:signature|system

+ permission:com.google.android.googleapps.permission.ACCESS_GOOGLE_PASSWORD

  package:com.google.android.gsf.login

  label:access to passwords for Google accounts

  description:Allows apps direct access to the passwords for the Google account(
      s) that you have set up.

  protectionLevel:signature

+ permission:android.permission.INVOKE_CARRIER_SETUP
```

```
  package:android

  label:invoke the carrier-provided configuration app

  description:Allows the holder to invoke the carrier-provided configuration app
      . Should never be needed for normal apps.

  protectionLevel:signature|system

+ permission:com.google.android.googleapps.permission.GOOGLE_AUTH.panoramio

  package:com.google.android.gsf

  label:Panoramio

  description:null

  protectionLevel:normal

+ permission:com.google.android.gms.permission.CAR

  package:com.google.android.gms

  label:Car Service

  description:Access to the car service.

  protectionLevel:signature

+ permission:com.android.vending.billing.BILLING_ACCOUNT_SERVICE

  package:com.android.vending

  label:null

  description:null

  protectionLevel:signature|system

+ permission:android.permission.BIND_TV_INPUT

  package:android

  label:bind to a TV input

  description:Allows the holder to bind to the top-level interface of a TV input
      . Should never be needed for normal apps.

  protectionLevel:signature|system

+ permission:com.google.android.apps.photos.permission.C2D_MESSAGE

  package:com.google.android.apps.photos

  label:null

  description:null

  protectionLevel:signature

+ permission:android.permission.MANAGE_VOICE_KEYPHRASES

  package:android

  label:manage voice key phrases
```

```
  description:Allows the holder to manage the key phrases for voice hotword
      detection. Should never be needed for normal apps.

  protectionLevel:signature|system

+ permission:android.permission.BIND_REMOTEVIEWS

  package:android

  label:bind to a widget service

  description:Allows the holder to bind to the top-level interface of a widget
      service. Should never be needed for normal apps.

  protectionLevel:signature|system

+ permission:com.google.android.partnersetup.permission.UPDATE_CLIENT_ID

  package:com.google.android.partnersetup

  label:null

  description:null

  protectionLevel:signature|system

+ permission:android.permission.LAUNCH_TRUST_AGENT_SETTINGS

  package:android

  label:Launch trust agent settings menu.

  description:Allows an application to launch an activity that changes the trust
       agent behaviour.

  protectionLevel:signature|system

+ permission:com.google.android.googleapps.permission.GOOGLE_AUTH.reader

  package:com.google.android.gsf

  label:Google Reader

  description:null

  protectionLevel:normal

+ permission:android.permission.SET_KEYBOARD_LAYOUT

  package:android

  label:change keyboard layout

  description:Allows the app to change the keyboard layout. Should never be
      needed for normal apps.

  protectionLevel:signature

+ permission:com.google.android.apps.magazines.permission.C2D_MESSAGE

  package:com.google.android.apps.magazines

  label:null

  description:null

  protectionLevel:signature
```

```
+ permission:com.android.permission.INJECT_OMADM_SETTINGS

  package:com.android.omadm.service

  label:null

  description:null

  protectionLevel:signature|system

+ permission:android.permission.ACCESS_SURFACE_FLINGER

  package:android

  label:access SurfaceFlinger

  description:Allows the app to use SurfaceFlinger low-level features.

  protectionLevel:signature

+ permission:android.permission.SHUTDOWN

  package:android

  label:partial shutdown

  description:Puts the activity manager into a shut-down state. Does not perform
       a complete shut down.

  protectionLevel:signature|system

+ permission:com.google.android.apps.enterprise.dmagent.permission.
    AutoSyncPermission

  package:com.google.android.apps.enterprise.dmagent

  label:null

  description:null

  protectionLevel:signature|system

+ permission:android.permission.ACCESS_DOWNLOAD_MANAGER

  package:com.android.providers.downloads

  label:Access download manager.

  description:Allows the app to access the download manager and to use it to
      download files. Malicious apps can use this to disrupt downloads and
      access private information.

  protectionLevel:signature|system

+ permission:android.permission.FACTORY_TEST

  package:android

  label:run in factory test mode

  description:Run as a low-level manufacturer test, allowing complete access to
      the phone hardware. Only available when a phone is running in
      manufacturer test mode.

  protectionLevel:signature
```

```
+ permission:android.permission.SET_INPUT_CALIBRATION
  package:android
  label:change input device calibration
  description:Allows the app to modify the calibration parameters of the touch
      screen. Should never be needed for normal apps.
  protectionLevel:signature
+ permission:com.google.android.videos.permission.C2D_MESSAGE
  package:com.google.android.videos
  label:null
  description:null
  protectionLevel:signature
+ permission:android.permission.SET_TIME
  package:android
  label:set time
  description:Allows the app to change the phone's clock time.
  protectionLevel:signature|system
+ permission:com.android.chrome.permission.C2D_MESSAGE
  package:com.android.chrome
  label:null
  description:null
  protectionLevel:signature
+ permission:android.permission.ACCESS_CACHE_FILESYSTEM
  package:android
  label:access the cache file system
  description:Allows the app to read and write the cache file system.
  protectionLevel:signature|system
+ permission:com.google.android.launcher.permission.RECEIVE_LAUNCH_BROADCASTS
  package:com.google.android.googlequicksearchbox
  label:null
  description:null
  protectionLevel:signature
+ permission:android.permission.ACCESS_NOTIFICATIONS
  package:android
  label:access notifications
```

```
  description:Allows the app to retrieve, examine, and clear notifications,
      including those posted by other apps.

  protectionLevel:signature|system

+ permission:android.permission.UPDATE_LOCK

  package:android

  label:discourage automatic device updates

  description:Allows the holder to offer information to the system about when
      would be a good time for a non-interactive reboot to upgrade the device.

  protectionLevel:signature|system

+ permission:android.permission.BIND_NOTIFICATION_LISTENER_SERVICE

  package:android

  label:bind to a notification listener service

  description:Allows the holder to bind to the top-level interface of a
      notification listener service. Should never be needed for normal apps.

  protectionLevel:signature

+ permission:com.google.android.apps.plus.permission.MAPS_RECEIVE

  package:com.google.android.apps.plus

  label:null

  description:null

  protectionLevel:signature

+ permission:android.permission.BIND_ACCESSIBILITY_SERVICE

  package:android

  label:bind to an accessibility service

  description:Allows the holder to bind to the top-level interface of an
      accessibility service. Should never be needed for normal apps.

  protectionLevel:signature

+ permission:com.google.android.gms.permission.GAMES_DEBUG_SETTINGS

  package:com.google.android.gms

  label:null

  description:null

  protectionLevel:signature

+ permission:com.google.android.gms.permission.INTERNAL_BROADCAST

  package:com.google.android.gms

  label:null

  description:null

  protectionLevel:signature
```

```
+ permission:android.permission.CRYPT_KEEPER
  package:android
  label:null
  description:null
  protectionLevel:signature|system
+ permission:com.android.chrome.TOS_ACKED
  package:com.android.chrome
  label:null
  description:null
  protectionLevel:signature|system
+ permission:com.android.vending.billing.IN_APP_NOTIFY.permission.C2D_MESSAGE
  package:com.android.vending
  label:null
  description:null
  protectionLevel:signature
+ permission:android.permission.BIND_VPN_SERVICE
  package:android
  label:bind to a VPN service
  description:Allows the holder to bind to the top-level interface of a Vpn
      service. Should never be needed for normal apps.
  protectionLevel:signature
+ permission:com.google.android.googlequicksearchbox.permission.FLUSH_LOGS
  package:com.google.android.googlequicksearchbox
  label:null
  description:null
  protectionLevel:signature
+ permission:com.android.certinstaller.INSTALL_AS_USER
  package:com.android.certinstaller
  label:null
  description:null
  protectionLevel:signature
+ permission:android.permission.BIND_WALLPAPER
  package:android
  label:bind to wallpaper
```

```
  description:Allows the holder to bind to the top-level interface of wallpaper.
       Should never be needed for normal applications.

  protectionLevel:signature|system

+ permission:android.permission.ACCESS_NETWORK_CONDITIONS

  package:android

  label:listen for observations on network conditions

  description:Allows an application to listen for observations on network
      conditions. Should never be needed for normal apps.

  protectionLevel:signature|system

+ permission:android.permission.DELETE_PACKAGES

  package:android

  label:delete apps

  description:Allows the app to delete Android packages. Malicious apps may use
      this to delete important apps.

  protectionLevel:signature|system

+ permission:com.google.android.googlequicksearchbox.permission.
    FINISH_GEL_ACTIVITY

  package:com.google.android.googlequicksearchbox

  label:null

  description:null

  protectionLevel:signature

+ permission:android.permission.REBOOT

  package:android

  label:force phone reboot

  description:Allows the app to force the phone to reboot.

  protectionLevel:signature|system

+ permission:android.permission.ALLOW_ANY_CODEC_FOR_PLAYBACK

  package:android

  label:use any media decoder for playback

  description:Allows the app to use any installed media decoder to decode for
      playback.

  protectionLevel:signature|system

+ permission:android.permission.BIND_CONDITION_PROVIDER_SERVICE

  package:android

  label:bind to a condition provider service
```

```
  description:Allows the holder to bind to the top—level interface of a
      condition provider service. Should never be needed for normal apps.

  protectionLevel:signature

+ permission:android.permission.BIND_JOB_SERVICE

  package:android

  label:run the application's scheduled background work

  description:This permission allows the Android system to run the application
      in the background when requested.

  protectionLevel:signature

+ permission:android.permission.CONFIRM_FULL_BACKUP

  package:android

  label:confirm a full backup or restore operation

  description:Allows the app to launch the full backup confirmation UI. Not to
      be used by any app.

  protectionLevel:signature

+ permission:com.android.printspooler.permission.ACCESS_ALL_PRINT_JOBS

  package:com.android.printspooler

  label:access all print jobs

  description:Allows the holder to access print jobs created by another app.
      Should never be needed for normal apps.

  protectionLevel:signature

+ permission:com.google.android.music.store.permission.C2D_MESSAGE

  package:com.google.android.music

  label:null

  description:null

  protectionLevel:signature

+ permission:android.intent.category.MASTER_CLEAR.permission.C2D_MESSAGE

  package:android

  label:null

  description:null

  protectionLevel:signature

+ permission:com.synaptics.permission.FINGERPRINT

  package:com.motorola.motocit

  label:Access fingerprint reader.

  description:Allows application to access fingerprint reader.

  protectionLevel:signature|system
```

```
+ permission:android.permission.BIND_PRINT_SPOOLER_SERVICE

  package:android

  label:bind to a print spooler service

  description:Allows the holder to bind to the top-level interface of a print
      spooler service. Should never be needed for normal apps.

  protectionLevel:signature

+ permission:android.permission.CAPTURE_SECURE_VIDEO_OUTPUT

  package:android

  label:capture secure video output

  description:Allows the app to capture and redirect secure video output.

  protectionLevel:signature|system

+ permission:android.permission.BIND_REMOTE_DISPLAY

  package:android

  label:bind to a remote display

  description:Allows the holder to bind to the top-level interface of a remote
      display. Should never be needed for normal apps.

  protectionLevel:signature

+ permission:android.permission.SET_ORIENTATION

  package:android

  label:change screen orientation

  description:Allows the app to change the rotation of the screen at any time.
      Should never be needed for normal apps.

  protectionLevel:signature

+ permission:com.google.android.googleapps.permission.GOOGLE_MAIL_SWITCH

  package:com.google.android.gsf.login

  label:select Gmail or Gmail branding

  description:Allows apps to change the displayed name between "Gmail" and "
      Google Mail" branding.

  protectionLevel:signature

+ permission:android.permission.REMOVE_DRM_CERTIFICATES

  package:android

  label:remove DRM certificates

  description:Allows an application to remove DRM certficates. Should never be
      needed for normal apps.

  protectionLevel:signature|system

+ permission:android.permission.CONFIGURE_WIFI_DISPLAY
```

```
  package:android
  label:configure Wi-Fi displays
  description:Allows the app to configure and connect to Wi-Fi displays.
  protectionLevel:signature
+ permission:android.permission.MOVE_PACKAGE
  package:android
  label:move app resources
  description:Allows the app to move app resources from internal to external
      media and vice versa.
  protectionLevel:signature|system
+ permission:com.google.android.launcher.permission.RECEIVE_FIRST_LOAD_BROADCAST
  package:com.google.android.googlequicksearchbox
  label:null
  description:null
  protectionLevel:signature|system
+ permission:com.android.chrome.permission.READ_WRITE_BOOKMARK_FOLDERS
  package:com.android.chrome
  label:null
  description:null
  protectionLevel:signature|system
+ permission:com.motorola.audiomonitor.permission.LOCAL
  package:com.motorola.triggerenroll
  label:null
  description:null
  protectionLevel:signature|system
+ permission:android.permission.ACCESS_CONTENT_PROVIDERS_EXTERNALLY
  package:android
  label:access content providers externally
  description:Allows the holder to access content providers from the shell.
      Should never be needed for normal apps.
  protectionLevel:signature
+ permission:android.permission.PACKAGE_USAGE_STATS
  package:android
  label:update component usage statistics
```

```
  description:Allows the app to modify collected component usage statistics. Not
       for use by normal apps.

  protectionLevel:signature|development|appop

+ permission:com.google.android.gsf.subscribedfeeds.permission.C2D_MESSAGE

  package:com.google.android.gsf

  label:null

  description:null

  protectionLevel:signature

+ permission:android.permission.RETRIEVE_WINDOW_TOKEN

  package:android

  label:retrieve window token

  description:Allows an application to retrieve the window token. Malicious apps
       may perform unauthorised interaction with the application window
      impersonating the system.

  protectionLevel:signature

+ permission:android.permission.MEDIA_CONTENT_CONTROL

  package:android

  label:control media playback and metadata access

  description:Allows the app to control media playback and access the media
      information (title, author...).

  protectionLevel:signature|system

+ permission:com.google.android.calendar.permission.C2D_MESSAGE

  package:com.google.android.calendar

  label:null

  description:null

  protectionLevel:signature

+ permission:android.permission.COPY_PROTECTED_DATA

  package:android

  label:copy content

  description:copy content

  protectionLevel:signature

+ permission:com.motorola.android.permission.TCMD_LOCAL

  package:com.motorola.motocit

  label:Use Test Commands

  description:Use Test Commands

  protectionLevel:signature|system
```

```
+ permission:com.google.android.videos.permission.INVALIDATE_AUTH_TOKENS
  package:com.google.android.videos
  label:null
  description:null
  protectionLevel:signature
+ permission:android.permission.PROVIDE_TRUST_AGENT
  package:android
  label:Provide a trust agent.
  description:Allows an application to provide a trust agent.
  protectionLevel:signature|system
+ permission:android.permission.DEVICE_POWER
  package:android
  label:turn phone on or off
  description:Allows the app to turn the phone on or off.
  protectionLevel:signature
+ permission:com.google.android.music.download.artwork.
    RECEIVE_BROADCAST_PERMISSION
  package:com.google.android.music
  label:null
  description:null
  protectionLevel:signature
+ permission:com.google.android.music.xdi.START_PLAYBACK
  package:com.google.android.music
  label:null
  description:null
  protectionLevel:signature
+ permission:com.motorola.audiomonitor.permission.STATE_CONTROL
  package:com.motorola.triggerenroll
  label:null
  description:null
  protectionLevel:signature|system
+ permission:android.permission.BIND_PACKAGE_VERIFIER
  package:android
  label:bind to a package verifier
```

```
  description:Allows the holder to make requests of package verifiers. Should
      never be needed for normal apps.

  protectionLevel:signature

+ permission:android.permission.HDMI_CEC

  package:android

  label:null

  description:null

  protectionLevel:signature|system

+ permission:android.permission.BIND_INPUT_METHOD

  package:android

  label:bind to an input method

  description:Allows the holder to bind to the top-level interface of an input
      method. Should never be needed for normal apps.

  protectionLevel:signature

+ permission:android.permission.GET_TOP_ACTIVITY_INFO

  package:android

  label:get current app info

  description:Allows the holder to retrieve private information about the
      current application in the foreground of the screen.

  protectionLevel:signature

+ permission:android.permission.FRAME_STATS

  package:android

  label:retrieve frame statistics

  description:Allows an application to collect frame statistics. Malicious apps
      may observe the frame statistics of windows from other apps.

  protectionLevel:signature

+ permission:android.permission.STATUS_BAR

  package:android

  label:disable or modify status bar

  description:Allows the app to disable the status bar or add and remove system
      icons.

  protectionLevel:signature|system

+ permission:android.permission.SET_ACTIVITY_WATCHER

  package:android

  label:monitor and control all app launching
```

```
  description:Allows the app to monitor and control how the system launches
      activities. Malicious apps may completely compromise the system. This
      permission is only needed for development, never for normal use.

  protectionLevel:signature

+ permission:com.google.android.apps.maps.permission.C2D_MESSAGE

  package:com.google.android.apps.maps

  label:null

  description:null

  protectionLevel:signature

+ permission:android.permission.ACCESS_ALL_DOWNLOADS

  package:com.android.providers.downloads

  label:Access all system downloads

  description:Allows the app to view and modify all downloads initiated by any
      app on the system.

  protectionLevel:signature

+ permission:com.motorola.audiomonitor.permission.BROADCAST_RECEIVER

  package:com.motorola.triggerenroll

  label:null

  description:null

  protectionLevel:signature|system

+ permission:com.android.server.telecom.permission.REGISTER_CONNECTION_MANAGER

  package:com.android.server.telecom

  label:Register CONNECTION_MANAGER PhoneAccount

  description:null

  protectionLevel:signature

+ permission:com.google.android.gms.permission.CHECKIN_NOW

  package:com.google.android.gms

  label:null

  description:null

  protectionLevel:signature

+ permission:com.google.android.launcher.permission.CONTENT_REDIRECT

  package:com.google.android.launcher

  label:null

  description:null

  protectionLevel:signature
```

```
+ permission:android.permission.STOP_APP_SWITCHES
  package:android
  label:prevent app switches
  description:Prevents the user from switching to another app.
  protectionLevel:signature|system
+ permission:com.google.android.gms.DRIVE
  package:com.google.android.gms
  label:null
  description:null
  protectionLevel:signature
+ permission:com.google.android.googlequicksearchbox.LAUNCH_FROM_DSP_HOTWORD
  package:com.google.android.googlequicksearchbox
  label:Launch voice search from DSP hotword
  description:Launch voice search from DSP hotword
  protectionLevel:signature|system
+ permission:android.permission.TEMPORARY_ENABLE_ACCESSIBILITY
  package:android
  label:temporary enable accessibility
  description:Allows an application to temporarily enable accessibility on the
      device. Malicious apps may enable accessibility without user consent.
  protectionLevel:signature
+ permission:com.android.chrome.permission.CHILD_SERVICE
  package:com.android.chrome
  label:null
  description:null
  protectionLevel:signature
+ permission:com.google.android.providers.gsf.permission.WRITE_GSERVICES
  package:com.google.android.gsf
  label:Modify Google service configuration
  description:Allows this app to modify Google service configuration data.
  protectionLevel:signature|system
+ permission:com.google.android.gms.auth.permission.GOOGLE_ACCOUNT_CHANGE
  package:com.google.android.gms
  label:null
```

```
  description:null

  protectionLevel:signature

+ permission:android.permission.CONTROL_KEYGUARD

  package:android

  label:Control displaying and hiding keyguard

  description:Allows an application to control keyguard.

  protectionLevel:signature

+ permission:com.android.server.telecom.permission.
    REGISTER_PROVIDER_OR_SUBSCRIPTION

  package:com.android.server.telecom

  label:Register CALL_PROVIDER or SIM_SUBSCRIPTION PhoneAccount

  description:null

  protectionLevel:signature

+ permission:com.android.vending.billing.ADD_CREDIT_CARD

  package:com.android.vending

  label:null

  description:null

  protectionLevel:signature|system

+ permission:android.permission.INTERNAL_SYSTEM_WINDOW

  package:android

  label:display unauthorised windows

  description:Allows the app to create windows that are intended to be used by
      the internal system user interface. Not for use by normal apps.

  protectionLevel:signature

+ permission:com.google.android.gm.email.permission.ACCESS_PROVIDER

  package:com.google.android.gm

  label:Access email provider data

  description:Allows the app to access your email database, including received
      messages, sent messages, usernames and passwords.

  protectionLevel:signature

+ permission:android.permission.DOWNLOAD_CACHE_NON_PURGEABLE

  package:com.android.providers.downloads

  label:Reserve space in the download cache

  description:Allows the app to download files to the download cache, which can'
      t be deleted automatically when the download manager needs more space.

  protectionLevel:signature|system
```

```
+ permission:org.simalliance.openmobileapi.SMARTCARD
  package:org.simalliance.openmobileapi.service
  label:SmartcardServicePermission label
  description:null
  protectionLevel:dangerous
+ permission:android.permission.MASTER_CLEAR
  package:android
  label:reset system to factory defaults
  description:Allows the app to completely reset the system to its factory
      settings, erasing all data, configuration and installed apps.
  protectionLevel:signature|system
+ permission:android.permission.FORCE_BACK
  package:android
  label:force app to close
  description:Allows the app to force any activity that is in the foreground to
      close and go back. Should never be needed for normal apps.
  protectionLevel:signature
+ permission:com.google.android.talk.permission.C2D_MESSAGE
  package:com.google.android.talk
  label:null
  description:null
  protectionLevel:signature
+ permission:android.permission.BIND_TRUST_AGENT
  package:android
  label:Bind to a trust agent service
  description:Allows an application to bind to a trust agent service.
  protectionLevel:signature
+ permission:com.google.android.apps.enterprise.dmagent.permission.
    NotificationBroadcastReceiverPermission
  package:com.google.android.apps.enterprise.dmagent
  label:null
  description:null
  protectionLevel:signature|system
+ permission:android.permission.CHANGE_COMPONENT_ENABLED_STATE
  package:android
```

```
  label:enable or disable app components

  description:Allows the app to change whether a component of another app is
      enabled or not. Malicious apps may use this to disable important phone
      capabilities. Care must be taken with this permission, as it is possible
      to get app components into an unusable, inconsistent or unstable state.

  protectionLevel:signature|system

+ permission:com.google.android.marvin.talkback.permission.LABELING

  package:com.google.android.marvin.talkback

  label:Manage TalkBack customised labels

  description:Permission to access, modify and delete customised labels spoken
      by TalkBack.

  protectionLevel:signature

+ permission:com.google.android.apps.now.CURRENT_ACCOUNT_ACCESS

  package:com.google.android.googlequicksearchbox

  label:null

  description:null

  protectionLevel:signature

+ permission:com.android.vending.permission.C2D_MESSAGE

  package:com.android.vending

  label:null

  description:null

  protectionLevel:signature

+ permission:android.permission.TRUST_LISTENER

  package:android

  label:Listen to trust state changes.

  description:Allows an application to listen for changes in trust state.

  protectionLevel:signature

+ permission:android.permission.BROADCAST_CALLLOG_INFO

  package:com.android.server.telecom

  label:Broadcast the call type/duration information

  description:null

  protectionLevel:signature|system

+ permission:android.permission.STATUS_BAR_SERVICE

  package:android

  label:status bar
```

```
  description:Allows the app to be the status bar.

  protectionLevel:signature

+ permission:android.permission.SERIAL_PORT

  package:android

  label:access serial ports

  description:Allows the holder to access serial ports using the SerialManager
      API.

  protectionLevel:signature|system

+ permission:android.permission.READ_INPUT_STATE

  package:android

  label:record what you type and actions that you take

  description:Allows the app to watch the keys that you press even when
      interacting with another app (such as typing a password). Should never be
       needed for normal apps.

  protectionLevel:signature

+ permission:android.permission.BIND_NFC_SERVICE

  package:android

  label:bind to NFC service

  description:Allows the holder to bind to applications that are emulating NFC
      cards. Should never be needed for normal apps.

  protectionLevel:signature

+ permission:android.permission.PACKAGE_VERIFICATION_AGENT

  package:android

  label:verify packages

  description:Allows the app to verify a package is installable.

  protectionLevel:signature|system

+ permission:com.google.android.gms.permission.BIND_NETWORK_TASK_SERVICE

  package:com.google.android.gms

  label:null

  description:Permission that must be required by any client service providing
      an endpoint to the Gcm Network Scheduler

  protectionLevel:signature

+ permission:android.permission.GRANT_REVOKE_PERMISSIONS

  package:android

  label:grant or revoke permissions

  description:Allows an application to grant or revoke specific permissions for
      it or other applications. Malicious applications may use this to access
```

```
    features for which you have not granted them permission.
  protectionLevel:signature
+ permission:com.android.permission.WHITELIST_BLUETOOTH_DEVICE
  package:com.android.bluetooth
  label:Whitelist bluetooth device access.
  description:Allows the app to temporarily whitelist a Bluetooth device,
      allowing that device to send files to this device without user
      confirmation.
  protectionLevel:signature
+ permission:android.permission.BROADCAST_SCORE_NETWORKS
  package:android
  label:send score networks broadcast
  description:Allows the app to broadcast a notification that networks need to
      be scored. Never needed for normal apps.
  protectionLevel:signature|system
+ permission:android.permission.CAPTURE_VIDEO_OUTPUT
  package:android
  label:capture video output
  description:Allows the app to capture and redirect video output.
  protectionLevel:signature|system
+ permission:com.google.android.gms.cloudsave.EVENT_BROADCAST
  package:com.google.android.gms
  label:null
  description:null
  protectionLevel:signature|system
+ permission:com.android.gallery3d.filtershow.permission.READ
  package:com.google.android.apps.plus
  label:null
  description:null
  protectionLevel:signature
+ permission:android.permission.MODIFY_PARENTAL_CONTROLS
  package:android
  label:modify parental controls
  description:Allows the holder to modify the system's parental controls data.
      Should never be needed for normal apps.
  protectionLevel:signature|system
```

```
+ permission:android.permission.MANAGE_NETWORK_POLICY

  package:android

  label:manage network policy

  description:Allows the app to manage network policies and define app-specific
      rules.

  protectionLevel:signature

+ permission:com.google.android.googleapps.permission.GOOGLE_AUTH.doraemon

  package:com.google.android.gsf

  label:Google Catalogs

  description:null

  protectionLevel:normal

+ permission:android.permission.CAPTURE_AUDIO_OUTPUT

  package:android

  label:capture audio output

  description:Allows the app to capture and redirect audio output.

  protectionLevel:signature|system

+ permission:android.permission.INSTALL_PACKAGES

  package:android

  label:directly install apps

  description:Allows the app to install new or updated Android packages.
      Malicious apps may use this to add new apps with arbitrarily powerful
      permissions.

  protectionLevel:signature|system

+ permission:android.permission.INSTALL_LOCATION_PROVIDER

  package:android

  label:permission to install a location provider

  description:Create mock location sources for testing or install a new location
       provider. This allows the app to override the location and/or status
      returned by other location sources such as GPS or location providers.

  protectionLevel:signature|system

+ permission:com.google.android.googlequicksearchbox.permission.C2D_MESSAGE

  package:com.google.android.googlequicksearchbox

  label:null

  description:null

  protectionLevel:signature

+ permission:com.google.android.voicesearch.AUDIO_FILE_ACCESS
```

```
  package:com.google.android.googlequicksearchbox

  label:Recorded audio access

  description:Can access the recorded audio utterances for notes to self and for
      raw audio analysis.

  protectionLevel:signature

+ permission:com.google.android.marvin.feedback.permission.TALKBACK

  package:com.google.android.marvin.talkback

  label:Control TalkBack

  description:Permission to send gestures to TalkBack and resume spoken feedback
      .

  protectionLevel:signature

+ permission:android.permission.ACCESS_DRM_CERTIFICATES

  package:android

  label:access DRM certificates

  description:Allows an application to provision and use DRM certficates. Should
      never be needed for normal apps.

  protectionLevel:signature|system
```