P2P SHARING 2.1: APLICACIÓN ANDROID P2P PARA COMPARTICIÓN DE ARCHIVOS

TRABAJO FIN DE GRADO

CURSO 2019-2020

GRADO EN INGENIERÍA INFORMÁTICA

FACULTAD DE INFORMÁTICA

UNIVERSIDAD COMPLUTENSE DE MADRID

AUTOR

ALEJANDRO MARTÍN RUEDA

DIRECTOR

PABLO RABANAL BASALO

P2P SHARING 2.1: ANDROID P2P APPLICATION FOR FILE SHARING

FINAL DEGREE PROJECT

ACADEMIC YEAR 2019-2020

DEGREE IN COMPUTER ENGINEERING

COMPUTER SCIENCE FACULTY

COMPLUTENSE UNIVERSITY OF MADRID

AUTHOR

ALEJANDRO MARTÍN RUEDA

DIRECTOR

PABLO RABANAL BASALO

Ni tú, ni yo, ni nadie golpea más fuerte que la vida,
pero no importa lo fuerte que golpeas, sino lo fuerte que pueden golpearte,
y lo aguantas mientras avanzas, hay que soportar sin dejar de avanzar,
así es como se gana.

Rocky (Sylvester Stallone)

No importa cuán estrecho sea el camino,
cuán cargado de castigos el viaje…
Soy el amo de mí destino,
soy el capitán de mí alma.

William Ernest Henley,
Invictus.

AGRADECIMIENTOS

Tengo que agradecer a mi tutor Pablo Rabanal su esfuerzo y paciencia a lo largo del proyecto para poder realizarlo.

A mi novia, porque desde que entró en mi vida me ha apoyado en todo, y sin ella y su motivación para continuar, no habría logrado sacar este proyecto.

Por último, dar las gracias y dedicar este proyecto y lo que conlleva, a mis padres, por todo el apoyo que me han dado, durante el desarrollo del trabajo y durante toda la carrera.

RESUMEN

El proyecto P2P Sharing 2.1 es una aplicación cuya finalidad es compartir archivos entre amigos mediante una conexión directa con cada uno ellos. Los objetivos del proyecto consisten en el desarrollo e implementación de nuevas características y funcionalidades para la ampliación y mejora de la aplicación P2P Sharing 2.0 [1], la cual se ha tomado como base para el proyecto.

Los principales puntos que se han desarrollado en el proyecto son la creación de la funcionalidad de grupos, la implementación de varias características de seguridad y la mejora de algunas características de la antigua aplicación.

Un grupo es una lista de amigos entre los cuales poder compartir archivos. De este modo, solo esos amigos que estén en el grupo podrán ver, compartir y descargar los archivos. Para los grupos se ha creado una nueva vista donde se gestionan todos los grupos que tiene un usuario. En esta vista se puede acceder a los archivos compartidos en cada grupo y a los amigos que lo componen. También puede realizarse la gestión de crear nuevos grupos, eliminar o salir de ellos.

Con respecto a la seguridad, se ha implementado un sistema criptográfico para cifrar y firmar el envío de archivos entre un usuario y otro. También se han realizado otras tareas de seguridad como la revisión del código para hacerlo más ininteligible.

Por último, se han realizado otras mejoras como la implementación de los buscadores en amigos y grupos o el uso de nuevos componentes más dinámicos en lugar de otros más anticuados.

Palabras clave

P2P, aplicación, Android, Java, grupos, seguridad, criptografía, compartir, archivos.

ABSTRACT

The P2P Sharing 2.1 project is an application for sharing files with friends through a direct connection with each other. The objectives of the project consist of the development and implementation of new features and functionalities for the extension and improvement of the P2P Sharing 2.0 [1] application, which has been taken as the base for the project.

The main points that have been developed in the project are the creation of group functionality, the implementation of several security features and the upgrade of some features of the old application.

A group is a list of friends among which you can share files. This way, only these friends who are in the group will be able to view, share and download the files. For groups, a new view has been created where all the groups of a user can be managed. In this view, you can access the files shared in each group and the friends in that group. You can also manage to create new groups, delete, or exit them.

With respect to security, a cryptographic system has been implemented to encrypt and sign the files sent between users. Other security tasks have also been performed such as vulnerabilities code analysis to make it more secure.

Finally, other improvements have been made such as the implementation of search tools on friends and groups or the use of new and more dynamic components in place of older ones.

Keywords

P2P, app, Android, Java, groups, security, cryptography, share, files.

ÍNDICE DE CONTENIDOS

ÍNDICE DE FIGURAS

CAPÍTULO 1. Introducción

En este capítulo se va a poner en contexto el trabajo realizado hablando de los antecedentes y motivación que han llevado a realizarlo, junto con los objetivos fijados y el plan seguido para conseguirlos.

1.1 Antecedentes

Las aplicaciones móviles son una herramienta que en la actualidad usamos a diario y sin las cuales nuestra forma de vida no sería como la conocemos. Desde el año 2008, que se creó la primera versión de Android [2], principal sistema operativo de smartphones, ha evolucionado mucho la tecnología móvil y se han refinado mucho las aplicaciones que contenían, realizando cada vez más funciones y con mayor complejidad. Antes de esta fecha, las únicas aplicaciones que había eran las alarmas, calendario, mensajes de texto y algún juego simple que venían ya predeterminados.

En la actualidad, se usan aplicaciones móviles en todos los ámbitos y con casi todas las finalidades imaginables. Una de las principales, son las comunicaciones entre usuarios, ya sea para chatear con amigos, como para compartir archivos y que otro usuario los vea o los pueda descargar. Ejemplos de ello son WhatsApp [3] o Drive [4].

De forma no tan conocida pero incluso antes del éxito de las aplicaciones móviles, ya existían las comunicaciones entre dispositivos, en este caso ordenadores, a través de redes P2P [5], cuya base consiste en no utilizar servidores centrales por donde pase toda la información, sino en que la comunicación entre dos usuarios sea directa, de forma que no esté centralizado todo el tráfico e información. Como ejemplo de programa que usan la tecnología P2P para este fin tenemos Emule [6].

Respecto a la constante evolución y desarrollo tecnológico mundial, no solo de la telefonía móvil, sino de todas las tecnologías existentes, no cabe duda de que proteger toda esa información personal que circula y se utiliza en ella es uno de los principales puntos más importantes para tener en cuenta en las empresas, ya que también lo es para los atacantes que quieren obtener esa información para poder

beneficiarse de forma ilegítima. Es por este motivo por el que la seguridad informática también ha cobrado una gran presencia en los últimos años. Hay multitud de puntos débiles a través de los cuales se puede poner en peligro la comunicación entre los usuarios y por ello es una de las ramas de la informática en la que más se está invirtiendo.

Por todos estos motivos, el usar como base para el proyecto una aplicación Android que comparte archivos entre usuarios a través de una comunicación P2P me pareció una idea muy buena e interesante. En ella se unen las 3 ramas descritas anteriormente: desarrollo en Android, comunicación entre usuarios a través de una comunicación P2P y la posibilidad de implementar técnicas de seguridad para la información entre los diferentes usuarios. De esta forma he tenido la oportunidad de trabajar estas 3 disciplinas en un mismo proyecto, las cuales son muy importantes en la actualidad.

1.2 Background

Apps are tools that we use every day at present, and without our way of life would not be as we know. Since 2008, when the first version of Android [2] was created, the main operating system for smartphones, mobile technology has developed a lot, and the applications it included have been greatly refined, performing more and more functions and with greater complexity. Before this date, the only applications that existed were the alarms, calendar, text messages and some simple games that by default.

Today, the apps are used in all areas and for almost every purpose imaginable. One of the main ones is communication between users, either to chat with friends or to share files with other user. Examples are WhatsApp [3] or Drive [4].

Not so well known but even before the success of these apps, there were already communications between devices, in this case, computers, through P2P networks [5], whose basis is not to use central servers where all information passes through, but to have direct communication between two users so that all traffic, and information is not centralized. An example of a program that uses P2P technology for this purpose is Emule [6].

With respect to the constant evolution, and technological development worldwide, not only of mobile technology, to all existing technologies, it is clear that protecting all that personal information that passes, and is used in it is one of the most important points to consider in enterprises because it is also for attackers who want to get that information to benefit illegitimately. It is for this reason that computer security has also become very important in the last years. There are many vulnerabilities that can compromise communication between users, and this is why it is one of the areas of information technology in which most investment is being made.

For all these reasons, using as a base for the project an Android application that shares files between users through P2P communication, i found it a very good and interesting idea. It joins the 3 branches described before: development in Android, communication between users through a P2P communication, possibility of implementing security techniques for information between different users. This way I had the possibility to work these 3 subjects in the same project, which are very important nowadays.

1.3 Motivación

Este proyecto me pareció una gran oportunidad para poder trabajar en varios aspectos muy interesantes de forma unificada y con un objetivo común, era mejorar una aplicación móvil con diversas características relevantes en la informática.

La principal tarea de trabajar sobre una aplicación ya desarrollada y funcional supone un reto, ya que además de desarrollar las nuevas funcionalidades, tienes que amoldarte a la estructura que tiene. En la universidad, en la mayoría de las asignaturas eres tú mismo el que desarrolla el código desde el inicio con unos objetivos de aprendizaje, pero pensando en el mundo laboral y profesional, normalmente no es así, trabajas sobre desarrollos anteriores.

Por este motivo, creo que es muy interesante desarrollar un proyecto que está precedido por otra aplicación desarrollada por terceros, pudiendo asemejarse a un entorno laboral común.

Todas las empresas tecnológicas deben tener en cuenta las posibles vulnerabilidades de sus productos. Por eso, otro factor muy relevante para el desarrollo de este proyecto fue la importancia de la seguridad. El poder llevar a cabo diferentes puntos de securización de una aplicación móvil es muy interesante por la importancia que tiene en la actualidad, en el día a día de todas las personas que los datos personales que usan estén protegidos.

1.4 Motivation

I think this project was a great opportunity to work on several, and very interesting aspects in a unified way and with a common goal, was to improve an app with different relevant characteristics in computer technology.

The main task of working on an already developed and functional application is a challenge because, in addition to developing new features, you have to adapt to the structure it has. At the university, in most subjects, you are the one who develops the code from the beginning with some learning objectives but thinking about the job, and professional context, usually, it is not like that, you have to work on previous developments.

For this reason, I think it is very interesting to develop a project that is preceded by another application developed by a third fragment, and it can be similar to a common working environment.

All technology enterprises must take into account the possible vulnerabilities of their products. For this reason, another very relevant factor for the development of this project was the importance of security. The possibility of implementing different points of securitization of an app is very exciting because of the importance of protecting the personal data of all the people who use it in their daily tasks.

1.5 Objetivos

El principal objetivo del proyecto es usar de base una aplicación desarrollada por terceros para ampliar su funcionalidad y mejorarla. Para ello es necesario estudiar y entender la aplicación para poder valorar qué aspectos hay que mejorar y qué nuevas funcionalidades pueden añadirse. Esto conlleva el análisis previo de la aplicación, la identificación de los puntos débiles o de mejora para crear nuevas funcionalidades que se puedan llevar a cabo en este proyecto. Para ello se utilizan los puntos de trabajo futuro de la memoria del TFG de la aplicación en la que me baso [7] y el análisis de las funcionalidades que tienen aplicaciones similares en el mercado.

La funcionalidad más importe es la creación de grupos. Esta consiste en poder juntar a varios amigos en un mismo espacio donde compartir archivos y que únicamente lo vean los integrantes del grupo. Además, elaborar una gestión del grupo por parte del administrador para organizarlo.

Otro objetivo es aprovechar estas mejoras, para implementar capas de seguridad para poner en prueba la teoría aprendida a lo largo de la carrera y ampliar conocimientos en esta área. Usar métodos de encriptación y firma de datos para la comunicación y envío de los ficheros entre los usuarios y así añadir más valor y seguridad al objetivo principal de la aplicación sobre la que nos basamos, la conexión P2P. También se podrán desarrollar otras técnicas como el análisis de código para encontrar vulnerabilidades que se puedan explotar de forma maliciosa.

Con respecto a los objetivos personales, los principales son: aprender a programar Android, el lenguaje móvil más usado y extendido, usando gran parte de los conocimientos previos que tenía en Java, ampliar mis conocimientos en seguridad y aprender a ponerlos en práctica sobre una aplicación, y, por último, enfrentarme a una situación similar a la que se puede dar en el ámbito profesional, que es trabajar sobre un desarrollo previo para realizar las diferentes tareas.

1.6 Objetives

The main objective of the project is to use as a base an application developed by third parties to extend its functionality and improve it. To do this, it is necessary to study, and understand the application to be able to evaluate which aspects need to be upgraded and which new functionalities can be introduced. This requires the previous analysis of the application, the identification of the weak points or improvement points to create new functionalities that can be implemented in this project. For this purpose, the future working points of the TFG memory of the application on which I base myself are used [7], and the analysis of the functions that similar applications on the market have.

The most important functionality is the creation of groups. This consists of joining several friends in the same place where they can share files, and they can only be seen by the members of the group. Besides, the administrator can manage the group to organize it.

Another objective is to take advantage of these upgrades, to implement security levels to test the theory learned throughout the degree and improve skills in this area. It will be used data encryption and signature methods for communication and sending files between users and add more value and security to the main objective of the application on which we are based the P2P connection. Other techniques can also be developed such as code analysis to find vulnerabilities that can be exploited maliciously.

Concerning personal goals, the main ones are: to learn how to program Android, the most used, and extended mobile language, using much of the previous skills I had in Java, to increase my security knowledge and learn how to put it into practice on an application, and finally, to face a situation similar to that which can exist in the professional world, which is to work on a previous development to perform the different tasks.

1.7 Plan de trabajo

A continuación, voy a detallar los principales pasos que se han seguido a lo largo del proyecto para desarrollar todo el trabajo:

- Estudio del código desarrollado previamente en la aplicación sobre la que se basa el proyecto. Dado que el lenguaje es Android, y está basado en Java, gran parte del código era fácilmente entendible. He tenido que poner más empeño en conocer las partes más puras de Android y cómo utilizarlas para el desarrollo.
- Selección de las funcionalidades que se van a implementar y priorizar un orden de desarrollo para alcanzar los objetivos del proyecto con éxito.
- Desarrollo de las funcionalidades siguiendo una lógica basada en la metodología Agile [8], consistente en dividir una funcionalidad en pequeñas tareas que puedan ser desarrolladas en un corto plazo de tiempo y que aporten valor a la aplicación de forma incremental hasta llegar a la funcionalidad completa establecida al inicio.

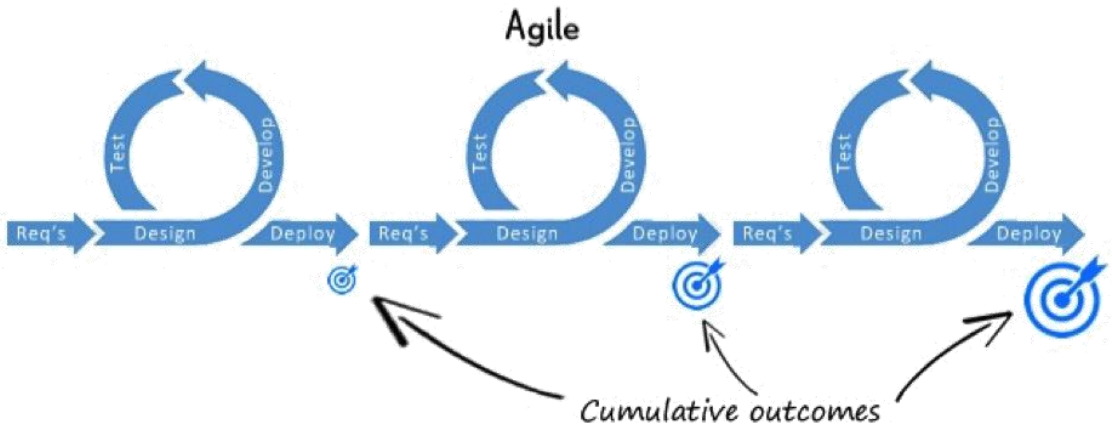

Figura 1-1: Metodología Agile [9]

Para ello se ha seguido el siguiente proceso en todas las funcionalidades que se explican en el capítulo 4 de esta memoria:

  - Establecer las principales tareas en las que se divide la funcionalidad.
  - Desarrollar una de las tareas.
  - Una vez terminada, comprobar su funcionamiento, siempre y cuando sea posible. En caso de necesitar de otras tareas adicionales aún no desarrolladas para comprobar si funciona, se puede depurar hasta ese punto para probar su correcto funcionamiento y seguir avanzando.
  - Una vez probado que funciona correctamente, pasar a la siguiente tarea.
  - Cuando estén completadas todas las tareas, probar la funcionalidad completa. Si está todo correcto, pasar a la siguiente funcionalidad. En caso de encontrar errores, identificar las tareas necesarias para solucionarlos, y volver a realizar los pasos anteriores para resolverlos.
  - Una vez realizadas todas las funcionalidades establecidas para el desarrollo del proyecto, realizar una batería de pruebas diseñada para comprobar los puntos más importantes y críticos de la aplicación y de los nuevos desarrollos. De esta forma, además de comprobar que nuestros desarrollos funcionan correctamente, probamos que ninguna de las funcionalidades anteriores deja de funcionar. Dentro de la metodología ágil, a esta tarea se le llama regresión.

- Realización de la memoria del proyecto explicando todo el proceso de elaboración de la aplicación desde su inicio.
- Preparación y realización de la exposición pública del trabajo realizado en el proyecto.

1.8 Work plan

Next, I will detail the main steps followed during the project to develop all the work:

- Studying the code previously developed in the application on which the project is based. Considering that the language is Android, and is based on Java, much of the code was easy to understand. I had to put more effort into knowing the purest parts of Android, and how to use them for development.
- Selecting the features to be implemented and prioritizing a development order to complete the project's goals.
- Development of the functionalities following a logic based on the Agile methodology [8]; It consist of dividing the functionality into small tasks that can be developed in a short period, and add value to the application in an incremental way until reaching the complete functionality established at the beginning.

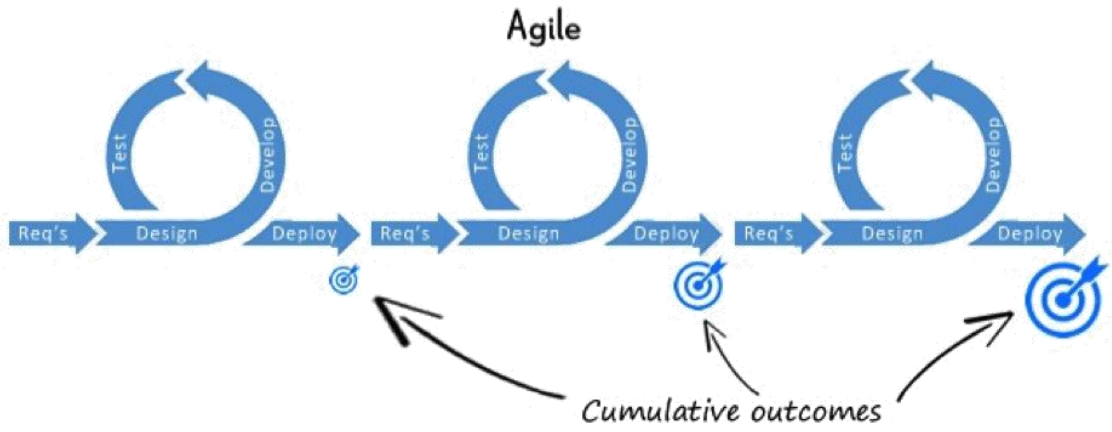

Figura 1-2: Agile Methodology [9]

To do that, the following process has been applied to all the functions explained in chapter 4 of this report:

- Establish the main tasks into which the functionality is divided.
- Develop one of the tasks.

    - Once it is finished, check its operation, if possible. In case you need additional tasks not yet developed to check if it works, you can debug it to that point to test its correct operation, and continue the progress.
    - Once it has been tested to work properly, move on to the next task.
    - When all the tasks are completed, test the full functionality. If everything is correct, move on to the next functionality. If errors are found, identify the tasks needed to fix them, and repeat the previous steps to solve them.
    - Once all the functionality established for the development of the project is completed, perform a battery of tests designed to check the most important and critical points of the application, and new developments. In this way, in addition to checking that our developments work correctly, we prove that none of the previous functionalities stops working. Within the agile methodology, this task is called regression.
- Making the project memory explaining the whole process of development of the application.
- Preparation and realization of the public presentation of the work done in the project.

1.9 Descripción de la memoria

Además de este capítulo introductorio al proyecto, en el cual se explican los antecedentes y motivos para realizarlo, como los objetivos propuestos y el plan para llevarlos a cabo, esta memoria se compone de otros 4 capítulos descritos a continuación.

En el capítulo 2 se exponen las aplicaciones en las que se ha basado gran parte de las ideas para crear las nuevas funcionalidades. Se muestran diferentes ejemplos con diferentes usos, pero con varios aspectos en común entre ellas mismas, como con la

aplicación desarrollada. Se explica brevemente cada una de ellas para tener una idea de su finalidad.

El capítulo 3 trata de poner en contexto el proyecto explicando las diferentes tecnologías que implican las funcionalidades que se han llevado a cabo en este trabajo. Se desarrolla la parte de Android y seguridad, y se explica brevemente las características de la tecnología P2P.

Durante el capítulo 4 se desarrolla todo el proceso y características de las funcionalidades llevadas a cabo para realizar el proyecto. En ellas se explica su lógica y funcionamiento dentro del flujo de la aplicación. Todas estas funcionalidades se han dividido en 3 grandes apartados:

- Grupos
- Seguridad
- Mejoras

El capítulo 5 recoge las conclusiones sacadas después de terminar el proyecto, una discusión crítica sobre los resultados obtenidos y una propuesta de mejoras en el proyecto pensando en un futuro trabajo.

Por último, se encuentra la bibliografía con todas las referencias consultadas y el material para poder realizar el proyecto.

1.10 Memory description

In addition to this introduction to the project, which explains the background and reasons for the project, such as the proposed objectives and the plan for carrying them out, this memory is composed of 4 other chapters described below.

Chapter 2 presents the applications on which many of the ideas for creating the new functionalities have been based. Different examples with different uses are shown, but with several aspects in common among them, as with the developed application. Each one of them is shortly explained in order to have an idea of its purpose.

Chapter 3 tries to put the project in context by explaining the different technologies that involve the functionalities that have been implemented in this work. The Android and security part is developed, and the characteristics of P2P technology are quickly explained.

During chapter 4 the full process and characteristics of the functionalities developed to realize the project are implemented. It explains their logic and operation within the application workflow. All these functionalities have been divided into 3 important sections:

- Groups
- Security
- Improvements

Chapter 5 collects the conclusions extracted after finishing the project, a critical discussion about the obtained results, and a propose of improvements in the project thinking in a future job.

Finally, there is the bibliography with all the consulted references, and the material to be able to carry out the project.

CAPÍTULO 2. Trabajo relacionado

En este capítulo vamos a ver una serie de aplicaciones que nos han servido de algún modo u otro, bien sea por su funcionalidad bien sea por sus características, para asentar una base de ideas y necesidades que podemos usar en nuestro proyecto. De esta lista, se han seleccionado las que se han estimado más convenientes y viables, tanto de desarrollar como de adaptar a la aplicación base, dándole mayor valor y consiguiendo los objetivos del proyecto.

Estas aplicaciones tienen finalidades diferentes entre ellas, pero muchas de sus características son comunes a la aplicación desarrollada en el proyecto. No es viable crear una aplicación con nuevas funcionalidades que no se haya hecho ya antes, porque es casi imposible con la variedad que hay actualmente, pero sí crear una aplicación que uniendo las características de otras similares dé a los usuarios una alternativa más completa o específica dependiendo de su fin.

Figura 2-1: Apps analizadas

2.1 P2PSharing 2.0

Esta primera aplicación [1] es la base del proyecto sobre la que se ha partido para realizar todas las tareas descritas en esta memoria. Se trata de una aplicación que comparte archivos entre amigos, usando una comunicación P2P a través de la unión de las tecnologías de conexión WebRTC [10] y PubNub [11] sin necesidad de servidores centrales donde pase toda la información.

Para ello el funcionamiento es simple, agregas amigos, compartes ficheros o carpetas, y todos tus amigos pueden ver lo que has compartido y descargárselo. Otras

de las características de la aplicación son la capacidad para bloquear usuarios, previsualizar los archivos sin necesidad de descargarlos por completo, ver el estado de la descarga o la opción de seleccionar el uso o no de los datos para la descarga.

P2PSharing 2.0 surge también de un proyecto de TFG basado en una aplicación desarrollada anteriormente, esta aplicación en la cual se basa es P2P Sharing [12].

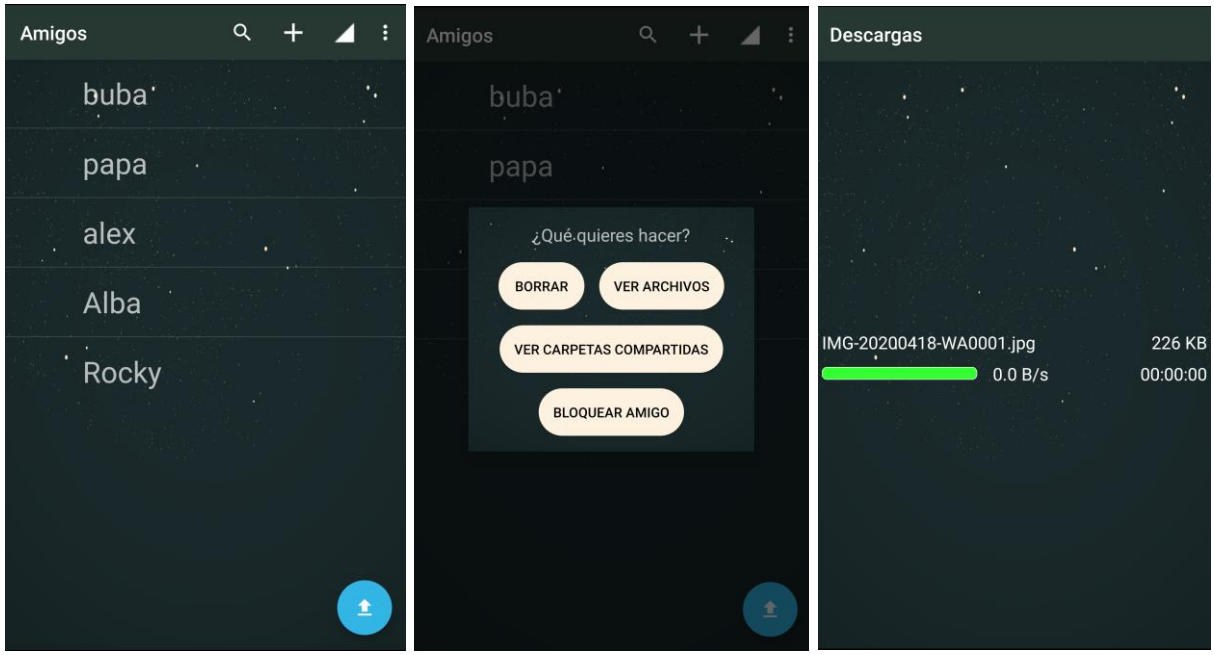

Figura 2-2: Funcionalidades P2P Sharing 2.0

2.2 WhatsApp

Es una de las aplicaciones [13] móviles estrella y que casi ningún usuario de smartphone no tiene instalada en su terminal. Su finalidad principal es muy clara, el envío de mensajes con otros amigos mediante un chat. Pero todas sus funcionalidades son muchísimas más y con el paso del tiempo se van añadiendo nuevas características a la amplia oferta que ya tiene. Entre ellas destacan los grupos de amigos, que se han incorporado a la aplicación siguiendo su idea, las llamadas y videollamadas, la compartición de fotos, archivos, contactos, audios, etc. con los amigos.

Incluso a nivel de seguridad, aunque sigue siendo una aplicación cuya conexión e información está centralizada en servidores propios, han incorporado cifrado entre los nodos, para proteger toda esa información que se intercambia entre los usuarios.

2.3 Google Drive

A diferencia de WhatsApp, que es una aplicación más centrada en el intercambio de mensajes por chat desde el inicio, Google Drive [4] se centra en el almacenamiento en la nube de archivos de forma fácil e intuitiva, con una organización similar a la de un escritorio de ordenador organizado en carpetas y pudiendo acceder a ella desde cualquier dispositivo.

Junto a esta finalidad, otra de sus grandes funcionalidades, y que ha servido de referencia en el proyecto, es la forma de compartir archivos. La forma de compartir archivos con amigos no es tan fácil como por ejemplo enviar un mensaje desde WhatsApp. Aquí es donde P2P Sharing 2.1 unifica la compartición de archivos que tiene Google Drive con el envío a otros amigos de forma fácil de WhatsApp.

2.4 ShareOnWifi

Esta aplicación [14] es la más similar que se puede encontrar a la aplicación del proyecto. Su principal finalidad es el intercambio de archivos mediante una conexión P2P de forma directa entre usuarios. Además, tiene otras funcionalidades en común como compartir carpetas, crear grupos o ver el estado de las descargas.

La principal diferencia de ShareOnWifi con la aplicación desarrollada es la necesidad de estar conectados los amigos a la misma red Wifi para realizar la conexión. Esto tiene el impedimento de no poder compartir archivos con personas que no estén cerca de ti. En este aspecto la aplicación P2P Sharing 2.1 va un paso más allá mejorando esta comunicación, permitiendo establecer la conexión a través de internet.

2.5 RetroShare (Tsunamic Democratic)

Es una plataforma [15] basada en una comunicación descentralizada, libre y de código abierto con una conexión P2P de forma segura con cifrado de extremo a extremo. De esta forma también garantiza un gran anonimato, más allá de los amigos con los que estés conectado [16].

Entre sus funcionalidades hay una gran variedad, como puede ser envío de texto e imágenes por chat, correo electrónico, compartir archivos, foros, etc. Además, es una aplicación multiplataforma, por lo que no solo está en Android, sino que también puedes tenerlo en otros sistemas operativos como Linux y Windows.

Un ejemplo muy conocido donde se usó esta aplicación fue en las protestas en Cataluña contra la sentencia del Tribunal Supremo del referéndum ilegal del 1-O. La plataforma Tsunami Democratic usó, para comunicarse y organizar los movimientos, una aplicación P2P anónima que estaba basada sobre la plataforma RetroShare. De esta forma consiguieron descentralizar la comunicación sin necesidad de servidores, y enviándose la información entre personas de confianza. Para poder acceder a la aplicación usaban códigos QR, que usaban como clave [17] .

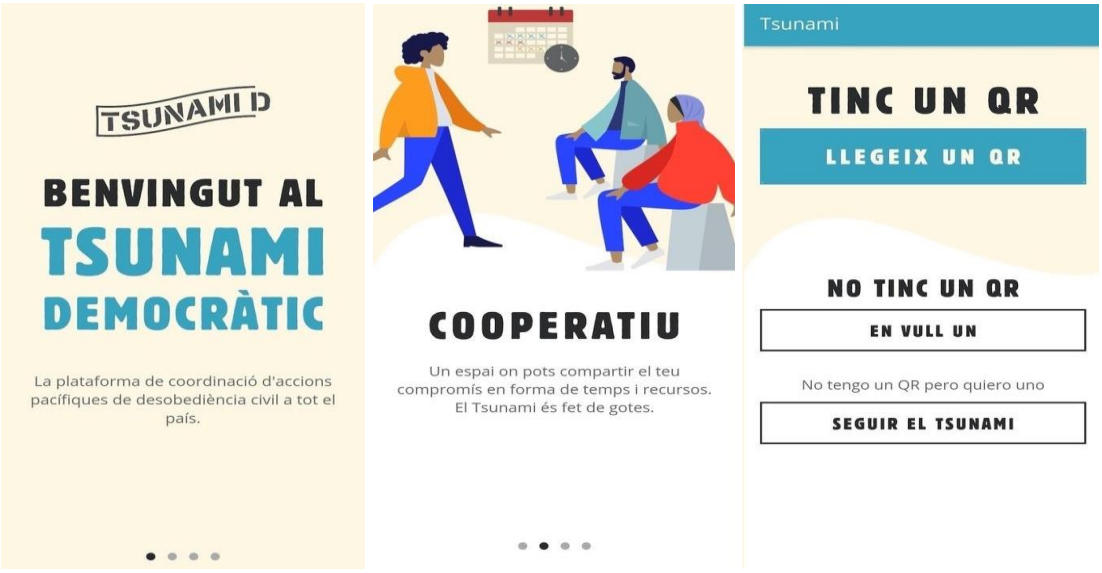

Figura 2-3: App Tsunamic Democratic

CAPÍTULO 3. Tecnologías empleadas

3.1 Redes P2P

"Una red peer-to-peer (P2P, por sus siglas en inglés) es una red de ordenadores en la que todos o algunos aspectos funcionan sin clientes ni servidores fijos, sino una serie de nodos que se comportan como iguales entre sí. Es decir, actúan simultáneamente como clientes y servidores respecto a los demás nodos de la red. Las redes P2P permiten el intercambio directo de información, en cualquier formato, entre los ordenadores interconectados." [18]

Tras esta primera definición, se puede observar la principal ventaja y el por qué se ha usado esta tecnología de comunicación como base de esta aplicación: permite realizar la comunicación directa entre dos dispositivos sin necesidad de atravesar un servidor, en el cual la información pueda ser leída o alterada, manteniendo la privacidad y dando cierta seguridad a esa comunicación.

Estas redes tienen varias finalidades y en cada una de ellas se pueden encontrar ejemplos de aplicaciones bastantes conocidas:

- Compartir música: Napster [19], primera aplicación en usar la tecnología P2P
- Compartir ficheros: Emule [6]
- Telefonía IP: Skype [20]
- Monedas virtuales: Bitcoin [21]
- Sistemas para proporcionar anonimato en la red: Tarzan P2P [22]

Con esta información se pueden entender fácilmente las características propias que debe tener una red P2P:

- Escalabilidad. Como la estructura de una red P2P depende de los nodos conectados a ella, cuantos más nodos, más recursos están disponibles para su funcionamiento. Comparándolo con el tipo de red servidor-cliente, cuantos

más nodos se conecten, más recursos debe tener ese servidor para abastecer a todos, pudiendo saturarse.

- Robustez. Al no estar centralizada toda la información en un servidor, y ser una estructura bastante distribuida, no es fácil que falle la conexión, ya que hay múltiples orígenes, aumentando la robustez de la red.
- Descentralización. Es la base de la red, todos los nodos tienen la misma importancia, y ninguno de ellos es imprescindible para que funcione.

En cuanto a los diferentes tipos de redes P2P se pueden clasificar según el grado de su centralización, clasificándolas en redes centralizadas, descentralizadas o hibridas. También se puede tener en cuenta otros parámetros como la estructura de la red, la generación a la que pertenezcan o el grado de protección de la identidad del usuario.

3.1.1 Tecnologías utilizadas

Para poder establecer y realizar las conexiones entre los diferentes usuarios, hay una gran variedad de tecnologías posibles para conseguir ese fin. Debido a la actual estructura, las que consiguen el objetivo que se quiere para la comunicación son las tecnologías que utilizan protocolos de NAT transversal.

Las dos tecnologías que se detallan a continuación son las que se eligieron en los proyectos anteriores sobre los que se desarrolla el proyecto, tras un análisis de todas las alternativas posibles para realizar la conexión entre pares.

3.1.1.1 WebRTC

"WebRTC, también conocido como Web Real-Time Communications, es un proyecto de código abierto – promovido por Google, Mozilla y otros – que permite comunicaciones en tiempo real sin plug-ins a través de una API Javascript. Facilita las aplicaciones de llamadas de voz, chat de vídeo y compartimiento de archivos entre navegadores." [23]

Este proyecto [10] también está disponible para clientes nativos como pueden ser aplicaciones desarrolladas para Android e IOS, utilizando las bibliotecas diseñadas para este fin con la misma funcionalidad.

Hay muchos casos de usos diferentes para WebRCT, desde aplicaciones web simples que únicamente usan componentes como la cámara o el micrófono, hasta aplicaciones más avanzadas donde se realizan videollamadas, compartición de pantalla o envío de archivos.

El funcionamiento de la comunicación a través de la aplicación de WebRTC se realiza por un flujo de aplicación común. Por este flujo se accederá a los dispositivos de medios, abrir las conexiones entre pares, descubrir los pares y comenzar a transmitir.

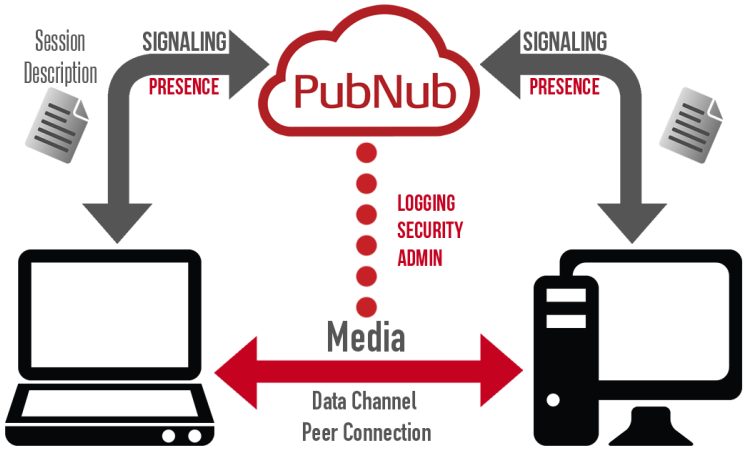

Figura 3-1: Lógica tecnología WebRTC con PubNub [24]

3.1.1.2 PubNub

Pubnub [11] es una compañía que ofrece un servicio de tiempo real a través de su infraestructura. Por esta, se proporciona una red de flujo de datos que se puede aprovechar para la conexión a lo largo de todo el mundo.

Su principal producto es una plataforma que proporciona la infraestructura, red, SDK y herramientas necesarias para crear aplicaciones que se ejecuten en tiempo real. Esta plataforma consiste en una API de publicación/suscripción para el flujo de datos en tiempo real y la señalización de dispositivos pudiendo sincronizarlos para crear una conexión, con una latencia inferior a 100ms en cualquier lugar del mundo. Es

multiplataforma y dispone de SDKs en diferentes lenguajes de programación y entornos, como por ejemplo el utilizado en el proyecto para Android entre otros.

Los componentes de este flujo son las claves de la API, los mensajes que se envían y los canales utilizados. Las claves que se utilizan son 2: una para la publicación o envío de mensajes, y otra para la suscripción o recibo de mensajes. Un cliente, al poder enviar y recibir mensajes puede tener ambas claves. Todos los mensajes que se envían se realizan a través de canales mediante el formato de datos JSON, pudiendo incluir en ellos diferentes formatos que son serializados.

Los canales se crean cuando se publica un mensaje, y para recibirlo es necesario suscribirse a ese mismo canal. Pueden tener diferentes diseños (Figura 3-2):

- Unicast, donde el canal es único para cada cliente en la comunicación uno a uno. Este diseño tiene la ventaja de un uso mínimo de la red, ya que solo recibe mensajes que sean necesarios y no necesita filtrarlos.
- Multicast, donde se utiliza un canal público para las comunicaciones globales entre todos los clientes que hay en la red. En este tipo es necesario realizar un filtrado de los mensajes para evitar recibir mensajes innecesarios.

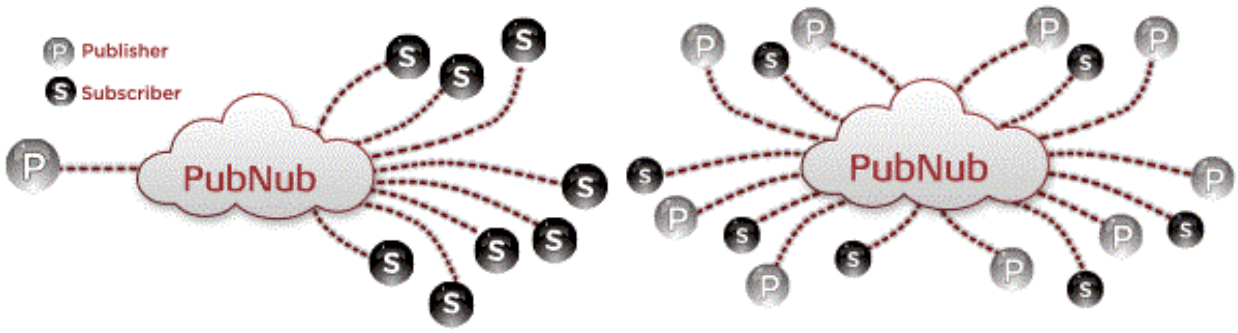

Figura 3-2: PubNub Unicast (a) y Multicast (b)

3.2 Android

Android [2] es un sistema operativo móvil basado en el kernel de Linux, un núcleo de sistema operativo libre, gratuito y multiplataforma. El sistema operativo proporciona las interfaces necesarias para desarrollar aplicaciones que usen las funciones del teléfono, como la cámara, el bluetooth, GPS o los contactos, de una forma fácil en un lenguaje de programación conocido como es Java, generando una gran cantidad de aplicaciones disponibles para todas las finalidades y experiencia del usuario con el dispositivo.

Android Inc. fue la empresa creadora de este sistema operativo, pero hasta el año 2005 que la compró Google no se dio a conocer a nivel mundial. Dos años después, en 2007, se creó la Open Handset Alliance, una agrupación de fabricantes de teléfonos móviles, hardware y software, más Google. La finalidad de esta agrupación era la de promover y avanzar en los estándares abiertos para los dispositivos móviles [25].

En septiembre de 2008 se lanzó la primera versión: Android 1.0 Apple Pie. Junto a ella se proporcionó el SDK necesario para que los programadores creasen sus aplicaciones para este sistema. Esta primera versión era muy simple, y contenía las aplicaciones de Google del sistema, Android Market, patrón de desbloqueo y aviso por batería baja.

Desde ese primer lanzamiento, hasta la última versión lanzada el 3 de septiembre de 2019, Android 10, se han sucedido 17 versiones a lo largo de más de 10 años. En cada versión se han ido añadiendo nuevas funcionalidades y mejorando y optimizando las ya existentes. Todo esto ha sido posible gracias al gran auge de la telefonía móvil en estos años, al gran uso por parte de todas las personas y a los grandes avances de la tecnología.

Android está formado por una estructura de diferente software de código abierto basado en Linux, con el fin de poder ser utilizado por una gran variedad de dispositivos y con diferentes formas. Con esta idea se consigue una gran compatibilidad, siendo posible llegar al mayor número de usuarios [26].

En la Figura 3-3 se observan los componentes principales de cada bloque de la plataforma Android.

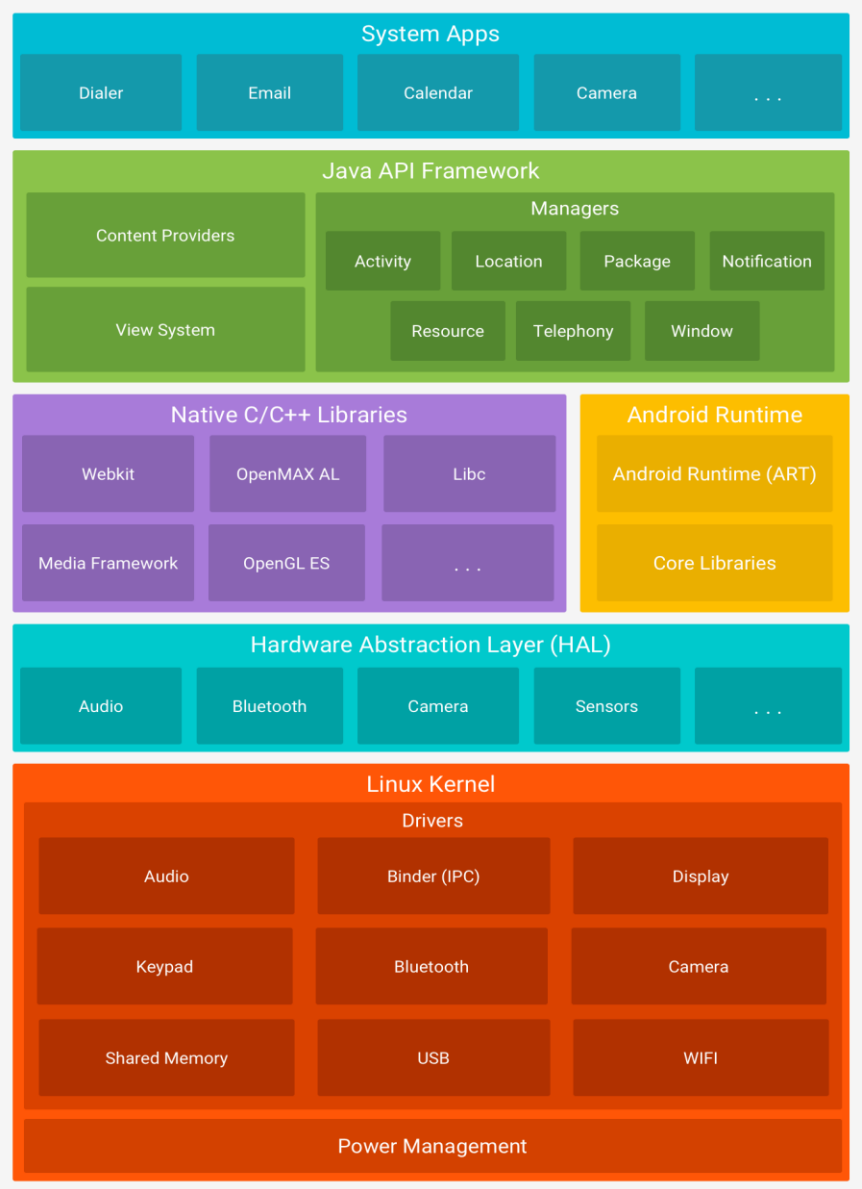

Figura 3-3: Arquitectura de Android

3.3 Técnicas de Seguridad

La seguridad en el área de la informática es uno de los objetivos más importantes de las empresas tecnológicas en los últimos años, debido a la evolución constante de la tecnología y su gran uso a nivel global, como a la importancia de los datos e información que se utiliza en ella [27]. Esta información puede ser muy valiosa y resultar peligrosa si cae en manos que no debe. En el caso de una aplicación móvil, como es el caso, cuya base se sustenta en la compartición de archivos por la red, estos datos e información son su activo más valioso, y donde hay que focalizar la seguridad.

Dentro de la seguridad podemos diferencias dos grandes tipos, que son difícil de diferenciar entre ellos, ya que ambos trabajan en armonía y de forma conjunta, son: la seguridad informática y la seguridad de la información.

"La seguridad informática, también conocida como ciberseguridad o seguridad de tecnología de la información, es el área relacionada con la informática y la telemática que se enfoca en la protección de la infraestructura computacional y todo lo relacionado con esta y, especialmente, la información contenida en una computadora o circulante a través de las redes de computadoras." [28]

"La seguridad de la información es el conjunto de medidas preventivas y reactivas de las organizaciones y sistemas tecnológicos que permiten resguardar y proteger la información buscando mantener la confidencialidad, la disponibilidad e integridad de datos." [29]

Como podemos observar en la Figura 3-4, y en las definiciones anteriores la seguridad de la información se centra en englobar un conjunto de técnicas y medidas para controlar los datos que se manejan dentro de una institución y asegurar que no se corrompan, es decir en la estrategia a seguir para asegurar la información. Por otro lado, la seguridad informática se enfoca en la protección de la infraestructura y todo lo relacionado con ella, como la información que contiene o cómo circula a través de la red. Por eso la seguridad informática se relaciona con la parte práctica de defender los ordenadores y servidores, los sistemas electrónicos y las redes.

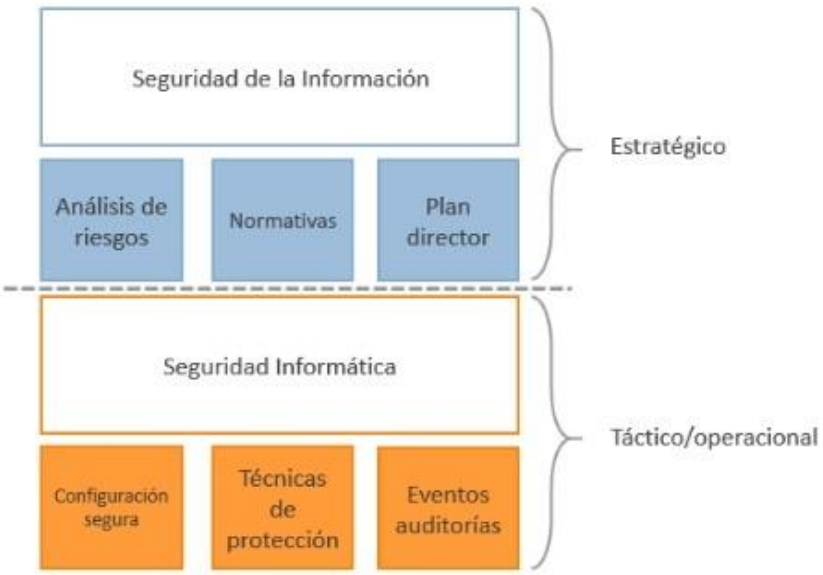

Figura 3-4: Seguridad Informática vs Seguridad de la Información

Indistintamente del término que se use, todos tienen una misma finalidad que es la de proteger los datos e información. Para ello es necesario asegurar tres aspectos fundamentales, conocidos como tríada CIA (por sus siglas en inglés): confidencialidad, integridad y disponibilidad. [30]

Figura 3-5: Principios de la seguridad

La confidencialidad es el principio encargado de prevenir la divulgación de los datos a personas que no están autorizadas a acceder a ellos, impidiendo la divulgación de información sensible y manteniéndolos privados controlando el acceso a ellos.

El principio de la integridad tiene como función garantizar que el contenido de la información debe permanecer inalterado, a no ser que sea modificado por alguien que esté autorizado. Mantener la integridad es importante ya que un fallo en ella puede generar modificaciones en los datos, sin conocimiento de ellos.

La disponibilidad es el tercer principio fundamental, basado en que los datos estén siempre disponibles para que los usuarios autorizados puedan acceder a ellos cuando deseen. Este principio se puede romper con uno de los ataques más conocidos como la denegación de servicio.

Adicionalmente, hay un cuarto principio, estrechamente relacionado con la integridad, que es la autenticación. Esta consiste en la identificación del creador de la información, asegurando que la información recibida es de la persona que lo ha mandado, y no una tercera suplantando su identidad.

Figura 3-6: Resumen áreas principales

3.3.1 Criptografía

Este apartado está dedicado a toda la parte de técnicas, herramientas y pasos necesarios para poder alcanzar 3 de los principios de la seguridad que se detallaban en el punto anterior: confidencialidad, integridad y autenticación.

Para poder avanzar más es necesario explicar algunos de los términos más importantes para poder entender y tener claro todo:

- **Criptología**. "La criptología es la disciplina que se dedica al estudio de la escritura secreta, es decir, estudia los mensajes que, procesados de cierta manera, se convierten en difíciles o imposibles de leer por entidades no autorizadas." [31]
- **Criptografía**. "La criptografía se ha definido, tradicionalmente, como el ámbito de la criptología que se ocupa de las técnicas de cifrado o codificado destinadas a alterar las representaciones lingüísticas de ciertos mensajes con el fin de hacerlos ininteligibles a receptores no autorizados." [32]

  Esta definición es la más genérica de la criptografía, pero si se centra en el área de la informática la criptografía es la encargada del estudio de los algoritmos y protocolos criptográficos y sistemas necesarios para proteger la información, dar seguridad a las comunicaciones y a las empresas o usuarios que se conectan entre ellos.
- **Cifrado**. Cifrar en criptografía es la acción de modificar un mensaje haciendo que su contenido sea ilegible para toda aquella persona que sean la receptora del mensaje. Para realizar el cifrado es necesario utilizar un algoritmo de cifrado con una clave, que juntos transforman el mensaje a su estado ilegible.
- **Algoritmo**. Es el conjunto de instrucciones y reglas que hay que seguir para poder transformar el mensaje original a cifrar, en el mensaje cifrado, mediante la realización de cómputos y proceso de datos.
- **Clave**. Es la parte fundamental que controla la acción de las operaciones del algoritmo de cifrado utilizado. La información que contiene la clave es la necesaria para especificar la transformación del mensaje original.

Teniendo claros estos conceptos, podemos diferenciar dos tipos de criptografía, en función del tipo de clave que se usen para los algoritmos. Cada tipo de criptografía tiene sus propias finalidades y algoritmos propios que se pueden utilizar para su función.

3.3.1.1 Criptografía Simétrica

Este tipo de criptografía usa una clave única entre el emisor y el receptor del mensaje, por lo tanto, esa misma clave es la encargada de cifrar como de descifrar la información. Tiene la ventaja de ser rápida en establecer y enviar los mensajes [33].

El principal inconveniente es que de alguna forma es necesario que ambos usuarios en la comunicación conozcan o estén en posesión de la clave, esto implica que tienen que enviársela por algún medio para realizar el cifrado, como puede verse en la Figura 3-7, y puede interceptarse esa comunicación de la clave, poniendo en peligro la privacidad de la información, al no necesitarse romper el mensaje cifrado.

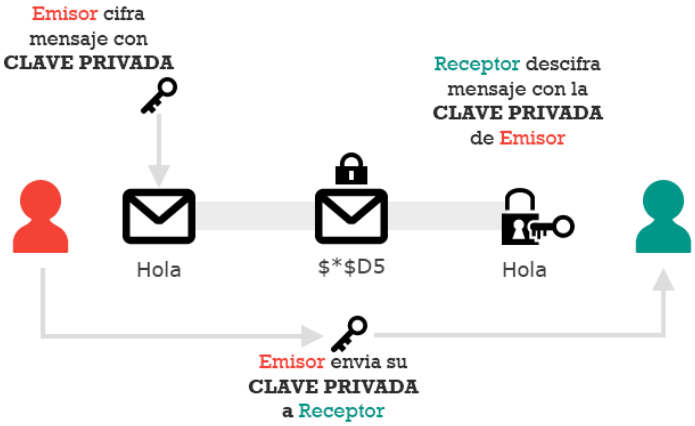

Figura 3-7: Criptografía simétrica [34]

Entre los principales algoritmos [35] que se pueden usar para el cifrado con clave simétrica se encuentran:

- **DES**: Es un algoritmo actualmente inseguro ya que es fácil de descifrar, aunque tuvo una gran aceptación cunado se generó, ya que llegó a ser escogido como un estándar por los Estados Unidos. Codifica en bloques de 64 bits, demasiados pequeños y tiene una clave corta, vulnerable ante fuerza bruta.
- **AES**: Es un cifrado estándar que soporta bloques y usa claves con longitud variable, además combina seguridad, eficiencia, velocidad, sencillez y flexibilidad.

- **BLOWFISH**: tiene una estructura sencilla, siendo fácil de utilizar, la longitud de la clave es variable y puede ser de gran tamaño, pudiendo elegir entre la velocidad y la seguridad del sistema.
- **RC5**: La clave tiene longitud variable, permitiendo variar entre seguridad y velocidad. Incorpora rotaciones circulares de bits que dependen de los datos introducidos, haciéndolo muy robusto.

3.3.1.2 Criptografía Asimétrica

La criptografía asimétrica emplea dos claves matemáticamente relacionadas entre sí: la clave pública, usada para cifrar la información por parte del emisor, y la clave privada, la encargada para descifrar la información por el receptor.

El principal problema es que son poco eficientes, ya que el tiempo que requiere en cifrar es bastante, debido a la longitud de las claves. También otro de los inconvenientes es que hay que guardar y proteger la clave privada, ya que si alguien la consigue ya no serían seguros los mensajes.

El proceso necesario para el cifrado en este tipo de criptografía es, como se muestra en la figura 3-8, el siguiente:

- Generación de par de claves por el receptor.
- Envío de la clave pública para cifrar al emisor.
- Cifrado del mensaje por el emisor con la clave pública.
- Envío del mensaje cifrado del emisor al receptor.
- Descifrado del mensaje con la clave privada por el receptor.

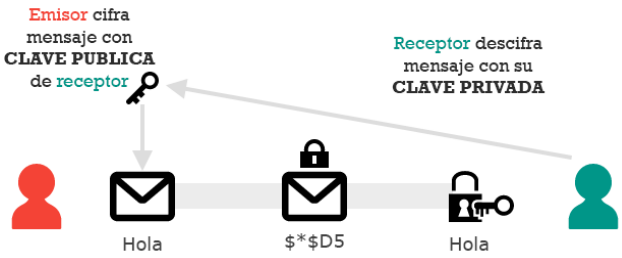

Figura 3-8: Criptografía Asimétrica [34]

Este método tiene la principal ventaja de que no es necesario pasar la clave publica de forma segura como en la simétrica, porque, aunque la intercepten no van a poder descifrar el mensaje, pues requiere de la clave privada, que solo tiene el receptor.

Dentro de la criptografía asimétrica [36] se encuentra la función de firma digital, la cual complementa al cifrado. La firma tiene la utilidad de autenticar al emisor mientras que el cifrado garantiza la confidencialidad de la comunicación. Ambas se complementan. El firmar un mensaje garantiza que el emisor de la información es quien dice ser generando una firma mediante una función hash a partir de la información original del mensaje que se envía.

Los pasos que hay que seguir para realizar la firma son los siguientes:

- Firmar el mensaje con la clave privada del emisor para obtener la firma digital.
- Envío del mensaje y de la firma al receptor.
- El receptor recibe el mensaje y la firma.
- Usa la clave pública del emisor y la firma para comprobar que el mensaje es correcto, y pertenece al emisor.

Como hemos comentado antes, ambas técnicas son compatibles, y se recomienda usarlas, de esta forma se consigue mayor seguridad, cubriendo la confidencialidad de la comunicación y la autentificación del emisor. Para ello hay que realizar una unión de cifrado del mensaje y firma de este (Figura 3-9). Los pasos que hay que realizar son:

- Cifrado del mensaje con la clave pública del receptor por el emisor.
- Firma del mensaje ya cifrado con la clave pública del emisor.
- Envío del mensaje cifrado y de la firma digital al receptor.
- El receptor recibe el mensaje cifrado y la firma digital.
- Usa la clave pública del emisor y la firma para comprobar que el mensaje, que está cifrado, es correcto y del emisor.
- Descifrado del mensaje con la clave privada del receptor.

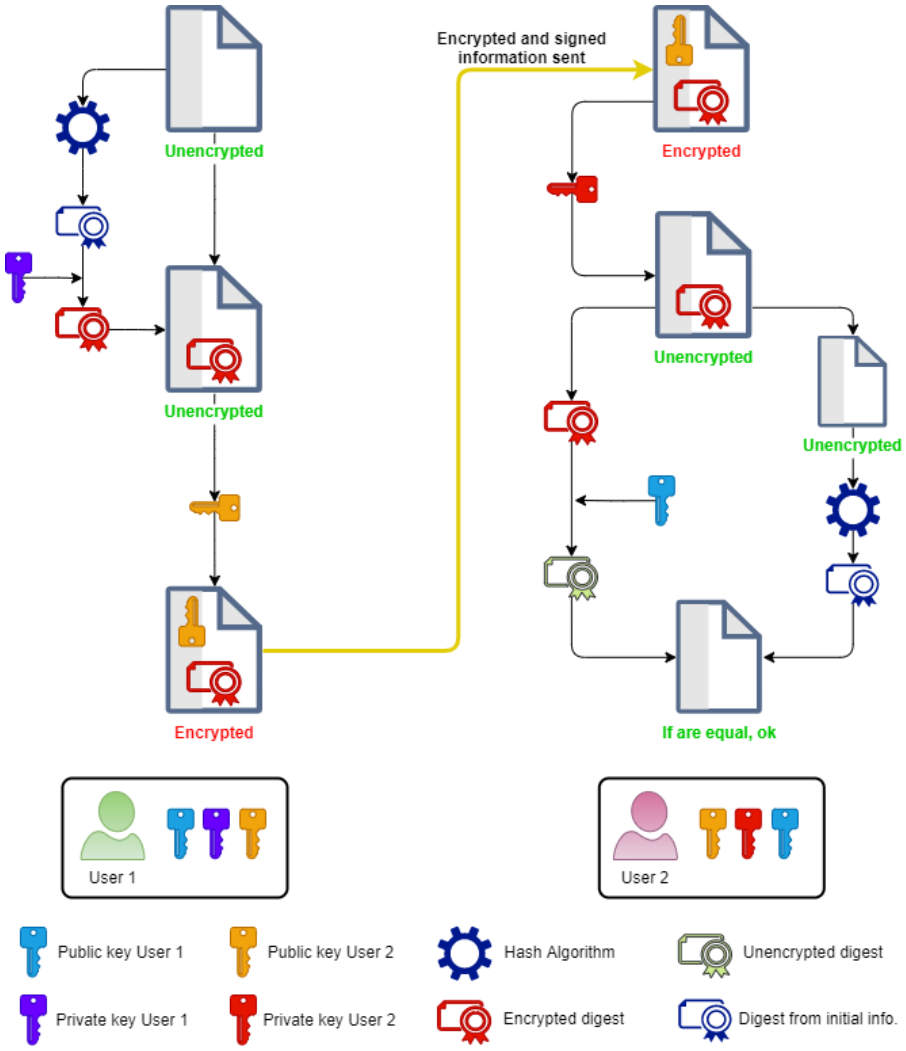

Figura 3-9: Cifrado y firma asimétrica [36]

Con respecto a los algoritmos que se pueden usar en la criptografía asimétrica, el más conocido es el protocolo RSA, que más adelante se detalla. Aunque hay una gran variedad de algoritmos, tecnologías e incluso protocolos que se basan en el cifrado y la firma asimétrica para sus comunicaciones, que son bastante conocidos por los usuarios como: Diffie-Hellman, criptografía de curva elíptica, GPG, SSH o SSL.

RSA [37] es el algoritmo más conocido y representativo de la criptografía de clave asimétrica. Fue desarrollado por Rivest, Shamir y Adleman, cuyas iniciales dan nombre al algoritmo en el MIT. El algoritmo se basa en la multiplicación de números primos muy grandes y la seguridad de este radica en la dificultad de encontrar los factores primos de números tan grandes. Estas son las 2 grandes características del sistema, la facilidad de hallar 2 números primos grandes y la dificultad de factorizar sus productos.

CAPÍTULO 4. Desarrollo del proyecto

En este capítulo se explican todas las funciones realizadas en el proyecto en profundidad, como los detalles técnicos que se han empleado y su utilidad. Estos desarrollos son obtenidos de los objetivos que se pusieron al principio del proyecto, y se han ido realizando de forma secuencial, para ir añadiendo nuevas funciones a la aplicación de forma incremental, dándole mayor valor con cada nueva característica incorporada.

Durante los siguientes apartados se explica en qué consiste la funcionalidad desarrollada, qué valor aporta a la aplicación o qué se mejora de ella. También se detalla el funcionamiento de esta y qué flujo hay que realizar para poder realizar esa funcionalidad y los pasos a seguir.

El capítulo lo dividimos en dos grandes apartados correspondientes a las 2 nuevas funciones incorporadas en la aplicación: creación de grupos e implementación de seguridad. Además, hay un tercer apartado donde explicaremos otras mejoras también hechas a lo largo del proyecto que se suman a las anteriores, pero cuyo impacto es menor y por ello se agrupan en uno.

El desarrollo de la aplicación se ha realizado a través del uso de 2 aplicaciones: GitKraken [38], interfaz gráfica para gestionar el control de versiones de los desarrollos e ir actualizando todos los cambios en Github [39]; Android Studio, entorno de desarrollo integrado (IDE) oficial de Android [40] con todas las características necesarias para poder trabajar en el desarrollo de todos los aspectos de la aplicación de forma unificada.

Todo el código de la aplicación se puede encontrar en la página de Github: https://github.com/alexmartin3/P2PSharing.2.1

4.1 Grupos

Esta característica es la más básica que se puede imaginar que tenga una aplicación en la que una de las principales características es tener amigos. Seguramente tengas amigos que no lo sean de tus otros amigos de la aplicación; por eso, pensando en compartir un archivo con ellos, no siempre quieres o tienes la necesidad de que todos los amigos que tienes en la aplicación puedan descargarse ese archivo que compartes.

Esta funcionalidad sirve para poder agrupar o unir con quién compartes cosas, creas un conjunto de amigos separado del resto, con el cual puedes compartir uno o varios archivos diferentes, y que solo sean vistos por los miembros. Además, también ellos pueden compartir archivos en ese grupo, eliminarlos o incluso añadir o eliminar amigos si tuvieran los permisos necesarios de administrador.

Dentro de esta funcionalidad, se establecen 2 partes: la primera es la encargada de crear y eliminar el grupo y la segunda es la que gestiona la interacción con un grupo ya creado en el apartado de amigos y archivos.

Para poder acceder a los grupos desde la pantalla principal de la aplicación, como se ve en la Figura 4-1a, se necesita pinchar en el icono "grupos" de la parte inferior izquierda, el cual lleva a una nueva vista con todos los grupos a los que pertenecemos.

En esta vista (Figura 4-1b) podemos observar las siguientes partes:

- **Barra de herramientas**. Muestra el nombre del usuario y la vista en la que se encuentra. En ella aparece la lupa para poder buscar entre los grupos.
- **Lista de grupos**. Todos los grupos a los que el usuario pertenece. Sobre cada ítem de esta lista se pueden realizar dos acciones:
    - Pulsar: Aparece un diálogo con las opciones que se pueden realizar para ese grupo: "Ver archivos" y "Ver amigos".
    - Mantener pulsado: Da la opción de borrar el grupo si el usuario es el administrador de él o salir del grupo si solo es un miembro.
- **Botón "Atrás".** Sirve para volver a la vista principal de la aplicación.

- **Botón "Crear Grupo".** Se inicia el proceso para crear un nuevo grupo.

Figura 4-1: Vista Profile (a) y vista grupos (b)

4.1.1 Crear y borrar un grupo

Estando en la vista de grupos, podemos gestionar la creación de nuevos grupos mediante el botón de "Crear Grupo" (Figura 4-2). Una vez se pulse sobre él aparece un diálogo para introducir el nombre que tendrá el grupo, el cual no podrá estar vacío, ni repetirse con otro grupo que ya tenga el usuario en su aplicación.

Después, pulsamos el botón "Añadir amigos" para pasar a la siguiente pantalla donde se mostrará la lista de amigos que tenemos. En ella se seleccionarán aquellos amigos que queremos añadir, resaltando los que pulsemos, y una vez elegidos, se pulsa el botón que hay en la esquina inferior derecha con un "check" para confirmar la creación del nuevo grupo. Automáticamente la aplicación volverá a la vista de grupos y mostrará la lista actualizada, y a su vez, comunicará a todos los amigos que se hayan añadido al grupo, la creación de este mismo con los integrantes de él.

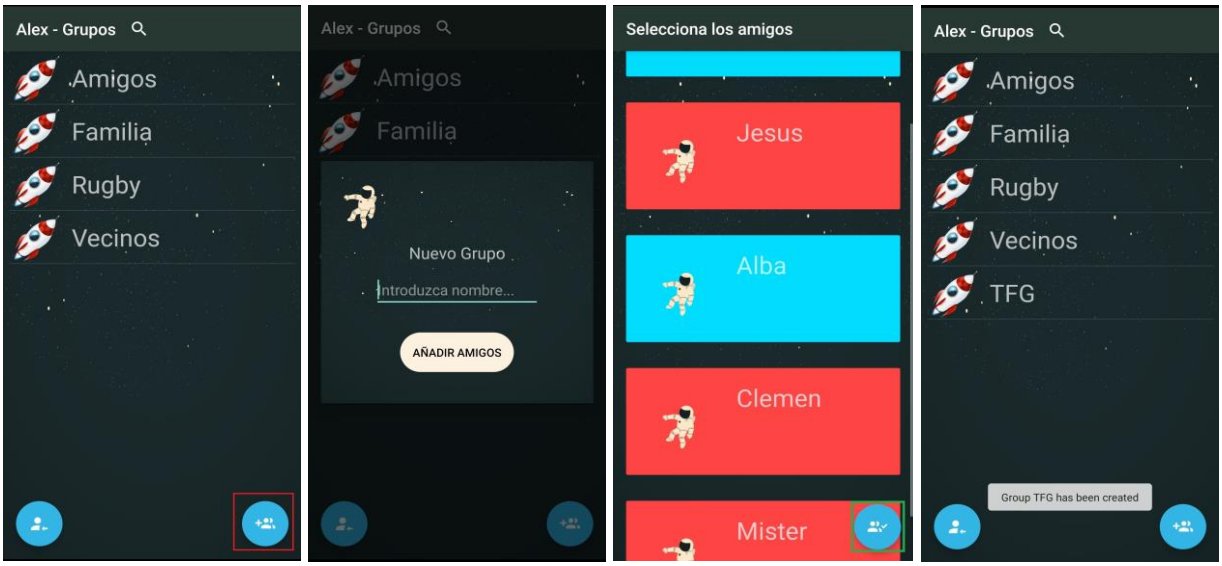

Figura 4-2: Crear grupo

Para eliminar un grupo el proceso es muy fácil, simplemente hay que mantener pulsado el grupo que se quiera eliminar, hasta que aparezca un diálogo para confirmarlo pulsando "Sí"; en caso contrario no se realiza ninguna acción.

En este diálogo el mensaje que se muestra es diferente si el usuario es el administrador del grupo o solo es un miembro de él, pues la acción es diferente:

- Si es administrador y borra el grupo, se elimina tanto del usuario como de todos los amigos que pertenecen al mismo.
- Si no es administrador, solo se le borra el grupo al usuario, permaneciendo para el resto de los integrantes. En este caso la lista de amigos del grupo se actualiza en todos, borrando al usuario de ella.

Por este motivo el mensaje del diálogo es diferente como se puede observar en la Figura 4-3 para el administrador y para el resto de los amigos.

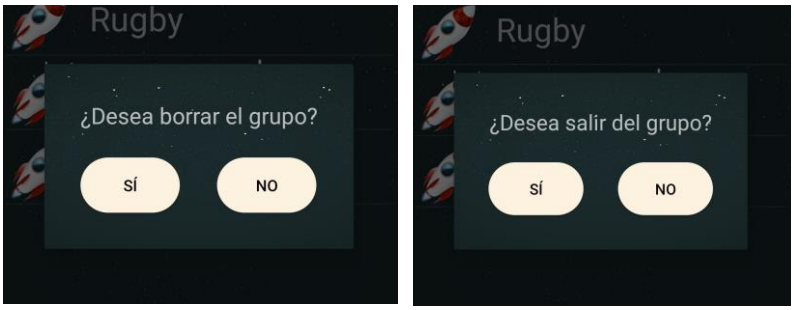

Figura 4-3: Mensaje eliminar grupo (a) admin, (b) amigos

A continuación, se va a mostrar el flujo de las 2 acciones explicadas anteriormente, crear un nuevo grupo, y eliminar o salir de un grupo al que se pertenece.

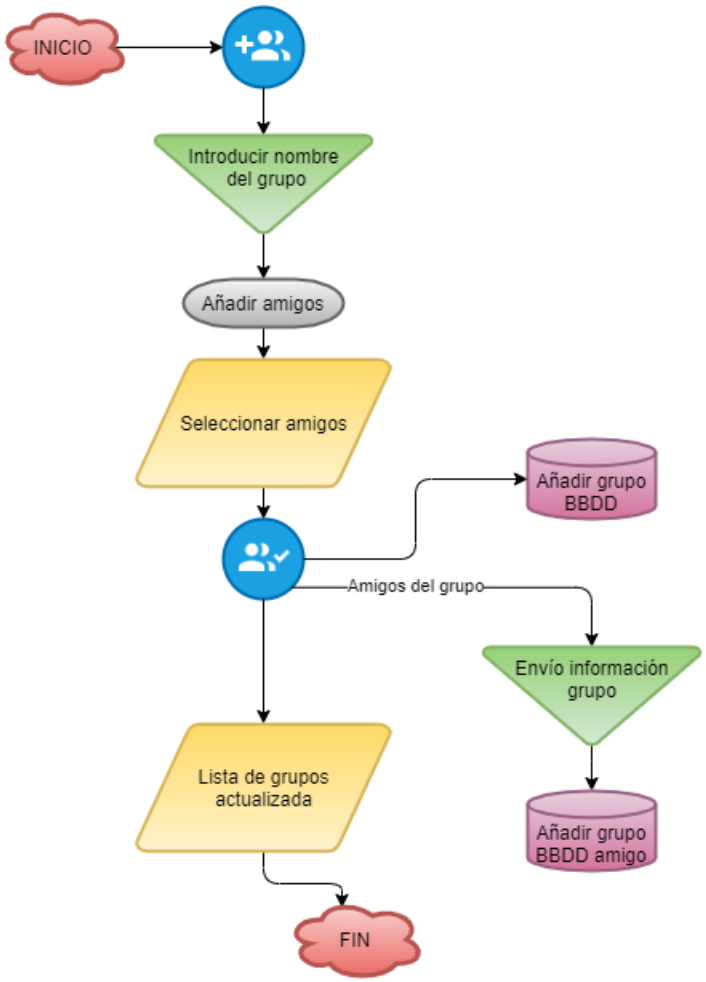

Figura 4-4: Flujo crear nuevo grupo

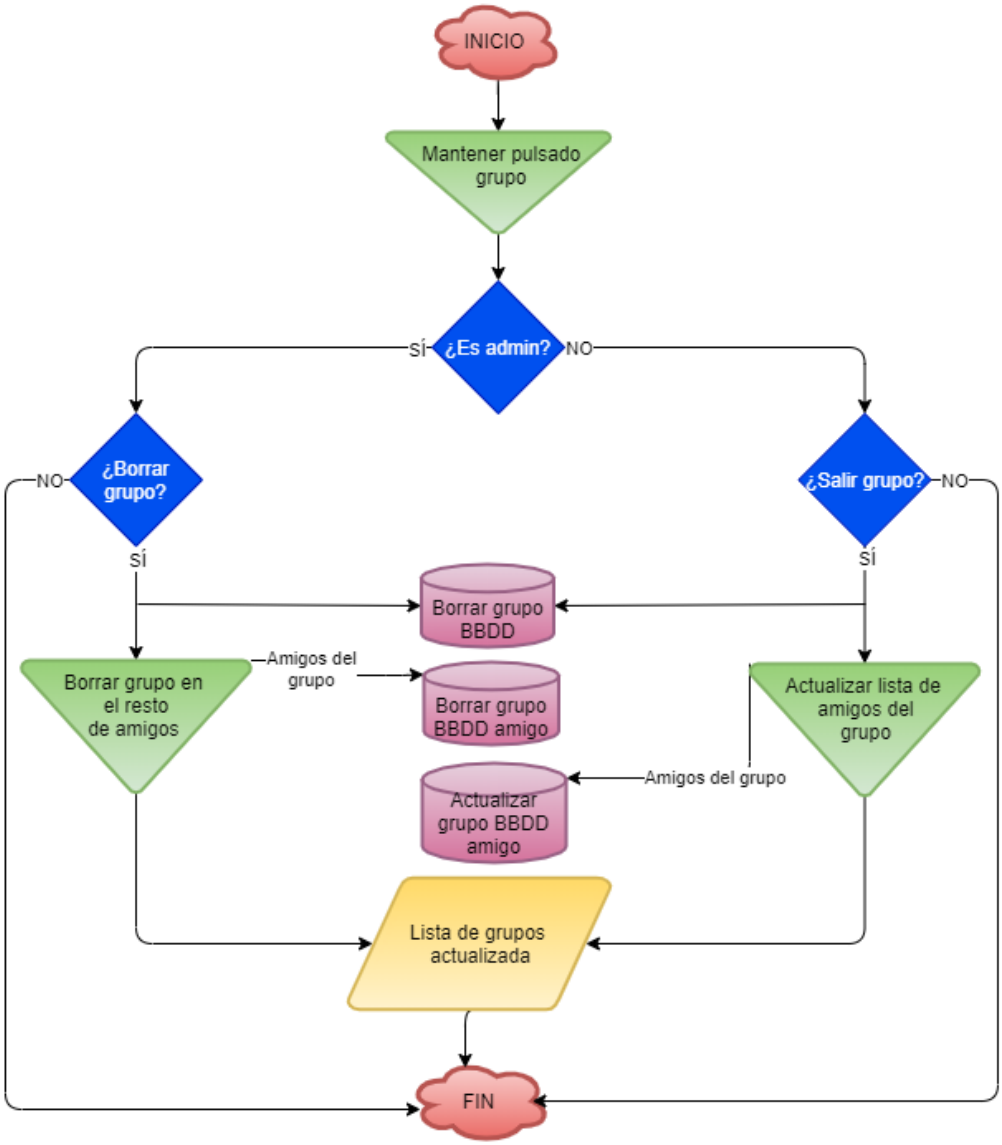

Figura 4-5: Flujo eliminar o salir de grupo

4.1.2 Administración de archivos

Para acceder a la vista de archivos es necesario seleccionar un grupo, y en el diálogo que aparece, elegir la opción "Ver archivos" para que lleve a la vista de los archivos de un grupo (Figura 4-6). Esta vista está formada por los siguientes elementos:

- **Barra de herramientas**. Muestra el nombre del grupo y la vista en la que se encuentra. En ella aparece a la derecha el icono de añadir archivos.
- **Lista de archivos**. Se muestran todos los ficheros que se han compartido por los integrantes del grupo. Sobre cada ítem de la lista se puede realizar una única acción, que depende de si el usuario es el dueño del archivo o no:
    - Si es el dueño. Se muestra un diálogo que da la opción de borrar el archivo si se confirma pulsando "Sí".
    - Si no es el dueño. Se muestra un diálogo para realizar la descarga del archivo seleccionado. La descarga tiene la opción de previsualización.
- **Botón "Atrás".** Sirve para volver a la vista de grupos. En caso de haber cambios en la lista y no se haya guardado, pide una confirmación.
- **Botón "Guardar".** Se encarga de guardar los cambios realizados sobre la lista de ficheros. Únicamente se activa si se realizan dichos cambios.

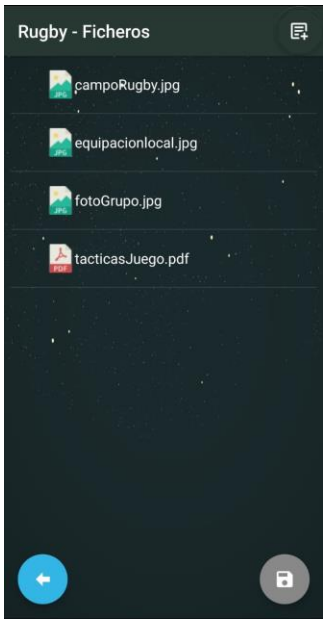

Figura 4-6: Vista de archivos de grupo

Cualquier usuario puede añadir un archivo al grupo. Para ello tiene que pulsar el botón "Añadir archivo" y en la nueva vista con el directorio de su móvil seleccionar un archivo y confirmar el diálogo que se muestra. Una vez se confirma, se vuelve a la lista de ficheros actualizados con el recién añadido. A su vez, el botón "Guardar" se ha activado en color rojo, ya que es necesario guardar el estado del grupo para confirmar, y subir el archivo al grupo. En caso de no pulsar el botón, los cambios desaparecerán. Al pulsar el botón "guardar" automáticamente se vuelve a la vista de grupos y se actualiza en la base de datos la lista de ficheros con los cambios realizados. También, se comunica a los amigos del grupo estos cambios, guardándolos en sus respectivas bases de datos.

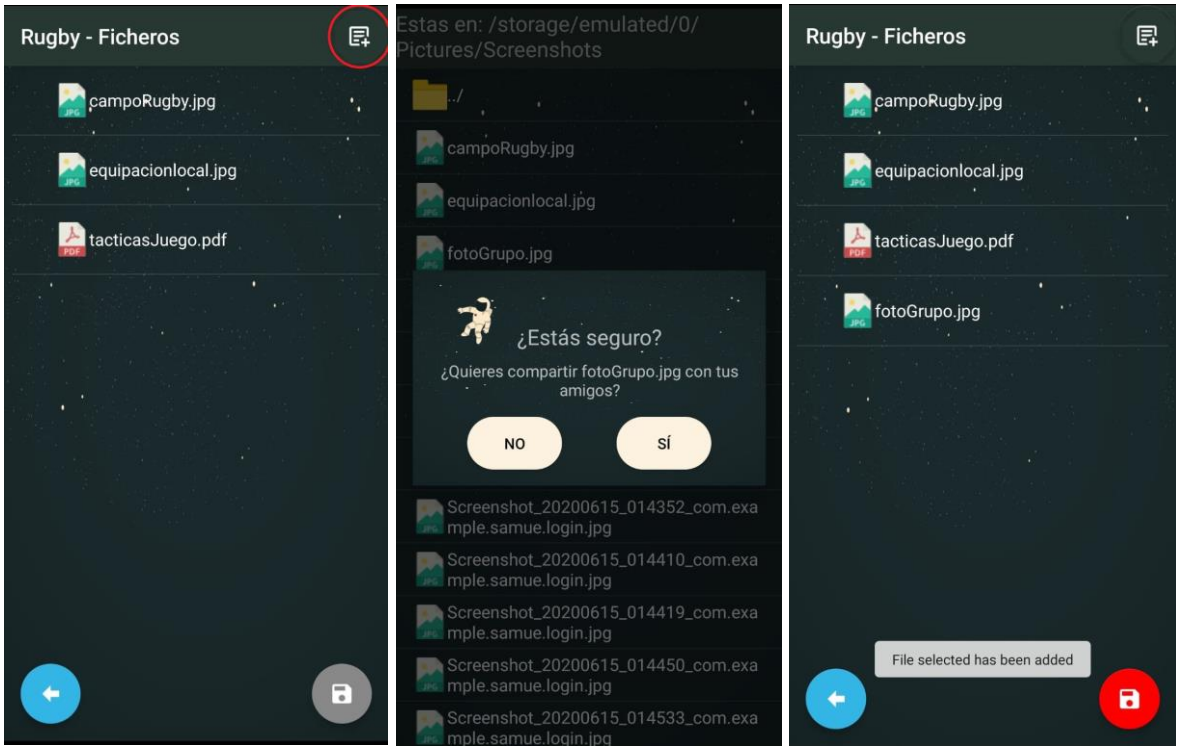

Figura 4-7: Pasos añadir un archivo al grupo

El proceso para borrar un archivo del grupo es similar y algo más simple a añadir un archivo. Únicamente hay que seleccionar de la lista un archivo que haya subido el usuario, es decir, que sea el dueño. Entonces, aparecerá un diálogo para confirmar si desea borrarlo. Tras confirmar, se actualiza la lista de ficheros y se activa el botón guardar, el cual es necesario para confirmar los cambios realizados al borrar. Al pulsarlo

se vuelve a la lista de grupos, se actualiza la base de datos sin el archivo y se comunica a todos los amigos del grupo la lista de archivos actualizada sin el fichero borrado.

La tercera acción que se puede realizar sobre los ficheros es la más importante, y en la que se basa la funcionalidad de la aplicación: descargar un fichero compartido. Para ello, hay que seleccionar un archivo, del cual no seamos dueños, y se mostrará un diálogo para su descarga (Figura 4-8). Si pulsamos la opción "No" no se realiza ninguna acción, pero si pulsamos una de las otras dos, "Sí" o "previsualizar" se inicia la descarga del fichero seleccionado abriendo la vista de descargas e iniciando el envío.

Este proceso es similar para ambas opciones con la diferencia de que, si se selecciona "previsualizar", al terminar la descarga automáticamente se abre el archivo con el programa predeterminado, es decir, si es una imagen se abre con el visor de imágenes, si es un PDF con un lector de PDFs.

Figura 4-8: Descargar archivo del grupo, opciones

En los siguientes diagramas se puede ver el flujo para añadir un archivo al grupo o para eliminar un archivo que se ha subido al grupo, siempre que el usuario sea el dueño de ese fichero. También, se puede observar el proceso de la descarga de uno de los ficheros que se encuentra en el grupo por uno de los amigos.

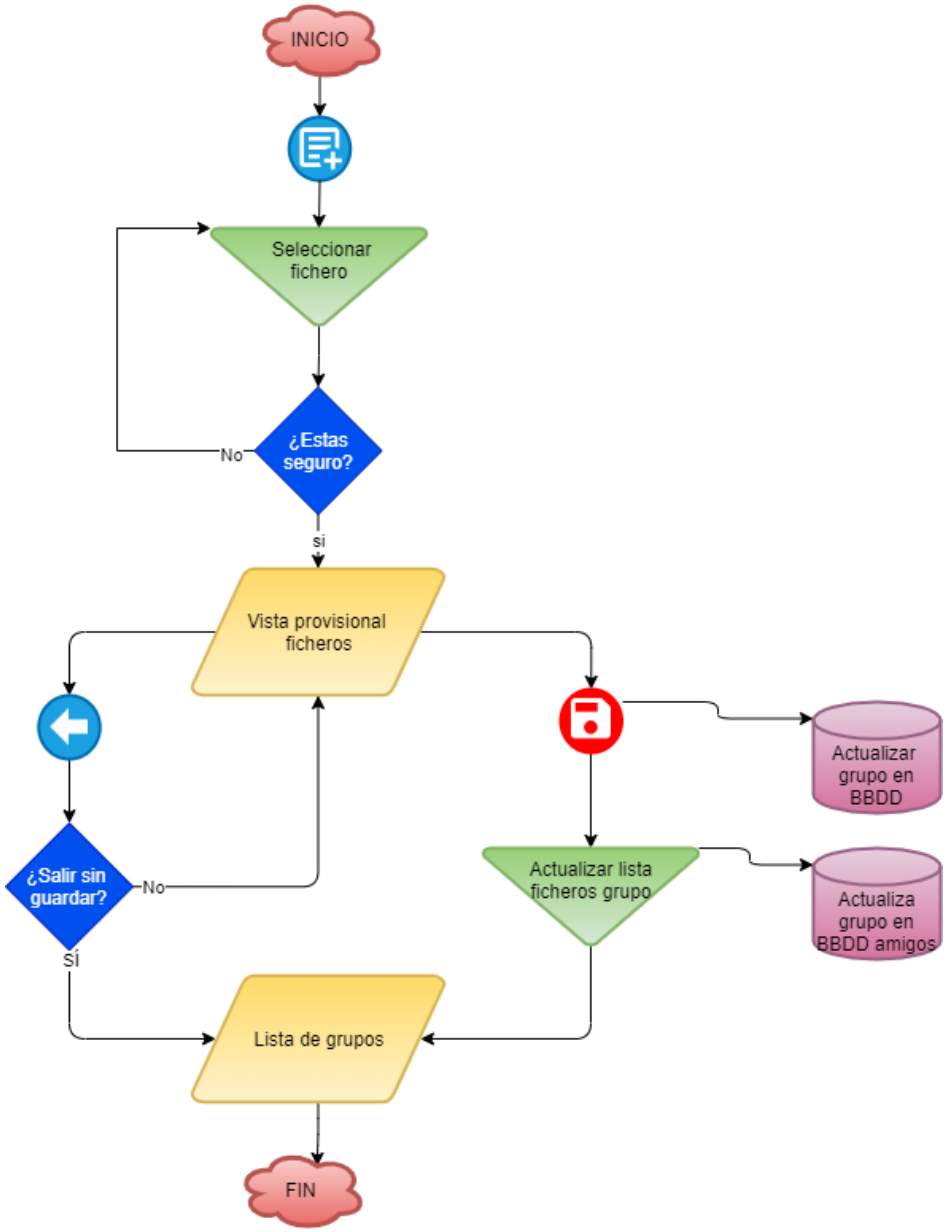

Figura 4-9: Flujo añadir archivo al grupo

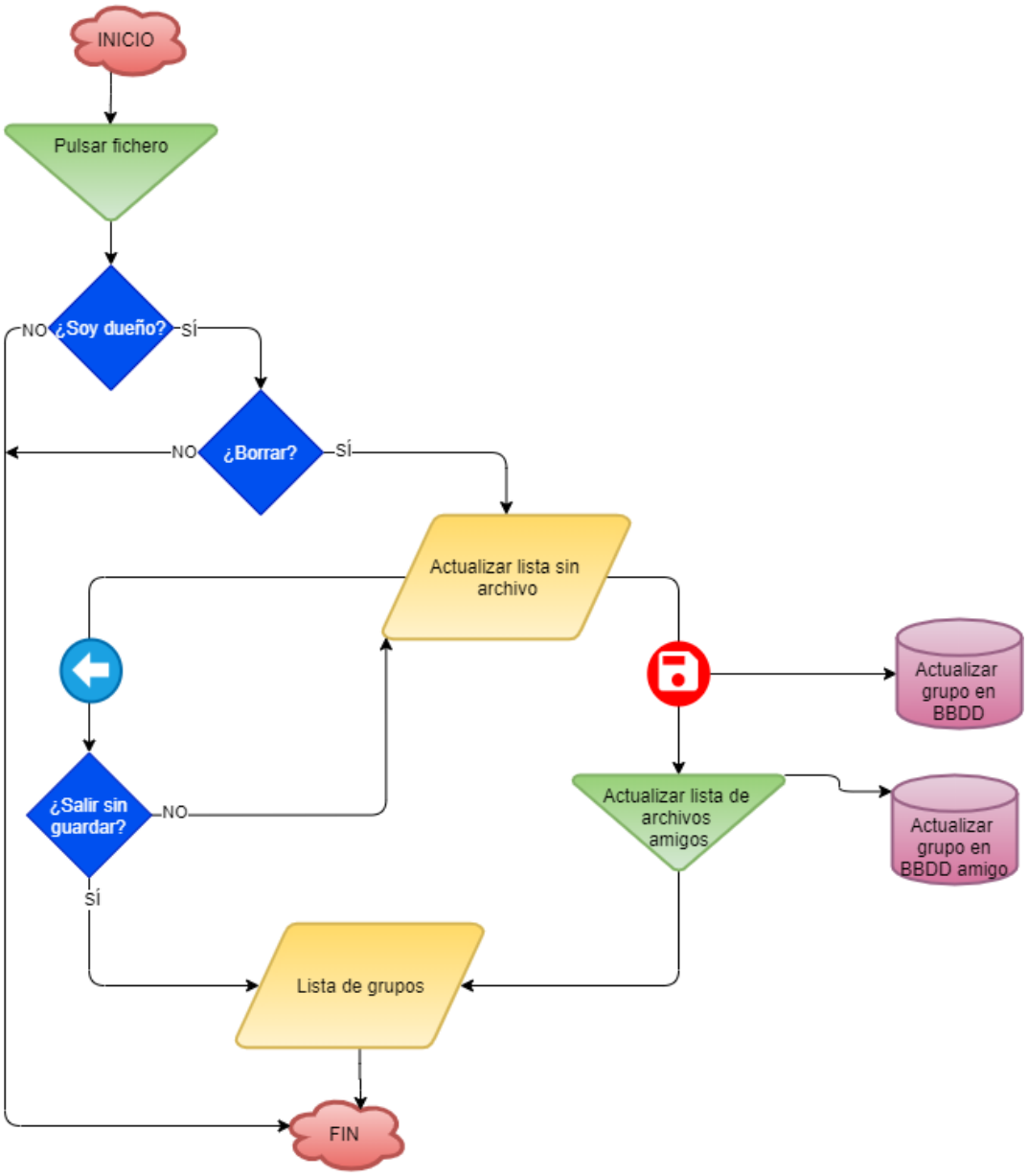

Figura 4-10: Flujo borrar archivo del grupo

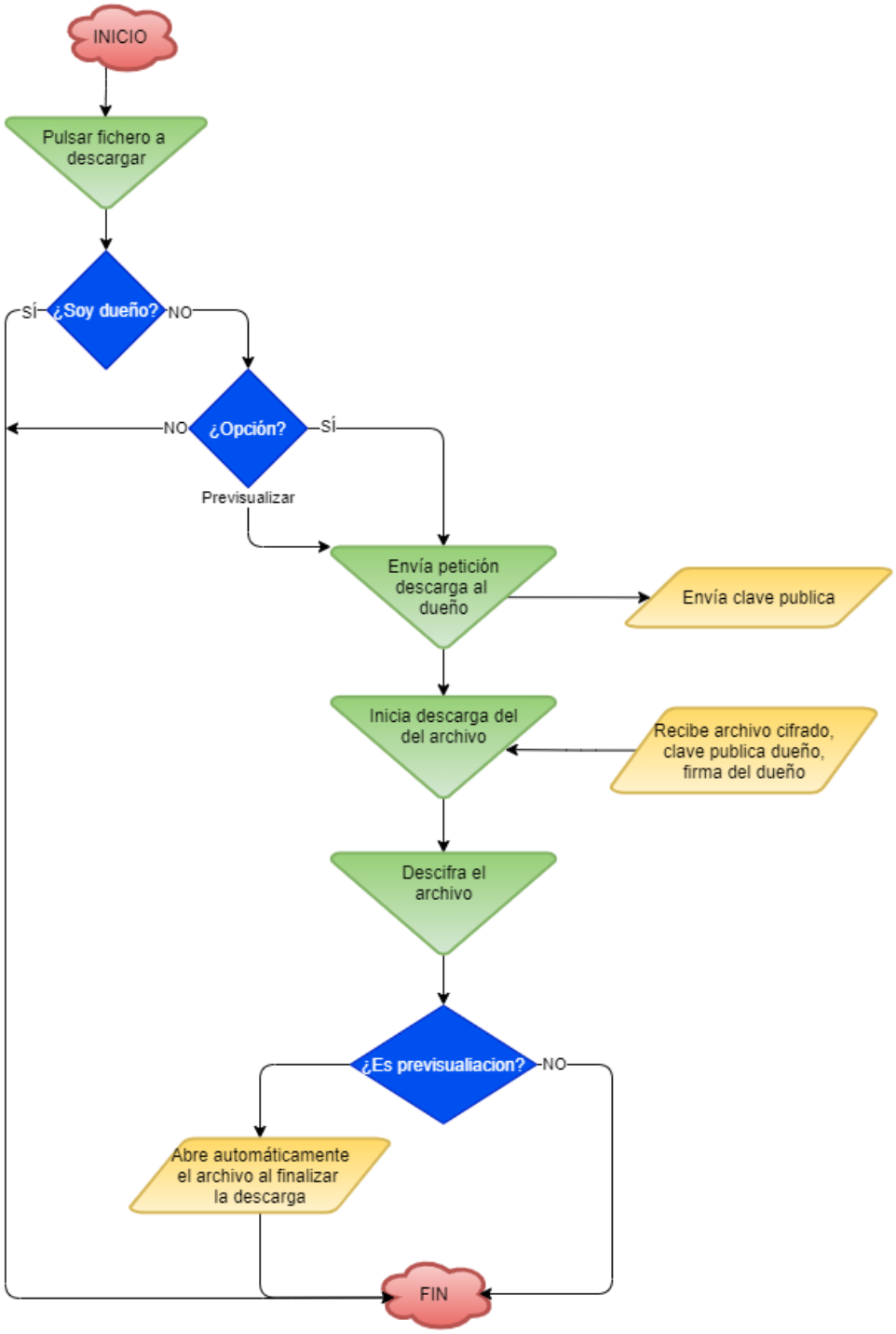

Figura 4-11: Flujo descargar/previsualizar archivo del grupo

4.1.3 Administración de amigos

Al igual que para acceder a los archivos, para ir a la vista de amigos y poder realizar todas las gestiones posibles, hay que seleccionar desde la lista de grupos el que queramos ver. A continuación, en el diálogo pulsamos "Ver amigos". En la Figura 4-12 podemos ver la vista de amigos de un grupo y sus elementos:

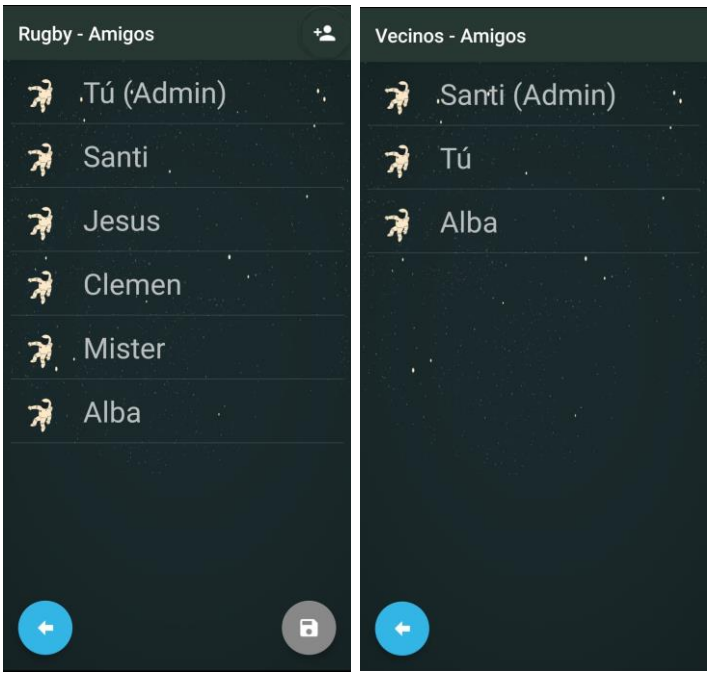

Figura 4-12: Vista de amigos de grupo (a) admin y (b) no admin

- **Barra de herramientas**. Muestra el nombre del grupo en el que estamos y la vista en la que se está. A la derecha hay un icono para añadir amigos que solo se muestra al usuario que es administrador del grupo, para el resto, está oculto.
- **Lista de amigos**. Está formada por todos los amigos que pertenecen al grupo. En dicha lista se indica entre paréntesis quién es el administrador, y en el ítem correspondiente al propio usuario aparece marcado como "Tú".

  En caso de seleccionar un amigo de la lista solo se podrá realizar una acción si se es administrador, que es borrarlo del grupo. Si el usuario no es administrador no podrá realizar ninguna acción sobre la lista. (Figura 4-12b)

- **Botón "Atrás".** Sirve para volver a la vista de grupos. En caso de haber cambios en la lista por parte del administrador del grupo y no se haya guardado, pide una confirmación.
- **Botón "Guardar".** Se encarga de guardar los cambios realizados sobre la lista de amigos. Únicamente se activa si estos se realizan.

Como se ha comentado, en esta vista solo se pueden realizar acciones si eres administrador del grupo, para el resto de los amigos del grupo que no lo son, esta vista es de información para saber todos los participantes que componen el grupo.

El administrador es el usuario que ha creado el grupo y tiene más privilegios sobre él al poder eliminar y añadir amigos.

Para añadir un amigo hay que pulsar el botón "Añadir amigo". En caso de que ya estén todos los amigos que tenemos, se mostrará un mensaje y no realizará ninguna acción. Si tiene más amigos, se llevará a una nueva vista donde verá todos los amigos que no pertenecen al grupo, para poder seleccionarlos. Después, se pulsa el botón de confirmar y se vuelve a la lista de amigos actualizada con los cambios. A su vez el botón "Guardar" se activa en rojo, indicando que se necesita confirmar los cambios y guardarlos. Si no se confirman y se sale del grupo sin guardar no se realizan.

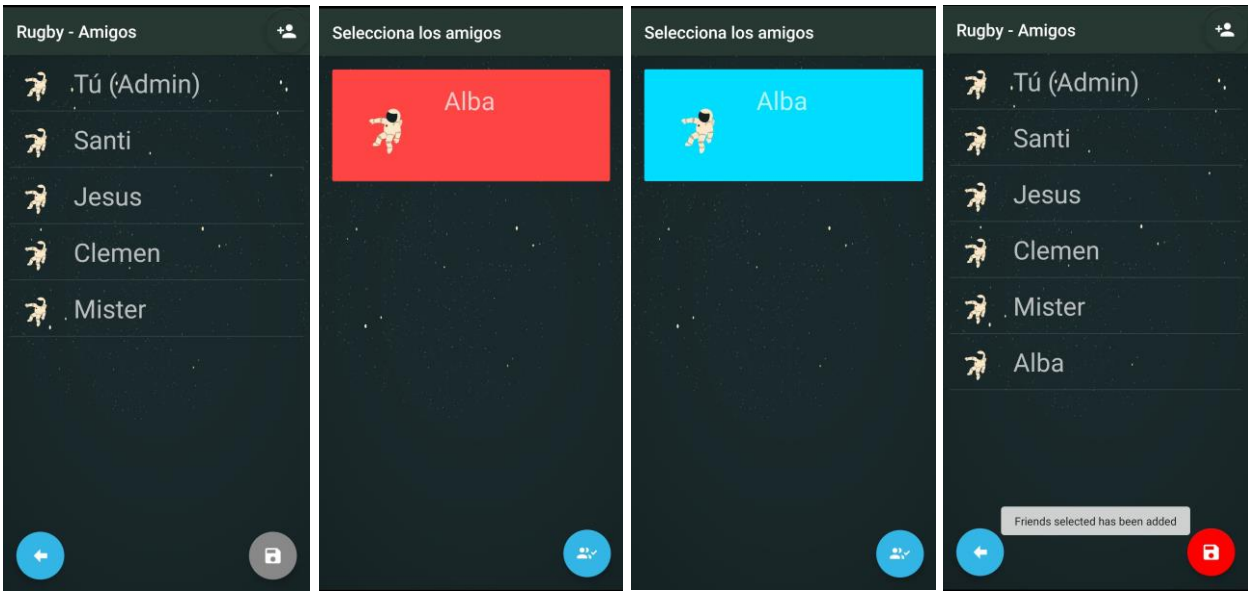

Figura 4-13: Pasos para añadir un amigo al grupo

Tras pulsar "Guardar" se vuelve a la lista de grupos, e internamente se realizan tres acciones: actualizar la base de datos con la nueva lista de amigos del grupo, enviar toda la información del grupo al nuevo amigo añadido, y actualizar la lista de amigos en el resto de los amigos que pertenecen al grupo.

Para borrar un amigo del grupo hay que pulsar sobre él y confirmar en el diálogo si se quiere borrar. Entonces se actualiza la lista de amigos y se activa el botón "Guardar" en rojo para la confirmación. Una vez se guarden los cambios, se vuelva a la lista de grupos, se actualiza la base de datos del usuario, se comunica al resto de amigos la nueva lista de amigos del grupo, y al amigo borrado, se le elimina el grupo de su BBDD.

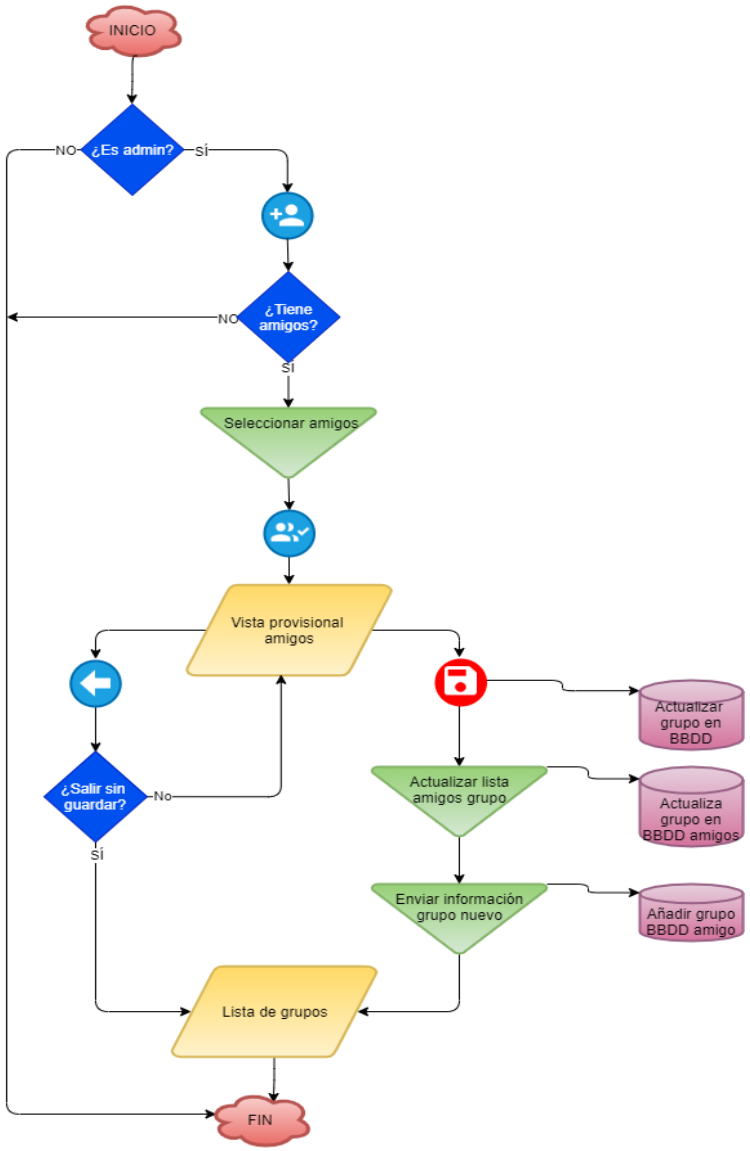

Figura 4-14: Flujo añadir amigo al grupo

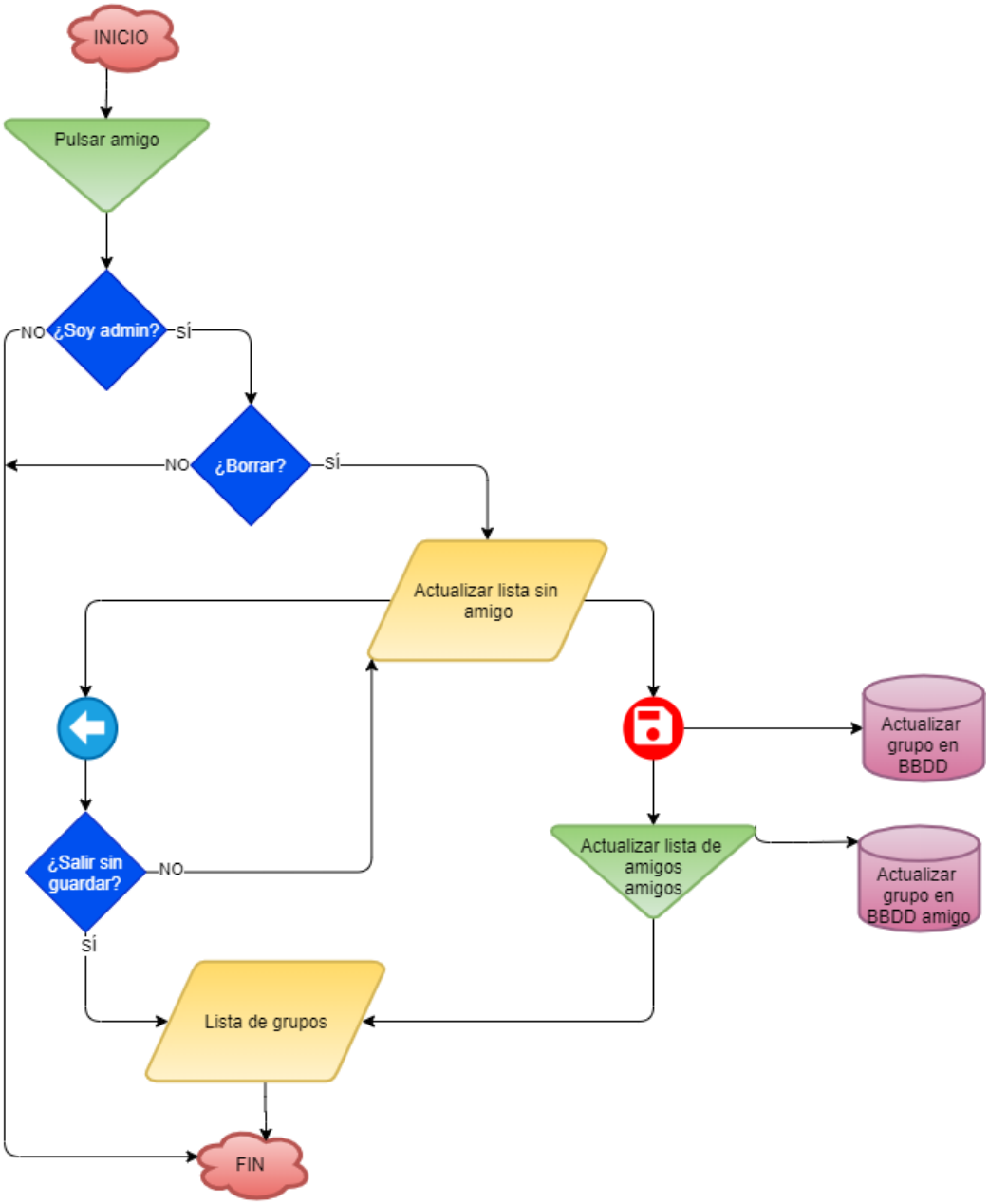

Figura 4-15: Flujo borrar amigo del grupo

4.2 Seguridad

Además de la posibilidad de crear grupos con amigos y poder gestionarlos, la cual era una funcionalidad que encajaba bastante bien con la aplicación y su uso, la otra gran carencia que se observó fue la falta de implementación de algunas medidas de seguridad mínimas. En una aplicación cuya base se basa en el uso de la tecnología de comunicación a través de desarrollos de conexión P2P, y además para compartir archivos, es algo extraño que no se envíe la información con algún nivel de cifrado para añadir más capas de seguridad, y poder proteger mejor esa información que se envía.

Por este motivo el otro gran foco del proyecto ha sido establecer las funciones necesarias para poder asegurar lo máximo posible esa comunicación. Para ello se han implementado dos tipos de cifrado: cifrado simétrico y cifrado asimétrico.

Con el cifrado simétrico, a partir de una clave se cifra la información antes de enviarse y una vez se ha recibido se descifra con esa misma clave. El cifrado asimétrico, consiste en generar 2 claves, una privada, la cual solo conoce el propio usuario, y una pública, que se comparte con los usuarios que quieres establecer una comunicación. Este segundo cifrado se usa para encriptar y firmar el envío de la clave que se usa en el cifrado simétrico entre los 2 usuarios. De esta forma se consiguen los 3 puntos básicos de la seguridad: confidencialidad, integridad y no repudio.

También, siguiendo los desarrollos en la rama de seguridad se ha querido profundizar en acciones dentro de otro ámbito que no sea la comunicación de los usuarios y la criptografía. Para ello se ha tomado como objetivo el propio código desarrollado de la aplicación, y las posibles vulnerabilidades que se pueden haber creado y no se hayan neutralizado, siendo un punto peligroso importante para la seguridad de la aplicación. Para mitigar estas vulnerabilidades se ha realizado un análisis de vulnerabilidades de código, el cual analiza la estructura y las dependencias de todos los elementos desarrollados, buscando posibles puntos críticos que puedan ser utilizados con fines maliciosos.

4.2.1 Cifrado y firma de los archivos enviados

Como se ha explicado en la introducción anterior, este apartado corresponde a las funciones desarrolladas para conseguir implementar unos niveles de encriptación lo suficientemente fuertes como para que la información que se comparte con otros amigos dentro de la aplicación se envíe de forma segura. Para ello se han realizado varias acciones, que se han ido implementando una tras otra, añadiendo cada vez más complejidad a este sistema.

Esta forma de trabajar corresponde con la explicada en el punto 1.7 del capítulo de introducción de esta memoria, siguiendo la filosofía de dividir la tarea principal, en este caso cifrar la información que se comunica en las conexiones, en tareas más pequeñas, que sean simples, y que según se desarrolle aporten un valor a la aplicación. En su conjunto completan la funcionalidad que se tenía fijada.

Para realizar todo el desarrollo se ha creado una nueva clase que agrupa todas las funciones de cifrado y descifrado, como la generación de la clave para el cifrado simétrico, como la generación del par de claves pública y privada. También incluye las funciones de cifrado asimétrico y de firma. Además, como los archivos se envían codificados como como array de bytes, se han incluido varias funciones que los transforman en Strings y viceversa para su utilización en los diferentes puntos necesarios de la aplicación.

Estos son los pasos, de manera más resumida, y de forma que sea entendible la lógica que se sigue entre dos usuarios que se envían un archivo, sin necesidad de saber el detalle de las funciones ni del código para comprender su funcionamiento:

- Ambos usuarios, user1 "el receptor" y user2 "el emisor", generan un par de claves asimétricas, basadas en el algoritmo RSA con una longitud de 2048 bits al iniciar la aplicación, dentro de Profile, las cuales serán las que usen durante el tiempo que estén con la aplicación activa.
- El user1 envía un mensaje al user2, diciéndole que quiere un archivo que él tiene. Este paso es común a partir de este momento para amigos como para grupos ya que usan el mismo flujo de comunicación y mensajes. En el mensaje añado la clave pública de user1.

- User2 recibe el nombre del archivo que quiere user1, y su clave pública para cifrarle el mensaje que se le envíe. En este punto, user2 realiza las siguientes tareas antes de enviar el nuevo mensaje a user1:
    - Genera una clave simétrica o clave secreta, basada en el algoritmo AES con un tamaño de 256 bits.
    - Con esta clave simétrica se cifra el archivo que se envía en el mensaje para user1. De esta forma si alguien ve la información del archivo estará cifrada.
    - A la vez que cifro el archivo, esta clave simétrica generada se cifra con la clave pública de user1, de este modo solo él podrá conocer la clave simétrica.
    - Con la clave privada de user2, uso la función de firma implementada en la clase de criptografía para obtener la firma (sign o signatura). De esta forma tengo el texto, que es la clave secreta cifrada y la firma del user2 sobre esa clave secreta cifrada.
    - En el mensaje inicial para user1 envío: la clave secreta usada para cifrar el archivo, cifrada con la clave pública de user1, la firma obtenida con la clave secreta cifrada y la clave privada de user2, la clave pública de user2, para que user1 compruebe la veracidad de la firma.
    - Por último, se envía el archivo que quiere user1 dividido en bloques, y todos ellos cifrados con la clave simétrica.
- User1 recibe el mensaje inicial de user2 con toda la información, y posteriormente el resto de los mensajes correspondientes a todo el archivo. Aquí el user1 tiene que hacer los siguientes pasos para poder obtener el archivo:
    - Recibir todos estos datos: clave secreta cifrada con su propia clave pública, la firma de la clave secreta con la clave privada de user2, la clave pública de user2, los datos cifrados del archivo.

- Compruebo que la clave simétrica cifrada es correcta y no ha sido alterada utilizando la clave pública de user2 y la firma correspondiente enviada. Para ello uso la función de verificación donde compruebo que usando la clave pública de user2 el texto de la clave simétrica cifrada es igual a la firma recibida.
- Una vez se tiene confirmada la clave simétrica cifrada, llega el momento de descifrar esa clave para poder usarla en la obtención de los datos del archivo. El proceso se realiza descifrando con la clave privada de user1 la clave simétrica cifrada.
- Teniendo ya la clave simétrica, se descifran todos los mensajes que se reciben de user2 de las diferentes partes del archivo, obteniendo el archivo completo de forma segura.

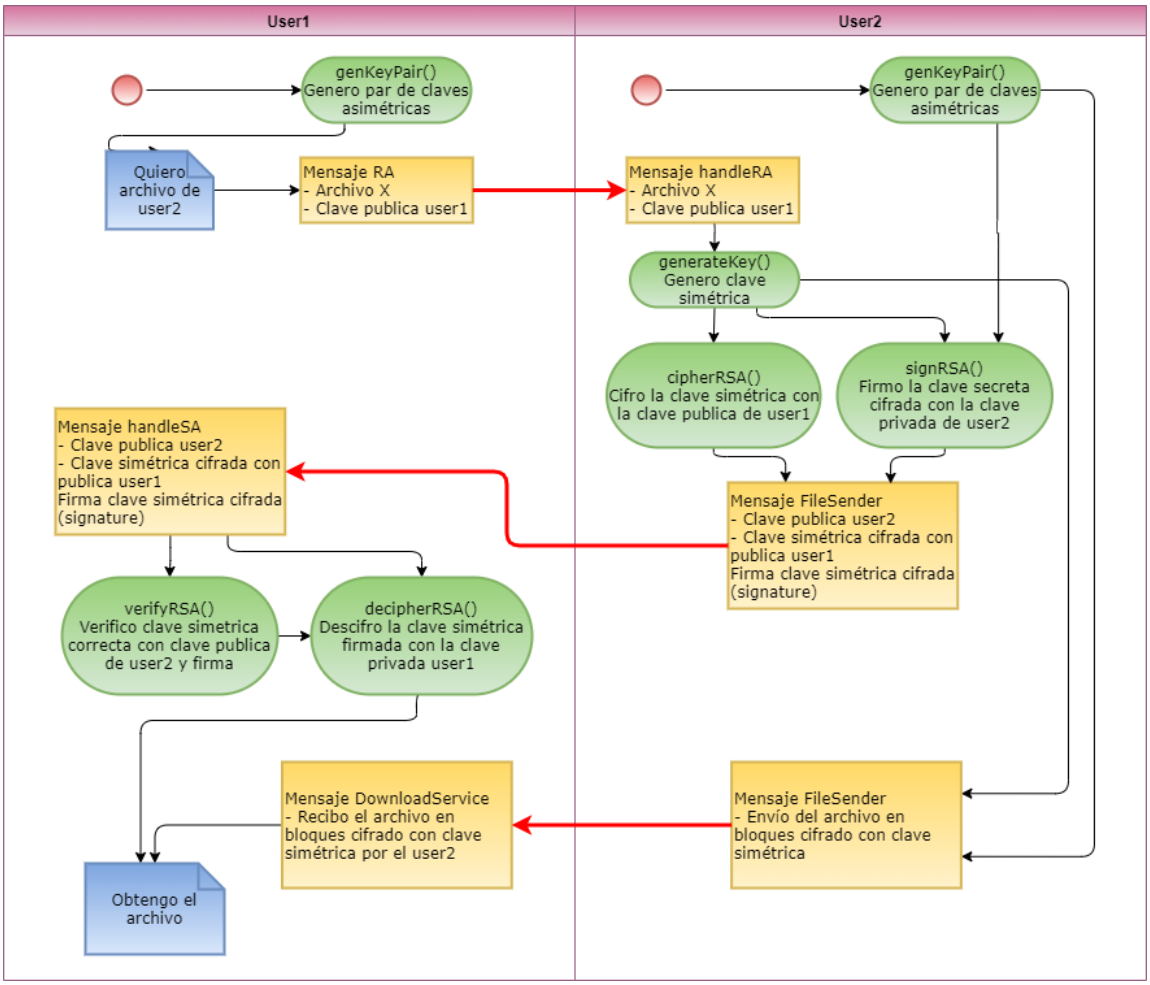

Figura 4-16: Cifrado y firma del envío de archivos

En el diagrama de actividad que se representa en la Figura 4-16 se explica todo el proceso en detalle que se produce en el intercambio de mensajes entre 2 usuarios que se envían información. Todo este desarrollo sirve tanto para conexiones entre 2 usuarios, como para los grupos, iniciándose todo en la fase común que tienen las conexiones para el envío de archivos.

4.2.2 Cifrado de contraseñas de conexión

Aprovechando la implementación de las funciones necesarias para el cifrado simétrico en el envío de los archivos, se han utilizados estas funciones para el cifrado de las contraseñas de conexión de una de las tecnologías de conexión empleadas en la aplicación, las claves de PubNub (API Keys). Estas comunicaciones funcionan usando unas claves que provee PubNub para asegurarlas y poder realizar la conexión de forma segura entre ambos usuarios. Para ello es necesario que las claves estén dentro de la aplicación, pero tener esa información escrita supone un peligro, ya que cualquiera que tenga la aplicación puede acceder a ellas y usarlas para su propio beneficio de forma simple.

Para ello se ha tomado la decisión de que, ya que es necesario tener las claves en la aplicación, no se presente en texto plano, sino que el texto que se use sea ilegible y necesario descifrar para usar, dificultando la obtención de las claves a aquellos que intenten conseguirlas, teniendo que realizar varios pasos por diferentes puntos del código.

Los pasos que se han llevado a cabo para intentar dificultar la obtención de las claves son los siguientes (Figura 4-17):

- Generación de una clave secreta con las funciones de cifrado simétrico. Con esta clave se han cifrado las claves de PubNub.
- Sustitución de las claves en texto plano por el texto cifrado en la aplicación. Ahora, el problema, es que donde se usan las claves de PubNub, no sirven estas variables, por lo tanto, se realizan más pasos.

- Guardar la clave secreta con la que se han cifrado las claves de PubNub en algún lugar del código. En este caso se ha guardado en la clase Utils, con un nombre común en la aplicación: PUBNUB.
- En las partes del código donde se usan las claves de PubNub, ahora se llama a la función de descifrado junto con la clave secreta necesaria para poder obtener el valor de las claves de PubNub originales, y así, poder usarlas para establecer la conexión.

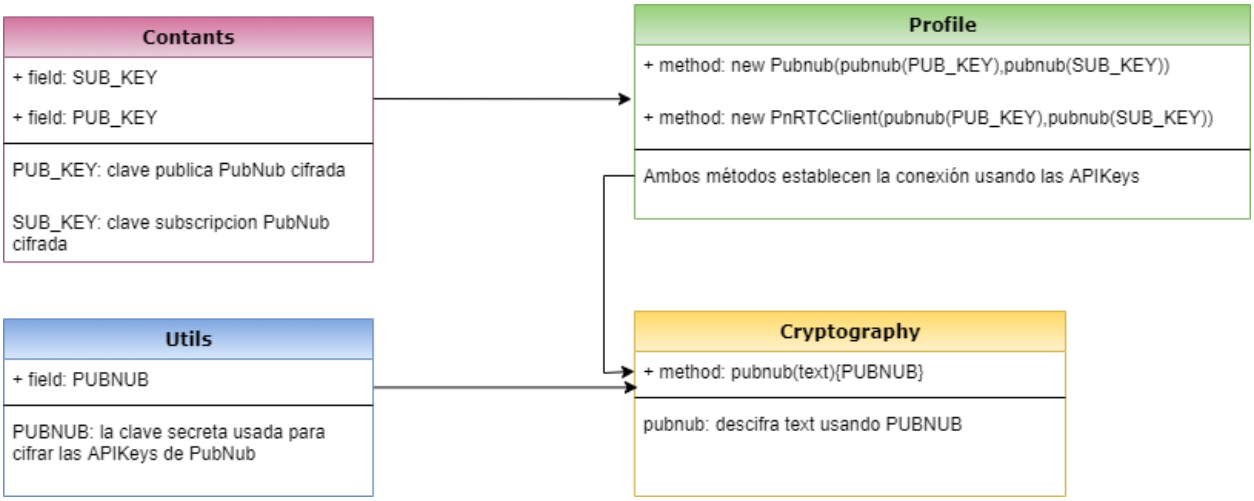

Figura 4-17: Cifrado APIKeys

4.2.3 Análisis de vulnerabilidades de código

En este apartado se va a explicar el proceso realizado para el análisis del código en busca de las posibles vulnerabilidades que se hayan creado a lo largo del proyecto. Se han utilizado dos herramientas diferentes, por un lado el inspector de código Lint [41] que trae la propia herramienta de desarrollo Android Studio, y por otro, una de las mejores herramientas de evaluación de calidad y seguridad de código fuente, cuyo software es libre, SonarQube [42].

El primer análisis de código se ha realizado tras terminar de desarrollar todas las funcionalidades del proyecto. Después de un pequeño análisis sobre qué opciones había para realizar análisis de código fuente y que fueran compatibles para Android, se

vio que una herramienta imprescindible debería ser Lint, que ya viene integrada con el Android Studio en la inspección de código, y su funcionamiento es muy simple. Para ello se siguieron estos pasos:

- Determinar el alcance de qué parte de la aplicación se iba a desarrollar. En este caso se ha realizado una selección de los ficheros nuevos generados por los desarrollos del proyecto, y aquellos que han sido ampliamente utilizados, como Profile (Clase principal) o los relacionados con la base de datos y la descarga de ficheros. En la figura 4.18 se pueden ver todos los ficheros analizados.

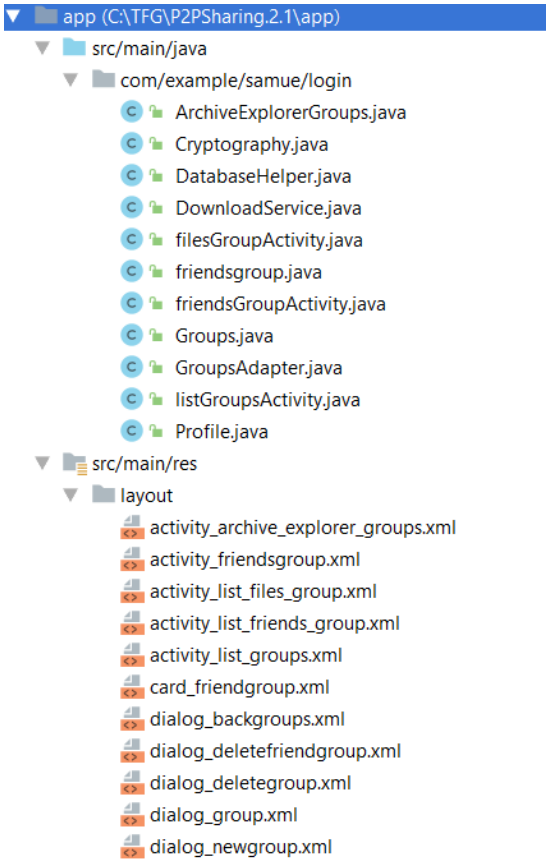

Figura 4-18: Alcance de los análisis de código

- Ejecutar la inspección de código sobre el alcance que hemos elegido (Figura 4-18). Tras su proceso, se muestra la información obtenida dividiendo todas las alertas por tipos de lenguajes de programación, y dentro de cada lenguaje agrupándolas por clases.

- Resolver todas las alertas posibles, siempre y cuando se sepan resolver sin afectar al adecuado funcionamiento de la aplicación. En algunos casos la solución propuesta por la herramienta no se considera razonable, o conlleva que la aplicación no se ejecute de la manera esperada. En estos casos, se asume la alerta y no se modifica. Para poder ver todos los cambios que se han realizado, se puede acceder al Github y ver el commit [43] completo donde se suben todos los cambios realizados.

  Estos cambios han sido: privatización de variables de la clase, guardar en variables strings textos para reutilizarlos y solo modificar en un lado, eliminación de funciones no utilizadas, simplificación de estructuras de código (if, else, return), llamadas a variables de clase sin crear objeto de ella y sustitución de variables globales por variables locales dentro de una función.

- Volver a lanzar la herramienta para ver todos los arreglos realizados, y comprobar que las vulnerabilidades se han solucionado. En la Figura 4-19 se puede ver la diferencia de las alertas que había antes y después de realizar las operaciones necesarias para solucionar los problemas detectados.

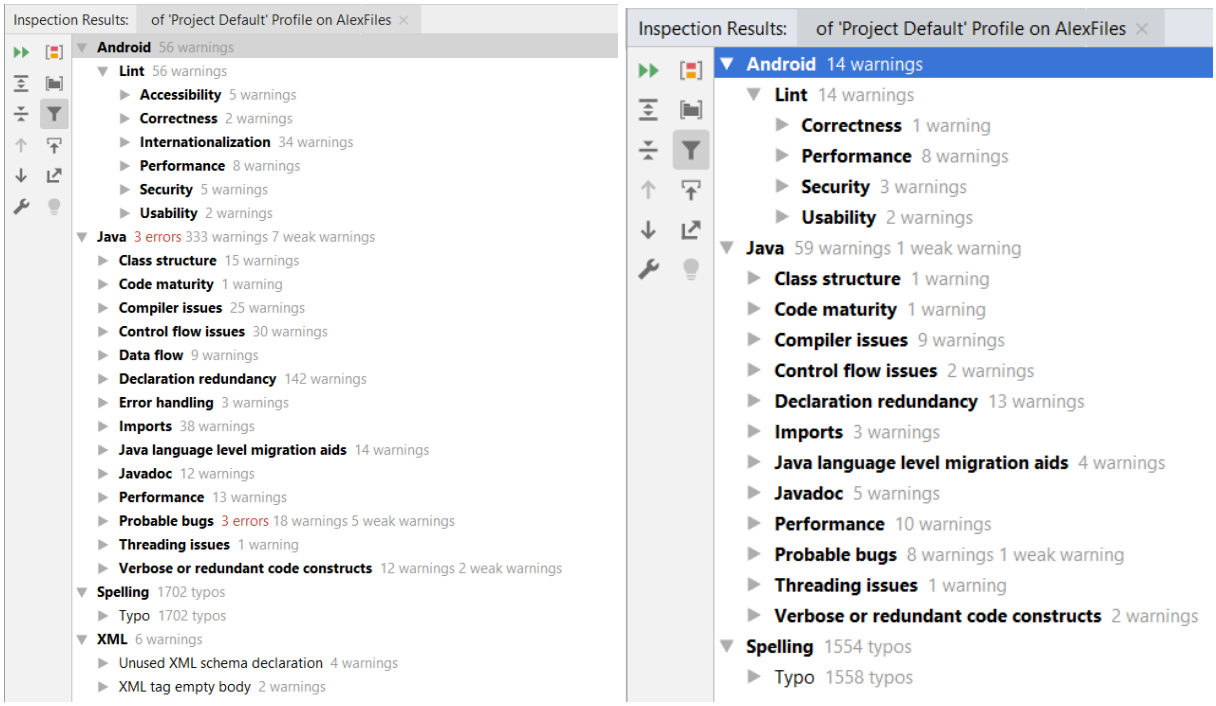

Figura 4-19: Resulltados Lint antes y después

Después del análisis realizado con Lint se decidió realizar otro análisis, pero más enfocado en la seguridad, ya que Lint se fija en aspectos más típicos de la programación, como la estructura de las clases, el uso de las variables y sus declaraciones, la redundancia, duplicación de código, etc. Para ello, se eligió Sonarqube, que es una herramienta enfocada al análisis de código buscando posibles vulnerabilidades de seguridad.

El proceso se explica a continuación:

- Instalar Sonarqube en el ordenador como un servidor al que poder acceder desde el navegador web.
- Añadir Sonarqube a AndroidStudio como plugin para ejecutar desde este el análisis. Para ello se configuró el archivo build.gradle del proyecto añadiendo la librería, las dependencias necesarias y el repositorio donde se volcaría toda la información del análisis.
- Ejecutar desde Android Studio el análisis a través del plugin instalado de Sonarqube, y acceder desde el navegador al informe para ver todos los datos obtenidos.

Una vez se obtuvieron los resultados, se observaba que la aplicación seguía teniendo vulnerabilidades de seguridad que podían ser explotadas. Al igual que en el anterior análisis, se utilizó un alcance con los archivos desarrollados y aquellos que se utilizaron. Tras resolver el mayor número de vulnerabilidades encontradas siguiendo un orden de criticidad, se volvió a realizar otro escaneo para ver las modificaciones, y como se observa en la Figura 4-20 se redujo el número hasta que la herramienta daba bueno el resultado del análisis. Al igual que antes con Lint, también se puede ver todo el código modificado en el commit [44] correspondiente a estos cambios en el GitHub del proyecto.

Entre esos nuevos cambios realizados se destaca la mejora de las sentencias de SQL en la conexión a BBDD quitando los textos literales que se repiten, el uso de los try-catch para realizar las comprobaciones que puedan dar error y utilizarlos mejor, uso de funciones get/set en lugar de usar los atributos directamente y aprovechar un objeto dentro de una clase para usar las variables, en lugar de hacerlas públicas en la clase.

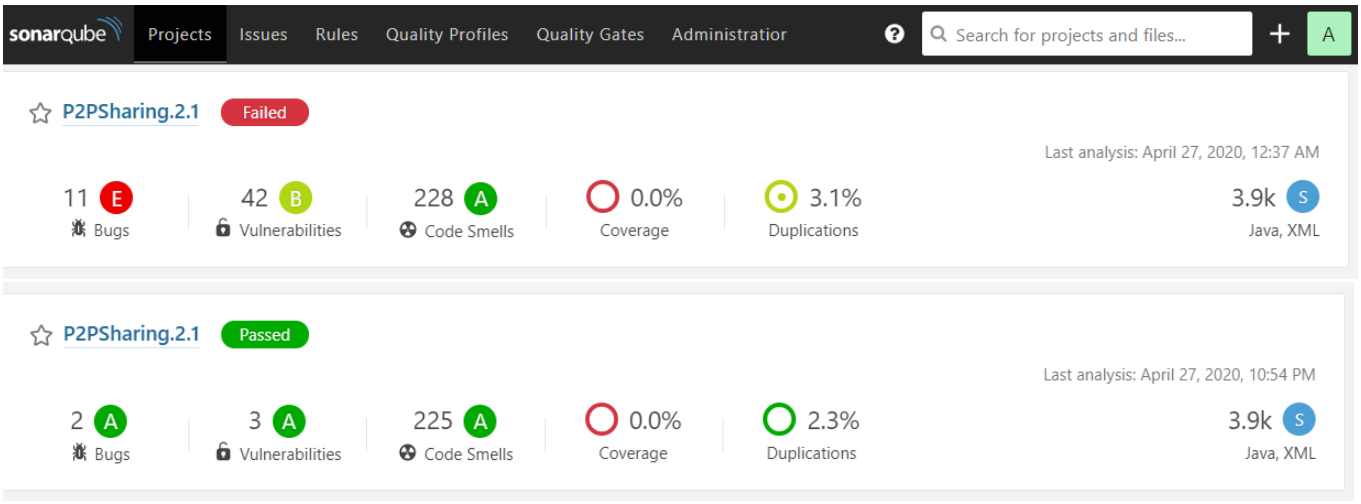

Figura 4-20: Resultados SonarQube antes y después

4.3 Mejoras

A lo largo del proyecto además de centrarse en los dos apartados más importantes explicados anteriormente, también se han realizado otras funciones y mejoras de menor tamaño o impacto en la aplicación, pero que sí suman valor a la aplicación. Por ello, en este apartado se van a explicar cómo funcionan y qué aportan.

4.3.1 Implementación buscadores

La mayoría de las aplicaciones que utilizamos en las cuales aparece un listado, ya sea de amigos, de archivos, de productos, etc., llevan una opción de búsqueda relacionada con estos elementos para poder filtrarlos.

Por su utilidad, su lógica y su gran uso cuando el número de elementos es grande, se ha decidido implementar esta opción en la aplicación. En este caso, se ha realizado en los dos apartados principales, la vista de amigos y la vista de grupos, que es donde tiene más sentido y utilidad usarlo. (Figura 4-21)

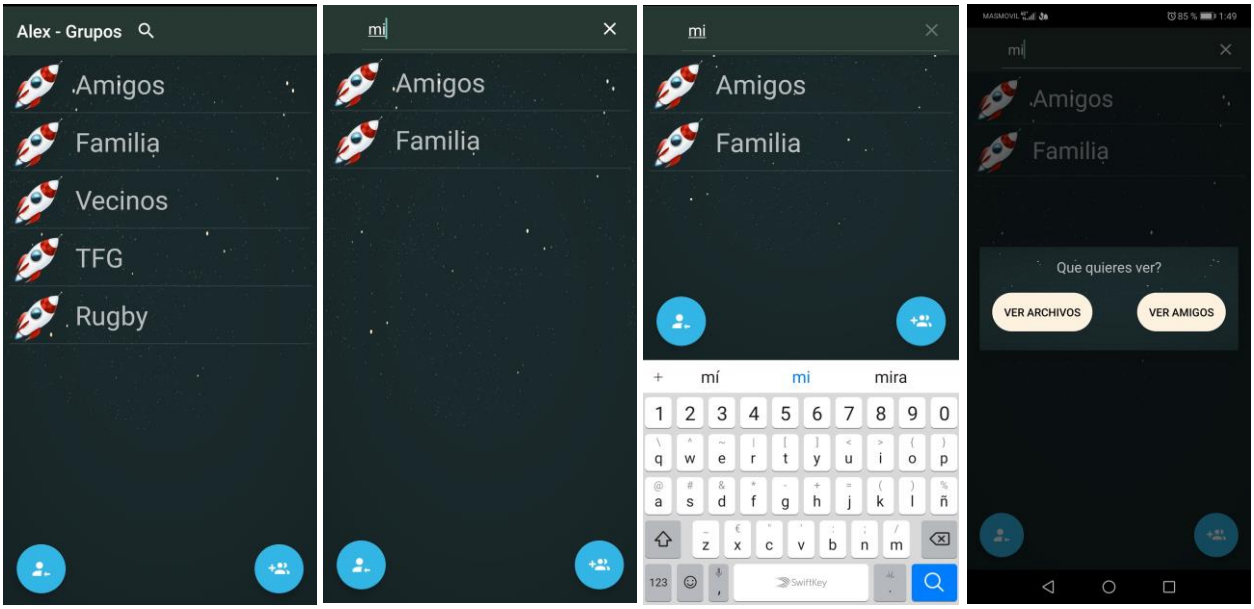

Figura 4-21: Buscador de amigos y grupos

Con respecto a su funcionamiento, sí es igual en ambas vistas. Una vez pulsamos en el icono de lupa:

- Se abre un cuadro de texto y aparece el teclado para poder escribir.
- En caso de cerrar el texto, se vuelve a la lista de amigos/grupos.
- Si se introduce uno o varios caracteres se hace un filtrado por todos los ítems, y se actualiza continuamente la lista con aquellos que contienen esos caracteres. No se distingue entre mayúsculas y minúsculas.
- En caso de no haber coincidencias no se muestra ningún elemento en la lista, y si se borran caracteres, y en algún momento coinciden con alguno, se actualiza la lista mostrándolos.
- Si durante la búsqueda se introducen caracteres y se pulsa para confirmar en el teclado, este desaparece y muestra la lista con las coincidencias.
- Durante la búsqueda, si se pulsa sobre cualquiera de los ítems, la funcionalidad es la misma que si no se estuviera en una búsqueda, apareciendo los diálogos correspondientes, y si se vuelve atrás, la búsqueda sigue activa.

Su implementación ha sido muy similar en ambas vistas, aunque no igual. En la vista de Profile de amigos, se utilizaba como un ítem de un menú usando la característica de "widget" soportado para los menús. En cambio, en la lista de grupos era un componente independiente más moderno del tipo SearchView [45].

4.3.2 Nuevo componente de visualización

En la mayoría de los elementos de la aplicación donde es necesario mostrar una lista con elementos, ya bien sean amigos, grupos, ficheros o descargas, se utiliza un elemento muy básico como es una lista. En todos estos casos el componente usado es un ListView [46], con un adaptador básico personalizado en cada caso para mostrar cada ítem. Este componente es muy simple y puede mostrar una lista de objetos de forma sencilla. El problema de este componente es que, aunque es un componente muy extendido por su gran uso, está ya bastante anticuado y desfasado, además de ser poco óptimo en el uso de memoria en el caso de tener una lista con muchos objetos.

En una de las partes nuevas del proyecto donde se ha necesitado mostrar una lista de amigos para seleccionar cuales incluir en el grupo, se ha desarrollado un nuevo componente, el RecyclerView [47] con CardViews [48], en lugar de reutilizar el ya mencionado como en el resto de los apartados de la aplicación.

De esta forma se ha aprendido a usar un componente más moderno y dinámico para mostrar listas de objetos, más personalizable y óptimo en el uso de la memoria, ya que como su propio nombre indica, en caso de tener que mostrar nuevos componentes porque no caben en la pantalla, recicla y reutiliza el espacio que usaban los ítems de los CardViews que ya no se ven, utilizando menos memoria del dispositivo. Como podemos ver en la Figura 4-22, vemos el uso que se ha hecho del RecyclerView, mostrando la lista de amigos que tiene el usuario y que puede añadir al grupo. Cada objeto, que son los amigos, se carga en un CardView independiente, dentro del RecyclerView.

Además, el adaptador que se ha usado para mostrar cada objeto se ha podido configurar para tener comportamientos diferentes cuando se hayan seleccionado. El RecyclerView se encarga de almacenar los ítems que han sido seleccionados, para así en la lógica del programa poder añadir aquellos que correspondan.

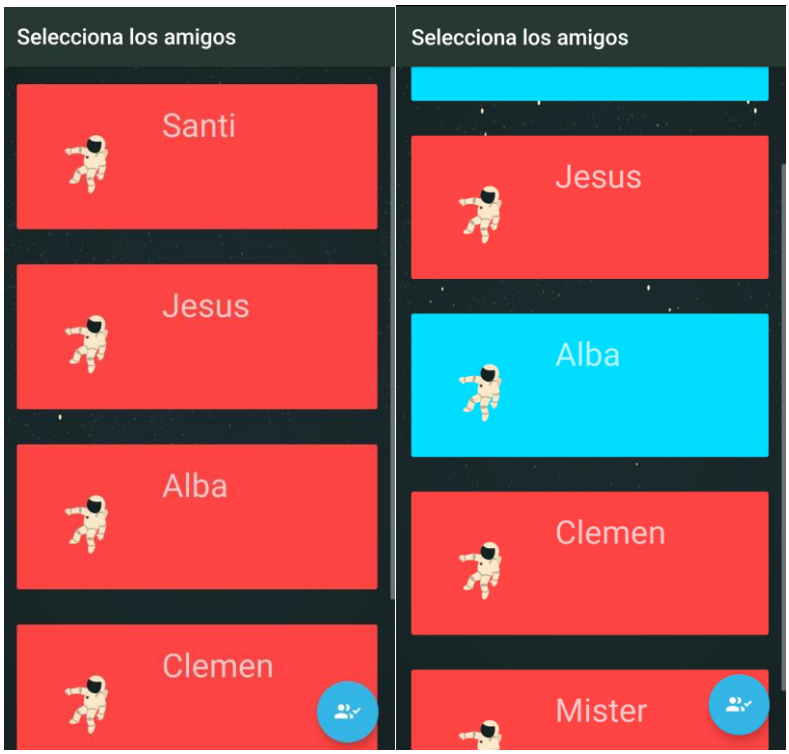

Figura 4-22: Componente RecyclerView y CardView

4.3.3 Imágenes de perfil en amigos y grupos

Para que la aplicación fuese más visual y no estuviera cargada solo por el texto correspondiente a los nombres de los amigos y de los grupos, se ha implementado la visualización de las imágenes correspondientes al perfil de los amigos y grupos cuando son visualizados.

La clase de amigos y grupos sí estaba implementada para tener imagen en su estructura, pero no se utilizaba, por lo que se ha aprovechado este desarrollo para mostrarla en las diferentes vistas donde se listan amigos y grupos. Al no personalizar el propio usuario su imagen, todos tienen la misma imagen, ya que se añade una imagen

por defecto para rellenar la clase al crear el objeto. En la Figura 4-23 se puede observar cómo era la vista antes y después.

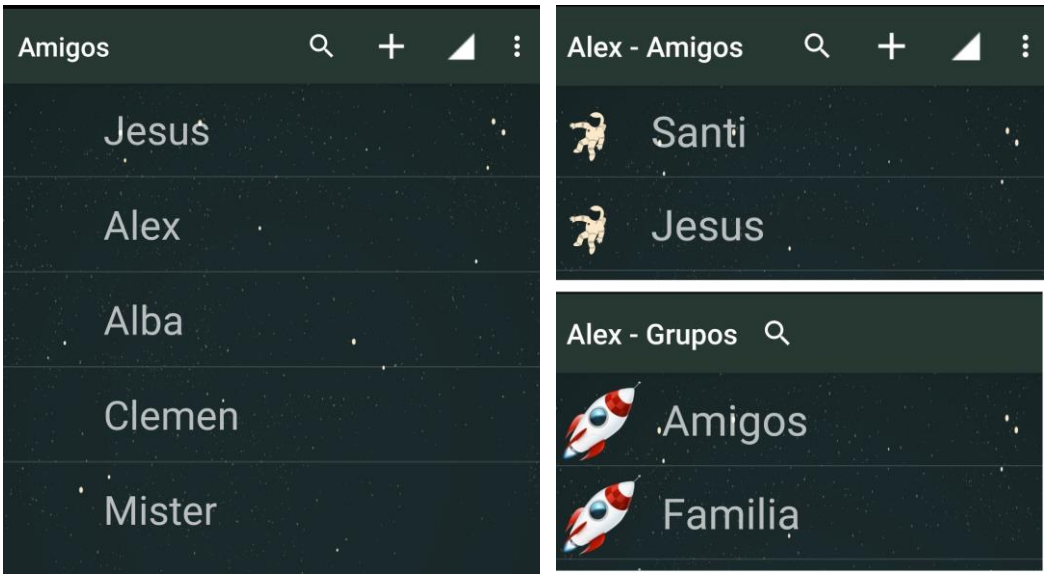

Figura 4-23: (a) Sin imagen y (b,c) con imagen de perfil

4.3.4 Mejoras visuales

Otras pequeñas mejoras que se han realizado es la visualización del nombre del usuario o del grupo y la vista en la que se encuentra. De esta forma se da más detalle y se ayuda al usuario a navegar por las ventanas de amigos y grupos o de archivos y amigos de un grupo. Este detalle se puede observar en la mayoría de las imágenes de este capítulo, en las diferentes barras de herramienta.

Además, al realizar una descarga, el único indicativo de si había terminado era que la barra de descarga estuviera llena y ya no se recibiese información. Esto también se ha mejorado, controlando el estado de la descarga y cuando ha finalizado mostrar un mensaje como se puede ver en la Figura 4-24.

Figura 4-24: Mensaje descarga finalizada

CAPÍTULO 5. Resultados, trabajo futuro y conclusiones

Este capítulo recoge de forma crítica y razonada un repaso por todo el plan de trabajo desarrollado anteriormente, revisando si se han conseguido todos los objetivos o qué más se podría realizar, así como una valoración propia y personal del trabajo realizado.

6.1 Resultados

Tras la realización del proyecto se ha conseguido una aplicación con una nueva funcionalidad bastante importante como la gestión de grupos, ampliando la capacidad de compartir archivos y gestionando cómo se comparten, pudiendo elegir con todos tus amigos, o con una selección de ellos.

También se ha añadido un nivel de seguridad necesario del que carecía la aplicación. Se ha conseguido encriptar el envío de los archivos para que nadie pueda saber qué se está compartiendo, solamente lo sepan el emisor y el receptor. Para ello se ha hecho efectivo el uso de 2 tipos de cifrado: cifrado simétrico y asimétrico.

Además, se han utilizado herramientas de análisis de código para encontrar aquellas vulnerabilidades a bajo nivel que pueden ser explotadas y así solucionarlas para evitarlo.

Por último, se han realizado pequeñas mejoras que aportan más valor a la aplicación, como la funcionalidad de búsqueda de amigos y grupos en ambas vistas, la mejora de componentes como la selección de amigos para crear un grupo, la visualización de la imagen de perfil en usuarios y grupos, y la ofuscación de código para dificultar la lectura del código con finalidad malintencionada.

6.2 Trabajo Futuro

En esta sección se mencionan algunas funcionalidades y características que se podrían realizar para añadir más valor a la aplicación y mejorarla si se pudiese seguir desarrollando el proyecto durante más tiempo.

De entre estas mejoras, algunas son de las funcionalidades ya disponibles en la aplicación y otras son nuevas características que podrían completar el funcionamiento de la aplicación.

Verificación de la integridad. Realizar la comprobación de que cuando se descarga un archivo de un usuario, al enviarse por partes, que se han recibido todas. En caso de que por el camino se hayan perdido algunos paquetes, solicitar el envío de ellos de forma automática para completar la descarga, o mostrar un mensaje de error al usuario.

Integración de la compartición de carpetas en grupos. Al igual que uno de los desarrollos anteriores como era la previsualización, que se ha añadido a la funcionalidad de los grupos, sería interesante añadir también la compartición de carpetas en un grupo para darle más valor a esta funcionalidad.

Funcionamiento en segundo plano. Esta característica sería muy interesante llevarla a cabo, para perder la menor información posible entre los usuarios. Ya que la aplicación funciona correctamente si están en línea los usuarios únicamente. Al tratarse de una conexión punto a punto, sin servidores de por medio, no es posible guardar la información que se envía en un servidor intermedio, que en caso de no estar conectado poder recuperar la información enviada cuando el otro usuario se conecte y estar actualizado. Si se pudiese mantener la funcionalidad en segundo plano, gran parte de la información que se envíe, aunque no se esté en línea en la aplicación, podría llegar y actualizarse.

Actualización de los ficheros compartidos. Comprobar al iniciar la aplicación que los ficheros que se han compartido siguen existiendo en el dispositivo. En caso contrario, borrarlo de los sitios donde se esté compartiendo, para no mostrar información que no existe y generar errores a la hora de descargarlos por otros usuarios.

Implementar la gestión de la imagen de perfil de usuario y grupo. Ya que en este proyecto se ha implementado la visualización de la imagen de perfil tanto de los usuarios como de los grupos, desarrollar la capacidad de poder seleccionar unas imágenes predeterminadas o subir una imagen propia, para personalizar los perfiles por parte de los usuarios.

Identificador único para usuarios. En la aplicación pueden existir dos usuarios con el mismo nombre, sin poder identificarse de forma única. Esto puede conllevar errores, tanto de confusión por parte de un usuario, como de envío de información en la comunicación. Se podría intentar generar un código aleatorio único al registrarse el usuario, que junto al nombre se envíe como información a los amigos. Así, en el apartado de las comunicaciones, no habría confusión de a qué usuario enviar los datos, ya que se usaría este código para verificar que se envía al usuario correcto.

Implementar guardado de claves de cifrado. Un paso más para aplicación aparte de toda la seguridad que se ha implementado, sería la gestión de todas las claves que se usan en las comunicaciones y con los diferentes amigos. Para ello podría crearse una nueva base de datos donde guardar todas las claves públicas de los amigos, que se intercambiarían en el proceso de añadir un amigo y así ya disponer de ellas. Con respecto a los grupos, podría añadirse un campo nuevo en la base de datos de grupo, donde se almacenasen las claves públicas de todos los amigos del grupo. Como alternativa, se podría usar la KeyStore de Android, funcionalidad disponible para la gestión de claves.

6.3 Conclusiones

Tras terminar el proyecto llega el momento de mirar atrás y ver si todos los objetivos que me propuse al inicio se han cumplido por completo o en su gran mayoría. Además, realizar una valoración de todo ese trabajo tanto de desarrollo como el esfuerzo personal y las metas alcanzadas.

El principal objetivo era coger una aplicación desarrollada por terceros, y sobre ella realizar nuevas funcionalidades y mejoras en esos puntos débiles que se analizaron al comienzo. Por lo tanto, tras ver todo el trabajo desarrollado, se puede decir que el objetivo se ha conseguido. Se ha creado una nueva funcionalidad muy conocida por los usuarios de aplicaciones móviles, que permite poder compartir archivos con los grupos de amigos que se quiera, y poder administrarlos. La creación de grupos aporta una gestión de la aplicación más amplia a la que tenía anteriormente.

Además de los grupos, el otro objetivo del proyecto que era mejorar la aplicación en la capa de seguridad, también se ha conseguido. Se han podido implementar varios tipos de cifrado para hacer más seguro el intercambio de información entre los usuarios, tanto de forma directa, como a través de los grupos con varios usuarios.

Por último, se ha logrado realizar más mejoras de menor importancia, con respecto a las 2 anteriores, que también aportan valor a la aplicación tanto en el apartado visual y funcional, como en el apartado de seguridad.

A nivel personal, también he conseguido los objetivos que me había propuesto. He sido capaz de aprender a desarrollar una aplicación Android, lo cual ha sido bastante satisfactorio, ya que, al tener una buena base de Java, he podido avanzar de forma regular, deteniéndome en las partes más puras de Android para aprender su uso e implementarlo.

Con respecto a la seguridad, es una rama que me apasiona, pero no había tenido la oportunidad de ponerla en práctica. Con este proyecto lo he conseguido, viendo desde un punto diferente, como poder proteger la aplicación de las principales amenazas. He podido implementar métodos de cifrado para proteger la información que se intercambian los usuarios, y realizado mejoras en el código frente a las amenazas surgidas por el desarrollo de la aplicación.

6.4 Conclusions

After finishing the project, it is time to look back and see if all the goals I set at the beginning have been met completely or most of them. Besides, to assess all that work, both development and personal, and goals are achieved.

The main objective was to take an application developed by a third party, and on it develop new features and improvements in those weak points that were analyzed at the beginning. So, after viewing all the work done, we can say that the objective has been completed. New functionality has been created, that is very well known by apps users, which allows them to share files with their friends and manage them. The creation of groups provides management of the application more extensive to the one that previously had.

In addition to the groups, the other objective of the project, which was to improve the application in the security level, has also been achieved. It has been possible to implement various types of encryption to make the transfer of information between users more secure, both directly and through groups with several users.

Finally, more minor improvements have been made, concerning the previous two, that also add value to the application both in the visual and functional section, as well as in the security section.

On a personal level, I have also achieved the objectives I had established. I have been able to learn how to develop an Android application, which has been very satisfactory, because, having a good Java base, I have been able to advance constantly, stopping in the purest parts of Android to learn its use and implement it.

Regarding security, it is a subject that I am fascinated with, but I had not had the opportunity to put it into practice. With this project I have made it, looking from a different point of view, how to protect the application from the main threats. I have been able to implement encryption methods to protect the information that users exchange and to make improvements to the code against the threats that arise from the development of the application.

BIBLIOGRAFÍA

[1] J. Galilea, «TFG P2P Sharing 2.0,» [En línea]: https://github.com/jotagalilea/P2P-Android-App.

[2] Android, «What is android,» [En línea]: https://www.android.com/intl/es_es/what-is-android/.

[3] W. Inc, «Whatsapp Funciones,» [En línea]: https://www.whatsapp.com/features/.

[4] G. LLC, «Google Drive,» [En línea]: https://www.google.com/intl/es_ALL/drive/.

[5] Wikipedia, «Peer to peer (P2P),» [En línea]: https://es.wikipedia.org/wiki/Peer-to-peer.

[6] Emule, «Bienvenidos Emule,» [En línea]: https://www.emule-project.net/home/perl/general.cgi?l=17.

[7] J. Galileo, «P2P Sharing 2.0,» GitHub, [En línea]: https://github.com/jotagalilea/P2P-Android-App.

[8] Zenkit, «Agile Methodology,» [En línea]: https://zenkit.com/en/blog/agile-methodology-an-overview/.

[9] Novis, «Una metodología ágil para las mejoras SAP,» [En línea]: https://www.novis.cl/noticias-novis/servicios-sap-2/una-metodologia-agil-para-las-mejoras-sap/.

[10] WebRTC, «WebRTC,» [En línea]: https://webrtc.org/.

[11] P. Inc, «PubNub Home,» [En línea]: https://www.pubnub.com/.

[12] J. y. Josué, «TFG P2P Sharing,» [En línea]: https://eprints.ucm.es/48889/.

[13] WhatsApp, «app WhatsApp,» [En línea]: https://www.whatsapp.com/?lang=es.

[14] G. Play, «ShareOnWifi,» [En línea]: https://play.google.com/store/apps/details?id=aaqib.shareonwifi.

[15] GitHub, «RetroShare,» [En línea]: https://github.com/RetroShare.

[16] RetroShare, «Home RetroShare,» [En línea]: https://retroshare.cc/.

[17] E. Confidencial, «App Tsunamic Democratic,» [En línea]: https://www.elconfidencial.com/tecnologia/2019-10-15/app-tsunami-democratic-supremo-sentencia-proces-cataluna-015_2283628/.

[18] Wikipedia, «Peer-to-peer,» [En línea]: https://es.wikipedia.org/wiki/Peer-to-peer.

[19] Rhapsody, «Napster,» [En línea]: https://es.napster.com/.

[20] Microsoft, «Skype,» [En línea]: https://www.skype.com/es/.

[21] B. Project, «Bitcoin,» [En línea]: https://bitcoin.org/es/.

[22] D. Library, «Introducing Tarzan,» [En línea]: https://dl.acm.org/doi/10.5555/646334.687802.

[23] 3CX, «¿Que es WebRTC?,» [En línea]: https://www.3cx.es/webrtc/que-es-webrtc/.

[24] pngflow, «WebRTC,» [En línea]: https://www.pngflow.com/en/free-transparent-png-rxuua.

[25] X. Android, «¿Qué es Android?,» [En línea]: https://www.xatakandroid.com/sistema-operativo/que-es-android.

[26] Android, «Arquitectura de la plataforma,» [En línea]: https://developer.android.com/guide/platform.

[27] BBVA, «Asi potencia la ciberseguridad un banco,» [En línea]: https://www.bbva.com/es/asi-trabaja-banco-potenciar-ciberseguridad/.

[28] Wikipedia, «Seguridad informatica,» [En línea]: https://es.wikipedia.org/wiki/Seguridad_informatica.

[29] Wikipedia, «Seguridad de la información,» [En línea]: https://es.wikipedia.org/wiki/Seguridad_de_la_informacion.

[30] PC-Solucion, «Principios fundamentales Seguridad,» [En línea]: https://pc-solucion.es/2017/06/16/aspectos-principios-fundamentales-la-seguridad-informatica/.

[31] Wikipedia, «Criptología,» [En línea]: https://es.wikipedia.org/wiki/Criptologia.

[32] Wikipedia, «Criptografía,» [En línea]: https://es.wikipedia.org/wiki/Criptografia.

[33] UPM, «Introducción a la criptografía,» [En línea]: http://www.dma.fi.upm.es/recursos/aplicaciones/matematica_discreta/web/aritmetica_modular/criptografia.html.

[34] BandaGeek, «¿Que es la criptografía?,» [En línea]: https://bandageek.com/2017/02/que-es-la-criptografia/.

[35] WebNode, «Algoritmos simétricos,» [En línea]: https://criptografia.webnode.es/algoritmos-simetricos/.

[36] Medium, «Cryptography,» [En línea]: https://medium.com/@ealtili/cryptography-encryption-hash-functions-and-digital-signature-101-298a03eb9462.

[37] RSA, «RSA,» [En línea]: https://www.rsa.com/.

[38] GitKraken, «GitKraken,» [En línea]: https://www.gitkraken.com/.

[39] Github, «Github,» [En línea]: https://github.com/.

[40] Android, «Android Studio,» [En línea]: https://developer.android.com/studio.

[41] D. Android, «Como mejorar tu código,» [En línea]: https://developer.android.com/studio/write/lint.

[42] SonarQube, «SonarQube,» [En línea]: https://www.sonarqube.org/.

[43] Github, «Lint Commit,» [En línea]: https://github.com/alexmartin3/P2PSharing.2.1/commit/31c156ea614753d638450416fa6a5624991c658b.

[44] GitHub, «SonarQube commit,» [En línea]: https://github.com/alexmartin3/P2PSharing.2.1/commit/00e15c994aad309aea58df71efe79250396e8665.

[45] D. Android, «SearchView,» [En línea]: https://developer.android.com/reference/android/widget/SearchView.

[46] D. Android, «ListView,» [En línea]: https://developer.android.com/reference/android/widget/ListView.

[47] D. Android, «RecyclerView,» [En línea]: https://developer.android.com/guide/topics/ui/layout/recyclerview.

[48] D. Android, «CardView,» [En línea]: https://developer.android.com/guide/topics/ui/layout/cardview.

[49] X. movil, «P2P, el autentico espiritu del file sharing,» [En línea]: https://www.xatakamovil.com/conectividad/metodos-para-compartir-archivos-y-contenidos-en-internet-iiip2p-el-autentico-espiritu-del-file-sharing.

[50] Wikipedia, «Android Caracteristicas,» [En línea]: https://es.wikipedia.org/wiki/Android#Aplicaciones.

[51] Android, «Android Open Source Project,» [En línea]: https://source.android.com/.

[52] Android, «Documentacion Guia,» [En línea]: https://developer.android.com/guide.

[53] Google, «Google Play,» [En línea]: https://play.google.com/store.