Universidad de Valladolid

ESCUELA DE INGENIERÍA INFORMÁTICA

GRADO EN INGENIERÍA INFORMÁTICA
Mención en Tecnologías de la Información

Creación de una botnet para la recopilación y extracción de información en dispositivos Android

---

Alumno: Nicolás Cea Porras

Tutor: Blas Torregrosa García

Dedicado a aquellos que junto a mí han trabajado para que esto sea posible

Agradecimientos

A mis padres que me han ofrecido la posibilidad de poder estudiar y formarme hasta conseguir ser ingeniero

A mis amigos que siempre han estado ahí, ayudándome de forma desinteresada

A los profesores que aún mantienen su vocación por enseñar

Y a mi mismo, por haber sido el principal responsable de que esto haya funcionado.

Resumen

Una botnet es una red de dispositivos secuestrados, infectados con malware y controlados remotamente por un actor malicioso. En el presente trabajo se muestra en detalle mi implementación del sistema, tomando como objetivos diferentes dispositivos móviles basados en android y utilizando como Bot master una máquina virtual Kali Linux. Ambas partes serán capaces de interactuar entre sí, utilizando una base de datos integrada en la nube. El objetivo de la red será recopilar información de carácter personal de los dispositivos que la componen cumpliendo así una función de espionaje, accediendo y recopilando, entre otros datos, la lista de contactos y sus números de teléfono, la lista de mensajes SMS recibidos, la ubicación GPS, el contenido del portapapeles, etc. La información se recopila a petición del Bot master que realizará solicitudes para obtener los datos robados. Para lograr su objetivo los bots cumplirán una serie de características que permitirán su persistencia, evasión y control basadas en el framework MITRE ATT&CK (mobile) [1].

Abstract

A botnet is a network of hijacked devices, infected with malware and remotely controlled by a malicious actor. This paper shows in detail my implementation of the system, taking as targets different android based mobile devices and using as Bot master a Kali Linux virtual machine. Both parts will be able to interact with each other, using an integrated database in the cloud. The objective of the network will be to collect personal information from the devices that compose it, thus fulfilling a spying function, accessing and collecting, among other data, the list of contacts and their phone numbers, the list of received SMS messages, GPS location, clipboard contents, etc. The information is collected at the request of the Bot master who will make requests to obtain the stolen data. To achieve their goal the bots will comply with a series of features that will enable their persistence, evasion and control based on the MITRE ATT&CK (mobile) framework [1].

Índice general

Lista de Figuras

Lista de Tablas

Capítulo 1

Introducción

1.1. Contexto

Una botnet es una red de dispositivos (ordenadores, sistemas móviles, dispositivos IoT) conectados a internet que se encuentran secuestrados, comúnmente como consecuencia de una infección por malware. Estos, se encuentran bajo el control de una única parte atacante llamada "bot master", o "bot herder", por otro lado, cada máquina individual que está formando parte de la red bajo el control de este se conoce como bot y pueden ser utilizados de manera conjunta a otros bots para realizar actividades maliciosas como por ejemplo ataques DDoS (Denegación de servicios distribuida), substracción de información confidencial o distribución de malware mediante Spam a servicios de mensajería electrónica. El propósito de la botnet presentada en el presente trabajo será el espionaje de sistemas móviles que utilizan android como sistema operativo.

Según ENISA (European Union Agency for Cybersecurity) [14], las botnet son consideradas una de las mayores amenazas para los dispositivos conectados a internet, de hecho, cada año el número de ataques aumenta. Además, suponen una inmensa amenaza de ciberseguridad tanto para los gobiernos como para empresas y particulares como ya se ha demostrado en varias ocasiones, por ejemplo, con el spyware "Pegasus"[55] un software espía desarrollado, comercializado y licenciado a gobiernos de todo el mundo siendo, quizás, el software espía más potente de la historia demostrando la capacidad de infectar a miles de millones de teléfonos con sistemas operativos iOS o Android para, posteriormente, realizar tareas de espionaje a gran escala.

Desde un punto de vista enfocado a la vida profesional, conocer los diferentes tipos de ataques, amenazas y retos para la ciberseguridad es una de las necesidades básicas de un ingeniero en informática que trabaje en esta rama (por ejemplo, para determinar el camino de ataque al realizar un análisis de amenazas y evaluación de riesgos), por lo que, aplicado con motivos educativos, el desarrollo de una botnet puede ofrecer una excelente experiencia para el aprendizaje y la formación de un estudiante.

1.2. Motivación

Principalmente mi mayor motivación durante la realización de este trabajo ha sido aprender, por lo que he pensado en dos enfoques diferentes a la hora de dirigir el trabajo. La primera decisión es realizar un trabajo sobre un tema, que para mi, resulte desconocido. La ventaja es que trabajaré con conceptos nuevos que me permiten tener mayor experiencia en distintas ramas de la informática para poder tomar una mejor decisión sobre la orientación de mi futura vida profesional, sin embargo, cuento con la desventaja de que la estimación de la dificultad, el esfuerzo y el alcance del proyecto posiblemente se vean perjudicados. Otra opción interesante es perfeccionar algún tema que ya hubiera tratado anteriormente, lo cual influye de forma positiva en el proceso de trabajo y en la especialización y formación mas avanzada sobre un tema, pero tiene como inconveniente un abanico menor de posibilidades. Finalmente me he inclinado por realizar un proyecto nuevo más enfocado en la exploración y aprendizaje de nuevas tecnologías como son la investigación y desarrollo de una botnet. Además a lo largo de la realización del grado en ingeniería informática he desarrollado un interés hacia el mundo de la ciberseguridad por lo que la oportunidad de participar en el y seguir formándome en este adentrándome un poco más con un proyecto importante de estrecha relación para el mismo es una opción que considero muy interesante de cara al futuro.

1.3. Objetivos y alcance del proyecto

Para poder considerar este trabajo como exitoso deberán cumplirse los siguientes objetivos de alto nivel:

1. El proyecto se desarrollará de forma exitosa de acuerdo con los requisitos y la planificación descritos en el capítulo 3

2. Las actividades mínimas que debe desempeñar un bot en un dispositivo son la persistencia, el mando y control mediante un servicio web bien conocido, la ejecución de tareas y su propósito principal el espionaje mediante extracción de información

3. Se utilizará como servidor central una máquina virtual Kali Linux encargada de ejecutar el proceso correspondiente al bot master, sin embargo se permite realizar pruebas en otros sistemas operativos evaluando esto como positivo.

4. El bot se deberá desarrollar de forma funcional para el sistema operativo android 12 permitiéndose realizar pruebas en otras versiones evaluando esto como positivo.

5. Las comunicaciones entre los bots y el bot master se realizará a través de una base de datos en la nube, firebase.

1.4. Objetivos personales

A través de la realización de este trabajo he establecido una serie de objetivos de carácter personal que, como futuro ingeniero, considero de gran importancia para el desarrollo de mis habilidades y competencias:

1. Investigar y aprender sobre las botnets así como sobre su funcionamiento, desarrollo, implementación, ciclo de vida y posibilidades.
2. Mejorar mi experiencia y conocimientos como desarrollador en Python (orientado a la creación de scripts) así como en java y kotlin (orientados al desarrollo de aplicaciones)
3. Ampliar mis conocimientos de administración de sistemas operativos y ciberseguridad
4. Aumentar mis conocimientos sobre el sistema operativo Android utilizado como víctima de la botnet
5. Aumentar mis conocimientos sobre Kali Linux utilizado como servidor central para el control de los bots
6. Utilizar scrum adaptado a proyectos individuales como metodología de trabajo
7. Aprender más sobre el proceso de ingeniería y el resto de habilidades adquiridas durante la realización del grado en ingeniería informática a las que de uso durante la realización del proyecto.

1.5. Estructura de la memoria

De acuerdo con los requisitos exigidos en la guía docente de la asignatura [50], este documento se estructura de la siguiente forma:

Capítulo 2 Estado del arte: Referencia el marco teórico en el que se basa la investigación y el estado actual del mercado en el que se encuentra el producto que se desarrollará durante la elaboración de este proyecto

Capítulo 3 Requisitos y planificación: Describe la lista de requisitos que moldearán el comportamiento del sistema. Además se realizará la planificación del proyecto teniendo en cuenta la metodología a seguir y los posibles riesgos encontrados.

Capítulo 4 Análisis: Describe el procedimiento de análisis que se ha seguido para la correcta implementación del sistema cubriendo aspectos como el desarrollo de los casos de uso y de las primeras fases de la ingeniería de Software

Capítulo 5 tecnologías utilizadas: Describe todas las tecnologías que han supuesto un papel importante para el desarrollo del proyecto, por ejemplo entre ellas encontraremos los lenguajes de programación utilizados o las herramientas para el seguimiento del proyecto.

Capítulo 6 Diseño: Describe las fases del proceso de ingeniería de software que han sido realizadas a lo largo del proyecto.

Capítulo 7 Implementación y pruebas: Describe como se han implementado algunos de los aspectos más característicos del sistema y se explica en detalle el plan de pruebas

Capítulo 8 Seguimiento del proyecto: Describe el éxito (o fracaso) del proyecto analizando en profundidad cada una de sus fases y el desempeño que se ha aplicado en cada una de estas

Capítulo 9 Conclusiones: Describe las conclusiones finales que se obtienen al dar el proyecto por finalizado.

Anexo A Manuales: Incluye manuales de instalación y despliegue y un manual de usuario para el software desarrollado.

Anexo B Resumen de enlaces adicionales: Incluye enlaces de interés sobre el proyecto, como el repositorio de código, ficheros .apk o contenido relativo al seguimiento.

Capítulo 2

Estado del arte

2.1. ¿Qué es realmente una botnet?

Antes de que hable brevemente de la historia de las botnets, es necesario ampliar la definición y lo que sabemos sobre ellas. Una botnet es una red de bots, tradicionalmente esto se refiere a una colección de ordenadores o servidores comprometidos (aunque también podemos encontrar dispositivos IoT y sistemas móviles), a menudo denominados zombis. Estos pueden haber sido infectados a través de diferentes escenarios como puede ser mediante la instalación de malware a través de P2P (por ejemplo, un cliente torrent) o cliente-servidor (por ejemplo, un enlace de descarga), mediante phising (por ejemplo, la estafa del CEO), o de una forma menos común, ser directamente introducidos en un sistema por un usuario del mismo (por ejemplo, por un empleado insatisfecho con su trabajo), cualquiera de estos enfoques permite a un atacante controlar los dispositivos infectados y realizar tareas en su nombre.

Actualmente los bots no siempre juegan el papel de delincuente, existen para ellos varios propósitos capaces de aportar un servicio o beneficio para el usuario, como por ejemplo, los chatbots que se utilizan en las paginas de comercio electrónico para ayudar al usuario a encontrar el producto deseado o los web crawlers encargados de analizar la red de internet. Sin embargo, el propósito de las botnets es generalmente malicioso y oscuro, el concepto es el mismo, realizar tareas que los humanos no pueden realizar, ya sea por tratarse de tareas repetitivas, porque la carga de trabajo es muy elevada o simplemente porque se necesita cierto grado de automatismo pero aplicado a la delincuencia, estafa, espionaje, robos... [8]

Para comprender el objetivo de una botnet vamos a definir una serie de funciones que comúnmente presentan:

- Infección: Se refiere la capacidad de conducir a su víctima, antes de que esta se encuentre bajo el control de la botnet hacia un destino infestado por botnets. Para ello se puede ayudar de técnicas como la esteganografía en imágenes para la inyección de código a

través de email.

- Propagación: Corresponde a la función de poder moverse de una víctima a otra acaparando cuantos más terminales infectados como sea posible. Esta función puede realizarse automáticamente, por ejemplo, a través del escaneo del puerto de sus vecinos combinado con un troyano de acceso remoto

- Mando y control: También llamado çanal C2"se utiliza para la comunicación de los bots con el bot master. Implementa numerosas técnicas como la migración de DNS (DNS fluxing), uso de proxys, uso de peer to peer, la utilización de la dark web y el uso de técnicas avanzas de encriptación. El propósito de esta función es ocultarse y ser sigilosas, camuflándose entre los protocolos comunes de internet.

- Propósito principal o ataque: Por lo general las botnets tienen un propósito económico de tal manera que la persona que la utiliza obtenga un beneficio ya sea de forma directa al utilizarla o indirecta al venderla o permitir su uso (on-demand software). Algunos ejemplos de este propósito pueden ser los ataques de denegación de servicios distribuidos DDoS, el espionaje de aplicaciones de mensajería o el minado de criptomonedas. [12]

Para la ciberseguridad las botnets presentan una serie de desafíos que las caracterizan frente a otro tipo de amenazas. Primero, las botnets son una amenaza que continuamente evoluciona y aumenta las posibilidades para la delincuencia que estas ofrece, adaptándose a los esfuerzos para mitigarlas. Otra de sus particularidades es que son relativamente fáciles de utilizar, sobre todo para un comprador que encarga el software a demanda para un propósito específico. Uno de los principales retos es la localización geográfica de sus bots, que al englobar comúnmente varios países, entra en conflicto la legislación y los esfuerzos colaborativos por contenerlas. Por último, el punto de partida de una botnet, es decir, un dispositivo vulnerable es fácil de encontrar, lo que permite un escalado horizontal que a menudo tiene menos medidas de seguridad que el acceso directo a un sistema bien defendido.

Para defendernos de las botnets existe una serie de indicaciones y metodologías que permite reducir la probabilidad de que se materialice una amenaza, la primera es la concienciación que permite a las partes implicadas (entiéndanse como afectadas) saber que hacer frente a este tipo de amenazas, como colaborar entre entidades afectadas y aplicar medidas para enfrentarlas. La siguiente consiste en aplicar un enfoque colaborativo en el que el intercambio de inteligencia, datos, buenas prácticas y opciones de mitigación se comparten entre varios colaboradores proactivos. [20]

2.2. Breve historia de las botnets y botnets de importancia histórica

Los bots no surgen directamente para el propósito de las botnets si no que han estado presentes en Internet desde finales de los años 80, realizando tareas como mantener vivos los servicios de los servidores de juegos y de chat, por ejemplo, en IRC (Internet Relay Chat) un protocolo de comunicación en tiempo real basado en texto que permite debates entre

dos o más personas. No obstante, en 1999 ya había múltiples botnets en acción, vigilando los canales de IRC en busca de máquinas vulnerables. Al año siguiente, la red IRC difundía "Gtbot", la primera red de bots de denegación de servicio. Para entonces, los bots también se habían expandido más allá del IRC a HTTP, ICMP y otros espacios. El primer bot de motor de búsqueda, WebCrawler, fue comprado y desplegado por AOL (America Online) en 1995.

Las botnets no lograron un impacto para las empresas hasta alrededor del año 2000, cuando Khan C. Smith, consiguió unos 3 millones de dólares de beneficios gracias a 1.250 millones de correos electrónicos maliciosos (phising) que le permitieron acceder a números de tarjetas de crédito y contraseñas robadas de usuarios de EarthLink, un proveedor de servicios de internet. Este ataque dio lugar a una de las mayores sentencias por spam fraudulento desde la creación de Internet, obligando al atacante a indemnizar a la compañía ISP con 25 millones de dólares por el delito.

2007 fue un año de gran movimiento para las botnets, Internet se vio afectado por el gusano Storm, que fue la primera red de bots P2P del mundo. Aunque se llamaba gusano (malware cuya función consiste en propagarse y afectar al mayor número de dispositivos posible), Storm era en realidad una red de bots que podía seguir las órdenes emitidas por un control central a manos de un criminal, es decir, el bot master. Storm era capaz de enviar spam, lanzar ataques DoS, etc. La red de bots Cutwail también se creó en 2007 y en un momento dado fue responsable de casi la mitad del spam a nivel global. Otra de estas redes de bots, Grum, operaba con varios servidores de control, lo que dificultaba a las fuerzas de seguridad localizar y desmantelar la red. Grum era responsable del 18 % del spam mundial en 2012.

Las botnets siguieron creciendo y haciéndose más inteligentes. Kraken llegó a Internet en 2008, infectó al 10 % de las empresas de la lista Fortune 500 (una lista publicada de forma anual por la revista Fortune que presenta las 500 mayores empresas estadounidenses de capital abierto a cualquier inversor) y fue la primera red de bots que se encontró utilizando técnicas de evasión que le permitían ocultarse del software antimalware, sistemas antivirus y firewalls. Esta fué desmantelada por las fuerzas de seguridad, pero volvió con sus capacidades de infección de dispositivos mejoradas con otro nombre, Butterfly. Esta también se utilizó para crear una red de bots formada por 12,7 millones de máquinas.

Mirai, una botnet enorme, se utilizó para derribar varios sitios web de elevadas magnitudes. En lugar de atacar los sitios directamente, Mirai atacó una pieza clave de la infraestructura de tal manera que incluso sitios como Reddit, Spotify y Twitter fueron inaccesibles. Mirai atacó a máquinas más allá del PC, teléfonos móviles y routers, los dispositivos de la Internet de las Cosas incluyendo las cámaras, los monitores de bebés y otros dispositivos inteligentes que se ejecutan en versiones reducidas de Linux.

Botnets como Methbot (2016) y Smominru (2017) utilizan métodos más novedosos de obtención de beneficios para sus propietarios. Methbot era la mayor red de bots orientada al fraude publicitario, esta, cuando fue desmantelada en 2016, generaba entre 3 y 6 millones de dólares al día en esquemas fraudulentos de publicidad de pago por clic (PPC). Smominru utilizaba EternalBlue y otros exploits para instalar software de minado de criptomonedas en máquinas vulnerables. Esta botnet había minado ilegalmente unos 2,5 millones de dólares

hasta mayo de 2018. Smominru es también la desagradable botnet que propagó el ataque WannaCry (un ransomware de cifrado, un tipo de software malicioso que los cibercriminales utilizan a fin de extorsionar a un usuario para que pague y desbloquear sus archivos) en mayo de 2017. [8]

2.3. Android

Otro de los pilares que caracteriza este trabajo es el sistema operativo android, presente principalmente en dispositivos móviles como teléfonos, tabletas, relojes y hasta coches inteligentes. Los componentes mas importantes del sistema operativo android son los siguientes: [22]

- Núcleo Linux: El kernel de Android es Linux y actúa como una capa de abstracción (oculta los detalles de implementación) entre el hardware del dispositivo y las aplicaciones instaladas. Es característico de Linux tanto la seguridad, como la gestión de memoria, como la gestión de procesos, pila de red, controladores, etc que hereda de este Android
- Runtime: Para ejecutar el conjunto de bibliotecas que ofrecen la mayoría de las funcionalidades con las que trabajará Android, cada aplicación tiene su propio proceso con su instancia a la máquina virtual. La máquinas virtuales le permite ejecutar una copia del sistema operativo en una ventana de independiente, entre otras cosas, se utiliza para ejecutar el software en un entorno seguro y aislado.
- Bibliotecas: El sistema operativo Android incluye un conjunto de bibliotecas de C o C++ que son utilizadas por varios componentes del sistema para ejecutar funcionalidades. Los desarrolladores las utilizan mediante el framework de Android. Entre estas bibliotecas encontramos System C, SQLite...
- Framework de aplicaciones: Todos los desarrolladores tienen acceso a las mismas APIs publicas y estas son utilizadas por las aplicaciones, la reutilización de componentes es algo recomendable. Cualquier aplicación puede publicar sus capacidades y que otras aplicaciones puedan reutilizarlas dentro de unas reglas de seguridad.
- Aplicaciones: Son los programas que ejecutan los usuarios para poder utilizar las funcionalidades de su dispositivo, suelen estar programadas en java o kotlin. Entre ellas encontramos funcionalidades como el correo electrónico, servicios de mensajería instantánea, galería de fotos, etc.

2.4. Estudio de mercado actual

De acuerdo con el articulo publicado por "Spamhaus"[43] en el cuarto trimestre de 2021 se produjo un aumento del 23 % en el número de nuevas botnets encontradas por los expertos de dicha organización, además, han observado una peculiaridad bastante alarmante,

normalmente los atacantes que utilizan las botnets realizaban sus comunicaciones utilizando el protocolo HTTPS, en este caso se ha encontrado que utilizan DNS sobre HTTPS (DoH) y abusan de grandes proveedores de DoH, como Google y Alibaba. Esto es un gran problema porque se pierde la visibilidad que se tenia sobre las peticiones realizadas por las botnets, DoH está pensado para proteger a los usuarios pero no se esperaba que los criminales pudieran usarlo para protegerse a ellos mismos, no se puede determinar su dirección IP y por tanto uno de los métodos mas utilizados que era bloquear la larga lista de IPs infectadas ya no es útil.

La mayoría de los servidores de botnets está alojada en Rusia conformando un 30 % aproximadamente del total seguido por Estados Unidos, y Alemania. Se incorporan a los puestos mas altos del ranking varios países de habla hispana latinoamericanos siendo México el más destacado. De momento España no se encuentra entre las 20 posiciones de la lista.

Rank	Country	Q3 2021	Q4 2021	% Change Q on Q
#1	Russia	381	854	124%
#2	United States	301	384	28%
#3	Germany	170	230	35%
#4	Mexico	182	186	2%
#5	Saudi Arabia	117	180	54%
#6	Uruguay	63	177	181%
#7	Netherlands	273	164	-40%
#8	Dominican Rep	96	110	15%
#9	Brazil	86	92	7%
#10	Latvia	58	69	19%

Rank	Country	Q3 2021	Q4 2021	% Change Q on Q
#11	Czech Republic	40	66	65%
#12	Ukraine	-	64	New Entry
#13	United Kingdom	39	61	56%
#14	France	123	60	-51%
#15	Bulgaria	-	56	New Entry
#16	Moldova	49	50	2%
#17	Seychelles	-	34	New Entry
#18	Hong Kong	-	28	New Entry
#19	Sweden	38	26	-32%
#20	Romania	33	24	-27%

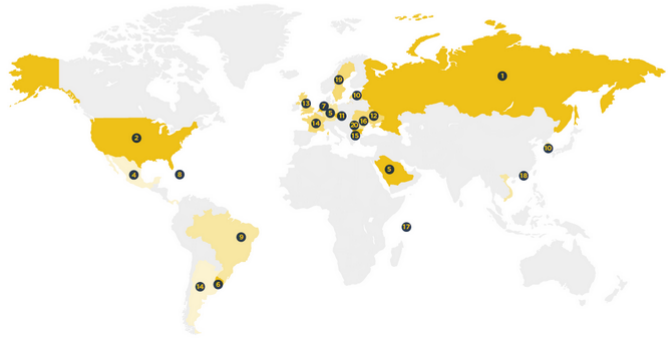

Figura 2.1: Numero de botnets detectadas por país en Q4 2021

Respecto al malware asociado al propósito de la botnet encontramos como fin más popular el robo de credenciales, seguido de herramientas de acceso remoto y dropper (instalación de puertas traseras para realizar ataques más complejos).

Rank	Q3 2021	Q4 2021	% Change	Malware Family	Description
#1	210	164	-22%	RedLine	Credential Stealer
#2	108	102	-6%	Loki	Credential Stealer
#3	121	91	-25%	AsyncRAT	Remote Access Tool (RAT)
#4	-	86	New Entry	GCleaner	Dropper
#5	93	75	-19%	Raccoon	Credential Stealer
#6	43	65	51%	VjwOrm	Remote Access Tool (RAT)
#7	41	43	5%	CryptBot	Backdoor
#8	136	37	-73%	BitRAT	Remote Access Tool (RAT)
#9	71	36	-49%	NjRAT	Remote Access Tool (RAT)
#10	-	32	New Entry	DCRat	Remote Access Tool (RAT)
#11	26	29	12%	Tofsee	Spambot
#11	40	29	-28%	Remocs	Remote Access Tool (RAT)
#13	50	28	-44%	Smoke Loader	Dropper
#14	-	27	New Entry	Arkei	Credential Stealer
#15	-	21	New Entry	TrickBot	Backdoor
#16	-	18	New Entry	Socelars	Credential Stealer
#16	55	18	-67%	CobaltStrike	Backdoor
#18	23	17	-26%	Gozi	E-banking Trojan
#18	37	17	-54%	NanoCore	Remote Access Tool (RAT)
#18	30	17	-43%	STRRAT	Remote Access Tool (RAT)

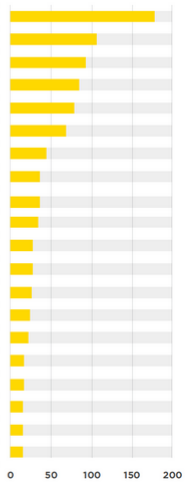

Figura 2.2: Malware más común como propósito de la botnet en Q4 2021

Los dominios más afectados por botnets son aquellos con las terminaciones .com, .top y .xyz siendo claramente dominante .com. Esta información nos permite aplicar medidas de protección frente a esta amenaza permitiendo filtrar el tráfico de una manera más eficiente

Rank	Q3 2021	Q4 2021	% Change	TLD	Note
#1	3730	3719	-0.2%	com	gTLD
#2	829	715	-14%	top	gTLD
#3	833	396	-52%	xyz	gTLD
#4	-	223	New Entry	xxx	gTLD
#5	132	143	8%	ga	Originally ccTLD, now effectively gTLD
#6	665	136	-80%	net	gTLD
#7	330	133	-60%	ru	ccTLD
#8	183	122	-33%	tk	Originally ccTLD, now effectively gTLD
#9	265	116	-56%	org	gTLD
#10	538	108	-80%	buzz	gTLD
#11	178	103	-42%	info	gTLD
#12	98	97	-1%	cf	Originally ccTLD, now effectively gTLD
#13	123	87	-29%	ml	Originally ccTLD, now effectively gTLD
#14	-	75	New Entry	site	gTLD
#15	-	70	New Entry	one	gTLD
#16	-	56	New Entry	gq	Originally ccTLD, now effectively gTLD
#17	82	52	-37%	cloud	gTLD
#18	-	51	New Entry	sbs	gTLD
#19	170	45	-74%	br	ccTLD
#20	-	44	New Entry	de	ccTLD

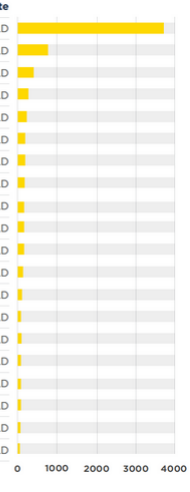

Figura 2.3: Dominios más comprometidos por botnets en Q4 2021

Hablando de Android, este es un sistema operativo que cuenta con un gran numero de versiones, según los datos del periódico el español a noviembre de 2021 [4] la versión 11 es utilizada por el 24.3 % y la versión 10 un 26.5 % . Esto es de utilidad para estudiar en que

versión es mas viable el desarrollo de la botnet

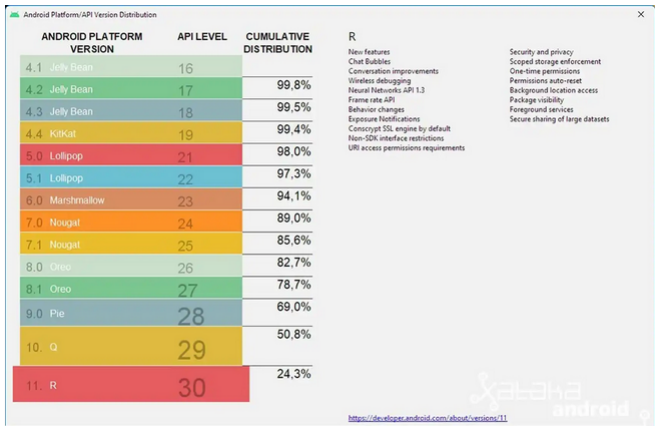

Figura 2.4: Distribución acumulada de versión de android hasta la versión 11 en 2021

Respecto al sistema operativo de los sistemas móviles android es el más popular destacando por su presencia en el 84.1 % de los terminales seguido por IOS en un 15.9 % segun "statista"[35]. Esto da un argumento valido para realizar la botnet enfocada a los dispositivos que portan este sistema operativo ya que se podrá infectar una mayor cantidad de dispositivos y se tendrá mas información a la hora de realizar tareas de investigación.

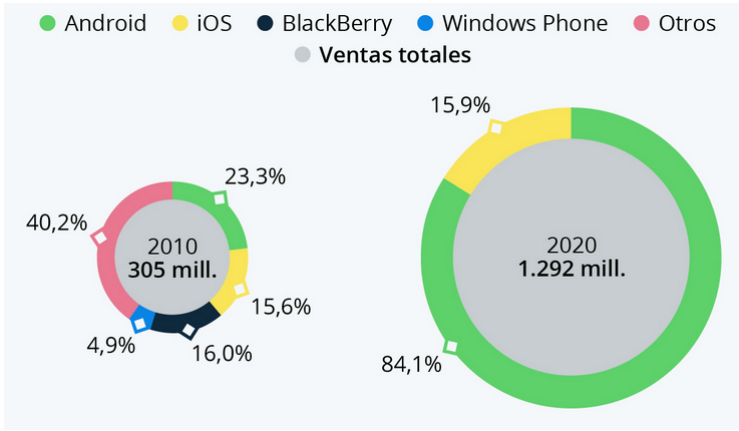

Figura 2.5: Cuota de mercado de SO en dispositivos móviles

Capítulo 3

Requisitos y Planificación

3.1. Introducción

En esta sección se enumeran los diferentes requisitos funcionales, requisitos no funcionales, requisitos de la información y requisitos de seguridad y privacidad que se persiguen en el proyecto así como la planificación y gestión del mismo. Entrando en detalle la definición de los diferentes requisitos puede verse como:

- Requisitos funcionales, es decir, las funcionalidades del sistema y el comportamiento que debe de presentar.
- Requisitos no funcionales, más enfocados a la implementación de las diferentes características del sistema.
- Requisitos de la información, estos especifican los detalles de la implementación a la hora de almacenar datos.
- Requisitos de seguridad y privacidad, en los que se detalla la explotación de las características del sistema operativo de la víctima para evadir las defensas de este

3.2. Requisitos funcionales

En esta sección se explica en detalle los servicios que deberá ser capaz de prestar el sistema. A partir de este punto se me referiré al software malicioso instalado en los dispositivos móviles como el bot y al software de control ubicado en el servidor C2 que se ejecuta en la máquina virtual Kali Linux como el bot master, de esta forma facilitando la lectura.

ID	Nombre	Descripción
RF-01	Tomar notas	El bot será capaz de tomar notas escritas por el usuario, imitando la funcionalidad de agenda
RF-02	Almacenar notas	El bot deberá permitir elegir la fecha en la que se almacenan las notas que escribe el usuario
RF-03	Ver notas	El bot permitirá al usuario visualizar las notas de texto que ha tomado previamente
RF-04	Eliminar notas	El bot permitirá al usuario eliminar una de las notas tomadas previamente
RF-05	Mando y control	El bot deberá poder comunicarse con el bot master
RF-06	Envío de ImAlive	El bot deberá poder notificar que está en funcionamiento y a la espera de recibir órdenes al bot master
RF-07	Envío de SMS	El bot deberá poder obtener la lista de SMS del dispositivo
RF-08	Envío de contactos	El bot deberá poder obtener la lista de contactos y sus números de teléfono existente en el dispositivo
RF-09	Envío de características	El bot deberá poder obtener las características del dispositivo, como por ejemplo, la versión de la API, la marca, el modelo y la versión de android
RF-10	Envío de localización	El bot deberá poder obtener la localización del dispositivo, es decir, las coordenadas GPS
RF-11	Envío de notas	El bot deberá ser capaz de obtener las notas que guarde el usuario en la funcionalidad integrada en la aplicación maliciosa
RF-12	Envío de clipboard	El bot deberá ser capaz de obtener el contenido copiado en el portapapeles del dispositivo
RF-13	Ejecución de comandos	El bot permitirá la ejecución remota de comandos abd
RF-14	Recibir orden	El bot deberá ser capaz de realizar sus funciones al recibir la orden que lo especifique
RF-15	Apagar bot	El bot deberá ser capaz de apagarse y dejar de realizar sus funcionalidades maliciosas siendo aun capaz de mostrar su interfaz

Tabla 3.1: Lista de requisitos funcionales del bot

ID	Nombre	Descripción
RF-16	Control de bots	El bot master deberá poder comunicarse con el bot
RF-17	Administración de información	El bot master permitirá coordinar todas las ordenes, bots e información existentes en el sistema
RF-18	Selección de bot	El bot master será capaz de elegir de que bot y que información quiere obtener o mostrar en pantalla
RF-19	Solicitar GPS	El bot master será capaz de solicitar los datos del GPS del dispositivo
RF-20	Solicitar contactos	El bot master será capaz de solicitar y mostrar al atacante la lista de contactos del dispositivo
RF-21	Solicitar SMS	El bot master será capaz de solicitar y mostrar al atacante la lista de SMS del dispositivo
RF-22	Solicitar características	El bot master será capaz de solicitar y mostrar al atacante las características del dispositivo
RF-23	Ejecutar comando remoto	El bot master será capaz de solicitar y mostrar las atacante la ejecución remota de un comando abd en el dispositivo
RF-24	Apagar bot remoto	El bot master será capaz de solicitar que el bot deje de realizar su funcionalidad maliciosa en segundo plano
RF-25	Ver mapa coordenadas	El bot master será capaz de mostrar los datos recibidos de una solicitud GPS enseñando en un mapa la geolocalización del dispositivo
RF-26	Ver bots vivos	El bot master será capaz de determinar que bots están disponibles
RF-27	Recuperar datos	El bot master será capaz de recuperar los datos almacenados en peticiones previas
RF-28	Nueva orden permanente	El bot master será capaz de enviar una orden global permanente para que todos los bots realicen de forma continua la misma acción
RF-29	Eliminar orden permanente	El bot master será capaz de eliminar una orden global permanente

Tabla 3.2: Lista de requisitos funcionales del bot master

ID	Nombre	Descripción
RNF-01	Pantalla apagada	El bot deberá seguir funcionando aunque la pantalla del dispositivo esté apagada
RNF-02	Vista inactiva	El bot deberá seguir funcionando aunque la pantalla muestre una vista distinta a la interfaz de la aplicación
RNF-03	App cerrada	El bot deberá seguir funcionando aunque el dispositivo tenga la aplicación maliciosa cerrada
RNF-04	Sobrevivir reinicio	El bot deberá seguir funcionando aunque el dispositivo se haya reiniciado
RNF-05	Segundo plano	El bot deberá persistir en segundo plano sin que el usuario pueda notar su presencia
RNF-06	Toma de permisos	El bot deberá ser capaz de recoger los permisos mínimos para la ejecución de las tareas maliciosas
RNF-07	Uso de servicios web	El bot debe comunicarse de forma segura con el bot master a través de un servicio web conocido, en este caso, firebase[2]
RNF-08	Sin notificaciones	El bot no deberá mostrar notificaciones ni alertas de ningún tipo alertando de sus servicios en segundo plano
RNF-09	Notificar bot vivo	El bot enviará una señal ImAlive cada 60 segundos aproximadamente para confirmar que sigue vivo
RNF-10	Realizar tareas	El bot comprobará y realizará sus tareas si se ha especificado la orden cada 60 segundos
RNF-11	Versión de android	El bot será capaz de funcionar en dispositivos con la versión 12 de android
RNF-12	Almacenamiento de información	Se utilizará una base de datos en la nube, no relacional, como es Firebase

Tabla 3.3: Lista de requisitos no funcionales del bot

ID	Nombre	Descripción
RNF-13	Ejecución en terminal	El bot master deberá poderse ejecutar en interfaces de consola como por ejemplo el cmd de windows o la terminal de kali linux
RNF-14	Menú	El bot master deberá poderse ejecutar de forma continuada y permitir la interacción mediante un menú de opciones
RNF-15	Navegación en mapa	Al recuperar los datos de las coordenadas GPS deberá permitir apreciarlas en un mapa de google maps
RNF-16	Progreso	El ejecutar acciones con una carga computacional pesada, el bot master mostrará una barra de progreso

Tabla 3.4: Lista de requisitos no funcionales del bot master

3.3. Requisitos de la información

ID	Nombre	Descripción
RI-01	ID bot	Se almacenará la información recopilada de cada bot con un identificador único
RI-02	Formato de almacenamiento	Se almacenará el identificador y la hora en tiempo real de todos los datos guardados
RI-03	Texto plano	Los datos se almacenarán en texto plano
RI-04	Uso de la BD	Se almacenará en diferentes colecciones los datos del GPS, SMS, Clipboard, características del dispositivo, contactos, comandos ejecutados, mensajes de ImAlive, las notas del usuario y las ordenes
RI-05	Nueva colección	Se almacenará en un nuevo documento cada petición de ejecución de comandos, envio de ImAlives, Coordenadas GPS, ordenes al dispositivo y las notas del usuario
RI-06	Actualizar colección	Se actualizarán los documentos que contienen los contactos, las características del dispositivo, el portapapeles y los SMS cada vez que se ejecute una orden que devuelva datos nuevos

Tabla 3.5: Lista de requisitos de la información

3.4. Requisitos de seguridad y privacidad

ID	Nombre	Descripción
RSP-01	Alarmas	El bot deberá ejecutar los procesos maliciosos en intervalos de tiempo fijo, en segundo plano y a través de un gestor de alarmas para evitar alertar al usuario
RSP-02	Interacción con el usuario	El usuario del dispositivo infectado con el bot solo podrá acceder a una vista y una funcionalidad completamente diferentes al propósito real del sistema, es decir, espionaje y robo de información
RSP-03	Toma de datos personales	Se permitirá al bot tomar datos personales notificando al usuario mediante las alertas que muestran los permisos
RSP-04	Almacenamiento de datos personales	El bot no notificará ni permitirá la detección de sus actividades durante el almacenamiento de datos personales
RSP-05	Uso de permisos	El bot influenciará en el usuario a la hora de aceptar los permisos en tiempo de ejecución necesarios en el dispositivo para la obtención de la información

Tabla 3.6: Lista de requisitos de privacidad y seguridad

3.5. Planificación y gestión

Para la planificación del proyecto he decidido seleccionar como metodología, Scrum. Esto se debe a las siguientes razones [7]:

- Scrum puede ayudar a los equipos a completar los resultados del proyecto de forma rápida y eficiente.
- Scrum garantiza un uso eficaz del tiempo y del dinero.
- Los grandes proyectos se dividen en sprints fácilmente manejables.
- Los desarrollos se codifican y prueban durante la revisión del sprint.
- Funciona bien para los proyectos de desarrollo rápidos.

Sin embargo, aplicar Scrum en este proyecto presenta una clara desventaja, es una metodología enfocada al trabajo en equipo y este proyecto es de carácter individual. Para solucionar este problema he decidido utilizar como metodología Scrum for One en la que entraremos en detalle a continuación.

3.6. Metodologías Ágiles y Scrum

Tanto Scrum como la adaptación utilizada, Scrum for One, toman sus fundamentos de las metodologías ágiles. Por definición, las metodologías ágiles son aquellas que permiten adaptar la forma de trabajo a las condiciones del proyecto, consiguiendo flexibilidad e inmediatez en la respuesta para amoldar el proyecto y su desarrollo a las circunstancias específicas del entorno [42].

Por tanto se utilizará una metodología ágil para el desarrollo de este proyecto ya que permitirá conducirlo de una manera más flexible, autónoma y eficaz reduciendo los costes e incrementando la productividad.

Las metodologías ágiles cuentan con 12 principios [38], estos son:

1. Satisfacer al cliente mediante la entrega temprana y continua del producto.
2. Posibilidad de modificar los requisitos, incluso en las últimas fases del desarrollo, de esta forma aprovechando la ventaja competitiva.
3. Entregas de software con frecuencia, normalmente esto puede variar desde 2 semanas hasta un par de meses.
4. El equipo debe trabajar conjuntamente a diario
5. Confiar en que las personas pueden realizar el trabajo si se les da el entorno y el apoyo necesario

6. La conversación cara a cara es el mejor método de transmitir información al equipo.
7. Se progresa con software funcional. No es necesario que todo esté perfecto para poder continuar.
8. Se mantendrá un ritmo constante de trabajo hasta la finalización de proyecto
9. La atención continua a la excelencia técnica y al buen diseño mejora la agilidad.
10. Construir el software de forma estratégica y con un propósito con buen fundamento dejando para más adelante tareas menos importantes
11. Las mejores arquitecturas, requisitos y diseños surgen de equipos que permiten a sus miembros expresarse y organizarse de forma autónoma
12. El equipo debe reflexionar sobre cómo ser más eficaz, y modificar su comportamiento en consecuencia.

En Scrum existen varios roles que se deben de tener en cuenta para la correcta implementación de la metodología[26]:

- Product Owner, puede verse como un gestor de requisitos o un cliente que se encarga de gestionar el Product Backlog, manteniendo este bien estructurado, detallado y priorizado. Además tiene que entender perfectamente cuál es la la dirección que se desea tomar para el desarrollo del producto en todo momento, debiendo poder explicar y trasmitir a las partes implicadas cuál es el valor de su inversión.
- Scrum Master, se encarga de gestionar y asegurar que el proceso Scrum se lleva a cabo correctamente, así como de facilitar la ejecución del proceso y sus mecánicas. Genera valor al aplicar correctamente la metodología. También se encarga de eliminar impedimentos que van surgiendo en la organización y que afectan a su capacidad para entregar valor, así como a la integridad de esta metodología. El Scrum Master debe ser el responsable de velar porque Scrum se lleve adelante, transmitiendo sus beneficios a la organización y facilitando su implementación
- El equipo de desarrollo, suele estar formado por entre 3 a 9 profesionales que se encargan de desarrollar el producto, auto-organizándose y auto-gestionándose para conseguir entregar un incremento de software al final del ciclo de desarrollo. El equipo de desarrollo se encargará de crear un incremento terminado a partir de los elementos del Product Backlog seleccionados (Sprint Backlog) durante el Sprint Planning.

En Scrum se realizan diferentes reuniones para ayudar a que se cumplan los diferentes procesos del proyecto, estas son [25]:

- El Sprint Planning al comienzo del Sprint, sirve para inspeccionar el Product Backlog y que el equipo de desarrollo seleccione los Product Backlog Items en los que va a trabajar durante el siguiente Sprint

- Daily Scrums a diario, es una reunión diaria de 15 minutos en la que participa exclusivamente el Development Team. Cada miembro cuenta que hizo el día anterior, que hace el dia actual y posibles impedimentos
- Un Sprint Review al final del Sprint para evaluar el trabajo realizado
- finalmente, una Retrospectiva para evaluar al equipo y proponer mejoras que se apliquen en el siguiente Sprint.
- Adicionalmente se puede incorporar una reunión de Grooming o Refinement, que sirve para, dentro del Sprint, afinar y aclarar ciertas historias de usuario que pudieron quedar pendientes durante el Sprint Planning.

La metodología Scrum pasa por diferentes fases que hacen posible que se lleve a cabo con éxito [41].

- Primero se realiza la planificación, es decir, el Product Backlog, esta es la fase en la que se establecen las tareas prioritarias y donde se obtiene información breve y detallada sobre el proyecto que se va a desarrollar, es necesario para empezar el primer Sprint y puede cambiar y crecer tantas veces como sea necesario. Esta fase la realiza el Product owner en conjunto con el equipo.
- La segunda fase es la de ejecución mediante Sprints, este es la fase de tiempo donde se produce el desarrollo de un producto que es entregable potencialmente. Se puede definir como un mini proyecto en el que todo el equipo de trabajo se focaliza en tareas para alcanzar el objetivo definido en la planificación.
- Por último la fase de control, conocida como Burn Down. En esta fase se mide el progreso del proyecto. En ella, el Scrum Master será el encargado de actualizar los gráficos cuando se finalice cada uno de los Sprint.

3.7. Scrum for One

Como ya se ha visto, los equipos en el ámbito del desarrollo de software, utilizan la metodología Scrum para gestionar el flujo de trabajo, entregar productos viables más rápidamente e iterar, sin embargo, esto no es posible trabajando solo. Es por ello que se necesita realizar una adaptación de la metodología para gestionar el flujo de trabajo y aumentar la productividad [28].

Al ser solo una persona trabajando en el proyecto he de asumir los 3 roles propios de Scrum, por tanto yo seré el Product Owner, el Scrum Master y el equipo de desarrollo, la forma de plantearlo ha sido la siguiente:

- Como Product Owner, definiré claramente el producto final. Identificaré las características y los requisitos, y los registraré con detalle para su posterior consulta. Al asumir este rol debo de ser capaz de analizar cuales son las actividades más importantes para

centrar el esfuerzo de desarrollo en estos dejando de lado aquellos que no sean críticos, siempre teniendo en cuenta que el objetivo final del proyecto es la entrega del producto.

- En general tomar el rol de Scrum master individualmente consiste en identificar los problemas y encontrar maneras de solucionarlos. Por ejemplo, para evitar distracciones aplicaré de la técnica Pomodoro 25-5[37].
- Tomar el rol de equipo de desarrollo es mas fácil ya que simplemente haré el trabajo que haría de todos modos. Sin embargo, un equipo de desarrollo de Scrum está auto gestionado y auto motivado para completar el sprint, por tanto debo adoptar esa actitud

Adicionalmente, realizaré un seguimiento de los fallos del proceso de Scrum para abordarlos al final del sprint y de esta forma generar un feedback como base para realizar mejoras. Para ello se llevará un registro a lo largo de los sprints.

Respecto a la planificación del proceso de Scrum las diferentes fases se realizarán de la siguiente forma:

- Primero se planificará el proyecto definiendo los requisitos como Product Owner, planificaré el proceso y crearé objetivos razonables para cada sprint como el Scrum Master y el equipo de desarrollo
- Después realizaré los Daily Scrum para revisar el trabajo del día anterior, sus éxitos y sus impedimentos. Además, actuando como Scrum Master propondré soluciones a los problemas identificados. Por último planificaré que se hará en el día actual.
- Por último se realizará la revisión del sprint para poner en orden los resultados y los problemas que se han encontrado durante el sprint y se alinearán los avances con los requisitos definidos en el proyecto.

A lo largo del proceso crearé los siguientes items de Scrum que me ayudarán a mejorar la calidad del producto final:

- El Product Backlog, se creará al principio de un proyecto. En el enumeraré todas las tareas que deben de estar realizadas al final del proyecto. Al estar adaptado de forma individual seré el único responsable de completar cada tarea. Sin embargo me permitiré reorganizar la lista de tareas pendientes según sea necesario.
- El tablero Scrum, es una referencia visual rápida de lo que ya se ha logrado y de las tareas que aún deben completarse en cada sprint. Evita que se produzcan cambios en medio del sprint
- Burndown charts, útiles para ver la eficiencia de los sprints. Estos gráficos hacen que sea más fácil ver el progreso con un vistazo rápido.

Finalmente me gustaría recalcar que al adaptar la metodología Scrum para gestionar el trabajo individual, es posible que no pueda recrear perfectamente un proceso de Scrum en equipo, pero podré beneficiarme de muchos de los puntos fuertes de Scrum.

3.8. Product Backlog

Como ya se ha comentado en las secciones previas a continuación se detalla el product backlog, existen herramientas, procedimientos y variaciones para crear este. Yo he seleccionado una bastante sencilla de entender rápidamente con tan solo echar un vistazo.

La planificación de proyecto consistirá en 8 sprints, cada uno de ellos con una longitud de 2 semanas. El proyecto comienza en la tercera semana de febrero y acabará a finales de la primera semana de junio, de acuerdo con el inicio del segundo cuatrimestre de la escuela de Ingeniería Informática y la fecha fin para presentar el TFG en la convocatoria ordinaria

SPRINT	FECHA INICIO	FECHA FIN
1	14/02/2022	27/02/2022
2	28/02/2022	13/03/2022
3	14/03/2022	27/03/2022
4	28/03/2022	10/04/2022
5	11/04/2022	24/04/2022
6	25/04/2022	08/05/2022
7	09/05/2022	22/05/2022
8	23/05/2022	05/06/2022

Figura 3.1: Duración y fechas de todos los sprints

En las siguientes subsecciones podremos ver el desglose del Product Backlog por sprints y como se ha compuesto el Scrum board particular de cada uno de ellos. Más adelante, en la fase de seguimiento se utilizarán estos tableros para calcular de una forma más sencilla el progreso en el proyecto y el éxito del mismo

3.8.1. Sprint 1

He decidido destinar este primer Sprint para hacerme a mi mismo una introducción al proyecto. Como suele ser común al acceder a un nuevo puesto laboral en el cual tengas que trabajar con un sistema que no conoces ha de realizarse una adaptación al mismo, esto suele realizarse leyendo la documentación disponible y entendiendo su propósito. Es por esto que las tareas que tendré que realizar durante estas 2 primeras semanas son aquellas destinadas a leer, informarme, documentarme y aprender todo lo relacionado con el software a desarrollar, sus posibles competidores y estado del arte en el mercado. Por otro lado al tratarse de un proyecto en el que el desarrollo de la memoria bibliográfica es crítica he añadido tareas para la redacción de la misma. En este Sprint toca redactar la introducción y el estado del arte, de acuerdo con los temas que se investigaran como tarea. A continuación veremos las tareas del product Backlog desglosadas y el Scrum board para el sprint.

IDENTIFICACIÓN					TAREAS
ID	SPRINT	FECHA INICIO	FECHA FIN	TEMA	TAREA
T-01	1	14/02/2022	27/02/2022	Investigación	Estudiar la viabilidad del proyecto para tomar una decisión sobre el enfoque del proyecto y comenzar a investigar información, referencias y fuentes de utilidad
T-02	1	14/02/2022	27/02/2022	Memoria	Redactar la introducción al proyecto
T-03	1	14/02/2022	27/02/2022	Memoria	Redactar el estado del arte del producto

Figura 3.2: Tareas del sprint 1

3.8.2. Sprint 2

El segundo sprint está enfocado en la planificación y la toma de requisitos, es por eso que entre las tareas que encontramos se encuentra la toma de todos los requisitos del proyecto, que podrían cambiar más adelante de acuerdo con las necesidades de este (pero no pueden cambiar antes de la finalización de un sprint). También se realiza toda la planificación, esto incluye, la selección de una metodología y aplicar sus correspondientes particularidades. Es importante realizar estas tareas lo más pronto posible entre las fases del proyecto porque de ellas depende en gran medida la distribución correcta y aprovechamiento del tiempo disponible. En la finalización de este sprint se debería tener desarrollada la planificación del proyecto, el análisis del comportamiento del sistema a partir de los requisitos y un análisis de riesgo del proyecto teniendo en cuenta la probabilidad y el riesgo de aquellos que se hayan encontrado. Además se documentará todo el proceso en la memoria para continuar del desarrollo de esta.

IDENTIFICACIÓN					TAREAS
ID	SPRINT	FECHA INICIO	FECHA FIN	TEMA	TAREA
T-04	2	28/02/2022	13/03/2022	Investigación	Investigar sobre metodos de planificación de proyectos para escoger un método de planificación optimo para mi proyecto y sus características
T-05	2	28/02/2022	13/03/2022	Planificación	Realizar la planificación del proyecto para definir los sprints, las tareas, los requisitos y los objetivos
T-06	2	28/02/2022	13/03/2022	Planificación	Desarrollar el analisis de riesgos del proyecto
T-07	2	28/02/2022	13/03/2022	Memoria	Continuar la memoria del proyecto para explicar por escrito como se enfocará la planificación del proyecto
T-08	2	28/02/2022	13/03/2022	Ingeniería de s	Realizar la toma de requisitos funcionales para definir las acciones que debe realizar el sistema
T-09	2	28/02/2022	13/03/2022	Ingeniería de s	Realizar la toma de requisitos no funcionales para definir el comportamiento del sistema
T-10	2	28/02/2022	13/03/2022	Ingeniería de s	Realizar la toma de requisitos de la información para definir el comportamiento del sistema durante el almacenamiento de información
T-11	2	28/02/2022	13/03/2022	Ingeniería de s	Realizar la toma de requisitos de seguridad y privacidad para definir el comportamiento del sistema a la hora de aprovechar características propias de la seguridad de los dispositivos
T-12	2	28/02/2022	13/03/2022	Memoria	Documentar la toma de requisitos para explicar estos por escrito

Figura 3.3: Tareas del sprint 2

3.8.3. Sprint 3

El tercer Sprint está dedicado a la primera fase de ingeniería de software del proyecto, en este caso se realizará la fase de análisis, esta incluye el diagrama de casos de uso, el desarrollo de los casos de uso, el modelo de dominio y los diagrama de secuencia. Esta es una fase fundamental en el desarrollo de cualquier producto software y es por esto que se realiza antes de comenzar la programación del sistema. Como tareas adicionales y para evitar quemarme durante el proceso he añadido tareas relacionadas con la investigación de tecnologías que utilizaré durante el desarrollo del proyecto. Estas últimas son menos prioritarias por lo que solo emplearé el tiempo sobrante del resto de tareas para comenzar dicha búsqueda. Si no soy capaz de completarlas me permitiré modificar la lista de tareas durante la revisión final del Sprint para continuar la búsqueda (también de manera poco prioritaria) en el siguiente.

Como es habitual, documentaré el proceso del sprint en la memoria en sus correspondientes secciones.

IDENTIFICACIÓN				TAREAS	
ID	SPRINT	FECHA INICIO	FECHA FIN	TEMA	TAREA
T-13	3	14/03/2022	27/03/2022	Investigación	Investigar tecnologías para Utilizarlas durante el desarrollo del software del proyecto
T-14	3	14/03/2022	27/03/2022	Memoria	Documentar las tecnologías encontradas para explicar su utilidad durante el desarrollo
T-15	3	14/03/2022	27/03/2022	Análisis	Desarrollar el diagrama de casos de uso para plasmar las acciones a las que responderá el sistema
T-16	3	14/03/2022	27/03/2022	Análisis	Desarrollar los casos de uso para extender la definición y el alcance de cada uno de ellos
T-17	3	14/03/2022	27/03/2022	Análisis	Desarrollo de un primer modelo de dominio para comenzar a definirlo como base para futuros sprints
T-18	3	14/03/2022	27/03/2022	Análisis	Desarrollar los diagramas de secuencia de los casos de uso
T-19	3	14/03/2022	27/03/2022	Memoria	Documentar el proceso de análisis para explicar por escrito todo el procedimiento realizado

Figura 3.4: Tareas del sprint 3

3.8.4. Sprint 4

He dejado, en mi opinión, la parte más complicada del proyecto para el cuarto Sprint, esto se debe a que es posible que cualquiera de estas tareas sean más complicadas de lo esperado y tenga que utilizar otro Sprint para continuarlas. Este periodo de dos semanas me enfocaré en la parte de diseño del sistema obteniendo el patrón de diseño, el diseño de la base de datos. También realizaré los diagramas de despliegue, arquitectura y componentes y actualizaré el modelo de dominio al correspondiente en esta fase. Es posible que sea necesario añadir una tarea de investigación-formación para poder preparar mejor los resultados obtenidos durante esta fase. Finalmente documentaré en la memoria todo el proceso realizado.

IDENTIFICACIÓN				TAREAS	
ID	SPRINT	FECHA INICIO	FECHA FIN	TEMA	TAREA
T-20	4	28/03/2022	10/04/2022	Diseño	Definir el patron de diseño seleccionado para el sistema
T-21	4	28/03/2022	10/04/2022	Diseño	Diseñar el diagrama de la base de datos
T-22	4	28/03/2022	10/04/2022	Diseño	Actualizar el modelo de dominio
T-23	4	28/03/2022	10/04/2022	Diseño	Desarrollar el diagrama de componentes
T-24	4	28/03/2022	10/04/2022	Diseño	Desarrollar el diagrama de arquitectura
T-25	4	28/03/2022	10/04/2022	Diseño	Desarrollar el diagrama de despliegue
T-26	4	28/03/2022	10/04/2022	Memoria	Documentar el proceso de diseño para expicar por escrito las decisiones tomadas

Figura 3.5: Tareas del sprint 4

3.8.5. Sprint 5

A partir de este Sprint comienza la fase de desarrollo del producto. Es necesario desarrollar tanto el bot como el bot master por lo que voy a separarlo en dos fases, una en este sprint y otra en el siguiente. Durante estas 2 semanas desarrollaré la interfaz, las funcionalidades del bot, las condiciones de persistencia, evasión, comunicación y recopilación de información. También desarrollare la funcionalidad no maliciosa y la implementación de la base de datos a partir del diagrama realizado en anteriores Sprints. Es posible que el desarrollo del bot se complique más de lo esperado por lo tanto este es el último Sprint con tanta carga de trabajo y contemplaré la posibilidad de dejar las tareas menos prioritarias para los siguientes ya que sus tareas llevarán menos tiempo. Se documentará en la memoria todo el proceso realizado.

IDENTIFICACIÓN				TAREAS	
ID	SPRINT	FECHA INICIO	FECHA FIN	TEMA	TAREA
T-27	5	11/04/2022	24/04/2022	Desarrollo	Desarrollar la interfaz de la aplicación movil
T-28	5	11/04/2022	24/04/2022	Desarrollo	Implementar la funcionalidad no maliciosa de la aplicación movil (agenda, toma de notas)
T-29	5	11/04/2022	24/04/2022	Desarrollo	Implementar la condición de persistencia en el bot
T-30	5	11/04/2022	24/04/2022	Desarrollo	Implementar la comunicación del bot con el bot master
T-31	5	11/04/2022	24/04/2022	Desarrollo	Implementar la funcionalidad de espionaje recopilando información del dispositivo
T-32	5	11/04/2022	24/04/2022	Desarrollo	Implementar la funcionalidad de control del bot
T-33	5	11/04/2022	24/04/2022	Desarrollo	Implementación de la base de datos
T-34	5	11/04/2022	24/04/2022	Memoria	Documentación de la implementación del bot en la memoria del proyecto

Figura 3.6: Tareas del sprint 5

3.8.6. Sprint 6

Una vez desarrollado el Bot, comenzaré con el Bot Master. El desarrollo de este debería ser más sencillo ya que la comunicación y la parte de búsqueda de vulnerabilidades en Android era la más complicada de desarrollar. El bot debe de cumplir varias funciones a partir de los datos almacenados en la base de datos y también debe ser capaz de enviar ordenes para controlar a los bots disponibles. Al ser una fase de desarrollo más sencilla me encargaré de terminar las tareas de anteriores sprints que no haya conseguido finalizar. Además documentaré en la memoria el proceso realizado.

IDENTIFICACIÓN				TAREAS	
ID	SPRINT	FECHA INICIO	FECHA FIN	TEMA	TAREA
T-35	6	25/04/2022	08/05/2022	Desarrollo	Implementación del control de los bots basado en la comunicación ya desarrollada
T-36	6	25/04/2022	08/05/2022	Desarrollo	Implementación de la recogida de información de un dispositivo
T-37	6	25/04/2022	08/05/2022	Desarrollo	Implementación de la selección de información almacenada
T-38	6	25/04/2022	08/05/2022	Desarrollo	Implementación de actualización de la lista de bots vivos en la red
T-39	6	25/04/2022	08/05/2022	Desarrollo	Implementación de la visualización de la información de cara al usuario
T-40	6	25/04/2022	08/05/2022	Memoria	Documentación de la implementación del botmaster en la memoria del proyecto

Figura 3.7: Tareas del sprint 6

3.8.7. Sprint 7

El séptimo Sprint lo dedicaré a realizar las pruebas de la aplicación. Para ello montaré una batería de pruebas con varios dispositivos móviles, ya sean físicos o emulados. Después haré correr la aplicación en ellos para analizar si cumple con las expectativas. Una vez recopilados los resultados realizaré los ajustes pertinentes en la aplicación para que el funcionamiento se ajuste a lo esperado. Todo el proceso se recopilará en la memoria

IDENTIFICACIÓN				TAREAS	
ID	SPRINT	FECHA INICIO	FECHA FIN	TEMA	TAREA
T-41	7	09/05/2022	22/05/2022	Pruebas	Diseño de la batería de pruebas
T-42	7	09/05/2022	22/05/2022	Pruebas	Lanzamiento de la batería de pruebas con el software desarrollado
T-43	7	09/05/2022	22/05/2022	Pruebas	Ajustes del software relativos a los resultados de las pruebas
T-44	7	09/05/2022	22/05/2022	Memoria	Redacción de los resultados de las pruebas en la memoria del proyecto

Figura 3.8: Tareas del sprint 7

3.8.8. Sprint 8

En el último Sprint se encuentra la menor carga de trabajo de todo el proyecto, esto se debe a que el tiempo extra que me sobre de esta fase se utilizará para compensar ciertos retrasos que puedan aparecer en los anteriores. Como tareas propias de esta fase, se documentarán los resultados del seguimiento del proyecto y se terminará la memoria del proyecto, esto incluye correcciones, retoques, análisis de la ortografía y gramática y se escribirá una pequeña conclusión del proyecto. Como puede verse es un Sprint de apoyo al resto basado en el seguimiento real del proyecto y la situación encontrada después de todas estas semanas.

IDENTIFICACIÓN				TAREAS	
ID	SPRINT	FECHA INICIO	FECHA FIN	TEMA	TAREA
T-45	8	23/05/2022	05/06/2022	Memoria	Redacción del seguimiento del proyecto en la memoria
T-46	8	23/05/2022	05/06/2022	Memoria	Redacción de las conclusiones del proyecto en la memoria

Figura 3.9: Tareas del sprint 8

3.9. Análisis de riesgos

En esta sección se definirá una matriz de riesgos como soporte a la planificación del proyecto, adicionalmente se identificarán posibles riesgos de este y se evaluarán de acuerdo con las métricas definidas.

3.9.1. Matriz de riesgos

Una matriz de riesgos es una herramienta que se utiliza para analizar los riesgos del proyecto en función de su probabilidad de materializarse e impacto una vez este ha sucedido. Una vez identificados los riesgos, se utilizarán los valores de estas variables para calcular el impacto general y otorgarle a cada riesgo la prioridad que le corresponda. Existen varios tipos de matrices de riesgo pero en mi caso voy a utilizar la propuesta por Asana Team ya que presenta un enfoque útil para la gestión de riesgos del proyecto [44], aunque podemos encontrar otras como por ejemplo la propuesta por el INCIBE más enfocada a las auditorías de ciberseguridad [19].

Como podemos ver en la figura 3.10, asignaremos a cada riesgo encontrado un impacto, estos pueden tomar los siguientes valores:

- Insignificante (1): Las consecuencias del riesgo no tienen importancia, por ejemplo, se aplaza la reunión con el tutor de prácticas 1 día.
- Menor (2): Las consecuencias del riesgo pueden ser tratadas sin problema, por ejemplo, error en una de las referencias de la memoria.
- Moderada (3): Las consecuencias del riesgo pueden tardar en mitigarse, por ejemplo, una baja médica por 1 día

- Importante (4): Las consecuencias son significativas y pueden causar problemas a largo plazo, por ejemplo, selección errónea del patrón de diseño de la aplicación.
- Catastrófica (5): Las consecuencias de este riesgo serán muy perjudiciales y puede resultar difícil recuperarse, por ejemplo, pérdida de la memoria del proyecto.

Para la probabilidad encontramos diferentes opciones para manejar los distintos rangos entre las que se encuentran:

- Muy improbable (1): El hecho de que este riesgo ocurra es una posibilidad remota, por ejemplo, cambio de la funcionalidad y propósito del proyecto.
- No es probable (2): Existe una gran probabilidad de que este riesgo no ocurra, por ejemplo, se necesita reemplazar el PC en el que se produce el desarrollo.
- Posible (3): Este riesgo podría ocurrir o no. Las probabilidades de que suceda son 50/50, por ejemplo, un retraso en el sprint 7 de la planificación.
- Probable (4): Existe una gran probabilidad de que este riesgo ocurra, por ejemplo, una baja médica de un día.
- Muy probable (5): Estamos seguros de que este riesgo ocurrirá en algún momento, por ejemplo, no puedo dedicar 7 días a la semana al proyecto debido a posibles compromisos que surjan a lo largo de la semana.

Una vez establecidos los valores de impacto y probabilidad podemos calcular el impacto del riesgo como resultado del producto de las variables. $Riesgo = Probabilidad * impacto$

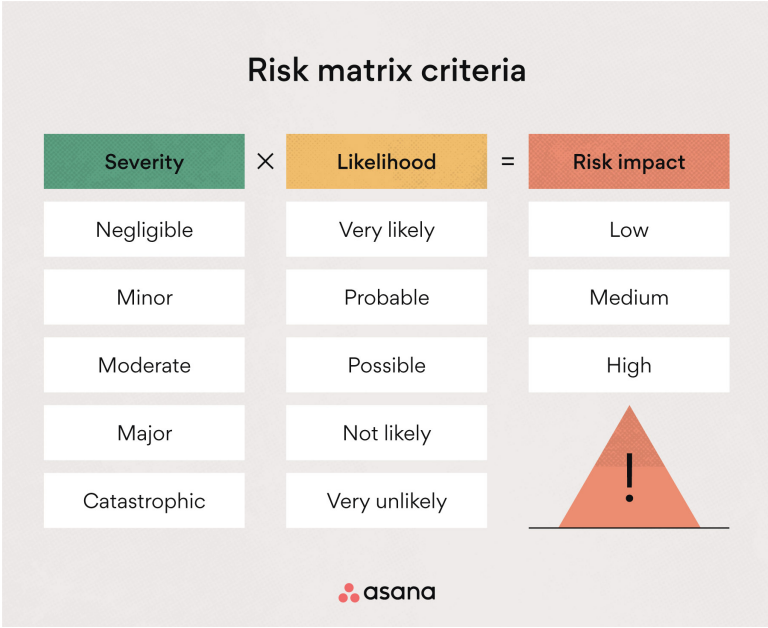

Figura 3.10: Matriz de riesgos propuesta por ASANA[44]

Según el valor obtenido al calcular el riesgo, podemos realizar una clasificación de su importancia. A mayor puntuación más prioridad debe darse y tendrá influencia en su manera de ser tratado, estas serán:

- Bajo, entre 1 y 6 puntos
- Medio, entre 7 y 12 puntos
- Alto, entre 13 y 25 puntos

A la hora de tratar un riesgo podemos tomar una serie de enfoques. Esto dependerá en gran medida de la clasificación que le haya sido dada [21]:

- Aceptar el riesgo y convivir con el, por tanto solo se realizarán acciones correctivas si sucede
- Evitar el riesgo, modificando la planificación o el proceso de la ingeniería para buscar alternativas
- Compartir el riesgo con otra persona u organización, por ejemplo, pagando por un servicio
- Mitigar el riesgo, reduciendo de esta forma la probabilidad y el impacto de que ocurra

3.9.2. Riesgos

A continuación se enumera la lista de riesgos, sus valores de probabilidad, impacto y valor. También se ofrece la conclusión de como se tratará y una serie de contra-medidas para su mitigación en el caso de que se considere necesario.

Descripción del campo	Valor
ID	Risk-01
Nombre	Baja laboral temporal
Descripción	Por causas externas al proyecto es necesario establecer una temporada de baja, por ejemplo, por enfermedad o por accidente laboral. Sufrir este riesgo implica que durante un periodo de tiempo asignado a uno de los sprints del proyecto nadie podrá trabajar en las tareas asignadas propuestas por lo que se efectuará un retraso en la planificación.
Probabilidad	Posible (3)
Impacto	Importante (4)
Valor riesgo	Medio (12)
Respuesta al riesgo	Reducir
Mitigación	La forma en la que este riesgo se mitigará es estableciendo un margen de tiempo libre en la planificación de los Sprints de forma que si uno de ellos se retrasa exista la posibilidad de remontarlo en el siguiente.

Tabla 3.7: Riesgo de proyecto nº1

Descripción del campo	Valor
ID	Risk-02
Nombre	Perdida de datos
Descripción	Por accidente o ataque se pierden datos importantes del proyecto, por ejemplo, el código fuente del software o la memoria del proyecto. A medida que el proyecto avanza este riesgo toma un mayor valor de impacto ya que la cantidad de información perdida aumenta.
Probabilidad	No es probable (2)
Impacto	Catastrófica (5)
Valor riesgo	Medio (10)
Respuesta al riesgo	Reducir
Mitigación	Se tomarán copias de seguridad frecuentemente (mínimo 1 vez al día) y se utilizará un software de control de versiones (github) para mantener el proyecto seguro

Tabla 3.8: Riesgo de proyecto nº2

Descripción del campo	Valor
ID	Risk-03
Nombre	Fecha de entrega imposible
Descripción	Es posible que durante alguno de los Sprints del proyecto por mucho esfuerzo que se aplique no sea posible llegar al final del sprint con todas las tareas completadas.
Probabilidad	Probable (4)
Impacto	Moderada (3)
Valor riesgo	Medio (12)
Respuesta al riesgo	Reducir
Mitigación	Se dejara una menor carga de trabajo en los últimos sprints del proyecto con el fin de utilizar ese margen de tiempo para compensar posibles retrasos que ocurran en alguno de los Sprints

Tabla 3.9: Riesgo de proyecto nº3

Descripción del campo	Valor
ID	Risk-04
Nombre	Productividad Baja
Descripción	A menudo en los proyectos de desarrollo de software la productividad de los empleados puede caer hasta llegar a un punto conocido como burnout en el cual los avances se reducen en gran medida.
Probabilidad	Probable (4)
Impacto	Moderada (3)
Valor riesgo	Medio (12)
Respuesta al riesgo	Reducir
Mitigación	Con el fin de mejorar la productividad he introducido tareas en la planificación que permiten variar entre varias tareas que se diferencian entre si (aunque tengan los objetivos comunes del proyecto). Trabajar por sprints ayuda a la motivación ya que mejora la sensación de que el producto ha de estar terminado para una fecha dividiéndolo en pequeñas tareas que pueden gestionarse más fácilmente

Tabla 3.10: Riesgo de proyecto nº4

Descripción del campo	Valor
ID	Risk-05
Nombre	Problemas con las partes implicadas
Descripción	Al ser un proyecto tutorizado existe más de un punto de vista para el producto entregado que puede diferir del planificado inicialmente. Es por esto que es importante estar preparado para posibles cambios debido al feedback obtenido.
Probabilidad	Posible (3)
Impacto	Moderada (3)
Valor riesgo	Medio (9)
Respuesta al riesgo	Reducir
Mitigación	Para evitar que el feedback obtenido impacte en el proyecto de manera negativa se procurará notificar de los avances de forma frecuente llegando a un acuerdo beneficioso para el proyecto

Tabla 3.11: Riesgo de proyecto nº5

Descripción del campo	Valor
ID	Risk-06
Nombre	Mal planteamiento
Descripción	Alguna de las fases del proyecto se ha interpretado de forma errónea llevando a un mal planteamiento de esta, por ejemplo, la interfaz de usuario tiene un diseño pobre que hace que junto a una explicación deficiente de la misma la aplicación no pueda ser utilizada ya que los usuarios no son capaces de entenderla.
Probabilidad	Posible (3)
Impacto	Moderada (4)
Valor riesgo	Medio (12)
Respuesta al riesgo	Reducir
Mitigación	El feedback, la fase de testing y las interacciones con compañeros ayudan a reconducir el proyecto antes de que sea demasiado costoso realizarlo

Tabla 3.12: Riesgo de proyecto nº6

Descripción del campo	Valor
ID	R-07
Nombre	Dedicación
Descripción	En aquellas fases del proyecto en las que la carga de trabajo no es muy elevada es fácil querer dedicar el tiempo libre a otras actividades ajenas al proyecto, como trabajo, ocio o descanso. Es posible que las actividades externas al proyecto consuman un mayor tiempo del esperado reduciendo las horas empleadas durante los sprints
Probabilidad	Posible (4)
Impacto	Moderada (4)
Valor riesgo	Alto (16)
Respuesta al riesgo	Mitigar
Mitigación	A veces es difícil obtener tiempo aunque se realizará un esfuerzo para conseguirlo reduciendo el numero de posibles actividades y enfocando el proyecto como un trabajo full time

Tabla 3.13: Riesgo de proyecto nº7

Finalmente, en la siguiente imagen se puede ver el resultado de todos los riesgos en la matriz definida anteriormente, para ello se ha utilizado la plantilla ofrecida por ASANA.

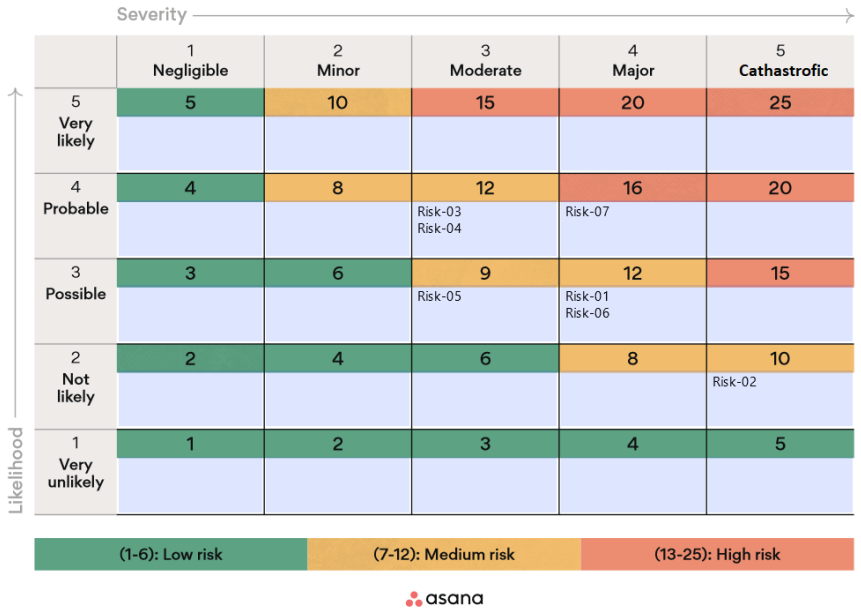

Figura 3.11: Matriz de riesgos completa[44]

Capítulo 4

Análisis

En este capítulo se presenta el proceso de análisis realizado en el proyecto. Esto incluye el diagrama de casos de uso, la secuencia de los casos de uso en detalle y el modelo de dominio del sistema para la fase de análisis, es decir, sin funciones y con la posibilidad de ser ligeramente modificado en siguientes secciones del proyecto.

4.1. Casos de uso

4.1.1. Introducción

A continuación se presentan los casos de uso del sistema, estos dan lugar a los resultados que producen las acciones del usuario al interactuar con el mismo. Los casos de uso se han extraído de los requisitos propuestos anteriormente y para facilitar su comprensión se han plasmado en un diagrama utilizando UML2 [5].

4.1.2. Diagrama de casos de uso

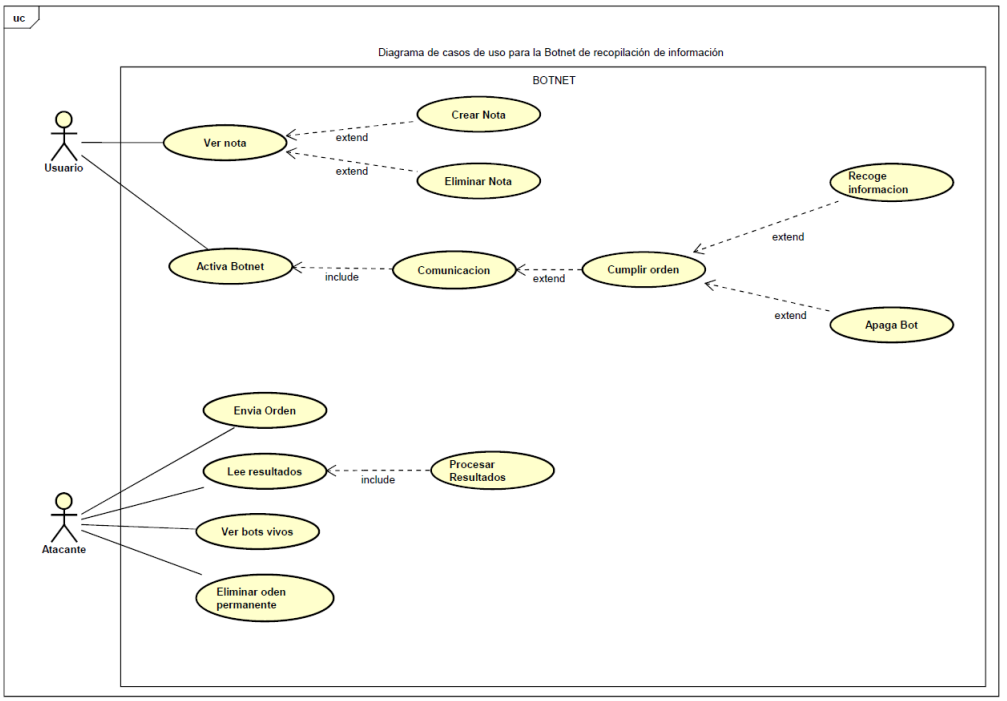

Figura 4.1: Diagrama de casos de uso para el sistema

4.1.3. Secuencia de casos de uso

Descripción del campo	Valor
ID	CU-01
Nombre	Ver Nota
Descripción	El usuario del bot selecciona una nota
Precondición	No
Dependencias	No
Secuencia	1. El usuario selecciona una fecha en el calendario 2. El sistema muestra la interfaz de edición de notas 3. El usuario ve la nota de texto almacenada en esa fecha
Excepciones	No

Tabla 4.1: Caso de uso Ver Nota

Descripción del campo	Valor
ID	CU-02
Nombre	Crear Nota
Descripción	El usuario del bot guarda una nota
Precondición	La interfaz de edición de notas está desplegada
Dependencias	CU-01, Ver Nota
Secuencia	1. El usuario selecciona el recuadro de edición 2. El sistema despliega el teclado 3. El usuario escribe una nota 4. El usuario pulsa el botón Aceptar 5. El sistema almacena la nota en la base de datos
Excepciones	No

Tabla 4.2: Caso de uso Crear nota

Descripción del campo	Valor
ID	CU-03
Nombre	Eliminar Nota
Descripción	El usuario del bot elimina una nota
Precondición	La interfaz de edición de notas está desplegada
Dependencias	CU-01, Ver Nota
Secuencia	1. El usuario pulsa el botón Eliminar 2. El sistema Elimina la nota de la base de datos 3. El sistema cierra la interfaz de edición
Excepciones	No

Tabla 4.3: Caso de uso Eliminar nota

Descripción del campo	Valor
ID	CU-04
Nombre	Activa botnet
Descripción	El bot comienza su actividad maliciosa
Precondición	El sistema móvil permite el trabajo en segundo plano
Dependencias	No
Secuencia	1. El usuario abre la aplicación 2. El sistema activa la condición de persistencia 3. El sistema comienza el ciclo de ejecución de código en segundo plano 4. El sistema activa la resistencia al reinicio
Excepciones	No

Tabla 4.4: Caso de uso Activar botnet

Descripción del campo	Valor
ID	CU-05
Nombre	Comunicación
Descripción	El bot realiza la comunicación con el servidor a través de la base de datos en la nube
Precondición	El trabajo en segundo plano del bot está activado
Dependencias	CU-04, Activa Botnet
Secuencia	1. El sistema comienza el ciclo de lectura 2. El sistema envía un mensaje de ImAlive 3. El sistema busca posibles ordenes
Excepciones	No

Tabla 4.5: Caso de uso Comunicación

Descripción del campo	Valor
ID	CU-06
Nombre	Cumplir orden
Descripción	El bot lee una orden dada por el Bot Master y la ejecuta
Precondición	El trabajo en segundo plano del bot está activado Se ha leído una orden
Dependencias	CU-04, Activa Botnet CU-05, Comunicación
Secuencia	1. El sistema ejecuta el código malicioso correspondiente a la primitiva de la orden 2. El sistema toma la información requerida en la primitiva 3. El sistema procesa los datos recogidos para adaptarse a la lectura del atacante en la botnet
Excepciones	No

Tabla 4.6: Caso de uso Cumplir orden

Descripción del campo	Valor
ID	CU-07
Nombre	Recoge Información
Descripción	El bot envía los datos tomados a la base de datos
Precondición	El trabajo en segundo plano del bot está activado Se ha leído una orden La primitiva leída es distinta de APAGAR
Dependencias	CU-04, Activa Botnet CU-05, Comunicación CU-06, Cumplir orden
Secuencia	1. El sistema se conecta con la base de datos 2. El sistema almacena los datos recopilados al ejecutarse la orden
Excepciones	No

Tabla 4.7: Caso de uso Recoge información

Descripción del campo	Valor
ID	CU-08
Nombre	Apaga Bot
Descripción	El bot se apaga y deja de ejecutarse el código malicioso
Precondición	El trabajo en segundo plano del bot está activado Se ha leído una orden La primitiva leída es APAGAR
Dependencias	CU-04, Activa Botnet CU-05, Comunicación CU-06, Cumplir orden
Secuencia	1. El sistema apaga la condición de persistencia 2. El sistema apaga el ciclo de ejecución malicioso
Excepciones	No

Tabla 4.8: Caso de uso Apaga Bot

Descripción del campo	Valor
ID	CU-09
Nombre	Envía Orden
Descripción	El atacante envía una orden a los bots desde el bot master
Precondición	No
Dependencias	No
Secuencia	1. El atacante selecciona la opción de envio de ordenes 2. El sistema muestra el menú de selección de primitivas 3. El atacante selecciona la orden a enviar 4. El sistema pregunta que tipo de orden desea almacenar 5. El atacante selecciona un tipo de orden 6. El sistema almacena la orden
Excepciones	El usuario no selecciona salir en el paso 1 El usuario no selecciona volver o salir en el paso 3

Tabla 4.9: Caso de uso Envía orden

Descripción del campo	Valor
ID	CU-10
Nombre	Lee resultados
Descripción	El atacante lee los datos recopilados por los bots
Precondición	No
Dependencias	No
Secuencia	1. El atacante selecciona la opción de obtener información 2. El sistema muestra el menú de selección de información 3. El atacante selecciona la información a recibir 4. El sistema conecta con la base de datos 5. El sistema recupera la información de la base de datos
Excepciones	El usuario no selecciona salir en el paso 1 El usuario no selecciona volver o salir en el paso 3

Tabla 4.10: Caso de uso Lee resultados

Descripción del campo	Valor
ID	CU-11
Nombre	Procesar Resultados
Descripción	El atacante Procesa los resultados obtenidos para visualizar información
Precondición	Los datos deseados han sido leídos de la base de datos
Dependencias	CU-10, Lee datos
Secuencia	1. El atacante selecciona aquellos datos que desea visualizar 2. El sistema ordena los datos de la forma en la que estos se mostrarán 3. El sistema imprime los datos por pantalla
Excepciones	El usuario no selecciona salir en el paso 1

Tabla 4.11: Caso de uso Procesar resultados

Descripción del campo	Valor
ID	CU-12
Nombre	Ver bots vivos
Descripción	El atacante ve los bots disponibles a los que mandar órdenes
Precondición	No
Dependencias	No
Secuencia	1. El atacante selecciona la opción ver bots vivos 2. El sistema recupera de la base de datos los bots que están participando en la botnet actualmente 3. El sistema imprime los datos por pantalla
Excepciones	No

Tabla 4.12: Caso de uso Ver bots vivos

Descripción del campo	Valor
ID	CU-13
Nombre	Eliminar orden permanente
Descripción	El atacante elimina una orden programada como permanente para que su ejecución cese
Precondición	No
Dependencias	No
Secuencia	1. El atacante selecciona la opción eliminar orden permanente 2. El sistema recupera de la base de datos las ordenes establecidas como permanentes 3. El sistema imprime la lista de ordenes permanentes por pantalla 4. El atacante selecciona la orden permanente que desea eliminar 5. El sistema elimina la orden permanente de la base de datos
Excepciones	No

Tabla 4.13: Caso de uso Eliminar orden permanente

4.2. Modelo de dominio

4.2.1. Introducción

En el modelo de dominio se agrupan todas las distintas entidades, sus atributos, papeles y relaciones, además de las restricciones que participan en el sistema.

4.2.2. Clases y relaciones

Para mi sistema he identificado las clases que podemos ver en la figura 4.2, en la parte izquierda podemos encontrar aquellas asociadas con el Bot Master y en la parte derecha las asociadas con el Bot, en la parte central encontramos los datos que comparten entre ellos.

El Bot Master se utilizará para seleccionar las distintas opciones que se le ofrecen al atacante para controlar la Botnet, como por ejemplo, enviar ordenes o comprobar los bots que están vivos en tiempo real.

- Utilizará un gestor de resultados que será el encargado de parsear los resultados y enseñárselos al atacante de una forma sencilla de leer para los humanos.
- El gestor de Bots Vivos se utiliza para buscar en la base de datos los bots que pueden recibir órdenes.
- Desde el gestor de ordenes se enviarán las Primitivas que ejecutarán acciones en los bots.

En la parte central del sistema tenemos la relación que se genera entre Bot y Bot Master, siendo esta a partir de clases que contienen datos. El Gestor de Ordenes pedirá al Bot que genere estos datos y este ejecutará el código necesario para conseguirlos gracias al Gestor de Alarmas, después estos se recogerán por el Bot Master para procesarlos y enviárselos al atacante. Entre los datos recopilados tenemos el GPS, las notas del usuario, los SMS, etc. Cada uno de ellos tiene sus atributos, pero todos tienen en común que utilizan un identificador único del Bot para poderlos distinguir más adelante y la fecha en la que se han almacenado. Los ImAlive no tienen atributos ya que solo se utilizan para determinar si un bot está vivo o no.

Para el Bot, el diagrama muestra la funcionalidad que se ejecuta en los dispositivos móviles, es decir, recopilar la información sin que el usuario se de cuenta mientras a este le muestra una interfaz ajena al verdadero propósito de la aplicación.

- El Calendario de notas será aquel en el que el usuario escribe las notas de texto construyendo así la funcionalidad tapadera de la aplicación.
- A partir de ahí es el gestor de alarmas el que se lleva la funcionalidad principal siendo el encargado de comunicarse con la base de datos y obteniendo los datos requeridos en las ordenes.
- También cuenta con un Gestor de inicio para despertar al Bot después de un reinicio.

4.2.3. Multiplicidad

La multiplicidad indica la forma en la que se asocia una instancia de una clase con otra. Podemos aplicar las siguientes explicaciones:

- El Bot Master puede tener o no el Gestor de resultados, el Gestor de órdenes y el Gestor de Bots Vivos pero estos siempre están asociados con el mismo Bot Master.
- Desde el Gestor de Órdenes cada uno de los resultados y ejecución de las ordenes que se generan pueden no estar como existir varias de ellas, lo mismo en sentido contrario ya que las ordenes pueden ser leídas desde varios Gestores utilizados desde servidores diferentes o no ser leídos por ninguno si el Bot Master está apagado
- Lo mismo ocurre para el Gestor de bots vivos y los ImAlive.
- Para el Gestor de alarmas podemos observar que la multiplicidad de los objetos generados por las ordenes siempre van asociadas a un Bot ya que se guarda la ID del que lo generó, en el caso de las órdenes, al poder ser para todos los bots utilizará una multiplicidad 1 o varios. Puede no haberse generado el resultado de una orden ya que no se haya requerido desde la botnet o haberse generado varias de estas.
- El gestor de alarmas siempre tiene asociado un Calendario de notas pero el calendario de notas puede no tener gestor de alarmas si el bot se ha apagado desde una orden
- El gestor de Alarmas puede tener activado o no el Inicio al reiniciar, pero este inicio siempre está asociado al mismo gestor de alarmas

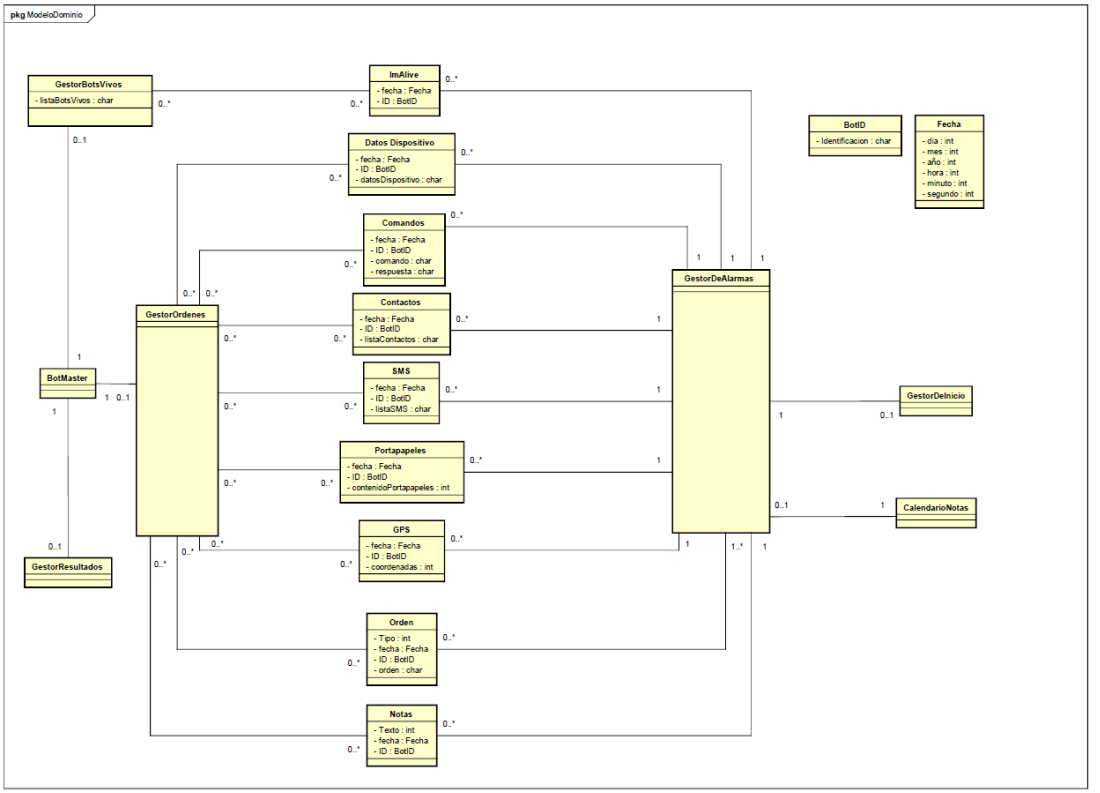

Figura 4.2: Modelo de dominio del sistema

Capítulo 5

Tecnologías utilizadas y plan de costes

En este capítulo se detalla cada una de las tecnologías utilizadas a lo largo del proyecto y se realiza un pequeño plan de costes derivado del uso de estas En este Capítulo se presenta un resumen de las tecnologías utilizadas en las diferentes secciones y capítulos de este documento. Además, de cara a ampliar la planificación, se realizará una estimación de los costes del proyecto.

5.1. Tecnologías utilizadas

Para cada tecnología se ofrecerá una descripción de la misma, se entrará en detalle en como se ha utilizado en el proyecto y finalmente se evaluará su coste de cara a un posterior plan de costes

5.1.1. MITRE ATT&CK: Mobile Matrix

¿En qué consiste esta tecnología? [47] [9]

MITRE ATT&CK (Adversarial, Tactics, Techniques, and Common Knowledge) es un framework, es decir, una estructura sobre la que se puede construir un software. Sirve como base, para que no empezar completamente de 0 a la hora de realizar un proyecto. En este caso, MITRE ATT&CK refleja las distintas fases del ciclo de vida de los ataques que puede realizar un actor malicioso en las diferentes plataformas objetivo (Empresa, móviles, ...). Este framework presenta tácticas y técnicas utilizadas por los atacantes a la hora de comprometer un sistema y puede ser utilizada tanto los lados ofensivos (tomando el rol de atacante, por ejemplo para realizar una auditoría de seguridad) como por el lado defensivo (tomando el

rol de defensa, por ejemplo, para programar software seguro) ya que también ofrece formas específicas de defenderse de estos ataques. Como podemos ver en la figura 5.1 el framework presentado por ATT&CK contiene los siguientes componentes básicos:

- Tácticas que denotan los objetivos a corto plazo del atacante durante un ataque, es decir, las columnas
- Técnicas que describen los medios por los que los atacantes logran sus objetivos, es decir, cada componente individual de la matriz
- Uso documentado de las técnicas por parte del atacante, al acceder dentro de cada elemento de la matriz

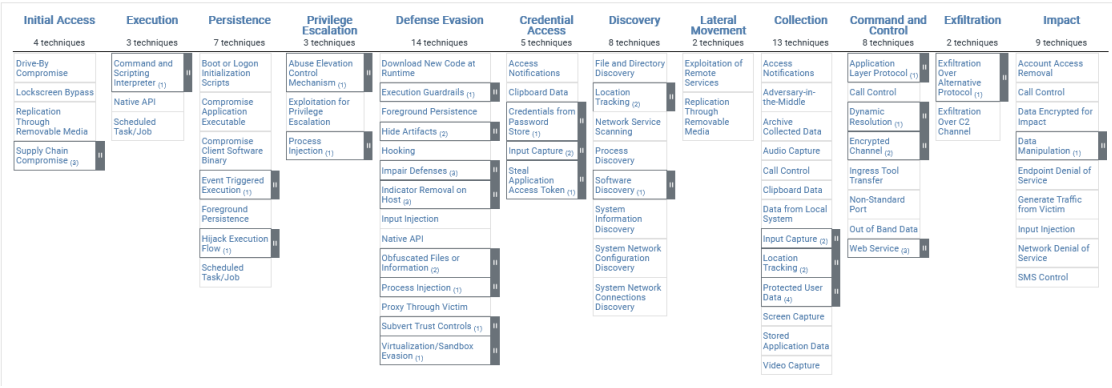

Figura 5.1: Matriz MITRE ATT&CK: Mobile

MITRE ATT&CK tiene tres variantes:

- ATT&CK para empresas, que no nos será de utilidad para el proyecto actual pero podría ser interesante para futuros trabajos.
- ATT&CK para ICS, tampoco será útil para este proyecto.
- ATT&CK para móviles, se centra en el comportamiento de los atacantes en los sistemas operativos iOS y Android. Es el que utilizaremos para este proyecto.

¿Cómo se ha utilizado en el proyecto? EL uso de esta tecnología se ha definido en el capítulo 7 de este documento por motivos de facilidad de comprensión y lectura, sin embargo si que se detallará el coste de la misma en el proyecto.

¿Cuál ha sido el coste de emplearla en el proyecto?

El coste total de utilizar MITRE ATT&CK: Mobile Matrix ha sido de 0€ ya que es un framework gratuito, por tanto no supondrá un incremento ni un inconveniente en el coste de desarrollo del proyecto.

5.1.2. Android

¿En qué consiste esta tecnología?

Consultar la sección 2.3 de este documento para obtener más información sobre esta tecnología.

¿Cómo se ha utilizado en el proyecto?

Una de las partes más interesantes de este proyecto será comprobar los límites a los que puede llegar la botnet a la hora de recopilar información dependiendo de las distintas versiones de Android. Principalmente se utilizará un dispositivo con Android 12.5.6 ya que es con el que cuento para las pruebas, además utilizare los emuladores que ofrece Android Studio para reducir la versión de Android y ampliar la batería de pruebas comprobando si el sistema funciona como se espera y a que podré acceder.

¿Cuál ha sido el coste de emplearla en el proyecto?

Para utilizar android en el proyecto podríamos apoyarnos únicamente en emuladores, para ello necesitamos un ordenador en el que ejecutar Android Studio, esto supone un coste que en mi caso ha sido un ordenador portátil con un valor de 599€. Por otro lado también utilizaré un dispositivo móvil para ejecutarlo en un entorno lo más fiel posible a la realidad a si que utilizaré un dispositivo móvil con un valor de 210€

5.1.3. Android Studio

¿En qué consiste esta tecnología?

Segun la web oficial de Android Studio [18], es un entorno de desarrollo integrado (IDE) oficial para el desarrollo de apps para Android. Android Studio ofrece funciones que aumentan la productividad al desarrollar apps para Android, como las siguientes:

- Un sistema de compilación flexible basado en Gradle
- Un emulador rápido y cargado de funciones
- Un entorno unificado donde puedes desarrollar para todos los dispositivos Android
- Aplicación de cambios para insertar cambios de código y recursos a la app en ejecución sin reiniciarla
- Integración con GitHub y plantillas de código para ayudarte a compilar funciones de apps comunes y también importar código de muestra
- Variedad de marcos de trabajo y herramientas de prueba

Además ofrece una integración completa tanto con Java como con Kotlin, dos de los lenguajes que se utilizarán para el desarrollo del proyecto.

¿Cómo se ha utilizado en el proyecto?

Android Studio ha sido el IDE principal utilizado en el proyecto, esto ha sido así ya que he podido beneficiarme de varias de las características que ofrece como pueden ser:

- Integración con Git, de esta forma se satisface la necesidad de utilizar un software de control de versiones ayudando a poder continuar el proyecto en remoto de mi puesto de trabajo habitual y a reducir el riesgo de perder parte del progreso
- Permite el uso de kotlin, mi lenguaje predeterminado para el proyecto. Además ofrece varias características que mejoran la productividad como por ejemplo el uso de atajos como control + espacio para ver todos los métodos posibles.
- Android Studio tambien permite convertir automáticamente el código Java en código Kotlin con un simple click.
- El Logcat es muy útil para el debug del proceso, además del modo debug como tal que permite la ejecución de la aplicación a través de distintos breakpoints.
- La gran variedad de emuladores que ofrece me permitirá un mejor proceso de testing. Esta característica es muy útil para comprobar en que dispositivos móviles puede funcionar la aplicación sin gastar dinero
- Importar paquetes y librerías externas es muy sencillo a través de Graddle Script
- Los proyectos en android son bastante complejos, por tanto la gestión de las actividades, paquetes, layouts, manifest y demás componentes que produce el IDE son prácticamente necesarios para el desarrollo de software móvil

¿Cuál ha sido el coste de emplearla en el proyecto?

Utilizar Android Studio no supone ningún coste ya que la licencia es gratuita. Los costes de la máquina que ejecutará el software ya han sido asumidos en el punto 5.1.3

5.1.4. Visual Studio Code

¿En qué consiste esta tecnología?

Visual Studio Code [53] es un editor de código fuente desarrollado por Microsoft para Windows, Linux, macOS y Web. Incluye soporte para la depuración, control integrado de Git, resaltado de sintaxis, finalización inteligente de código, fragmentos y refactorización de código. También es personalizable, por lo que los usuarios pueden cambiar el tema del editor, los atajos de teclado y las preferencias. Es gratuito y de código abierto, aunque la descarga oficial está bajo software privativo.

¿Cómo se ha utilizado en el proyecto?

Para la programación en Python en la parte del Bot Master se ha utilizado Visual Studio Code como IDE principal. Esto se debe a que los plugins de python que pueden descargarse

directamente desde el editor de texto son bastante potentes y permiten acelerar la fase de desarrollo del proyecto.

¿Cuál ha sido el coste de emplearla en el proyecto?

Visual Studio Code es gratuito y por tanto no supone coste para el proyecto. Además los costes de la máquina en la que se programa y ejecuta ya han sido contemplados.

5.1.5. Java

¿En qué consiste esta tecnología?

La edición móvil de Java se llama Java ME. Java ME se basa en Java SE y es compatible con la mayoría de los teléfonos inteligentes y tabletas. La Java Platform Micro Edition (Java ME) proporciona un entorno flexible y seguro para crear y ejecutar aplicaciones dirigidas a dispositivos móviles e integrados. Las aplicaciones que se construyen con Java ME son portátiles, seguras y pueden aprovechar las capacidades nativas del dispositivo. Java ME aborda las limitaciones que conlleva la creación de aplicaciones destinadas a dispositivos móviles. En esencia, Java ME aborda el reto de ejecutar aplicaciones en dispositivos con poca memoria, pantalla y energía disponibles. [24]

¿Cómo se ha utilizado en el proyecto?

Java se ha utilizado poco durante el desarrollo del proyecto, en torno al 2 % del código producido es en este lenguaje. Sin embargo muchas de las referencias consultadas para crear código han sido cogidas desde Java y transformadas a kotlin desde el IDE.

¿Cuál ha sido el coste de emplearla en el proyecto?

Java es gratuito y por tanto no supone coste para el proyecto. Además los costes de la máquina en la que se programa y ejecuta ya han sido contemplados.

5.1.6. Kotlin

¿En qué consiste esta tecnología?

Kotlin [45] es el lenguaje recomendado por Google para construir aplicaciones en Android. Este ahorra tiempo a los desarrolladores, ya que el lenguaje es menos verboso y por tanto proporciona un código más breve y menos redundante.

En muchos sentidos, Kotlin se considera un sustituto de Java. Aunque no es compatible con la sintaxis, es interoperable con el código y las bibliotecas de Java. Kotlin también tiene sus propias bibliotecas y una API para aplicaciones Android.

En Java, mucha redundancia da lugar a un código verboso y, por tanto, más largo. Kotlin es más moderno y se ha simplificado, lo que facilita su aprendizaje. Kotlin se centra en un

código funcional y simplificado y evita el código repetitivo. El lenguaje cuenta con seguridad contra las excepciones de NullPointerException. Los puntos y coma al final de cada línea no son necesarios, aunque pueden utilizarse.

¿Cómo se ha utilizado en el proyecto?

He escogido Kotlin porque al tratarse de un proyecto que necesita un desarrollo rápido este lenguaje me permite reducir el riesgo de no llegar a tiempo a la finalización de un Sprint. En torno al 51 % del código escrito ha sido en este lenguaje. Entre sus características me gustaría destacar su similitud con Java, un lenguaje que ya conocía, pero mejorando su sintaxis enormemente.

¿Cuál ha sido el coste de emplearla en el proyecto?

Kotlin es gratuito y por tanto no supone coste para el proyecto. Además los costes de la máquina en la que se programa y ejecuta ya han sido contemplados.

5.1.7. Python

¿En qué consiste esta tecnología?

Es un lenguaje de programación utilizado para crear software, automatizar tareas y realizar análisis de datos. Python es un lenguaje de propósito general, lo que significa que se puede utilizar para crear una variedad de programas diferentes y no está especializado en ningún problema específico. Esta versatilidad, junto con su facilidad para los principiantes, lo ha convertido en uno de los lenguajes de programación más utilizados en la actualidad[11]

Dos de sus características es que es un lenguaje interpretado con tipado dinámico, esto significa que Python ejecuta directamente el código línea por línea. En caso de cualquier error, detiene la ejecución e informa del error que se ha producido. Python sólo muestra un error aunque el programa tenga varios. Esto facilita la depuración. El tipado dinámico significa que Python no conoce el tipo de variable hasta que ejecutamos el código. Asigna automáticamente el tipo de datos durante la ejecución. El programador no necesita preocuparse de declarar las variables y sus tipos de datos.[46]

¿Cómo se ha utilizado en el proyecto?

En torno al 47 % del código del proyecto se ha realizado en python. Se ha utilizado para programar toda la parte del Servidor en la que se encuentra el Bot Master. He decidido utilizar Python porque permite desarrollar el código rápidamente, de forma limpia y no necesita ser compilado por lo que para ejecutarlo en varios servidores distintos, por ejemplo, Ubuntu, Windows 10 o el que yo he utilizado Kali Linux es más fácil. También he utilizado pip para instalar sus paquetes de forma sencilla.

¿Cuál ha sido el coste de emplearla en el proyecto?

Python es gratuito y por tanto no supone coste para el proyecto. Además los costes de la máquina en la que se programa y ejecuta ya han sido contemplados.

5.1.8. Virtualbox

¿En qué consiste esta tecnología?

VirtualBox[10] es un software de código abierto para virtualizar la arquitectura informática x86. Actúa como hipervisor, creando una VM (máquina virtual) donde el usuario puede ejecutar otro SO (sistema operativo).

El sistema operativo en el que se ejecuta VirtualBox se denomina SO .ªnfitrión". El sistema operativo que se ejecuta en la VM se denomina SO ïnvitado". VirtualBox admite Windows, Linux o macOS como sistema operativo anfitrión.

Al configurar una máquina virtual, el usuario puede especificar cuántos núcleos de CPU y cuánta memoria RAM y espacio en disco debe dedicar a la VM. Cuando la VM se está ejecutando, puede ser "pausada". La ejecución del sistema se congela en ese momento, y el usuario puede reanudar su uso más tarde. Por tanto este software ofrece unas condiciones para virtualizar una máquina de manera sencilla y eficiente que puede adaptarse a la mayoría de las máquinas que lo ejecuten

¿Cómo se ha utilizado en el proyecto?

Para un sistema como es una Botnet, utilizar un sistema operativo virtualizado como puede ser Kali Linux (en este proyecto) es una gran ventaja ya que te permite configurar varios parámetros de la máquina, sus conexiones, sus recursos, incluso te permite cambiar el SO sobre la marcha. Además en caso de error o fallo, común en sistemas operativos Linux, se puede descargar una copia nueva y utilizarla rápidamente.

¿Cuál ha sido el coste de emplearla en el proyecto?

Virtualbox es gratuito y por tanto no supone coste para el proyecto. Además los costes de la máquina en la que se ejecuta ya han sido contemplados.

5.1.9. Kali Linux

¿En qué consiste esta tecnología?

Kali Linux[36] es una distribución de Linux basada en Debian. Es un sistema operativo meticulosamente elaborado que atiende específicamente a los gustos de los analistas de redes, pentesters y auditores de ciberseguridad. Contiene una gran variedad de herramientas que vienen preinstaladas con Kali lo transforma en una navaja suiza para los hackers. Algunas de las ventajas de Kali son las siguientes:

- Libre y gratuito por tanto puede utilizarse sin miedo a incrementar los costes.
- Más de 600 diferentes herramientas relacionadas con el análisis de seguridad.
- Sigue un modelo de código abierto, su desarrollo es públicamente visible en Git y todo el código está disponible ajustarlo.

- Completamente personalízable, incluido el kernel.

Alguna de las desventajas de Kali Linux es que no está pensado para durar, es posible que con el tiempo su rendimiento se degrade notablemente, es por ello que su instalación en una máquina virtual es tan sencilla como descargar un archivo de su web oficial [27]

¿Cómo se ha utilizado en el proyecto?

Si bien es cierto que el código del servidor podría ejecutarse en cualquier sistema operativo con Python3 instalado, se ha utilizado Kali Linux para ejecutarlo. He decidido que esto sea así ya que podría utilizarse para futuras mejoras del proyecto en las que fuera necesario utilizar alguna de las herramientas que ofrece.

¿Cuál ha sido el coste de emplearla en el proyecto?

Kali Linux es gratuito y por tanto no supone coste para el proyecto. Además los costes de la máquina en la que se ejecuta ya han sido contemplados.

5.1.10. LaTeX (Overleaf)

¿En qué consiste esta tecnología?

Latex [52] es un sistema de composición de textos, orientado a la creación de documentos escritos que presenten una alta calidad tipográfica. Por sus características y posibilidades, es usado de forma especialmente intensa en la generación de artículos y libros científicos que incluyen, entre otros elementos, expresiones matemáticas.

Entre sus ventajas podemos ver:[54]

- Es software libre
- Existen toda una serie de plantillas que ofrecen un resultado profesional
- Inserción de fórmulas matemáticas y código
- No hay que preocuparse de cómo queda
- BibTeX: muchas bibliografías online
- LaTeX fuerza al autor(a) a estructurar sus textos
- Fácil de exportar a PDF
- Funciona en texto plano en cualquier máquina

Y entre sus desventajas nos podemos encontrar

- No tan intuitivo como office 365

- Requiere un aprendizaje con una curva de dificultad más elevada
- Necesita software adicional para compilar los documentos

¿Cómo se ha utilizado en el proyecto?

Para remediar las desventajas en el proyecto se utilizará un editor de Latex online, Overleaf [49]. Esta es una herramienta de publicación y redacción colaborativa en línea que hace que todo el proceso de redacción, edición y publicación de documentos científicos sea mucho más rápido y sencillo. Overleaf brinda la conveniencia de un editor LaTeX fácil de usar con colaboración en tiempo real y la salida totalmente compilada producida automáticamente en segundo plano a medida que escribe. Además ofrece un sistema de control de versiones y gestión de proyectos bastante útil. Toda la documentación del proyecto ha sido generada a partir de esta herramienta.

¿Cuál ha sido el coste de emplearla en el proyecto?

Latex y Overleaf es gratuito y por tanto no supone coste para el proyecto. Además los costes de la máquina en la que se ejecuta ya han sido contemplados.

5.1.11. Astah UML

¿En qué consiste esta tecnología?

Astah es una herramienta de modelado de software que permite crear y visualizar las ideas y diseños de software. Permite construir rápidamente y sin complicaciones diagramas para entender el funcionamiento del sistema. Generalmente se utiliza para generar diagramas UML, ER, diagramas de flujo, mapas conceptuales.

¿Cómo se ha utilizado en el proyecto?

Se ha utilizado Astah UML para crear todos los productos creados en las fases de ingeniería de software, es decir, análisis y diseño. En concreto he escogido esta herramienta para el proyecto porque me parece más fácil e intuitiva de utilizar que otras semejantes a esta como podría ser visual paradigm.

¿Cuál ha sido el coste de emplearla en el proyecto?

Este precio de este software varía dependiendo de la versión que escojamos utilizar, en mi caso al utilizar la versión Academic License for Students no es necesario añadir coste al proyecto ya que esta es gratuita. [6]

5.1.12. Draw.io

¿En qué consiste esta tecnología?

Draw.io[23] es una herramienta de creación de diagramas, como, por ejemplo, diagramas de flujo, diagramas UML, etc o cualquier tipo de mapa conceptual, esquema, representación gráficas, como por ejemplo, diagramas de jerarquía o de conjuntos. Entre sus ventajas encontramos:

- Se puede utilizar desde la propia web, permitiendo la descarga o el almacenamiento en la nube de los diagramas generados.
- La herramienta dispone de una serie de plantillas que hacen el trabajo más sencillo y más rápido en las que se incluyen formas como líneas, dibujos e iconos
- La interfaz es sencilla, utiliza el método drag and drop y permite exportar fácilmente el trabajo a diferentes formatos lo cual mejora su productividad

¿Cómo se ha utilizado en el proyecto?

Se ha utilizado draw.io para añadir imágenes en la memoria del proyecto cuando ha sido necesario generar un diagrama y no ha podido crearse a partir de la herramienta Astah UML, como por ejemplo, el diagrama de la base de datos.

¿Cuál ha sido el coste de emplearla en el proyecto?

Draw.io es gratuito y por tanto no supone coste para el proyecto. Además los costes de la máquina en la que se ejecuta ya han sido contemplados.

5.1.13. Excel

¿En qué consiste esta tecnología?

Excel[3] es el programa de hojas de cálculo de Microsoft. Este permite a los usuarios formatear, organizar y calcular datos. Su objetivo es facilitar la visualización de la información a medida que se añaden o modifican los datos. El uso más habitual de Excel es en el ámbito empresarial. Por ejemplo, se utiliza en el análisis empresarial, la gestión de recursos humanos, la gestión de proyectos y la elaboración de informes de rendimiento. Excel utiliza una gran colección de celdas con formato para organizar y manipular datos y resolver funciones matemáticas. Además, Excel, permite organizar los datos en la hoja de cálculo mediante herramientas gráficas, tablas dinámicas y fórmulas.

¿Cómo se ha utilizado en el proyecto?

Se ha utilizado excel para la fase de planificación del proyecto ordenando las distintas actividades en tablas y permitiendo un seguimiento eficiente de las mismas a través de diagramas y dando uso a las funciones matemáticas que ofrece, por ejemplo, para la obtención de estadísticas.

¿Cuál ha sido el coste de emplearla en el proyecto?

Excel forma parte de las suites Microsoft Office y Office 365. Estas no son gratuitas, sin embargo, al utilizarse en un proyecto realizado por un estudiante, he utilizado una licencia gratuita concedida por la universidad con un coste de 0€

5.1.14. Firebase Firestore

¿En qué consiste esta tecnología?

Firebase[17], Cloud Firestore es una base de datos NoSQL alojada en la nube a la que se puede acceder desde las aplicaciones de un proyecto. Al ser una base de datos NoSQL los datos se almacenan en documentos que contienen campos a los que se asignan a valores. Estos documentos se almacenan en colecciones, que son contenedores de documentos y que se pueden utilizar para organizar los datos y compilar consultas. Los documentos admiten varios tipos de datos diferentes, desde strings y números simples, hasta objetos anidados complejos. También se pueden crear subcolecciones dentro de documentos y crear estructuras de datos jerárquicas que se ajustan a escala a medida que la base de datos crece.

Se pueden crear consultas superficiales para recuperar datos en el nivel del documento, sin la necesidad de recuperar la colección completa ni las subcolecciones anidadas. También se pueden añadir filtros a las consultas para paginar los resultados. Firestore también ofrece la opción de escucha en tiempo real de los datos para mantenerlos actualizados siempre que se produzca un cambio

¿Cómo se ha utilizado en el proyecto?

Se ha utilizado Firebase Firestore como base de datos para el proyecto ya que permite cumplir varias condiciones necesarias para el funcionamiento de la Botnet de una forma muy sencilla. Además la facilidad de implementación que ofrecen las herramientas en la nube influyen de forma positiva en el resultado final del proyecto

¿Cuál ha sido el coste de emplearla en el proyecto?

Firebase Firestore no es una tecnología gratuita, aunque esto depende del uso que se la de. De esta forma, si se desea utilizar la base de datos de forma gratuita[16], no se podrá exceder de:

- Datos almacenados 1 GiB
- Operaciones de lectura de documentos 50,000 por día
- Operaciones de escritura de documentos 20,000 por día
- Operaciones de eliminación de documentos 20,000 por día
- Salida de red 10 GiB por mes

De otra forma los precios por su uso serán aquellos plasmados en la figura 5.2 [15]

Europa (multirregión)	
	Precios posteriores a la cuota gratuita
Lecturas de documentos	$0.06 por 100,000 documentos
Escrituras de documentos	$0.18 por 100,000 documentos
Eliminaciones de documentos	$0.02 por 100,000 documentos
Datos almacenados	$0.18 por GiB al mes

Figura 5.2: Precios Firestore

Al no haber excedido la cuota gratuita el precio que implicará Firebase Firestore (de momento) es de 0€ y por tanto no influirá en el coste del proyecto.

5.1.15. Trello

¿En qué consiste esta tecnología?

Trello [48] es una herramienta visual que permite a los equipos gestionar proyectos, supervisar una lista de ideas y organizar el trabajo que ha de ser desarrollado. Entre otras funciones permite añadir checklists o incluso automatizaciones, es capaz de ser personalízalo al completo según las necesidades del proyecto. Un tablero de trello se divide en 3 partes clave:

- Los tableros mantienen las tareas organizadas y ayudan a que el trabajo avance, permiten ver todo el progreso de un solo vistazo.
- Cada columna representa una fase de trabajo, estas pueden ser, por ejemplo, la columna en la que se depositan las actividades que estarán en curso durante un Sprint
- Las tarjetas representan las tareas y pueden incluir toda la información necesaria para hacer el trabajo.

¿Cómo se ha utilizado en el proyecto?

En la figura 5.3 puede verse el Scrum board utilizado durante el desarrollo del proyecto, este se ha generado utilizando exclusivamente esta herramienta. En la primera columna se encuentran todas las tareas que necesitan completarse para dar por finalizado el proyecto. Durante los 8 Sprint que componen el proyecto se seleccionarán las tarjetas que las representan y se moverán a la segunda columna, de esta forma las tarjetas reflejan que el trabajo esta pensado para completarse en el sprint en curso. Al empezar a trabajar en ellas estas se moverán a la columna de En progreso y una vez estén completadas a la columna Tareas finalizadas este sprint donde permanecerán hasta que finalice el Sprint y estas se den por completadas moviéndolas a la ultima de las columnas.

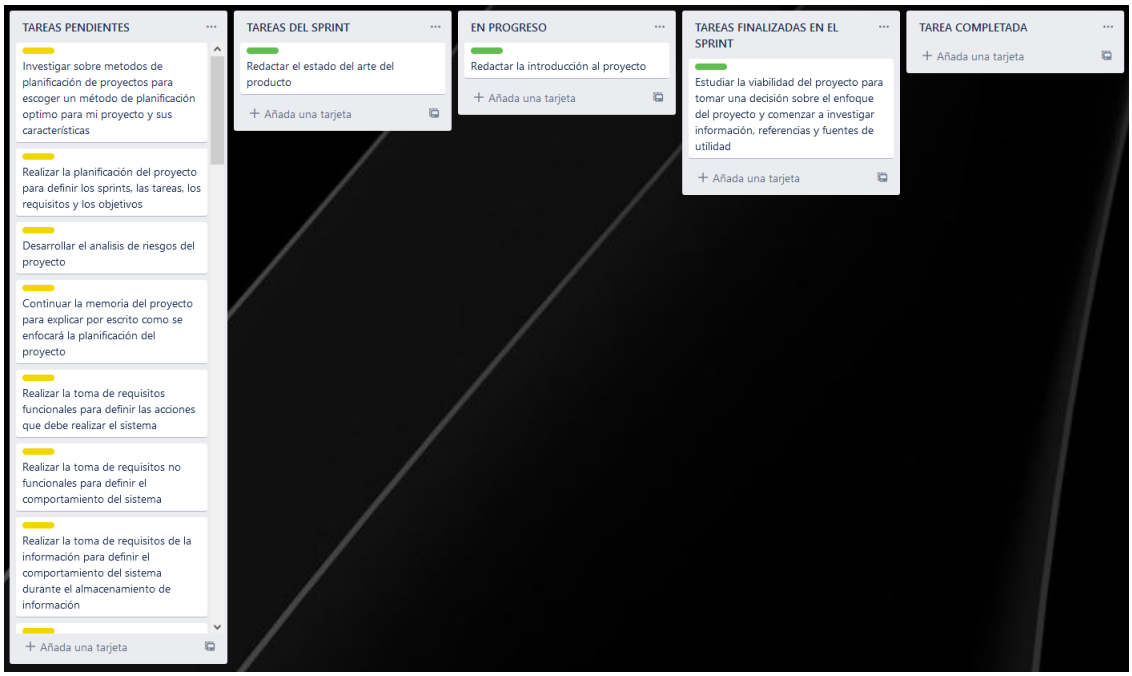

Figura 5.3: Explicación sobre la utilización del Scrum Board

¿Cual ha sido el coste de emplearla en el proyecto?

Trello es gratuito y por tanto no supone coste para el proyecto. Además los costes de la máquina en la que se ejecuta ya han sido contemplados.

5.2. Plan de costes

El plan de costes es útil para ayudar a gestionar los costes del proyecto al tiempo que se obtiene el mayor rendimiento de la inversión u otro tipo de beneficios. En un proyecto, como ya hemos visto, se utilizan herramientas que tienen un coste previsto. Para evitar tomar decisiones a la ligera, se puede minimizar el riesgo de acabar con costes adicionales siendo metódico y ordenado. En este caso ya hemos detallado el coste de cada herramienta en el proyecto y el resultado es el siguiente:

Herramienta	Precio en el proyecto
MITRE ATT&CK: Mobile Matrix	0 €
Android	0 €
Android Studio	0 €
Visual Studio Code	0 €
Java	0 €
Kotlin	0 €
Python	0 €
Virtualbox	0 €
Kali Linux	0 €
LaTeX (Overleaf)	0 €
Astah UML	0 €
Draw.io	0 €
Excel	0 €
Firebase Firestore	0 €
Trello	0 €
Ordenador personal (+ Windows 10)	599 €
Dispositivo Movil	210 €
Total	809 €

Figura 5.4: Lista de costes

Capítulo 6

Diseño

En esta capítulo se define el patrón de diseño utilizado para el desarrollo de la aplicación, se presentará el diagrama de la base de datos, se crearán los diagramas de componentes, arquitectura y despliegue y se finalizará con el modelo de dominio y los diagramas de secuencia de la aplicación

6.1. Patrones de diseño

Los Patrones de diseño en programación [40] son soluciones a problemas recurrentes de diseño aplicado al software, estos permiten tener una guía a la hora de establecer la estructura de un programa, hacerlo más flexible y reusable. Además, también permite solucionar problemas con una metodología ya testeada y no hacerlo desde cero.

6.1.1. Patrón de diseño en el Bot: Modelo vista controlador

El patrón de diseño modelo vista controlador[39] es capaz de separar una aplicación en tres responsabilidades diferenciadas entre si:

- El modelo serán los datos que la aplicación maneje, por ejemplo, las clases que crean objetos de datos para almacenar información en la base de datos, no estará vinculado a la vista ni al controlador, simplemente expresa datos y no tiene operaciones.
- La vista presenta la interfaz que utilizará el usuario al ejecutar la aplicación, esta, se comunicará con el usuario cuando se interactúe con los elementos que la componen, por ejemplo, botones, campos de texto, o pantallas que desplieguen información. La vista no tiene conocimiento del modelo, no comprende su estado y no sabe que hacer cuando el usuario interactúa con ella. Cuando menos sepa la vista de la aplicación

menos acoplada se encuentra y por tanto mas flexible es en el caso de que sea necesario reutilizarla

- El controlador determina la funcionalidad de la aplicación, funciona al recibir mensajes de la Vista y este realiza las operaciones necesarias, después envía los resultados, si es necesario, al modelo. También será el controlador el encargado de actualizar la vista cuando lo considere necesario. Un ejemplo del controlador es lo que en Android se llama Actividad

Aplicando el patrón descrito a la aplicación del Bot ejecutada en los dispositivos móviles, podemos distribuir las clases en la tres responsabilidades definidas:

- En la vista solo encontramos un único elemento, la interfaz de la aplicación que da al usuario la sensación de que es una aplicación como otra cualquiera. En el código de esta encontramos definido en formato .xml los botones, los campos de texto y el calendario con los que el usuario interactuará.
- En el controlador encontramos las clases de código kotlin que interactúan con la vista cuando el usuario pulsa los botones o acede a los campos de texto. Estas se encargan de realizar todo el trabajo, operaciones y lógica de la aplicación, incluida tanto la funcionalidad legítima de agenda de notas como la funcionalidad maliciosa de recopilación de información. También creará los objeto del modelo para enviarlos a la base de datos. Esta compuesta por el calendario de notas que manipula la vista, la alarma que recopila información del dispositivo y la envía a la base de datos, el servicio de la alarma para su inicialización y el AutoStart para iniciar la aplicación automáticamente al reiniciar el dispositivo.
- En el modelo encontramos todas las clases .java de la aplicación que forman los objetos generados para almacenar los datos en la BD. Se utilizará uno para cada elemento de información que se recopila en el dispositivo móvil.

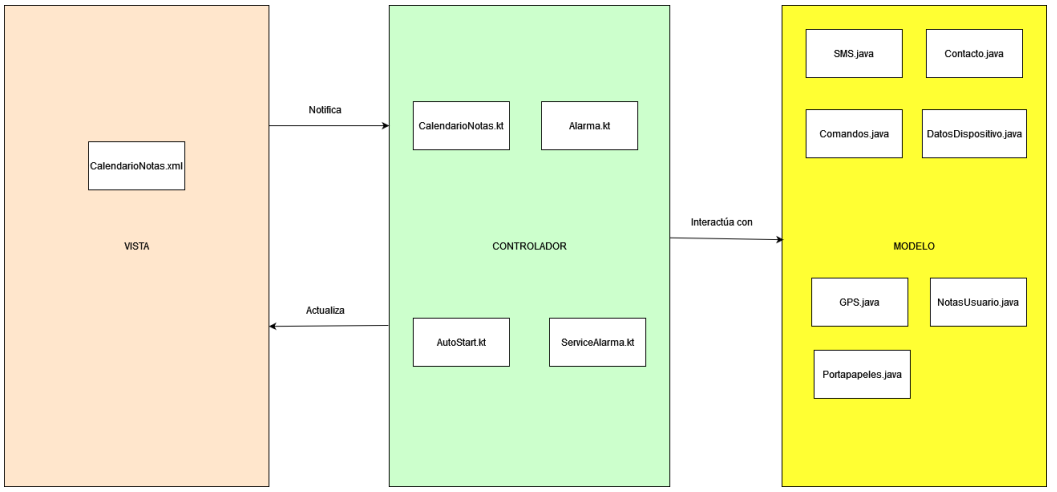

Figura 6.1: Patrón de diseño MVC aplicado al Bot

6.1.2. Patrón de diseño del Bot Master: Transaction Script

El patrón de diseño Transaction Script [13] organiza la lógica de negocio como un conjunto de procedimientos donde cada procedimiento maneja una sola solicitud de la capa de presentación, es un patrón de diseño que destaca por su simplicidad. Organizar la lógica del sistema en el que se aplica es muy sencillo ya que implica muy poca sobrecarga tanto en el rendimiento de la aplicación como en la comprensión del código, siempre y cuando, no se esté diseñando un sistema excesivamente complejo.

Este patrón propone que la lógica de negocio se vea como un conjunto de transacciones. Cada caso de uso puede interpretarse como una transacción, o romperse en varias transacciones si son complejos. Dependiendo de la complejidad de las transacciones se pueden agrupar varias dentro de una misma clase o dedicar una clase para encapsular cada una de ellas.

Una transacción es una operación, normalmente formada por varios procesos, que se trata como una operación atómica, es decir, que se completa al 100 % o no se completa en absoluto y además deja el sistema en un estado consistente.

Una de sus mayores desventajas es que al aplicar este patrón se tiende a utilizar mucho código duplicado, esto sucede ya que las transacciones pueden tener características en común unas con otras. Además no es un patrón útil si se desea implementar un diseño orientado a objetos o un sistema con una gran escalabilidad

6.2. Diagrama de arquitectura física de alto nivel

Me parece interesante presentar un diagrama de la arquitectura física del sistema de alto nivel para ofrecer una perspectiva general y sencilla del funcionamiento del sistema. Como se puede ver en la figura 6.2 podemos dividir el trabajo en 3 secciones:

- Primero, el lado atacante o Bot Master. Este se encuentra ejecutándose en un sistema kali linux. Se comunicará con la base de datos para enviar ordenes a los Bots y para recuperar y mostrar al actor atacante los resultados obtenidos de la extracción de información.
- Segundo, la base de datos. Sus dos principales características es que está alojada en la nube, por lo que la implementación en el proyecto es muy sencilla y además es NoSQL por lo que para manipularla habrá que trabajar con Colecciones, Documentos y Elementos. Se encargará de almacenar todos los resultados recopilados y servirá como canal de comunicación para el Bot y el Bot Master ya que se trata de un servicio web conocido y por ello cumple con el framework MITRE
- Por último, el Bot (o los Bots). Estos se encuentran en los dispositivos móviles infectados con el malware desarrollado en el proyecto. Su principal función es pasar desapercibidos una vez se hayan instalado correctamente en su máquina host y ejecutarán su función de recopilación de información cuando les llegue una orden a través de la base de datos, todo esto sin levantar sospechas. Una vez hayan recopilado los datos

que se les pida, lo enviarán a la base de datos para su almacenamiento y por tanto, comunicación al Bot Master.

Figura 6.2: Diagrama de la arquitectura de proyecto de alto nivel

6.3. Diagrama de la base de datos

Como base de datos del proyecto se ha utilizado una Base de datos en la nube, Firebase Firestore. Es servicio cuenta con un par de características especiales, estas son su carácter asíncrono y NoSQL. Por tanto la base de datos se divide en:

- Colecciones: Agrupa una serie de documentos, en este caso he utilizado una colección por cada elemento del modelo (MVC) definido en 6.1.1 y adicionalmente las órdenes.

- Documentos: Contienen una serie de campos, en este caso los documentos se han utilizado cada vez que se almacena un nuevo objeto, esto significa que, por ejemplo, cada ImAlive que se envié generará un documento ImAlive con sus características y campos.

- Campos: Contienen cada uno de los valores extraídos en la recopilación de información, por ejemplo, el BOT_ID en formato String y la hora en formato String como puede verse en el Field contenido en el documento ImAlive en la figura 6.3

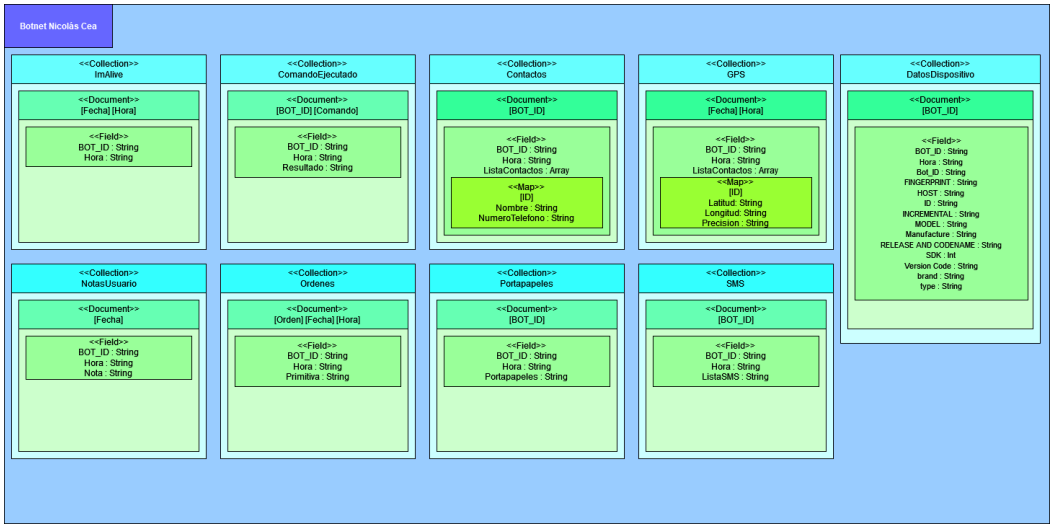

Figura 6.3: Diagrama de la base de datos del proyecto

6.4. Modelo de dominio

Sobre el modelo de dominio presentado en la fase de diseño se explicarán sus principales diferencias respecto al que ya se detallo durante el análisis:

- Se han añadido algunas de las clases necesarias para aplicar los patrones de diseño, por ejemplo, la interfaz del Calendario de notas del Bot, el service y el autostart necesarios para la ejecución del código en segundo plano y reinicio seguro de la aplicación. Por otro lado se ha eliminado el gestor de resultados del Bot ya que no se consideraba necesario al aplicar el Transaction Script. Sera el gestor de Ordenes el que se encargue de realizar todas las transacciones mediante llamadas a funciones descritas en su código.

- Se han añadido las funciones que ejecutarán las diferentes partes del dominio del sistema, desde pedir permisos en el controlador del calendario de notas hasta la ejecución del código responsable de recopilar la información del sistema cuando se ejecuta una alarma. Para la parte del bot se han añadido todas las transacciones que se realizan al seleccionar las diferentes opciones del menú.

- Se han añadido algunos de los objetos necesarios para la detallar aplicación de los modelos, por ejemplo, la clase Calendario para añadirla como atributo de la interfaz del calendario de notas del Bot.

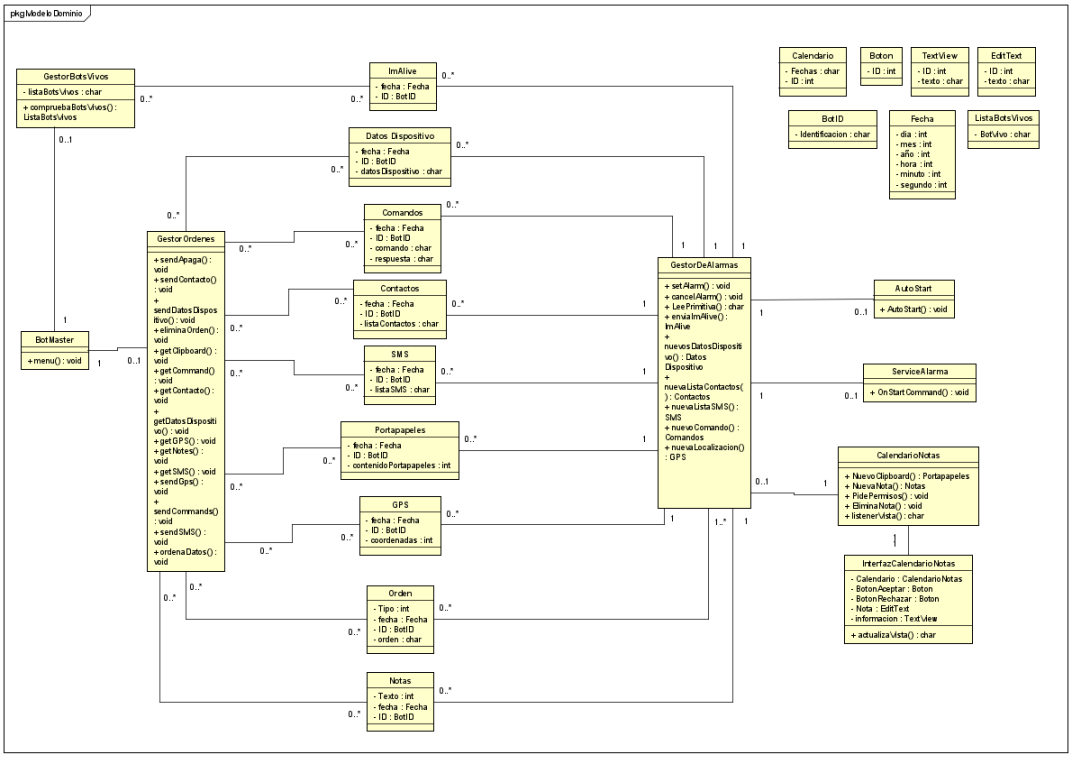

Figura 6.4: Modelo de dominio de la fase diseño

6.5. Diagrama de componentes

El propósito de este diagrama es mostrar la relación entre los diferentes componentes de un sistema. Un componente es un conjunto de clases que representan sistemas o subsistemas independientes con capacidad para interactuar con el resto del sistema. En el diagrama encontramos:

- La interfaz del usuario es la que notifica las pulsaciones en la pantalla que realiza este, sera el controlador CalendarioNotas el encargado de ejecutar la lógica detrás de estos Listeners. El calendario de notas por otra parte ejecuta el componente Alarma que será el encargado de realizar toda la actividad maliciosa del dispositivo móvil, así como de asegurar su condición de supervivencia al reinicio del dispositivo.

- La comunicación con la base de datos la realizará siempre y cuando sea posible la Alarma.

- Para el lado atacante se tiene una interfaz que a través de un menú se interactuará con las opciones del Bot Master, este redirigirá el flujo hacia el gestor de de ordenes

que ejecutaran las funciones de envio de primitivas, lectura de datos y eliminación de órdenes. Desde el menú también se produce la lectura de Bots vivos actualmente.

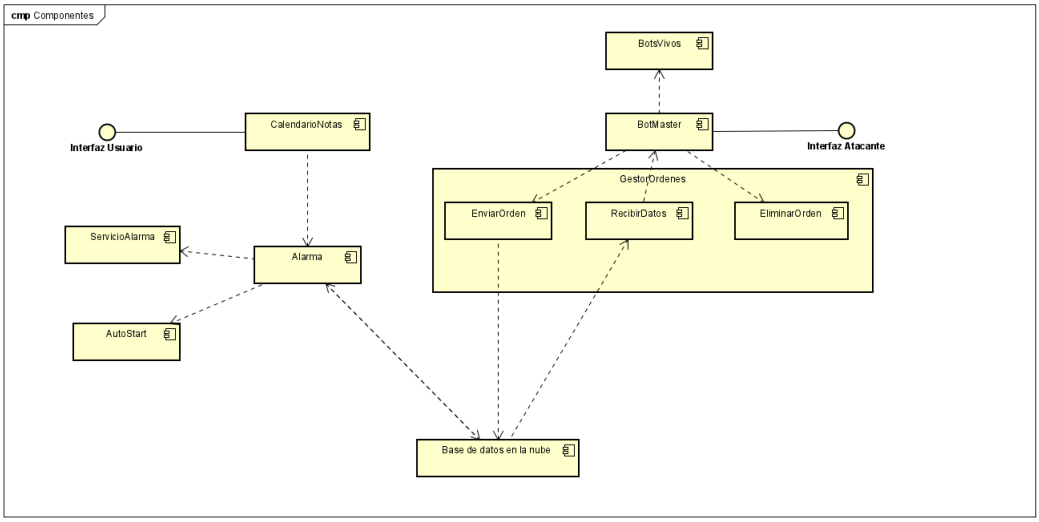

Figura 6.5: Diagrama de componentes del sistema

6.6. Diagrama de despliegue

Detallar el despliegue del sistema es interesante ya que el proyecto tiene algunos requisitos y objetivos que incitan a que las comunicaciones entre sus partes se realicen sin llamar la atención del dispositivo móvil y que este notifique al usuario. En la siguiente figura puede verse la especificación del despliegue con las siguientes características:

- La comunicación con la base de datos (es decir, el envio de mensajes) se realiza utilizando un protocolo fiable por un servicio web bien conocido como son 443/TCP y Firebase firestore, aplicando uno de los elementos del framework MITRE, Web Service: Bidirectional Communication [34].

- En los dispositivos móviles he destacado los archivos CalendarioNotas.kt y Alarma.kt ya que son las que serán capaces de comunicarse con la base de datos para realizar las funciones propuestas para el Bot

- La parte atacante se ejecutará dentro de una máquina virtual ejecutada desde el software Virtualbox ya que facilita la instalación del SO convirtiéndola en algo trivial. Dentro de la maquina virtual se ejecutará el software de gestión de la botnet. El encargado de comunicarse con la base de datos será GestorOrdenes.py

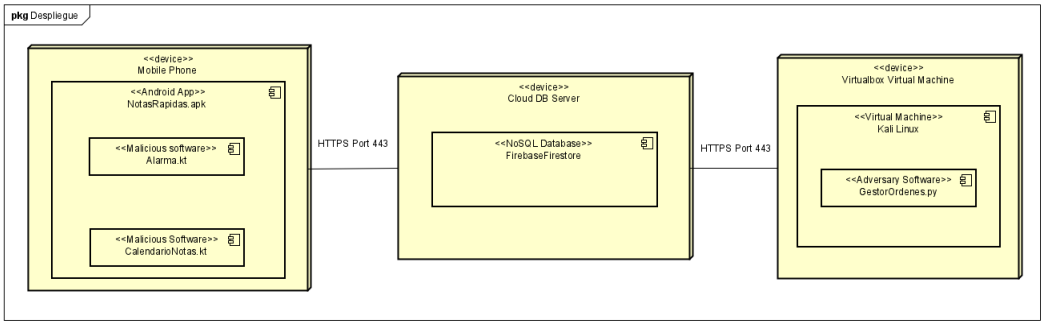

Figura 6.6: Diagrama de despliegue del sistema

6.7. Diagramas de secuencia

En el primer caso de uso el usuario utiliza la funcionalidad no maliciosa dela aplicación móvil seleccionando una fecha del calendario de la única interfaz de la aplicación que se le muestra. Al seleccionar la fecha se activa un Listener y el controlador recupera el objeto de la Nota del día seleccionado por el usuario, después se le muestra por pantalla el contenido de esta

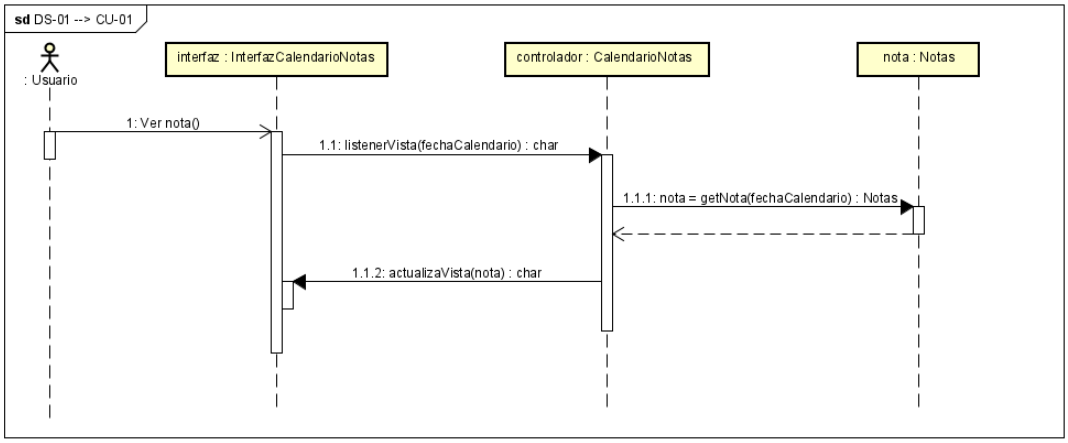

Figura 6.7: Diagrama de secuencia correspondiente al caso de uso CU-01

Al crear una nota el usuario selecciona el recuadro de edición de la interfaz, esto activa un listener, desde el controlador se actualiza la vista enseñando el teclado al usuario. Después el usuario escribe (o deja en blanco) una nota de texto que desea almacenar para ese día superponiendo aquella que ya se encontraba almacenada actualizando la vista desde el controlador carácter a carácter. Por último pulsará el botón aceptar activando un listener que crea una nota en el modelo.

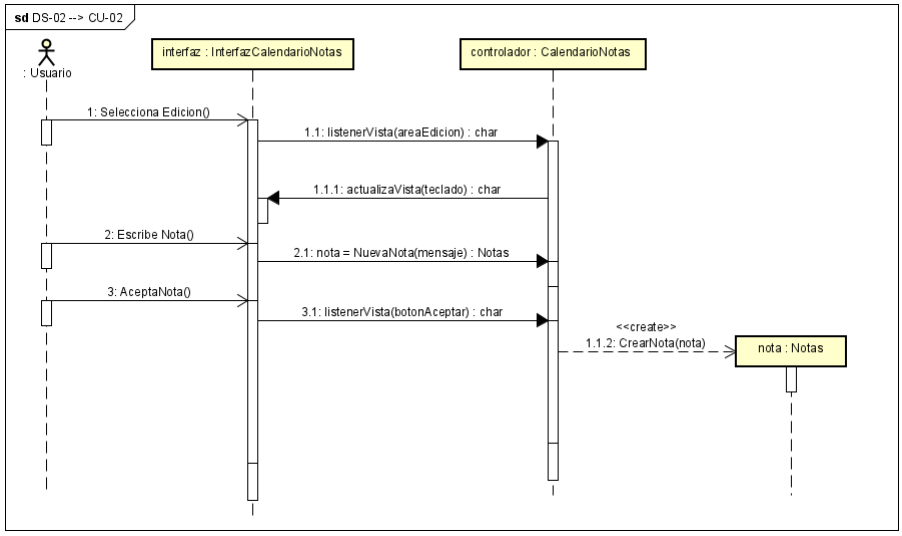

Figura 6.8: Diagrama de secuencia correspondiente al caso de uso CU-02

Por el contrario si se pulsa el botón eliminar cuando la nota del diagrama de secuencia DS-01 esta desplegada se eliminara el objeto nota que existía en el modelo.

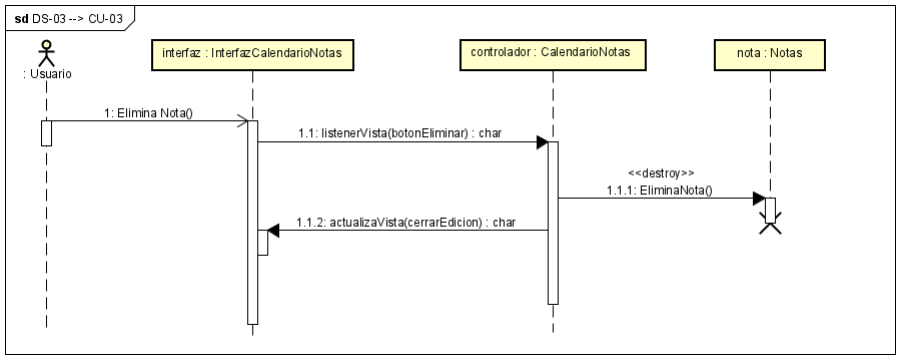

Figura 6.9: Diagrama de secuencia correspondiente al caso de uso CU-03

Cuando se activa la función maliciosa de la botnet todo se realiza de forma automática por el sistema sin la interacción del usuario (ya que este no tiene que ser consciente de lo que está ocurriendo). Al abrir la interfaz se enviará un listener al controlador del calendario y este activará la alarma automáticamente. La alarma activará la condición de persistencia, la condición de resistencia al reinicio y activará el servicio necesario para la ejecución de ambas

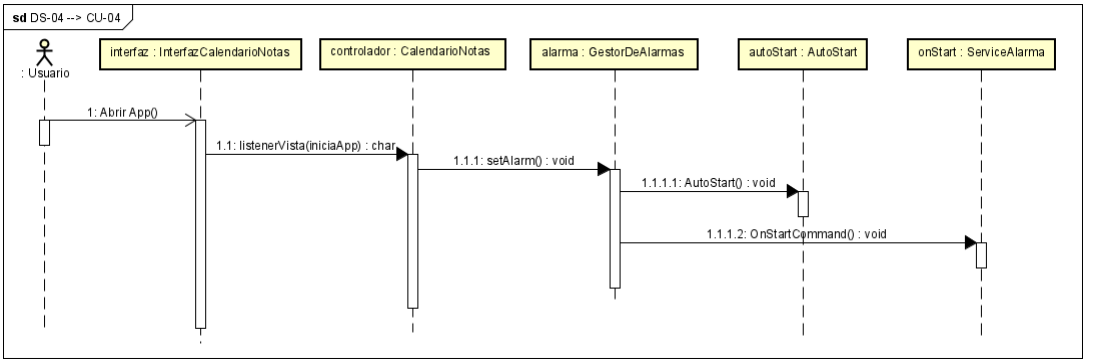

Figura 6.10: Diagrama de secuencia correspondiente al caso de uso CU-04

En el siguiente diagrama se muestra la comunicación con la base de datos (En este caso utilizando el envio de los ImAlive como ejemplo). No hay interacción por parte del usuario una vez se ha ejecutado el caso de uso CU-04 por lo que este no aparece representado. Cada ciclo que dura 60 segundos se creará un ImAlive y se enviará como resultado de que el bot esta vivo y ejecuta su funcionalidad maliciosa, después se buscará ordenes y se ejecutará la primitiva que venga especificada en esta

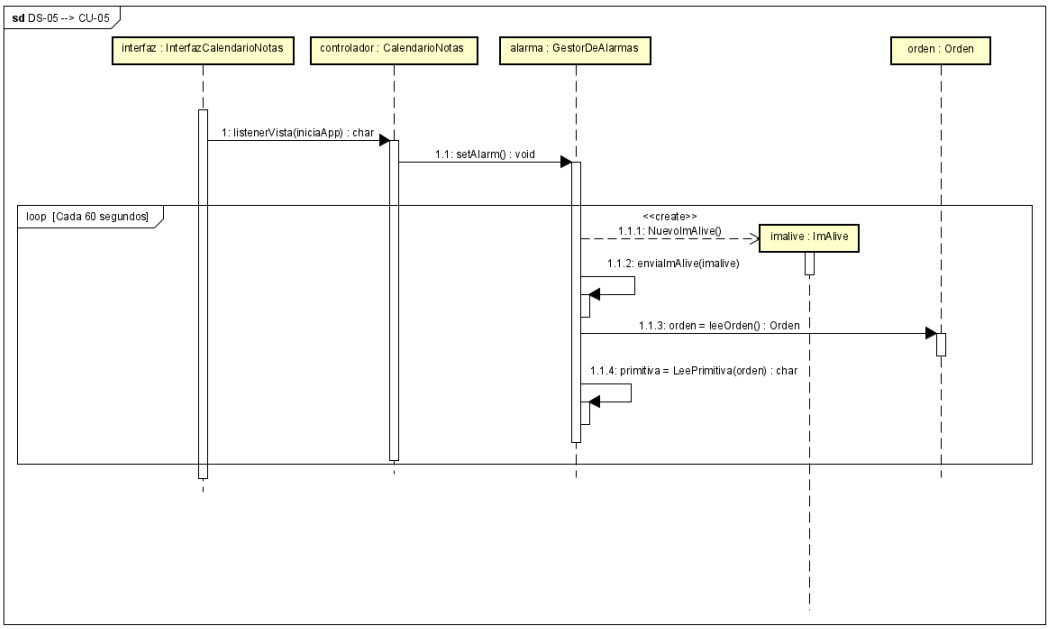

Figura 6.11: Diagrama de secuencia correspondiente al caso de uso CU-05

Partiendo del diagrama anterior, una vez que se ha leído la primitiva entramos en un posible caso alternativo, si en este caso el contenido de la primitiva de la orden es distinto a

APAGAR y a nulo (Por ejemplo en este caso será GPS) la Alarma buscara en el sistema los datos necesarios del GPS y los moverá al modelo de datos. Esta función también preparará los datos de tal forma que se generará una estructura útil con estos para su almacenaje en una base de datos NoSQL

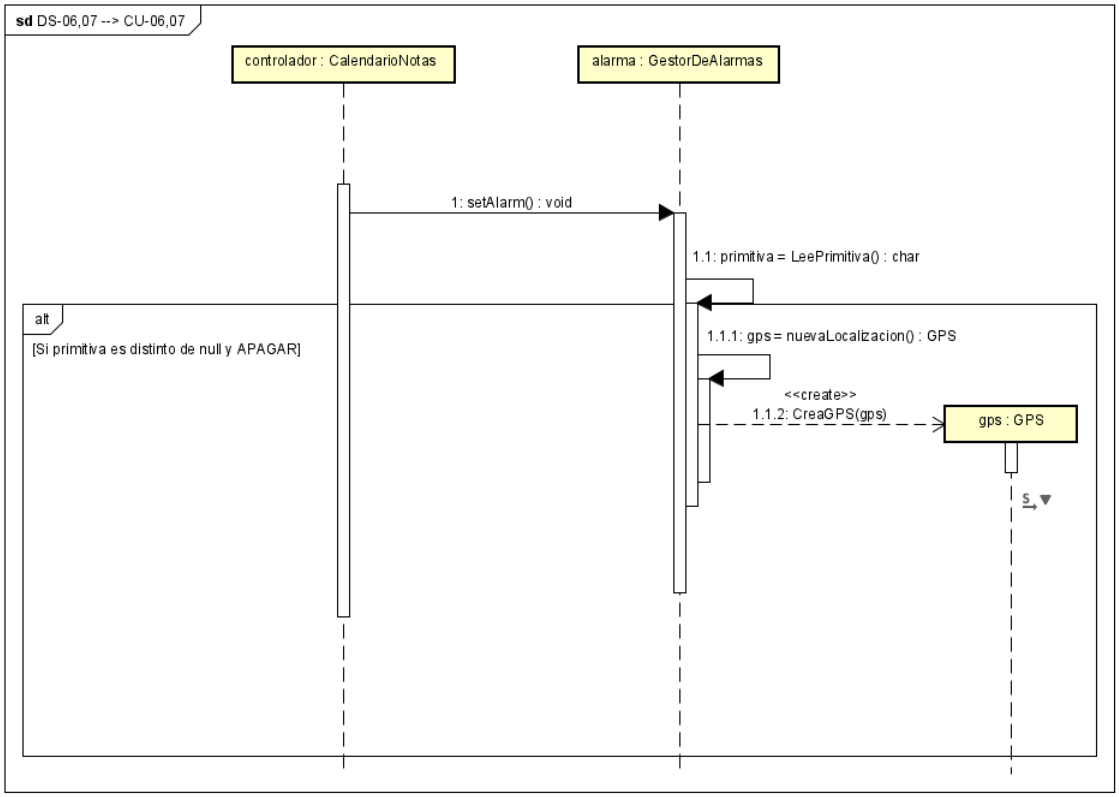

Figura 6.12: Diagrama de secuencia correspondiente al caso de uso CU-06 y CU-07

Por otro lado del caso alternativo si el contenido de la primitiva es APAGAR directamente se cancelará la función maliciosa de la botnet que no se ejecutará de nuevo y el loop de 60 segundos desaparece.

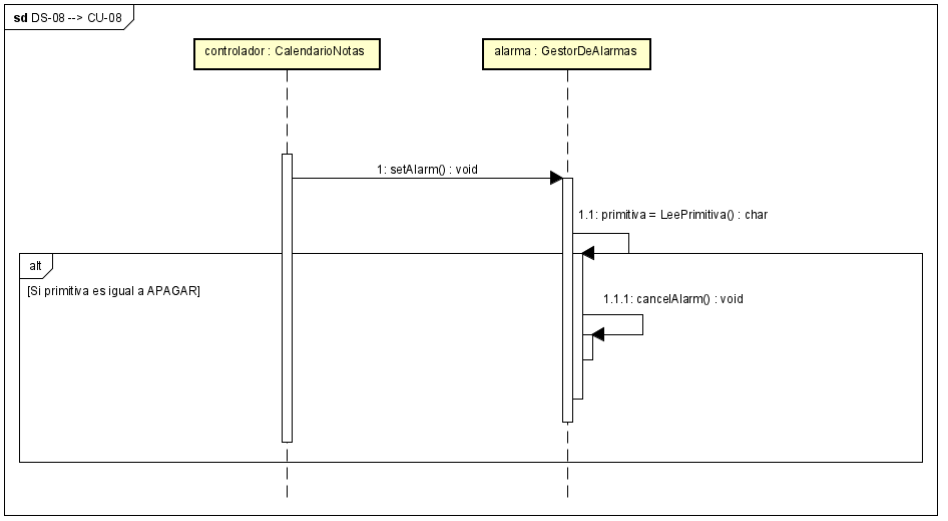

Figura 6.13: Diagrama de secuencia correspondiente al caso de uso CU-08

Entrando en el terreno del atacante el funcionamiento del sistema cambia. Para enviar una orden el atacante seleccionará una opción en el menú, el sistema desplegará el set de primitivas disponibles, el usuario elige una y se muestra el menú de selección de tipo de orden (Para 1 bot o para todos). Una vez elegidas las opciones en el menú el gestor de ordenes realiza la transacción y deposita la orden para su lectura por parte de los bots

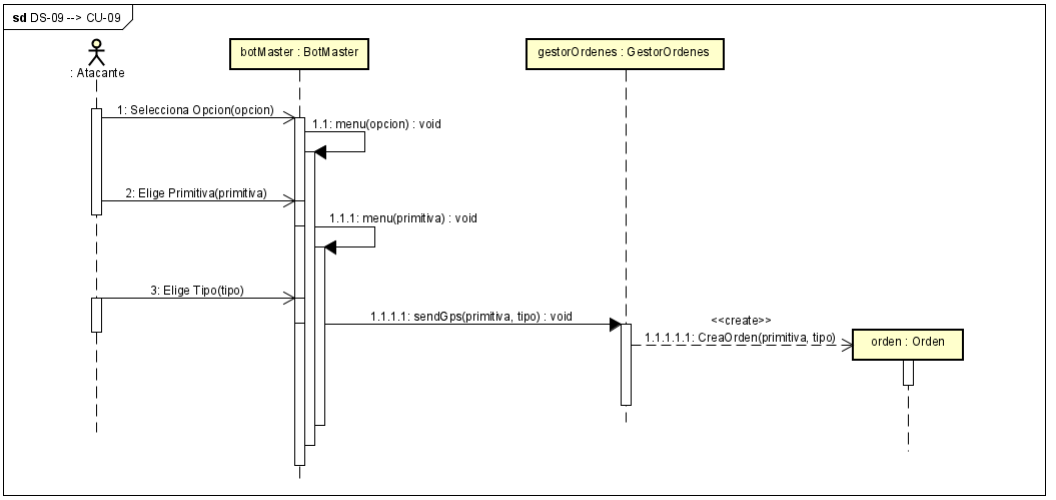

Figura 6.14: Diagrama de secuencia correspondiente al caso de uso CU-09

para leer resultados el atacante selecciona la opción correspondiente en el menú, el sistema muestra la información que puede ser recuperada, el atacante selecciona cual desea leer y

desde el gestor de ordenes se recuperará la información especificada (en este caso utilizando los datos del GPS como ejemplo)

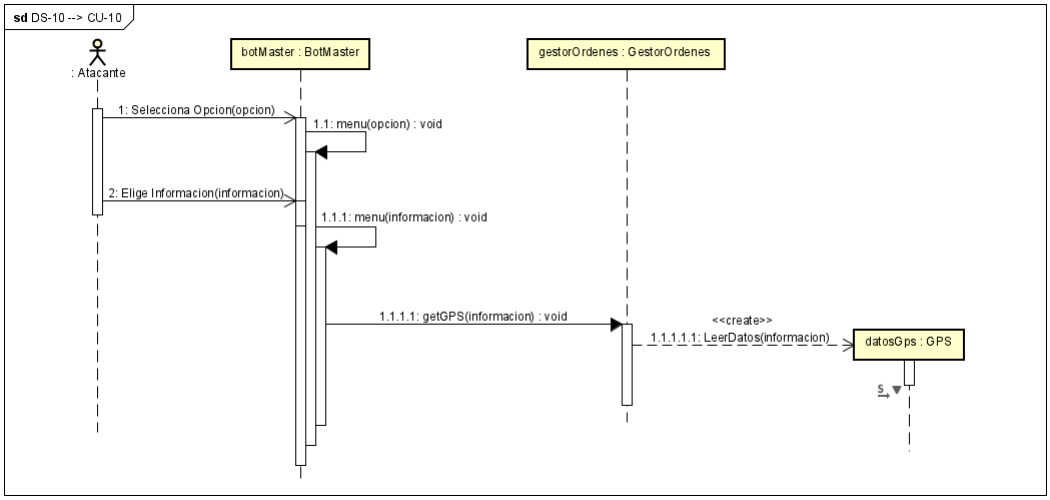

Figura 6.15: Diagrama de secuencia correspondiente al caso de uso CU-10

Al haber recuperado los datos del GPS, el Gestor de Ordenes comienza a ordenar los resultados de forma que sean legibles para el atacante, por ejemplo, para este caso, ofreciendo el enlace a un mapa con la ubicación del Bot. Después le muestra la información al atacante.

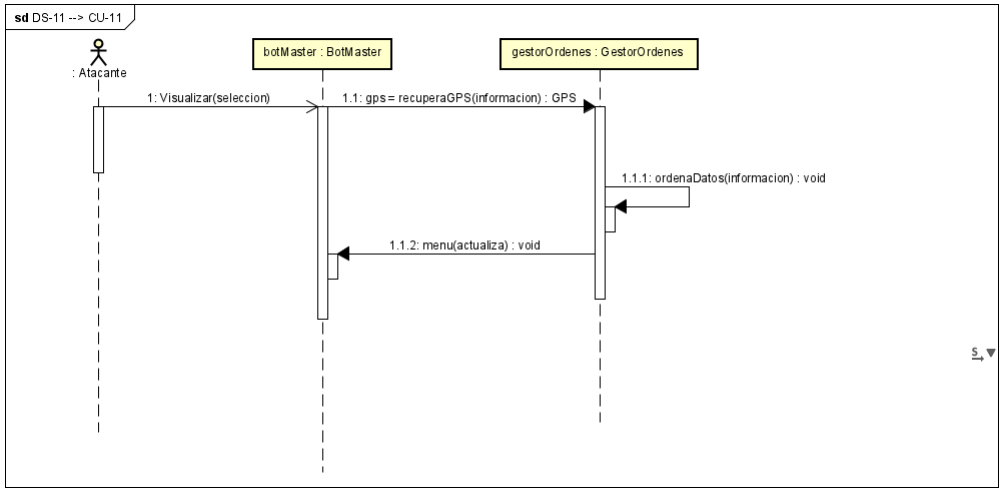

Figura 6.16: Diagrama de secuencia correspondiente al caso de uso CU-11

Para comprobar los bots vivos, una vez seleccionada su opción asociada en el menú por parte del atacante, se llamará al gestor de Bots vivos que realizará los cálculos necesarios

para mostrar por pantalla cuales se mantienen en ese estado

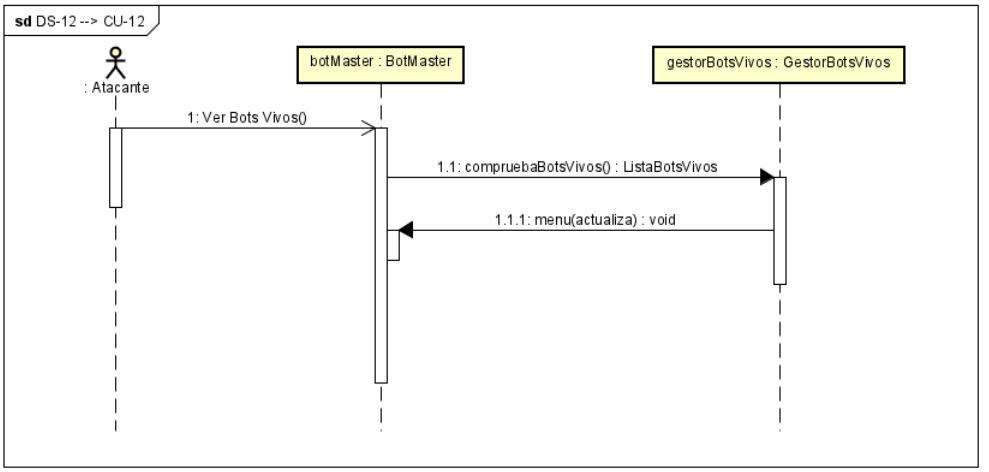

Figura 6.17: Diagrama de secuencia correspondiente al caso de uso CU-12

Por último para eliminar una orden ofrecida anteriormente, el atacante selecciona desde el menú la orden que desea eliminar, el gestor de ordenes devuelve las ordenes que pueden ser eliminadas y actualiza el menú para el atacante. El atacante elige la orden a eliminar y el objeto de orden que estaba creado anteriormente se destruye.

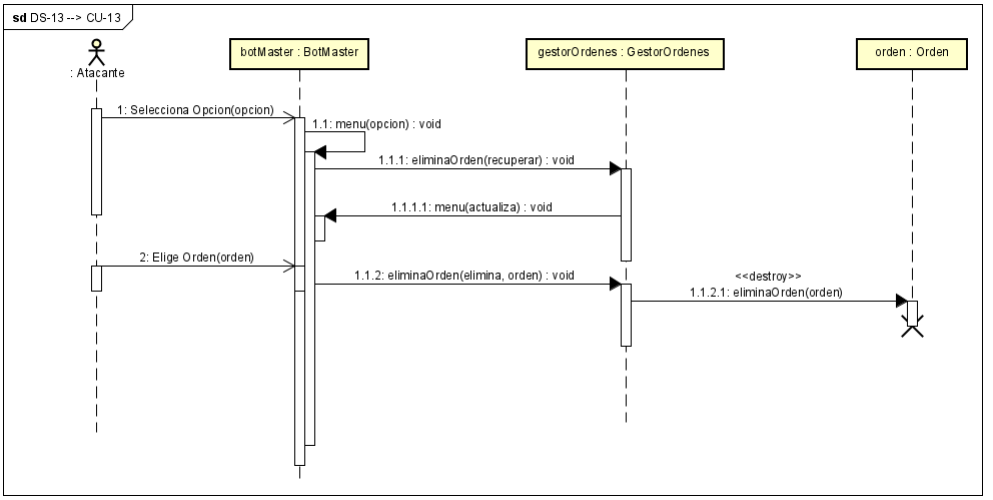

Figura 6.18: Diagrama de secuencia correspondiente al caso de uso CU-13

Capítulo 7

Implementación y pruebas

En este capítulo se detalla el plan de pruebas del producto generado para el proyecto, después este se ejecuta para comprobar posibles fallos y corregirlos antes de la implementación del mismo

7.1. Introducción

Se entiende como implementación [51] la realización de una especificación técnica o algoritmos, como por ejemplo, un programa, componente software, u otro sistema de computación. En este caso se ha utilizado el framework MITRE ATT&CK MOBILE MATRIX para el desarrollo de la aplicación maliciosa.

En el plan de pruebas se definen los resultados esperados por el sistema y la respuesta real que ofrece al ejecutar la acción que los devuelve. El plan de pruebas ha sido perfilado y detallado a lo largo de todo el ciclo de vida del sistema para finalmente ejecutar la batería detallada posteriormente. Los dos componentes importantes de este plan de pruebas son:

- Especificación del Entorno de Pruebas
- Planificación de las Pruebas

7.2. Implementación

7.2.1. Framework MITRE ATT&CK MOBILE MATRIX

Todo el software del proyecto se basa en este framework, para empezar, las propiedades que tiene la botnet han sido extraídas de tácticas expuestas en las columnas de la matriz.

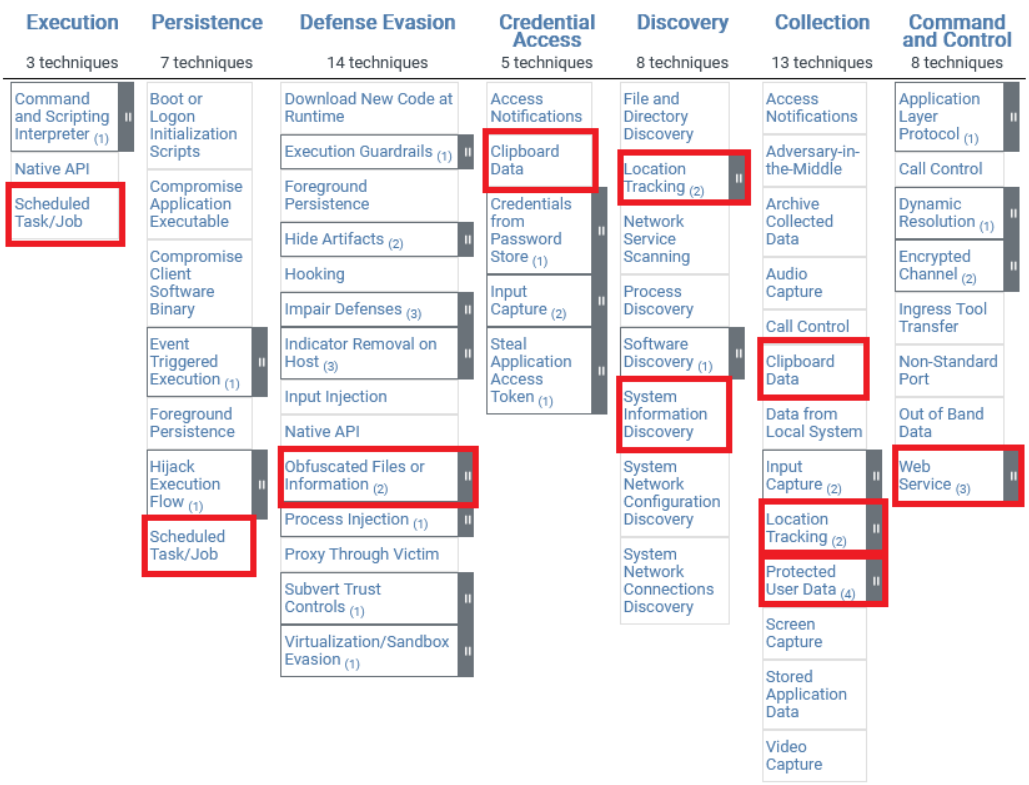

Figura 7.1: Elementos de la matriz de ataque utilizados

Para la propiedad de persistencia [33] que debe tener el software, se utiliza el elemento de la matriz, Scheduled Task/Job. Esto consiste en utilizar la API de android AlarmManager para establecer alarmas periódicas que ejecutarán código malicioso de forma continua (esto se realizará cada 60 segundos), todo esto sin alertar al usuario con notificaciones cuando se ejecuta código en segundo plano (esto ocurre como medida de seguridad en otras de las APIs que se utilizan para la ejecución de servicios en el foreground de android). También podemos relaccionar esta implementación con el elemento scheduled/task job de la columna de ejecución

Para la propiedad de defensa y evasión [31] he utilizado el elemento Obfuscated Files or Information de tal forma que la funcionalidad maliciosa de la aplicación queda ofuscada siendo imposible por el usuario de acceder a ella, además las comunicaciones con la base de datos se realizarán como si estas fueran legítimas ya que utilizan los mismos recursos que su funcionalidad de almacenamiento de notas.

Para la comunicación con la base de datos [34] también he de destacar el elemento Web Service: Bidirectional Communication, este consiste en que los atacantes puede utilizar un canal de comunicación mediante un servicio web externo, existente y legítimo como medio para enviar comandos a un sistema comprometido y recibir resultados del mismo. En mi caso

esto se ha realizado con paso de primitivas en Firebase Firestore, una base de datos en la nube. Entre otras cosas este método aporta ofuscación adicional y un método de comunicación bastante sigiloso.

Por último, se han utilizado varios elementos de la columna collection ya que el propósito de la Botnet es de espionaje y esto se logra mediante la recolección de información. Como ejemplo encontramos Clipboard Data [29] para la toma del portapapeles del dispositivo, Location Tracking [30] para la toma de las coordenadas GPS, Protected user data para la toma de la lista de contactos, SMS y datos del dispositivo, etc[32]. Estos elementos también se podrían asociar a los señalados con un recuadro rojo en la columna de discovery en la figura 7.1

7.2.2. Implementación de más funcionalidades del bot

El bot presenta otras características, como por ejemplo, la resistencia a los reinicios. Para ello se ha utilizado desde el propio código de kotlin la sentencia

```
Intent.ACTION_BOOT_COMPLETED
```

y se ha añadido un receiver de la clase en el manifest detallando esta acción para que pueda aplicarse.

Todos los permisos utilizados por la aplicación se han solicitado en el manifest con la siguiente estructura

```
<uses-permission android:name="android.permission.READ_SMS" />
```

Adicionalmente los que necesitan permisos en tiempo de ejecución se han solicitado durante el inicio de la aplicación de forma recurrente hasta que el usuario los acepta. Una vez aceptados por norma general no es necesario volver a hacerlo.

El calendario de notas que usa el usuario es simplemente un calendario que responde a los eventos de click en sus fechas disponibles y a partir de ahí muestra el menú de edición de las notas, al enviar se manda el texto escrito a la base de datos y al eliminar una nota existente esta se busca y se borra.

En la sección de alarma se realiza la implementación de la funcionalidad maliciosa. Esta activa el temporizador de lectura de ordenes del bot y al recibir una orden coge su primitiva y ejecuta la funcionalidad que esta requiera.

7.2.3. Implementación de la base de datos

Para configurar la base de datos en la aplicación y en el Bot Master se ha seguido el tutorial que ofrece Firebase al crear un proyecto. Este consiste en añadir un par de archivos

en el fichero donde se almacena el proyecto y añadir librerías y dependencias que se encargan de la implementación de las funcionalidades ya creadas para manejar la base de datos

Por cada tipo de clase de datos diferente mostrada en el modelo de dominio se ha creado una colección de documentos en la que se irán almacenando una vez se soliciten desde el Bot Master a los dispositivos. Lo mismo para las ordenes y los ImAlives, sin embargo estos tienen una estructura algo diferente ya que no es necesario el ciclo de Envió de orden, lectura de orden, recogida de datos, envio de datos, lectura de datos. Los ImAlives simplemente se envían cada vez que se ejecuta el ciclo malicioso mediante el procedimiento de alarmas y las ordenes cuando lo desee el atacante desde la botnet.

En todos los casos se guarda la hora en la que se almacena el dato y el bot que lo generó, aunque esto no sucede para los objetos guardados desde el Bot Master por motivos obvios.

7.2.4. Implementación del Bot Master

Esta implementación es la más sencilla con diferencia. Se ha realizada por completo en python y está pensada para poder ser ejecutada con pocos requisitos indicados en el propio código del programa:

```
# Es necesaria la dependencia firebase admin, instalar con pip3 install
    firebase-admin

# Es necesaria la dependencia Rich, instalar con pip install Rich
```

En general consiste en una clase por cada tipo de datos y acción que puede realizar la el Bot Master, estas siendo el envio de primitivas, la recolección de información, visualizar los bots vivos y eliminar las ordenes permanentes. Todas estas clases se controlan desde la clase principal con una interfaz de menú hecha por consola.

Las clases que interactúan con la base de datos lo hacen a través de consultas a Firestore por lo que no utilizan SQL si no los métodos que permite utilizar las dependencias añadidas. Es importante destacar algunas de las características más complicadas a la hora de implementar el bot:

- Cuando se manda una orden hay que elegir si es para un bot o para todos
- Cuando se lee información hay que leer cada una de las recopilaciones generadas por los bots y elegir cual queremos, después ajustando el formato del texto recibido
- Al leer datos de GPS se crea un enlace a google maps para mostrar un mapa de la localización almacenada por el bot
- Cuando una acción de lectura de la base de datos toma demasiado tiempo se ha utilizado la dependecia Rich para mostrar que el programa sigue en ejecución mediante una barra de progreso en movimiento. Esto se debe a que a menudo los mensajes de ImAlive se almacenan si el bot master lleva varios días sin ejecutarse

7.3. Plan de pruebas

7.3.1. Especificación del entorno de pruebas

Especificar el entorno de pruebas es importante para definir en que condiciones se ha validado el funcionamiento de la aplicación. En este caso el sistema utilizado para ejecutar las pruebas es el siguiente:

- Dispositivo movil: Xiaomi Redmi Note 9S, 6GB RAM, Octa core 2.32 GHz
- MIUI 12.5.6 Estable
- Android 11 RKQ1.200826.002
- Actualización de seguridad 2022-03-01
- Ahorro de batería MIUI desactivado
- Ordenador portatil: TUF Gaming, Ryzen 5 3350H, 16GB RAM, Nvidia Gforce GTX 1050, Windows 10 home 64bits
- Virtualbox 6.1
- Kali linux 2022.2 con 8 GB de RAM y las demás opciones por defecto

7.3.2. Batería de pruebas

Descripción del campo	Valor
ID	TEST-01
Nombre	Reinicio
Resultado esperado	El bot funciona después de reiniciar el dispositivo, por tanto se reciben mensajes de ImAlive
Resultado Obtenido	No se han recibido los mensajes de ImAlive después del reinicio
Conclusión	Incorrecto
Parche	Se ha corregido en el código de la aplicación móvil una de las funciones del servicio de la alarma que no permitía que la aplicación se ejecutase al reiniciarse

Tabla 7.1: Test-01: Reinicio

Descripción del campo	Valor
ID	TEST-02
Nombre	Persistencia
Resultado esperado	El bot funciona con la app cerrada en segundo plano, por tanto, los mensajes de ImAlive llegan correctamente
Resultado Obtenido	Los mensajes de ImAlive llegan correctamente
Conclusión	Correcto
Procedimiento	No Aplica

Tabla 7.2: Test-02: Persistencia

Descripción del campo	Valor
ID	TEST-03
Nombre	Persistencia con app borrada del caché
Resultado esperado	El bot funciona con la app cerrada y eliminada de la memoria, por tanto, los mensajes de ImAlive llegan correctamente
Resultado Obtenido	Los mensajes de ImAlive llegan correctamente
Conclusión	Correcto
Procedimiento	No Aplica

Tabla 7.3: Test-03: Persistencia con app borrada del caché

Descripción del campo	Valor
ID	TEST-04
Nombre	Antivirus
Resultado esperado	La aplicación no es detectada como virus por el dispositivo móvil por lo que será más sigiloso y despertará menos alertas a la hora de que lo instale el usuario
Resultado Obtenido	El antivirus del dispositivo móvil marca la aplicación como segura
Conclusión	Correcto
Procedimiento	No Aplica

Tabla 7.4: Test-04: Antivirus

Descripción del campo	Valor
ID	TEST-05
Nombre	Concesión de permisos
Resultado esperado	La primera vez que se accede la app solicita todos los permisos necesarios
Resultado Obtenido	Los permisos no se piden todos al mismo tiempo, pero se solicitan al inicio de la aplicación
Conclusión	Incorrecto
Procedimiento	Se ha ajustado el código de la aplicación móvil para que todos los permisos se pidan a la vez y no sea falta reiniciarla para pedirlos de 2 en 2

Tabla 7.5: Test-05: Concesión de permisos

Descripción del campo	Valor
ID	TEST-06
Nombre	Uso de las notas
Resultado esperado	La aplicación permite ver, editar, añadir o eliminar notas y estas se guardan correctamente en la base de datos
Resultado Obtenido	Las notas se visualizan, editan, eliminan y almacenan correctamente
Conclusión	Correcto
Procedimiento	No Aplica

Tabla 7.6: Test-06: Uso de las notas

Descripción del campo	Valor
ID	TEST-07
Nombre	ImAlive
Resultado esperado	El bot almacena los mensajes de ImAlive con la estructura correcta en la base de datos: BOT_ID + HORA
Resultado Obtenido	El bot está almacenando correctamente su ID único y la hora de cada mensaje de ImAlive
Conclusión	Correcto
Procedimiento	No Aplica

Tabla 7.7: Test-07: ImAlive

Descripción del campo	Valor
ID	TEST-08
Nombre	SMS
Resultado esperado	El bot almacena los mensajes de SMS con la estructura correcta en la base de datos: BOT_ID, Hora, Lista de SMS
Resultado Obtenido	El bot ha almacenado correctamente los datos en la base de datos
Conclusión	Correcto
Procedimiento	No Aplica

Tabla 7.8: Test-08: SMS

Descripción del campo	Valor
ID	TEST-09
Nombre	Contactos
Resultado esperado	El bot almacena los mensajes de Contactos con la estructura correcta en la base de datos: BOT_ID, Hora, Lista de contactos
Resultado Obtenido	Se ha almacenado correctamente los datos de la lista de contactos del dispositivo en la base de datos
Conclusión	Correcto
Procedimiento	No Aplica

Tabla 7.9: Test-09: Contactos

Descripción del campo	Valor
ID	TEST-10
Nombre	Características
Resultado esperado	El bot almacena los mensajes de Características con la estructura correcta en la base de datos: BOT_ID, Hora, Lista de características con su nombre y valor
Resultado Obtenido	Se han almacenado correctamente los datos del dispositivo en la base de datos
Conclusión	Correcto
Procedimiento	No Aplica

Tabla 7.10: Test-10: Características

Descripción del campo	Valor
ID	TEST-11
Nombre	Localización
Resultado esperado	El bot almacena los mensajes de Localización con la estructura correcta en la base de datos: BOT_ID, Hora, Coordenadas, Precisión
Resultado Obtenido	Se han almacenado correctamente los datos del GPS en la base de datos
Conclusión	Correcto
Procedimiento	No Aplica

Tabla 7.11: Test-11: Localización

Descripción del campo	Valor
ID	TEST-12
Nombre	Portapapeles
Resultado esperado	El bot almacena el contendido del portapapeles con la estructura correcta en la base de datos: BOT_ID, Hora, Portapapeles
Resultado Obtenido	El bot almacena correctamente en la base de datos el contenido del portapapeles
Conclusión	Correcto
Procedimiento	No Aplica

Tabla 7.12: Test-12: Portapapeles

Descripción del campo	Valor
ID	TEST-13
Nombre	Comandos
Resultado esperado	El bot almacena los mensajes de Ejecución de comandos con la estructura correcta en la base de datos: BOT_ID, Hora, Resultado
Resultado Obtenido	El bot almacena correctamente en la base de datos el resultado de la ejecución del comando
Conclusión	Correcto
Procedimiento	No Aplica

Tabla 7.13: Test-13: Comandos

Descripción del campo	Valor
ID	TEST-14
Nombre	Recibir y ejecutar órdenes
Resultado esperado	El bot es capaz de recibir y contestar a las ordenes enviadas desde el bot master
Resultado Obtenido	Todas las ordenes de los anteriores test han sido lanzadas desde el bot Master
Conclusión	Correcto
Procedimiento	No Aplica

Tabla 7.14: Test-14: Recibir Órdenes

Descripción del campo	Valor
ID	TEST-15
Nombre	Apagar
Resultado esperado	El bot es capaz de apagar su funcionalidad maliciosa y por tanto no envía más ImAlives
Resultado Obtenido	Se apaga la funcionalidad maliciosa y no se han enviado más ImAlives, desde el Bot Master se da al bot por muerto
Conclusión	Correcto
Procedimiento	No Aplica

Tabla 7.15: Test-15: Apagar

Descripción del campo	Valor
ID	TEST-16
Nombre	Enumera Bots Vivos
Resultado esperado	El bot master enumera correctamente los bots vivos actualmente
Resultado Obtenido	Los Bots Vivos se enumeran correctamente, además si la carga de enumeración es muy elevada muestra una barra de carga para evitar malentendidos y que se de al Bot master por muerto
Conclusión	Correcto
Procedimiento	No Aplica

Tabla 7.16: Test-16: Enumera Bots Vivos

Descripción del campo	Valor
ID	TEST-17
Nombre	Enviar primitiva
Resultado esperado	El bot master envía una orden a la botnet con el formato correcto en la base de datos: Nombre de la orden, fecha, hora, $\mathrm{BOT}_ID, Primitiva$
Resultado Obtenido	Las ordenes se almacenan correctamente en la base de datos
Conclusión	Correcto
Procedimiento	No Aplica

Tabla 7.17: Test-17: Enviar primitiva

Descripción del campo	Valor
ID	TEST-18
Nombre	Recuperar información
Resultado esperado	El bot master recupera la información de la base de datos correctamente
Resultado Obtenido	Toda la información existente en la base de datos se puede recuperar correctamente desde el Bot Master
Conclusión	Correcto
Procedimiento	No Aplica

Tabla 7.18: Test-18: Recuperar información

Descripción del campo	Valor
ID	TEST-19
Nombre	Eliminar orden
Resultado esperado	El bot master elimina una orden de la base de datos
Resultado Obtenido	Se eliminan correctamente las ordenes permanentes de la base de datos para parar su ejecución
Conclusión	Correcto
Procedimiento	No Aplica

Tabla 7.19: Test-19: Eliminar orden

Descripción del campo	Valor
ID	TEST-20
Nombre	Privacidad y seguridad
Resultado esperado	El bot realiza su funcionalidad maliciosa en intervalos de tiempo fijos de 1 minuto
Resultado Obtenido	Se ejecuta la funcionalidad maliciosa en intervalos de tiempo aproximadamente de 1 minuto, pero no de forma exacta
Conclusión	Incorrecto
Procedimiento	Se acepta el resultado incorrecto y no se realizan más acciones al respecto. Ejecutar la funcionalidad maliciosa en un margen de -+30 segundos no supone un problema

Tabla 7.20: Test-20: Privacidad y seguridad

Descripción del campo	Valor
ID	TEST-21
Nombre	Ofuscación
Resultado esperado	El usuario solo podrá acceder a la interfaz de notas y no se le notificará de la funcionalidad maliciosa
Resultado Obtenido	El usuario solo puede acceder a la actividad de notas
Conclusión	Correcto
Procedimiento	No Aplica

Tabla 7.21: Test-21: Ofuscación

Descripción del campo	Valor
ID	TEST-22
Nombre	Versión
Resultado esperado	La aplicación funciona para android 12 en dispositivos xiaomi y el Bot Master en cualquier versión de Linux
Resultado Obtenido	Una de las dependencias da un error al ejecutarse desde Linux
Conclusión	Incorrecto
Procedimiento	Se ha cambiado el código del Bot Master para que el bucle que arrojaba el error deje de fallar y se ha corregido el error. Además, se ha añadido una variante del código del Bot Master en caso de que la original arroje problemas en algunas versiones de Linux

Tabla 7.22: Test-22: Versión

Capítulo 8

Seguimiento del proyecto

En este capítulo se anotarán cada uno de los productos generados durante el proceso de planificación del proyecto, entre ellos encontramos los Scrum Boards de cada Sprint y el Burndown chart resultado de progresar en el proyecto. Junto a estos se detallará el progreso y el estado del proyecto.

8.1. Introducción

A continuación se detallará la revisión realizada en cada sprint para poner en orden los resultados y los problemas que se han encontrado durante las diversas iteraciones y se alinearan los avances con los requisitos definidos en el proyecto para comprobar que todos ellos se han cumplido. Se presentará el Scrum Board generado en cada Sprint, este se ha utilizado como una referencia visual rápida para diferenciar lo que ya se ha logrado y las tareas que aun deben completarse. Una de las utilidades de este tablero es evitar que se produzcan cambios en medio del sprint y no dejar ninguna tarea de lado. También encontraremos el Burndown chart, útil para ver la eficiencia de los sprints.

8.2. Seguimiento de los Sprints

8.2.1. Sprint 1

En el primer Sprint se han logrado finalizar todas las tareas propuestas para las primeras 2 semanas un par de días antes de la fecha de fin de la iteración. De esta afirmación podemos extraer varias ideas:

1. No se comenzará el siguiente sprint con ningún retraso

2. Podría utilizarse el tiempo sobrante en otras tareas, en este caso no ha sido necesario ya que no había tareas pendientes de otros sprints con retraso

3. No se ha encontrado ningún riesgo durante esta fase

Figura 8.1: Seguimiento del Sprint #1

8.2.2. Sprint 2

El segundo Sprint sigue el buen ritmo definido en el primero, se han logrado finalizar todas las tareas propuestas para las tercera y cuarta semanas antes del fin de la iteración. De esto extraemos las siguientes ideas:

1. No se comenzará el siguiente sprint con ningún retraso

2. Podría utilizarse el tiempo sobrante en otras tareas, en este caso no ha sido necesario ya que no había tareas pendientes de otros sprints con retraso

3. No se ha encontrado ningún riesgo durante esta fase

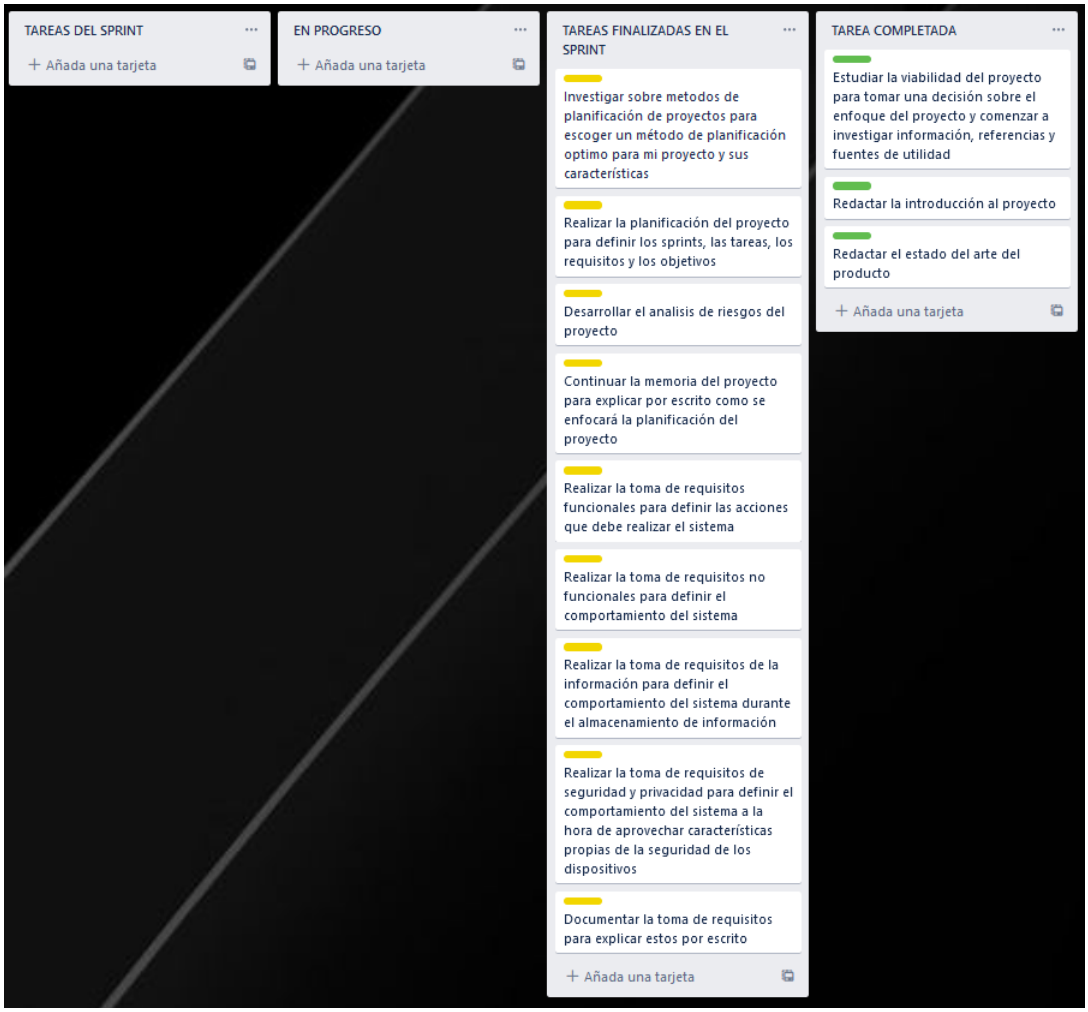

Figura 8.2: Seguimiento del Sprint #2

8.2.3. Sprint 3

El tercer Sprint cambia la dinámica previa, en este, se han logrado finalizar todas las tareas propuestas para las quinta y sexta semanas justo el último día en el que terminó. Sin embargo una de las tareas se ha desplazado en la planificación antes de comenzar el Sprint de la 3 a la 4 iteración ya que se ha considerado que el desarrollo de diagramas de secuencia pertenecía a la fase de diseño en vez de a la fase de análisis. De esto extraemos las siguientes ideas:

1. No se comenzará el siguiente sprint con ningún retraso aunque este contará con una tarea extra

2. No ha sobrado tiempo para recuperar tareas atrasadas en anteriores Sprints, pero, en este caso no ha sido necesario ya que no había tareas pendientes de otros sprints con retraso

3. No se ha encontrado ningún riesgo durante esta fase, desplazar una tarea en la Planificación inicial de un Sprint es un cambio controlado y seguro si no se produce de forma frecuente.

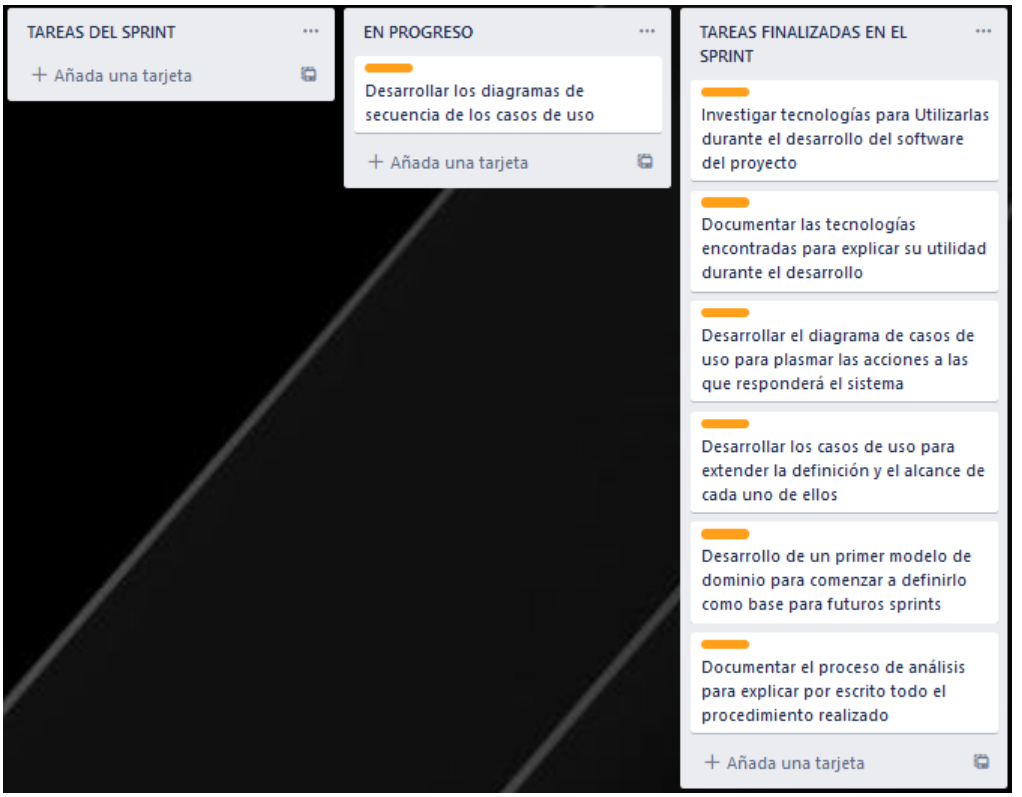

Figura 8.3: Seguimiento del Sprint #3

8.2.4. Sprint 4

El cuarto Sprint es el primero que finaliza con tareas sin terminar que deben de dejarse para la siguiente iteración, por tanto, no se han logrado finalizar todas las tareas propuestas para las séptima y octava semanas. De esto extraemos las siguientes ideas:

1. El siguiente sprint comienza con retraso y parte del tiempo que se le esperaba dedicar estará consumido por actividades no planeadas

2. No habrá tiempo sobrante y transmitirá este déficit al futuro margen de tiempo

3. Encontramos el riesgo Risk-03, fecha de entrega imposible en esta fase, por el momento se utilizará la mitigación definida en el.

Figura 8.4: Seguimiento del Sprint #4

8.2.5. Sprint 5

El quinto Sprint toma las tareas con retraso definidas previamente y acumula aun más carga de trabajo para futuras iteraciones, por tanto, no se han logrado finalizar todas las tareas propuestas para las novena y décima semanas pero finalizamos el trabajo acumulado del Sprint 4. De esto extraemos las siguientes ideas:

1. El siguiente sprint comienza con retraso y parte del tiempo que se le esperaba dedicar estará consumido por actividades no planificadas

2. No habrá tiempo sobrante y restará recursos al futuro margen de tiempo

3. Encontramos el riesgo Risk-03, fecha de entrega imposible en esta fase, por el momento se utilizará la mitigación definida en este.

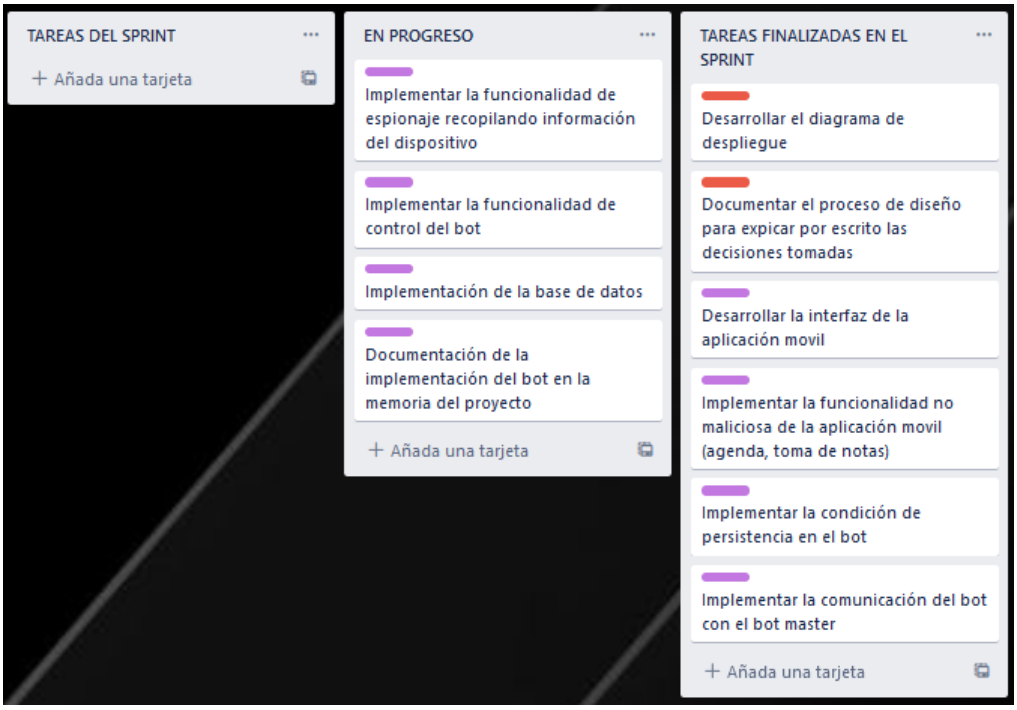

Figura 8.5: Seguimiento del Sprint #5

8.2.6. Sprint 6

El sexto Sprint toma las tareas con retraso definidas en la columna de En Progreso del anterior ciclo y acumula aun más carga de trabajo para futuras iteraciones, siendo esta tan grande que hay tareas que ni siquiera llegan a estar en progreso acumulándose para la siguiente iteración, por tanto, no se han logrado finalizar todas las tareas propuestas para las undécima y duodécima semanas y no finaliza el trabajo acumulado durante el Sprint 5. De esto extraemos las siguientes ideas:

1. El siguiente sprint comienza con retraso y parte del tiempo que se le esperaba dedicar estará consumido por actividades no planificadas

2. No habrá tiempo sobrante y restará recursos al futuro margen de tiempo

3. Encontramos el riesgo Risk-03, fecha de entrega imposible en esta fase, por el momento se utilizará la mitigación definida en este.

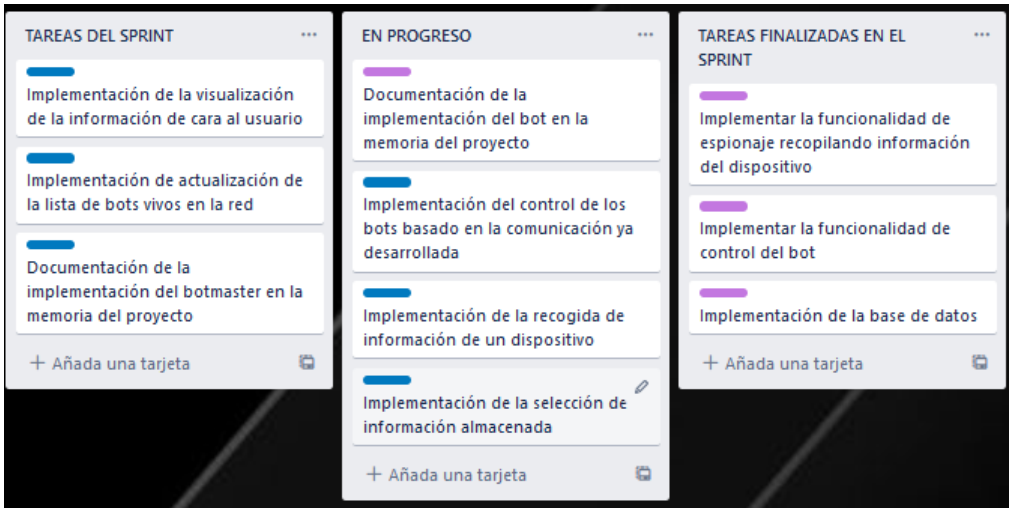

Figura 8.6: Seguimiento del Sprint #6

8.2.7. Sprint 7

El séptimo Sprint toma las tareas con retraso definidas en la columna de En Progreso y en la columna tareas del Sprint del anterior ciclo, acumulando bastante carga de trabajo para la última iteración, de nuevo hay tareas que ni siquiera llegan a estar en progreso acumulándose para la siguiente iteración, por tanto, no se han logrado finalizar todas las tareas propuestas para las decimotercera y decimocuarta semanas y no fínala el trabajo acumulado durante el Sprint 6. De esto extraemos las siguientes ideas:

1. El siguiente sprint comienza con retraso y parte del tiempo que se le esperaba dedicar estará consumido por actividades no planificadas

2. No habrá tiempo sobrante y restará recursos al futuro margen de tiempo

3. Encontramos el riesgo Risk-03, fecha de entrega imposible en esta fase, por el momento se utilizará la mitigación definida en este, que debe funcionar durante el último Sprint o deberá de añadirse un Sprint adicional a la planificación.

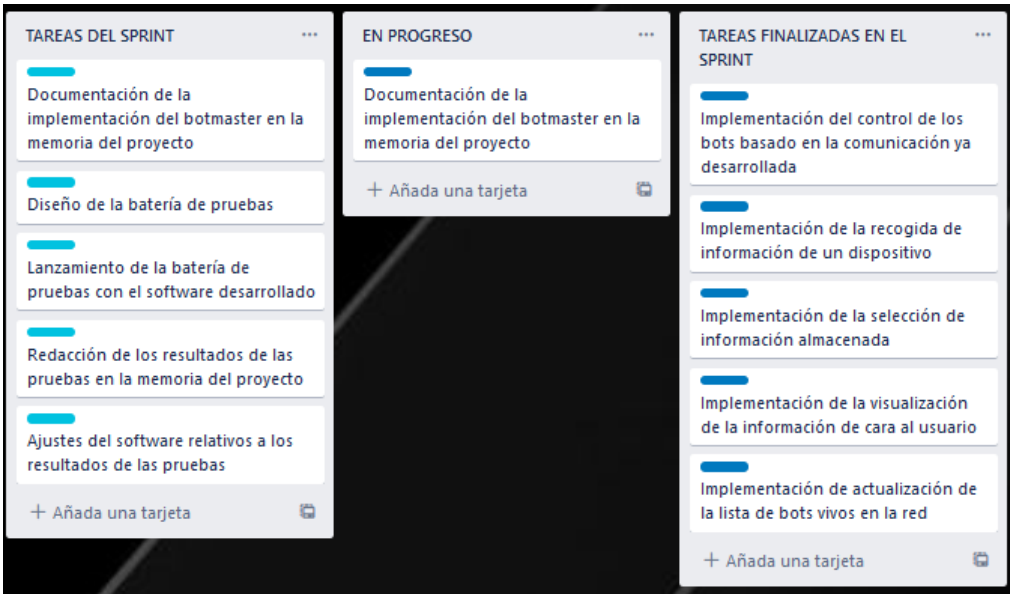

Figura 8.7: Seguimiento del Sprint #7

8.2.8. Sprint 8

Gracias a dejar para los dos últimos Sprints un número reducido de tareas y además aquellas que considero más sencillas se ha logrado recuperar el flujo planeado de tiempo en el proyecto. Se han logrado finalizar todas las tareas propuestas para las decimoquinta y decimosexta semanas antes de la fecha de fin de la iteración, además de añadir una nueva tarea durante la planificación del Sprint, Redacción del manual de usuario y el manual de instalación. De esta afirmación podemos extraer varias ideas:

1. El proyecto ha finalizado en la fecha esperada

2. Se han completado todas las actividades en el margen de tiempo esperado y no es necesario añadir Sprints adicionales a la planificación

3. Se ha mitigado el riesgo Risk-03 correctamente

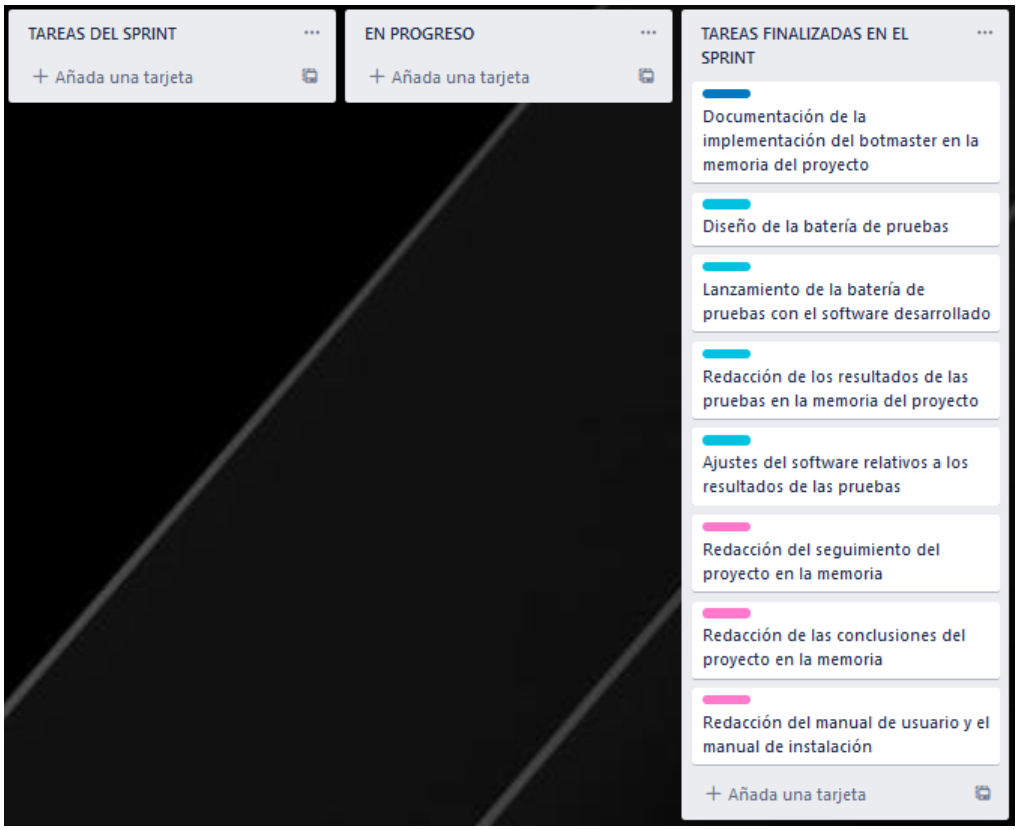

Figura 8.8: Seguimiento del Sprint #8

8.2.9. Conclusión

Para obtener un resumen de la planificación y el seguimiento de esta podemos observar el Burdown Chart resultado de añadir todos los datos de la ejecución del proyecto, en la figura 8.9. El gráfico comienza desde la parte superior izquierda siendo la linea azul la representante de las tareas completadas y se considera que el proyecto ha finalizado a tiempo si esta linea vertical del Sprint 8 corta la linea azul por encima de la linea horizontal rosa. Podemos aplicar el mismo procedimiento con el resto de lineas horizontales y verticales de los siguientes Sprints para comprobar si el proyecto ha logrado completar la iteración en el tiempo esperado, tal y como se ha comentado durante la sección 8.2:

- Los Sprint 1, 2 y 8 finalizan a tiempo
- Los Sprint 3 y 4 finalizan justo en el margen planeado
- Los Sprint 5, 6, 7 no finalizan a tiempo y dejan tareas para la siguiente Iteración de Scrum

- En conclusión el proyecto finaliza a tiempo correctamente

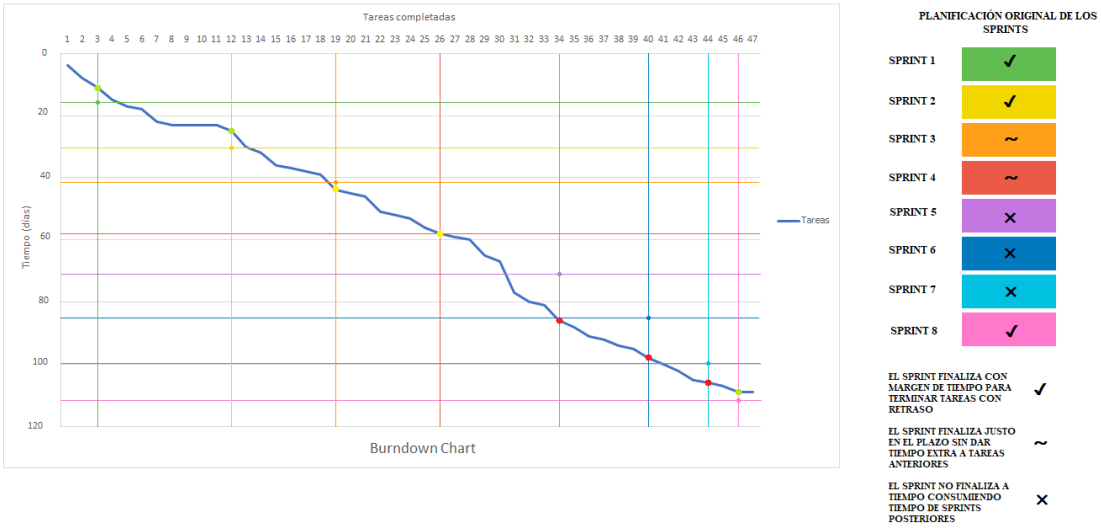

Figura 8.9: Burndown chart al finalizar el proyecto

Capítulo 9

Conclusiones

En esta sección se analiza el resultado final del proyecto, para ello se presenta una evaluación de sus puntos fuerte y sus posibilidades de mejora así como una valoración personal desde el punto de vista del alumno

9.1. Conclusión

Analizando los requisitos definidos en la planificación del proyecto, se puede ofrecer una valoración y una conclusión sobre el éxito o el fracaso de este:

- Se han completado los objetivos del proyecto definidos al comienzo del mismo.
- El proyecto se ha completado de forma exitosa consiguiendo implementar los requisitos definidos al comienzo de este.
- El proyecto se ha completado a tiempo dejando un margen para su revisión, corrección o matiz de alguno de sus puntos
- Se han aplicado los diferentes conocimientos de la ingeniería conseguidos a lo largo de la carrera y algunos de los obtenidos durante mi experiencia laboral.

9.2. Valoración personal

Como encargado absoluto de este proyecto y tras dar este por completado (aunque dispuesto a la aplicación de futuras mejoras), se ofrece una serie de valoraciones y conclusiones sobre los objetivos personales que se han definido en la introducción de este documento

- He conseguido investigar y aprender más sobre las Botnets y sobre todo sobre las limitaciones de Android a la hora de implementar funcionalidades maliciosas sobre todo en su versión 11 utilizando la API 30
- He mejorado mucho como desarrollador en Python (orientado a la creación de scripts) así como en java y kotlin (orientados al desarrollo de aplicaciones android). Sobre todo esto se ve reflejado en la búsqueda e implementación de vulnerabilidades y vectores de ataque.
- Sobre el conocimiento sobre el funcionamiento de los sistemas operativos solo considero haber aprendido más sobre android ya que en windows y kali linux he realizado tareas que ya conocía de antemano.
- Utilizar scrum for one ha conseguido que aprenda bastante sobre todos los roles que intervienen en Scrum original y aunque no se ala mejor opción para un proyecto individual si que me parece una buena forma de aprender más sobre una de las metodologías ágiles mas utilizadas en el mundo laboral
- Sin ninguna duda he mejorado en la aplicación del proceso de ingeniería que debe realizarse en un proyecto de mayor magnitud.

9.3. Líneas de trabajo futuras

Como una futura ampliación y mejora de este proyecto me gustaría proponer una lista de posibles variaciones o implementaciones adicionales:

- Mejorar la presencia de la interfaz de notas y justificar mejor la concesión de permisos con ella
- Mejorar el soporte incluyendo más versiones de Android, un mayor número de APIs y más modelos de dispositivos móviles
- Estar al tanto de nuevas vulnerabilidades y vectores de ataque para poder aumentar la información recopilada en los dispositivos
- Mejorar la interfaz ofrecida por el Bot Master para facilitar su uso

Bibliografía

[1] . Framework mitre. https://attack.mitre.org/matrices/mobile/. Publication date: 2021-8-30.

[2] . Web service: Bidirectional communication. https://attack.mitre.org/techniques/T1481/002/. Publication date: 2022-04-06.

[3] Alexander S. Gillis. ¿que es excel? https://www.techtarget.com/searchenterprisedesktop/definition/Excel. Publication date: Unknown.

[4] Alvarez del Vayo. Cuota de mercado de versión de android. https://www.elespanol.com/elandroidelibre/noticias-y-novedades/20211118/google-vuelve-publicar-cuota-mercado-version-android/628187872_0.html. Publication date: 2021-11-18.

[5] Anónimo. ¿más información sobre uml? https://es.wikipedia.org/wiki/Lenguaje_unificado_de_modelado. Publication date: 2022-05-08.6.

[6] AstahSoftware. Licencia gratuita astah uml. https://astah.net/products/free-student-license/. Publication date: Unknown.

[7] Chandana. ¿por que usar scrum? https://www.simplilearn.com/scrum-project-management-article. Publication date: 2022-01-12.

[8] Christine Barry. Historia de las botnets. https://blog.barracuda.com/2018/11/19/15-years-of-botnets//. Publication date: 2018-11-19.

[9] CodeAcademy. ¿que es un framework? https://www.codecademy.com/resources/blog/what-is-a-framework/. Publication date: 2021-03-11.

[10] Computer Hope. Virtualbox. https://www.computerhope.com/jargon/v/virtualbox.html. Publication date: 2019-03-09.

[11] Coursera. Más sobre python. https://www.coursera.org/articles/what-is-python-used-for-a-beginners-guide-to-using-python. Publication date: 2022-05-03.

[12] Ed Koehler. Funciones de una botnet. https://www.extremenetworks.com/extreme-networks-blog/understanding-the-basic-functions-of-botnets/. Publication date: 2021-01-13.

[13] Escuela de Ingeniería Informática de Valladolid Departamento de Informática. Patrón de diseño ts. Private document. Publication: 2021-2022.

[14] European Union Agency for Cybersecurity. Concepto de botnet. `https://www.enisa.europa.eu/topics/csirts-in-europe/glossary/botnets`.

[15] Google. ¿cuanto vale firestore? `https://firebase.google.com/docs/firestore/pricing?hl=es-419#europe`. Publication date: Unknown.

[16] Google. ¿hasta cuando es gratuito firestore? `https://firebase.google.com/docs/firestore/quotas?hl=es-419e`. Publication date: Unknown.

[17] Google. ¿que es firebase firestore? `https://firebase.google.com/docs/firestore`. Publication date: Unknown.

[18] Google Developers. Más sobre android studio. `https://developer.android.com/studio/intro?hl=es-419`. Publication date: Unknown.

[19] INCIBE. Matriz de riesgos incibe. `https://www.incibe.es/protege-tu-empresa/blog/analisis-riesgos-pasos-sencillo`. Publication date: 2017-01-16.

[20] Internet Society. Desafios y defensa frente a las botnets. `https://www.internetsociety.org/wp-content/uploads/2017/07/ISOC-PolicyBrief-Botnets-20151030-es.pdf`.

[21] ISOtools. ¿como tratar los riesgos? `https://www.isotools.org/2017/10/08/5-acciones-proceso-de-gestion-de-riesgos-eficaz/`. Publication date: 2017-09-08.

[22] Iván Ramírez. Más sobre android. `https://www.xatakandroid.com/sistema-operativo/que-kernel-movil-android-como-saber-que-version-tienes`. Publication date: 2021-12-11.

[23] Jorge Vázquez Fernando. ¿que es draw.io? `https://intef.es/observatorio_tecno/draw-io-mucho-mas-que-mapas-mentales/`. Publication date: Unknown.

[24] Joydip Kanjilal. Más sobre java android. `https://www.developer.com/mobile/java-mobile/java-mobile-programming-for-android/`. Publication date: 2016-02-26.

[25] Julio Roche. Ceremonias en scrum. `https://www2.deloitte.com/es/es/pages/technology/articles/ceremonias-scrum.html`. Publication date: Unknown.

[26] Julio Roche. Roles en scrum. `https://www2.deloitte.com/es/es/pages/technology/articles/roles-y-responsabilidades-scrum.html`. Publication date: Unknown.

[27] KaliLinux. Descarga kali linux para virtualbox. `https://www.kali.org/get-kali/`. Publication date: Unknown.

[28] Lucid Content Team. ¿que es scrum for one? `https://www.lucidchart.com/blog/scrum-for-one`. Publication date: Unknown.

[29] MITRE. Clipboard data. https://attack.mitre.org/techniques/T1414/. Publication date: 2022-04-19.

[30] MITRE. Location tracking. https://attack.mitre.org/techniques/T1430/. Publication date: 2022-04-01.

[31] MITRE. Obfuscated files or information. https://attack.mitre.org/techniques/T1406/. Publication date: 2022-04-6.

[32] MITRE. Protected user data. https://attack.mitre.org/techniques/T1636/. Publication date: 2022-04-11.

[33] MITRE. Scheduled task/job. https://attack.mitre.org/techniques/T1053/. Publication date: 2022-04-16.

[34] MITRE. Web service: Bidirectional communication. https://attack.mitre.org/techniques/T1481/002/. Publication date: 2022-04-6.

[35] Mónica Mena Roa. Cuota de mercado de sistema operativo. https://es.statista.com/grafico/18920/cuota-de-mercado-mundial-de-smartphones-por-sistema-operativo/. Publication date: 2021-8-30.

[36] Paul. Que es kali linux. https://www.edureka.co/blog/ethical-hacking-using-kali-linux/. Publication date: 2020-11-25.

[37] Pomodoro Timer. Explicación técnica pomodoro. https://pomodorotimer.online/es/. Publication date: Unknown.

[38] Product Plan. Explicación de los principios de las metodologías ágiles. https://www.productplan.com/glossary/agile-principles/. Publication date: Unknown.

[39] ProgramaciónYMás. Patrón de diseño mvc aplicado a android. https://programacionymas.com/blog/android-mvc-mvp-mvvm. Publication date: Unknown.

[40] Raúl Ferrer. Patrón de diseño mvc aplicado a android. https://www.raulferrergarcia.com/patrones-de-diseno-en-programacion/. Publication: 2020-02-01.

[41] Redacción APD. Explicación de los principios de las metodologías ágiles. https://www.apd.es/metodologia-scrum-que-es/. Publication date: 2022-01-13.

[42] Sandra Garrido Sotomayor. Que son las metodologías agiles. https://www.iebschool.com/blog/que-son-metodologias-agiles-agile-scrum/. Publication date: 2021-12-09.

[43] Spamhaus Malware Team. Estadísticas de mercado de las botnets en q4 2021. https://www.spamhaus.org/news/article/817/spamhaus-botnet-threat-update-q4-2021. Publication date: 2022-01-20.

[44] Team Asana. Matriz de riesgos de proyecto de asana. https://asana.com/es/resources/risk-matrix-template. Publication date: 2022-03-16.

[45] TechTarget Contributor. Más sobre kotlin. https://www.techtarget.com/whatis/definition/Kotlin. Publication date: Unknown.

[46] TechVidvan Target Contributor. Ventajas y desventajas de python. https://techvidvan.com/tutorials/python-advantages-and-disadvantages/. Publication date: Unknown.

[47] Trellix. ¿que es mitre att&ck? https://www.trellix.com/en-us/security-awareness/cybersecurity/what-is-mitre-attack-framework.html. Publication date: Unknown.

[48] Trello. ¿que es trello? https://trello.com/es/tour. Publication date: Unknown.

[49] uc3m. ¿que es overleaf? https://www.uc3m.es/sdic/servicios/overleaf. Publication date: Unknown.

[50] Universidad de Valladolid. Proyecto docente del trabajo de fin de grado 2019-2020 (Mención Ingeniería de Software). https://albergueweb1.uva.es/guia_docente/uploads/2021/545/46977/1/Documento.pdf.

[51] Wikipedia. ¿que es la implementación de un sistema? https://es.wikipedia.org/wiki/Implementaci\unhbox\voidb@x\bgroup\let\unhbox\voidb@x\setbox\@tempboxa\hbox{o\global\mathchardef\accent@spacefactor\spacefactor}\let\begingroup\let\typeout\protect\begingroup\def\MessageBreak{Ω(Font)}\let\protect\immediate\write\m@ne{LaTeXFontInfo:\def{}oninputline327.}\endgroup\endgroup\relax\let\ignorespaces\relax\accent19o\egroup\spacefactor\accent@spacefactor\futurelet\@let@token\protect\penalty\@M\hskip\z@skipn. Publication date: Unknown.

[52] Wikipedia. ¿que es latex? https://es.wikipedia.org/wiki/LaTeX. Publication date: Unknown.

[53] Wikipedia. ¿que es visual studio code? https://es.wikipedia.org/wiki/Visual_Studio_Code. Publication date: Unknown. Accessed: 2022-05-18.

[54] Xabier E. Barandiaran. Ventajas de latex. https://sindominio.net/xabier/old/curso_latex/clase1.html. Publication date: Unknown.

[55] Zen Chan. Entendiendo pegasus, como rastrear lo inrastreable. https://medium.com/technology-hits/takeaways-from-the-mighty-pegasus-the-nso-group-spyware-524e6fc3a698. Publication date: 2021-07-21.

Apéndice A

Manuales

En este apéndice se ofrecerá a los lectores una guía rápida de instalación y uso del software en formato tutorial acompañado con imágenes.

A.1. Manual de despliegue e instalación

Para poder instalar la aplicación móvil, es decir, el bot, es necesario conseguir la misma en formato .apk (Esto puede hacerse desde el apéndice B). Una vez la hayamos descargado debemos buscar en que carpeta se ha almacenado, en mi caso (Figura A.1) en la carpeta Downloads dentro del almacenamiento interno del dispositivo. Pulsamos en el archivo NotasRapidas.apk para comenzar la instalación.

Generalmente en la mayoría de dispositivos móviles, las instalaciones externas a Google play están bloqueadas por motivos de seguridad por lo que tendremos que habilitar la instalación desde orígenes dispositivos tal y como se muestra en la Figura A.1 y A.2

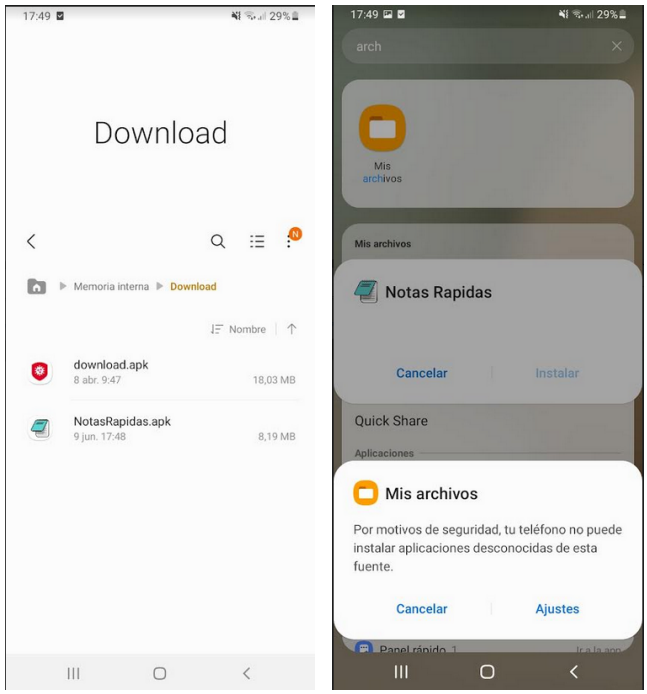

Figura A.1: Búsqueda del archivo .apk y notificación orígenes desconocidos

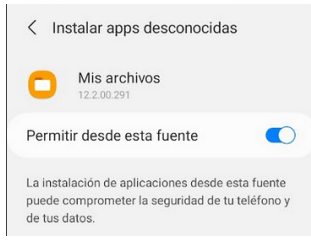

Figura A.2: Permiso Origenes Desconocidos

Por último el flujo de instalación vuelve a la pantalla original pero esta vez, si que se nos permitirá pulsar el botón instalar que comenzará el proceso. Es en este punto en el que si nuestro dispositivo cuenta con un antivirus, puede que se realice un escaneo de seguridad en la aplicación instalada, sin embargo no se detectará nada fuera de lo común y se permitirá el uso normal de la misma

Figura A.3: Pantalla final de instalación

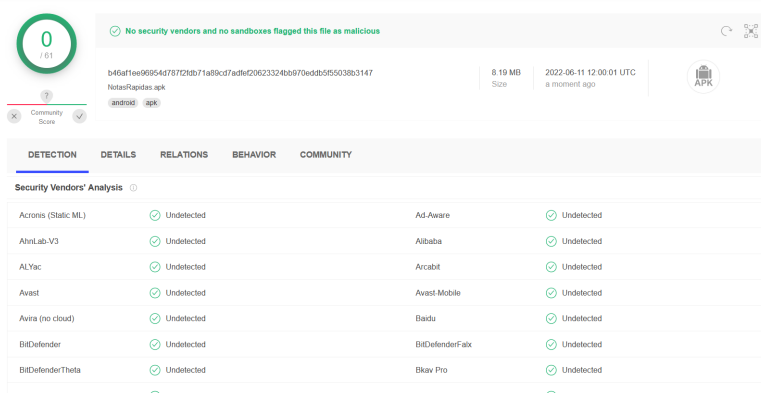

Figura A.4: Archivo .apk analizado por virustotal.com

Por otro lado para instalar el software relativo al BotMaster primero accederemos a su enlace de descarga que se encuentra en el repositorio de GitHub, aquí podemos descargar todo el software como se ve en la Figura A.5 o por otro lado descargar solo la carpeta C2Server (Hay 2 versiones una compatible con windows y otra compatible con Linux)

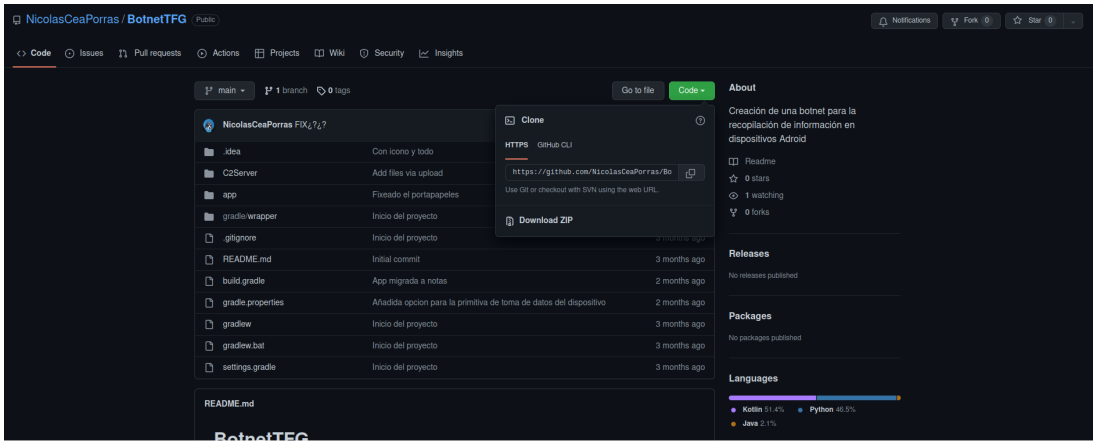

Figura A.5: Descarga del software desde GitHub

Una vez descargado el sofware habrá que extraerlo utilizando el gestor de archivos de kali linux (Basta con arrastrar el archivo fuera del .zip descargado). En la carpeta de C2Server abriremos una consola de comandos para comenzar la ejecución.

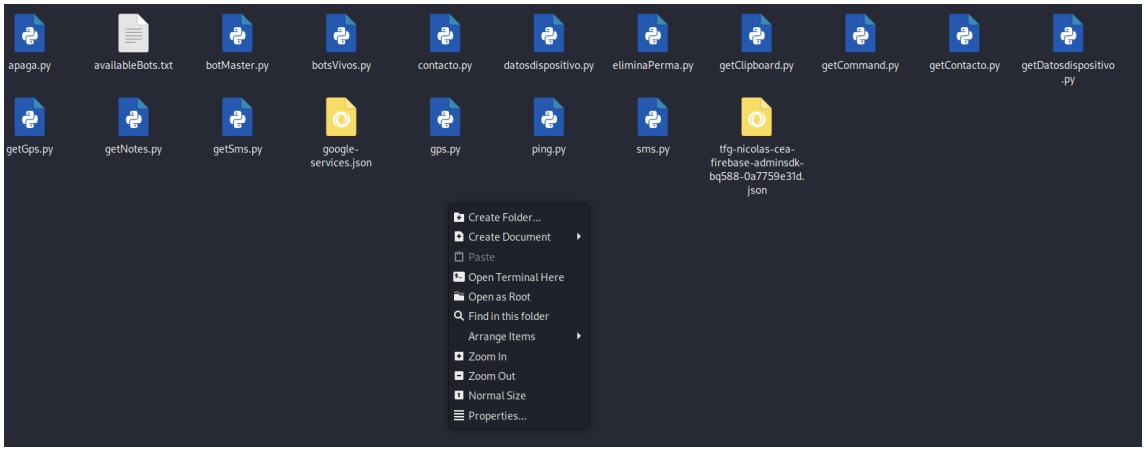

Figura A.6: Apertura de consola de comandos

Será necesaria la instalación de python3 y de un par de dependencias para que todo funcione correctamente. En las Figuras A.7 y A.8 podemos ver los comandos ejecutados para que estas que instalen correctamente en la máquina virtual (Para Windows + cmd o powershell el proceso sería igual).

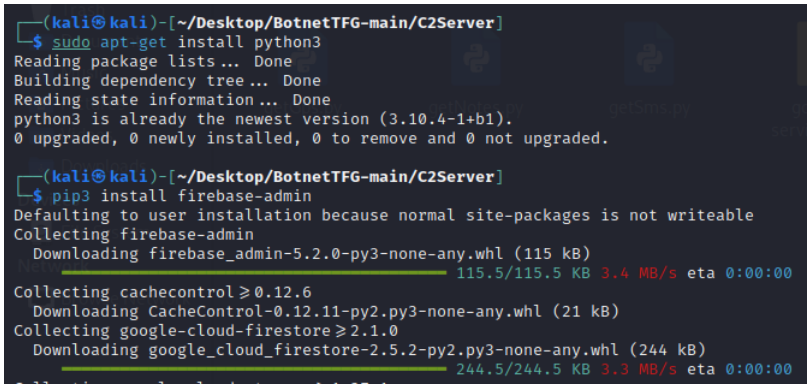

Figura A.7: Instalación de dependencias python3 parte 1

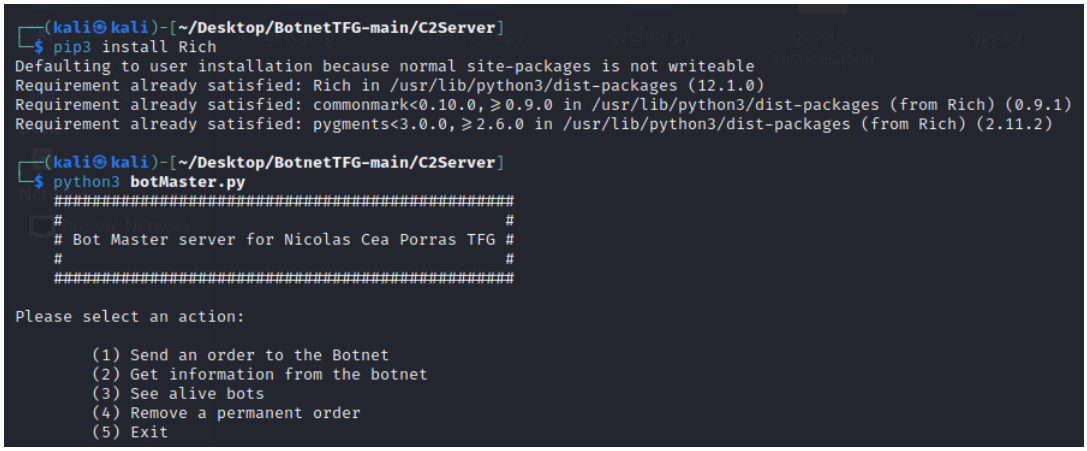

Figura A.8: Instalación de dependencias python3 parte 2

A.2. Manual de usuario

El manual de usuario para la aplicación móvil es muy sencillo ya que lo interesante de esta es el código malicioso que se ejecuta por detrás sin notificar al usuario. Simplemente como podemos ver en la figura A.9 contamos con un calendario, al pulsar alguna de sus fechas se despliega el menú de notas. Si para el día seleccionado hay alguna nota guardada esta se podrá ver en la sección de notas, en caso contrario la aplicación avisará de que no existe ninguna nota almacenada para tal día. Si editamos el texto de la nota podemos elegir almacenarlo pulsando el botón verde, por el contrario pulsando el botón rojo, la nota desaparecerá para siempre.

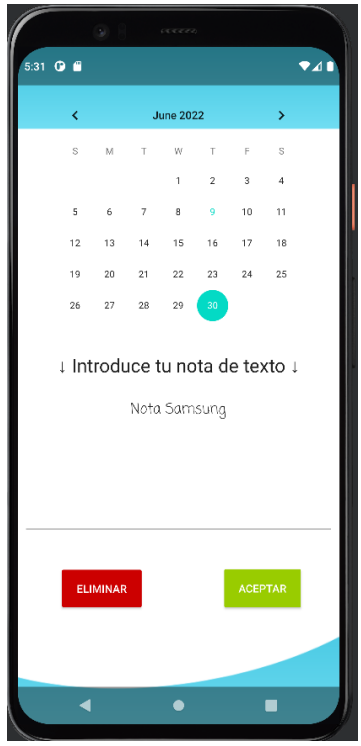

Figura A.9: Única interfaz disponible para el usuario en la aplicación

El menú del Bot Master es algo más complejo, la aplicación se ejecutará utilizando el comando

```
$ python3 botMaster.py
```

Se mostrará la interfaz en forma de menú de consola de comandos, aquí se podrá elegir entre 4 opciones para gestionar la Botnet.

La primera de ellas es enviar una orden, para ello pulsaremos el 1 y se nos informará de todas las ordenes que pueden enviarse. Al elegir una orden pulsando el numero que esta pide, se nos informará de a que Bots puede enviarse está, también se ofrece la posibilidad de elegir todos y de esta manera establecer una orden permanente. Una vez elegido el bot la orden se envía y en el siguiente ciclo de ejecución de código malicioso del bot, esta se ejecuta.

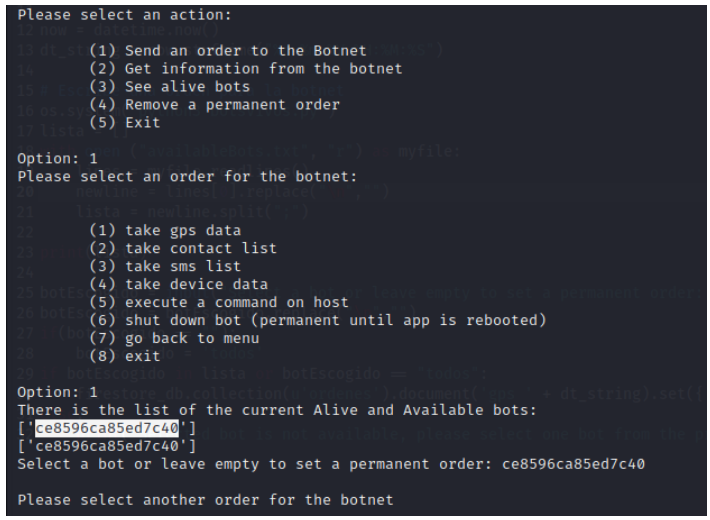

Figura A.10: Envió de la orden GPS a un bot para recuperar sus datos de GPS

Otra de las opciones es comprobar el listado de Bots vivos en la Botnet, para ellos seleccionamos la opción y se nos mostrará la lista de aquellos a los que podremos enviar órdenes. En el caso de Windows, se utilizará una barra de progreso para indicar la velocidad a la que se está realizando la búsqueda de bots.

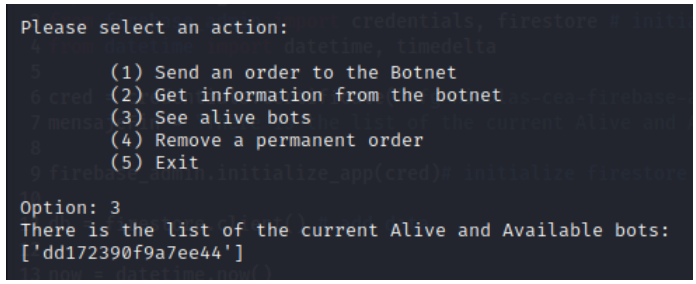

Figura A.11: Listado de bots vivos

Para leer datos recopilados, utilizaré el ejemplo de los datos GPS, simplemente desde el menú principal escogeremos la 2 opción para obtener la lista de datos recopilados que podemos visualizar, después escogemos GPS e introducimos el bot del que queremos obtener la información. Nos pedirá el día y la hora en la cual se tomaron datos y al escoger una de ellas se imprime todo aquello que pudo recopilarse, si las coordenadas de esa fecha son validas, se ofrecerá un link a google maps donde podremos visualizar la calle donde se ubica la víctima. Podemos ver todo esto en las figuras A.12 y A.13

Figura A.12: Lectura de los datos de un bot usando como ejemplo GPS

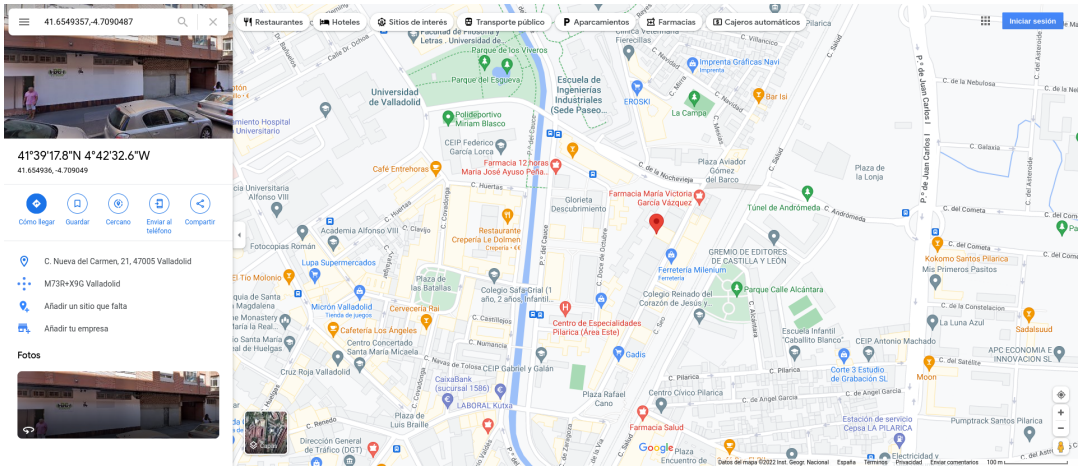

Figura A.13: Mapa desplegado al pulsar el link

Por último podemos eliminar las ordenes permanentes que habíamos enviado a la botnet, estas se ejecutan de forma continuada en todos los bots hasta que se eliminan, simplemente seleccionaremos la 4 opción del menú y la orden a eliminar.

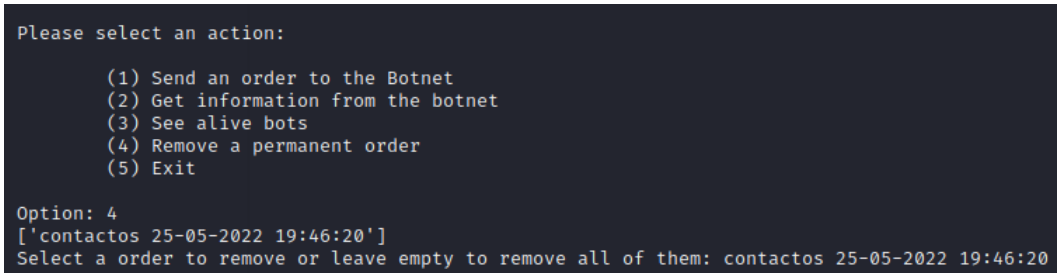

Figura A.14: Eliminar las ordenes permanentes fijadas en la base de datos

Apéndice B

Resumen de enlaces adicionales

Los enlaces útiles de interés en este Trabajo Fin de Grado son:

- Repositorio del código y APK: `https://github.com/NicolasCeaPorras/BotnetTFG`.
- Tablero Trello para el Scrum Board: `https://trello.com/b/cIultUte/scrumboard`.