UNIVERSIDAD AUTONOMA DE MADRID

ESCUELA POLITECNICA SUPERIOR

Grado en ingeniería informática

TRABAJO FIN DE GRADO

Ransomware: análisis y contramedidas

Jorge Antonio Ruiz Muñoz
Tutor: David Arroyo Guardeño

Junio 2017

Ransomware: análisis y contramedidas

AUTOR: Jorge Antonio Ruiz Muñoz
TUTOR: David Arroyo Guardeño

Dpto. Ingeniería Informática
Escuela Politécnica Superior
Universidad Autónoma de Madrid

Junio de 2017

Resumen (castellano)

La proliferación de los ataques *ransomware* alrededor del planeta y el considerable aumento del alcance de los mismos, en especial aquellos que se basan en el cifrado de archivos, supone una de las amenazas digitales más graves en la actualidad. El número de personas y organizaciones afectadas y la cantidad de dinero perdido relativa está aumentando considerablemente. Por ello se torna necesario la utilización de herramientas para neutralizar estas amenazas.

Debido a esta necesidad nace el planteamiento de realizar un estudio y un análisis del funcionamiento general del *ransomware* y de las posibles formas de contrarrestarlo. Este estudio ha comprendido la historia del *ransomware*, desde su primera aparición en 1989 hasta su estado actual. También se ha analizado su modo de actuación, poniendo especial énfasis en las soluciones potenciales. El estudio del *ransomware* concluye con el análisis de su impacto en la sociedad actual.

Este análisis ha derivado en la creación de un sistema capaz de hacer frente a las amenazas anteriormente citadas. El sistema, que se implementará en Linux, se encargará de dotar al usuario de las herramientas necesarias para protegerse eficazmente contra los efectos de los *ransomware*. Dicho sistema consta de dos componentes:

Un módulo para la generación de *backups* de forma segura, consiguiendo que el usuario pueda salvaguardar sus ficheros. Además, también controlará que el usuario realice *backups* de forma periódica.

El segundo módulo consiste en un monitor que podrá escanear el sistema del usuario, analizando en tiempo real los programas que se están ejecutando, lo que habilitará la detección de programas con comportamiento similar al del *ransomware*. De modo más específico, esta metodología de detección de *ransomware* identifica cambios en ficheros mediante *fuzzy hashing* y dictamina si esos ficheros han sido cifrados evaluando la entropía de los mismos. La combinación del *fuzzy hashing* y del cálculo de entropía permite definir una métrica en base a la cual se concluye si hay un ataque de ransomware. En caso afirmativo, se elimina el proceso que cifró los ficheros y se deshabilita la conexión de red. De esta forma se impide la propagación del malware y se habilita la recuperación del sistema mediante el *backup*.

Finalmente, el presente proyecto sigue la metodología *open source* de desarrollo con objeto de proporcionar una solución frente al *ransomware* y, al mismo tiempo, facilitar la evaluación de la herramienta y su ulterior mejora.

Palabras clave (castellano)

ransomware, criptografía, criptovirología, copia de seguridad, backup, cifrado, entropía, fuzzy hashing.

Abstract (English)

The *ransomware* attacks that spread around the world and the significant rise of their scope, especially those based on file encryption, are meant to be one of the most serious digital threats we have nowadays. The number of people and organizations affected and the amount of money loss are increasing substantially. Hence, it becomes necessary to use in order to counter these threats.

Due to this necessity, an approach to perform a study and an analysis of its general behavior and the possible countermeasures, to be protected against it effects, is created. Starting from its first emergence in 1989, to its present state, as well as it mode of operation, focusing on the potential solutions and to conclude with the effect of ransomware in present society.

This analysis has resulted in the building of a system capable of facing the abovementioned threats. The system, which will be deployed in Linux, will be responsible for the equipment of the user with the necessary tools in order to protect himself against the ransomware effects. To achieve so, it will be divided into two parts.

The first will be the backup safety generator, allowing the user to save his files. Furthermore, it will also control that the user makes these backups regularly.

The second, consisting of a monitor, will be able to scan the user´s system in real time and analyze the programmes which are running to finally detect those whose behavior is close to the ransomware´s. More specifically, this ransomware detection methodology recognizes file changes using fuzzy hashing and decide if that files have been ciphered evaluating their entropy. The combination of the entropy measure and the fuzzy hashing enables the creation of a metric to decide if the system is under a ransomware attack. If so, the cipher process will be removed safety, disabling the network. In this way, the system prevents the ransomware spread from continuing and it also allows the system to be recovered using the backup.

Finally, this project has been conducted according to the specifics of the open source methodology in order to foster protection means against ransomware and to enable the improvement of the provided tool by the open source community.

Keywords (inglés)

- ransomware, cryptography, cryptovirology, backup, encryption, entropy, fuzzy hashing

Agradecimientos

Primero, quiero agradecer a mis padres, a mis abuelos y a toda mi familia en general, por todo el apoyo brindado, sin el que no hubiera sido posible llegar hasta aquí.

A Luis, Alberto, Pablo y Jose. Desde que tuvimos la suerte de conocernos, hemos permanecido unidos. Juntos hemos superado todos los retos que nos hemos ido encontrando y juntos hemos disfrutado de los instantes más felices. Tampoco puedo olvidarme de todos los maravillosos compañeros que he conocido en la carrera.

Dar las gracias a todos los profesores que me han guiado durante la carrera, haciendo posible que llegue a ser la persona que soy hoy en día. Y en especial, a mi tutor David, por todo lo que me ha enseñado durante la realización de este trabajo y por toda la ayuda recibida.

INDICE DE CONTENIDOS

INDICE DE FIGURAS

INDICE DE TABLAS

1 Introducción

1.1 Motivación

El primer *ransomware* surge en 1989. A partir de 2005 los ransomware comienzan a extorsionar a víctimas por todo el mundo, exigiendo importantes sumas de dinero como rescate. Desde entonces el número de ataques ransomware, víctimas y la complejidad de los mismos ha aumentado a un ritmo vertiginoso.

Esto ha originado una necesidad imperiosa de desarrollar y utilizar herramientas y procedimientos para poder protegerse contra amenazas de este tipo. Que sean capaces de detener estos ataques, o al menos, reducir el impacto al máximo posible.

De ahí la existencia de un gran interés en intentar entender cómo actúan realmente los *ransomware*, y sobre todo las posibles contramedidas que puedan existir para impedir o paliar sus efectos. En este sentido, el primer paso está orientado a estimar el impacto real que tiene el *ransomware* en la sociedad de forma cuantitativa y cualitativa.

En cuanto al ámbito personal, hay que destacar el interés por desarrollar conocimientos en el campo de la seguridad informática, campo que no ha tenido una gran presencia a lo largo de la carrera. De este modo, este proyecto se presenta como una oportunidad perfecta para adquirir esos conocimientos.

Este TFG también ha permitido consolidar las habilidades de administración y gestión de sistemas operativos Linux. Más en concreto, se ha profundizado en el manejo de la terminal mediante comandos *(shell scripting)*.

1.2 Objetivos

EL principal objetivo de este TFG ha sido estudiar y analizar el comportamiento de los *ransomware* existentes, para intentar comprender su funcionamiento, modo de propagación y posibles deficiencias. Además, se ha estudiado su impacto y las motivaciones detrás de estos ataques.

A partir de este estudio se ha planteado el diseño y la implementación de una solución para Linux que permita la monitorización en tiempo real del sistema, con el objetivo de poder detectar y eliminar *ransomwares* intentando reducir al mínimo su impacto. También se tiene como reto el proporcionar una herramienta capaz de realizar *backups* periódicamente para permitir a las victimas recuperar los datos en caso de sufrir un ataque.

1.3 Organización de la memoria

Se ha considerado dividir el presente documento en seis capítulos.

El primero, realiza una introducción del proyecto y presenta los principales objetivos del mismo, remarcando la motivación general y personal que ha llevado a su realización .

El segundo capítulo, se centra en el estudio y análisis del *ransomware* realizado empezando desde un punto de vista general, hasta estudiar las contramedidas y analizar su impacto.

El tercer capítulo, describe los aspectos fundamentales del diseño y la arquitectura propias del sistema presentado, justificando las elecciones de tecnología.

El cuarto capítulo, tiene como objetivo detallar los aspectos de la implementación y el desarrollo presentados en el capítulo anterior.

En el quinto capítulo, se presentarán las pruebas realizadas para probar la funcionalidad y del sistema presentado.

En el capítulo sexto, se realizará la conclusión del presente trabajo, incluyendo los posibles trabajos futuros como mejoras del mismo.

2 Estado del arte

2.1 Ransomware

El *ransomware* es un tipo de *malware* cuyo objetivo es privar al usuario del acceso a un tipo de recurso de su sistema, comúnmente a sus archivos. Esto se consigue encriptándolos (*crypto ransomware*). O el impedir el acceso al del sistema al completo mediante un bloqueo de la sesión (*locker ransomware*). Es muy común que para eliminar este bloqueo o desencriptar los archivos afectados se pida a la víctima pagar un rescate (*ransom)* [1], en forma de dinero, normalmente se suelen utilizar *criptomonedas* para evitar el rastreo.

El presente proyecto se ha centrado en el estudio de los *ransomware* de tipo *crypto*, los cuales se centran en cifrar la información del usuario. Esta decisión ha sido tomada teniendo en cuenta que los ransomware de este tipo han predominado en los últimos años [2], como queda plasmado en las Figuras 1 y 2.

En la Figura 1 podemos ver cómo destaca el porcentaje de ransomware a partir de 2013, siendo mayoritarios a partir de 2014.

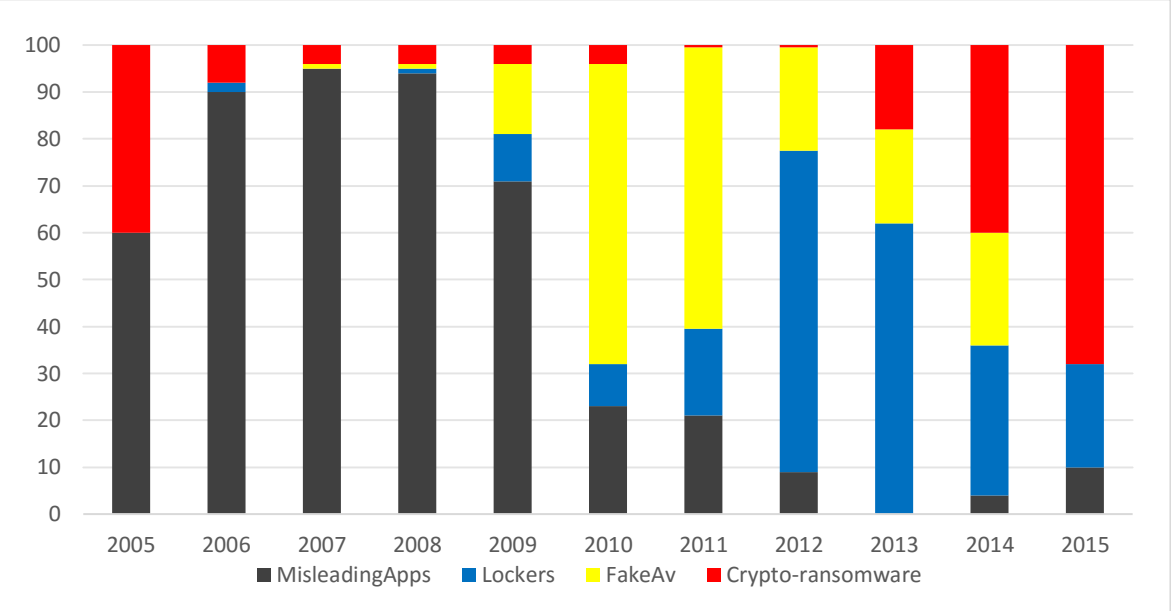

Figura 1. Porcentaje de ataque ransomware según tipo y año en el período 2005-2015. Imagen extraída de [3]**.**

En Figura 2 podemos ver una comparativa del número de ataques ransomware de tipo *locker*, frente a los de tipo *crypto*. Se puede apreciar que desde el comienzo de 2015 los de tipo crypto experimentan una tendencia alcista frente los de tipo *locker* que adoptan la posición contraria, siendo estos últimos superados en la segunda mitad del 2015.

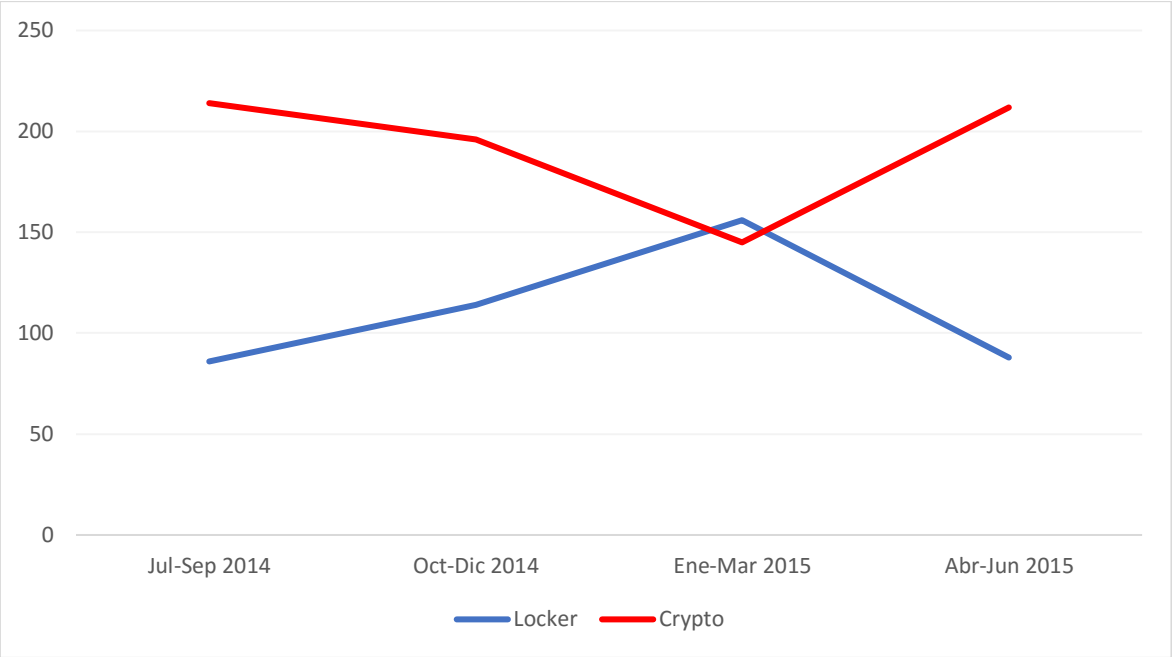

Figura 2. Número de ataques ransomware por tipo en el periodo Jul 2014-Jun 2015. Imagen extraída de [4]**.**

2.2 Criptografía

El objetivo de este capítulo es presentar, de forma breve y concisa, una visión general de que es la criptografía y las nociones básicas importantes en referencia a los *crypto ransomware*

La criptografía puede considerarse como la ciencia, conjunto de herramientas y procedimientos que permiten asegurar una información, haciéndola ilegible para aquellos que no tengan permisos para acceder a la misma [5]. Esto normalmente se consigue mediante el uso de una o varias claves, que mediante una serie de procedimientos y algoritmos consiguen cambiar el texto original suficientemente, para que no pueda ser legible. En el caso de los *ransomware*, en vez de para proteger unos datos, la criptografía se usa para encriptar los archivos de la víctima y vetarla el acceso a los mismos [5].

2.2.1 Criptografía de clave simétrica

La criptografía de clave simétrica [6], se basa en la utilización de una clave para cifrar (convertir el testo plano en cifrado) y para descifrar (convertir el texto cifrado en plano). Por lo que ambas partes, quien ha cifrado el texto y quien quiera descifrarlo, deben conocer la clave que ha sido utilizada. Lo que supone una restricción y un problema añadido, debido a la necesidad de planificar una política de gestión y distribución de claves.

La referencia en los algoritmos de criptografía simétrica es AES [7]. El cual consiste en una serie de rondas, en las que se realizan sustituciones y permutaciones en las que se va utilizando como entrada unas subclaves derivadas de la clave original con base matemática. Una de las propiedades más atractivas del algoritmo AES es que un archivo cifrado un número de rondas con una secuencia de subclaves se puede descifrar aplicando las mismas rondas, utilizando las claves en el sentido inverso. Los tamaños de clave pueden variar entre 128, 192 y 256 bits, haciendo necesario el cambiar la clave periódicamente para evitar ataques por fuerza bruta o búsqueda exhaustiva [7].

2.2.2 Criptografía de clave pública

La criptografía de clave pública [6] supera la debilidad de la comunicación de la clave entre las partes, propia del cifrado simétrico. En este caso, la criptografía pública se basa en la existencia de pares de claves, una pública y otra privada. La información cifrada por la clave pública, podrá ser descifrada por la clave privada, sin necesidad de una comunicación entre partes. Además el conocimiento de la clave pública, no compromete al sistema. El algoritmo referencia de clave pública es RSA, cuya seguridad se basa en la dificultad de factorizar números grandes .

Como podemos ver en el siguiente gráfico, basado en el estudio llevado a cabo por el Global Journal of Computer Sciente and Technology Network [8], los algoritmos de clave simétrica, en este caso AES, encriptan mucho más rápido que los de clave pública, RSA, además de consumir menos capacidad de procesamiento.

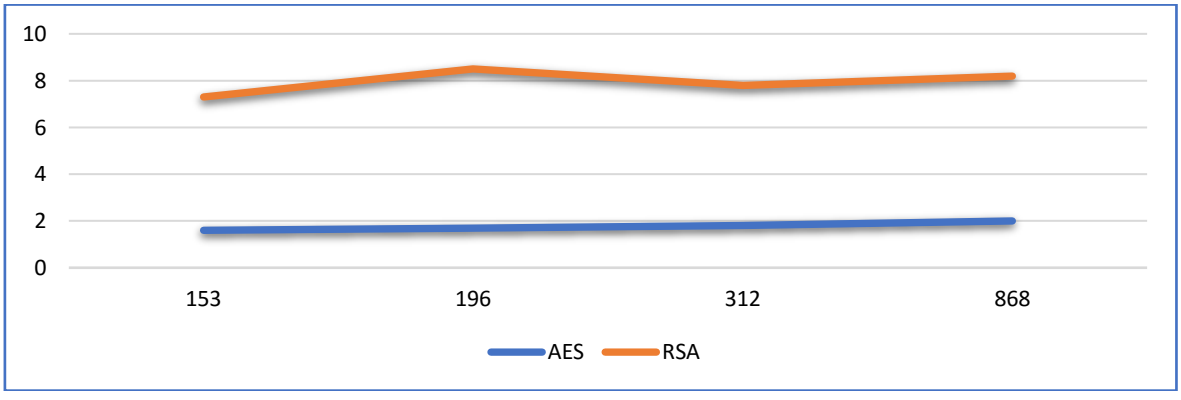

Figura 3. Tiempo de encriptación. Imagen extraída de [8]**.**

De acuerdo a lo anterior, es fácil entender que los creadores de *ransomware* prefieran utilizar algoritmos de clave simétrica, para lograr cifrar los archivos en el menor tiempo posible. A pesar de esto, las ventajas que otorga la clave pública, en cuanto a la distribución y gestión de la contraseña no pueden ser despreciadas. La aproximación más común, es la de cifrar los archivos de la víctima con un algoritmo de clave simétrica y la encriptación de esas claves simétricas mediante un algoritmo de clave pública, así se consigue aunar las ventajas de ambos métodos [3][8].

2.3 Historia del ransomware

El nacimiento del *ransomware* se remite a 1989 [10], cuando el doctor Joseph Popp, experto en biología, crea el *malware* AIDS (Aids info disk). Este *malware* fue distribuido en forma de 20,000 disquetes infectados en una conferencia de la Organización Mundial de la Salud [11].

Este *malware* es un tipo de troyano cuyo funcionamiento es el siguiente. Primero remplaza el fichero AUTOEXEC.BAT, que es usado por AIDS para contabilizar el número de reinicios del sistema. Una vez se alcanza la cifra de 90 reinicios, AIDS oculta directorios y encripta los nombres de todos los ficheros contenidos por la unidad C. Para esta encriptación, se utilizaba criptografía simétrica. Fue posible la recuperación de una gran parte de los archivos afectados, utilizando herramientas de la época.

Finalmente se pide al usuario un pago, que se presenta como una renovación de una licencia, en este caso fue de 189$, una cifra más que respetable para la época. Pese a tener un funcionamiento muy primitivo, AIDS tenía todos los elementos característicos para sentar las bases de lo que posteriormente sería conocido como *ransomware*.

Para encontrar el siguiente paso importante en la historia del *ransomware*, debemos remitirnos a 2006, donde el *malware* Archiveus, otro tipo de troyano, fue publicado. Su funcionamiento era muy similar al de AIDS, encriptando archivos del usuario. Este *malware* se focalizaba en los sistemas Windows, que eran los más utilizados en la época y en concreto en la carpeta My documents. Previsiblemente esta carpeta contenía los archivos más sensibles para las víctimas. Al igual que su predecesor, una vez los archivos eran encriptados, se pedía un rescate a cambio de desbloquearlos. Esta vez mediante una serie de compras en una farmacia online. La gran evolución de Archiveus fue que, además de ser distribuido a través de internet, lo que hacía que su propagación y el número de víctimas potenciales fuera mucho mayor, incorporaba encriptación asimétrica, específicamente RSA, para la distribución de las claves simétricas, que se utilizaban para cifrar los archivos [12].

A partir de 2007, comienzan a aparecer los primeros ransomware de tipo *locker*, cuyo funcionamiento es el impedimento al usuario del acceso al sistema. Uno de los más famosos en esta época fue WinLock, que bloqueaba el sistema con imágenes pornográficas y requería del envío de un SMS con un coste de 10$ para volver a permitir el acceso al usuario.
Durante los años posteriores, los *ransomware* se sofistican y se vuelven más complejos debido fundamentalmente a los propios avances en criptografía.

Posteriormente los *ransomware* siguen expandiéndose y evolucionando hasta que en 2011, aparece un tipo de troyano, sin nombre, que imita la notificación de renovación de producto de Windows. Además incorpora una opción online para renovarla y solucionar el problema, opción que realmente no realizaba esta tarea. Al final, se solicita al usuario realizar una llamada para solucionar el problema. Esta era una llamada internacional que repercutía en grandes costes para la víctima. También comienzan a ser utilizados los métodos de pago online anónimos, que facilitan la recaudación del dinero por parte de los *hackers*.

En 2012 el crimen en internet alcanza otro nivel, más en concreto con Citadel, un conjunto de herramientas y redes de *bots* para distribuir *malware*, que alcanza un gran número de sistemas (más de 100.000 en el primer cuarto de 2012).

Citadel permite mediante el pago de una pequeña cantidad, que los cibercriminales instalen cualquier tipo de *malware* en los sistemas de las víctimas. Aprovechando esta situación surge Reveton, que se basa en el funcionamiento de un *malware* denominado Lyposit y utiliza los servicios de Citadel para infectar a un gran número de víctimas. Su funcionamiento es simple: se bloquea el sistema y se notifica al usuario de que ha realizado algún tipo de comportamiento ilegal, afirmando ser un organismo oficial encargado de cumplir las leyes. El crimen en concreto y el organismo dependían de la ubicación del usuario. En algunos casos se llegaba a emitir contenido de la webcam de la víctima para simular que está siendo grabada. Posteriormente se requería al usuario pagar una multa correspondiente al delito, mediante algunos de los sistemas de pago más utilizados de la época, como Ukash, Paysafe o MoneyPak.

Reveton sienta un precedente y posteriormente nacen y se desarrollan muchos *ransomware* que al igual que Reveton simulan ser algún tipo de organismo oficial. El organismo más comúnmente imitado es la policía, también surgen varios que suplantan la identidad del FBI. El propio Reveton evoluciona en varias versiones y vertientes.

En 2013 surge uno de los *ransomware* más conocidos, Criptolocker. El cual Cambió la forma de expandirse, intentando infectar a las víctimas mediante descargas en webs controladas y mediante el envío de mails a profesionales de ciertos sectores haciéndose pasar por quejas de clientes.

Criptolocker utilizaba una encriptación AES-256 para cifrar, como en los casos anteriores, centrándose en ciertos ficheros del usuario. Esta vez incorporando la novedad de actuar sobre archivos con cierta extensión. Posteriormente encriptaba con un algoritmo RSA de 2048 bits, cuya clave era generada por un servidor *command and control*. Finalmente, al igual que en los casos anteriores, se pedía un pago a los usuarios afectados, a cambio de recuperar los ficheros encriptados, esta vez el método de pago más utilizado fueron los Bitcoins.

Posteriormente se desarrollarían nuevas versiones de Criptolocker como la 2.0, la cual se centraría también en encriptar archivos de imagen, música y video, los Bitcoins eran el único método de pago aceptado.

A comienzos de 2014 aparece CriptoDefense, utiliza la red Tor y de nuevo los Bitcoins como método de pago y RSA-2048 como método de encriptación. La novedad fue que usaba la API de Windows para encriptar, lo que hacía que el *software* tuviera que mantener la clave como texto plano en el sistema afectado.

Más tarde, los creadores de CriptoDefense, lanzaron una nueva versión denominada CriptoWall, esta vez la clave se guardaba inaccesible para el usuario. Entre marzo y agosto se estima que más de 600.000 sistemas y 5 mil millones de archivos fueron encriptados por Criptowall.

Durante 2014 siguen apareciendo nuevas variantes de *ransomware*, como Koler, un tipo de troyano que afectaba a usuarios Android o Cryptoblocker, centrado solo en pequeños archivos.

En 2015 los *ransomware* pasan al siguiente nivel gracias a la red Tor, la famosa red de servicios criminales. Pudiendo ser utilizada para propagar un ransomware a cambio de un porcentaje de las ganancias. Bajo esta premisa nace TeslaCrypt. Al igual que alguno de sus predecesores utiliza un algoritmo de clave simétrica para cifrar los archivos (AES-256) y uno de clave pública para encriptar esta clave simétrica (RSA-4096). TeslaCrypt incorporaba una gran capacidad para mantenerse en los ordenadores infectados.

En los siguientes años, hasta la actualidad, se han sucedido nuevos ransomware, cada vez más sofisticados y especializados en servicios como Wordpress alcanzado cada vez más usuarios y sistemas. Estos ransomware modernos incorporan variantes de ingeniería social más sofisticadas. El punto de infección más importante son los enlaces en páginas de descarga [13].

Más recientemente podemos destacar a Mamba [14], que fue descubierto en septiembre de 2016 por un grupo de investigadores brasileños. Su principal novedad es que, a diferencia de los objetivos de cifrado que pudieran tener los ransomware citados anteriormente, Mamba encripta todo un disco del sistema afectado. Este tipo de ataques abre un nuevo abanico de posibilidades. Finalmente se requiere que el usuario pague el rescate con Bitcoins,. Este método es el más utilizado por los cibercriminales, en este caso se debía pagar 1 Bitcoin [10].

Uno de los últimos ataques más importantes ha sido el de Wannacry, que fue globalmente conocido el pasado 12 de mayo de 2017. Este día se realizó un ataque masivo a varias organizaciones. Wannacry es un *ransomware* cuya propagación es la propia de un virus de tipo gusano. Al igual que la mayoría de ransomwares se encarga de cifrar los archivos del sistema de la víctima para después pedir un rescate a cambio de recuperarlos, la cuantía de este rescate va desde los 300$ a los 600$ [15]. Actualmente existen softwares capaces de recuperar los archivos afectados por Wannacry, como la herramienta de Sensorstechforum [16].

2.4 Contramedidas y vulnerabilidades

En esta sección se va a hablar de los distintos tipos de medidas existentes, para luchar contra los ransomware y contrarrestar sus efectos. Al igual que en el resto del estudio, nos centraremos en el tipo de *ransomware* cuyo principal efecto es el cifrado de archivos. Diferenciaremos entre medidas de prevención, mitigación y recuperación del sistema, una vez ha sido afectado. También se comparará la efectividad de cada medida.

2.4.1 Prevención

Pueden considerarse cómo prevención, aquellas medidas que se asumen antes de que se produzca el ataque, con el objetivo de evitarlo o de reducir sus efectos.
La prevención es posiblemente la medida más eficaz, puesto que, si se consigue evitar el ataque o se consigue estar preparado para recuperar la totalidad de los archivos afectados de un el ataque no habrá tenido ninguna consecuencia.

Contramedidas generales:

Aquí podemos diferenciar entre las contramedidas generales para cualquier tipo de software malicioso que son igual de recomendables para un *ransomware*:

Mantener actualizado el Sistema Operativo y todos los programas, configurar el *firewall*, instalar un antivirus, restricción del acceso a redes WIFI, utilizar contraseñas seguras, protecciones del correo (antispam, filtros) [17]. Es recomendable poner énfasis, en las medidas de protección en la red y en el correo, puesto que, como se detalla en la sección 2.5 y en especial en la Figura 6, son la principal fuente de infección [18].

Backup:

En esta subsección vamos a centrarnos en la medida de prevención más importante y eficaz a la hora de proteger nuestro sistema contra un ataque *ransomware*. Como hemos explicado anteriormente, el ataque del *ransomware* se centra en encriptar los archivos del usuario para hacerlos inaccesibles. Es lógico pensar, que la forma más efectiva de prepararnos contra un ataque de este tipo, es tener una copia protegida de los datos, que nos permita recuperar la totalidad de los mismos. Esto es exactamente la función que cumplen los *backups* [19].

Un *backup* nos permite realizar una copia de seguridad de todos los archivos que queramos, para posteriormente recuperarlos, esto la convierte, sin duda, en la contramedida más efectiva que existe, ya que además de permitirnos recuperar el 100% de los ficheros afectados (siempre que estén respaldados), supone un coste inferior tanto en consumo como en medios frente a los otros tipos de contramedidas.

Una vez se ha realizado el *backup*, es muy importante guardar los datos de una forma segura, ya que en caso de que sean afectados no podremos recuperarlos de ninguna forma. Por ejemplo, si el *backup* se encontrara en el mismo dispositivo de almacenamiento, unidad o disco que está siendo objetivo del *ransomware*, podría ser encriptado, imposibilitando la recuperación. Se pueden elegir varios lugares para contener la copia de seguridad, como pueden ser dispositivos físicos externos, como un pendrive, cinta magnética, etc. Otra de las posibilidades es usar servicios online externos, ya pueden ser, bases de datos convencionales o servicios en la nube, que nos otorguen disponibilidad total en cualquier momento. Teniendo en cuenta que en este tipo de servicios la seguridad deriva a los proveedores del mismo.

La política de *backup* (cuando se debe hacer, de que archivos, con cuanta frecuencia), es determinante a la hora de defenderse de un ataque. Esto es debido a que este ataque podría producirse existiendo datos críticos que aún no han sido respaldados. A continuación, hablaremos de los distintos tipos de backup basándonos en el primer capítulo del libro de Steven Nelson [20].

Primero, hablaremos de *backup* completo. El cual, consiste, en el respaldo total de todos los datos que queremos preservar, es nuestro punto de partida. Después de este *backup* inicial, se pueden diferenciar dos tipos de *backups* parciales que trabajan sobre él:

El *backup* diferencial, se aplica a un *backup* completo o total y guarda los cambios existentes entre la copia inicial y los datos actuales. No trabaja sobre otros *backups* parciales, por lo que, si hacemos un respaldo diferencial de un *backup* completo y unos días después repetimos el proceso, este último *backup* diferencial contendrá todos los cambios guardados por el primero además de los que se hayan procedido posteriormente.

El *backup* incremental, del mismo modo que el diferencial, guarda los cambios con respecto al *backup* completo. Pero a diferencia del diferencial, si puede trabajar sobre otros *backups* incrementales previos, guardando solo los cambios entre ellos.

Por lo tanto, seguir una estrategia basada en *backups* diferenciales, hará que ocupemos más espacio que en una incremental, puesto que cada uno de ellos guardará más datos. A su vez, esto supondrá más facilidad para restaurar los ficheros, además de ser menos vulnerables, ya que, solo necesitaremos el respaldo completo y el último *backup* diferencial para hacer una restauración completa. Mientras que, con el incremental, necesitaremos todos los *backups* incrementales intermedios y la pérdida de uno de ellos, puede suponer pérdidas derivadas en los *backups* siguientes. Por lo tanto, la estrategia a seguir, dependerá de las necesidades y capacidades que tenga el sistema:

La capacidad para guardar los datos, si estamos hablando de una empresa con miles de dispositivos, es sensato pensar que se prefieran *backups* incrementales, para reducir al mínimo la cantidad de datos guardados. Pero a su vez, es muy importante llevar a cabo test de recuperación para intentar estimar los costes operativos, ya que recuperar los datos puede suponer un coste muy grande, más aun suponiendo *backup* incremental, ya que necesitaremos todas las copias parciales para restaurar los datos, sobre todo si no se tiene definido y probado un plan de restauración, a veces incluso supone un coste superior al del propio rescate que nos piden los criminales, aunque nunca es recomendable ceder al chantaje, ya que pagar no nos asegura que vayamos a recuperar los archivos.

Por eso cada compañía, dependiendo de su tamaño, el tipo de datos con los que trabaje y sus necesidades, deberá elegir la política que más le convenga. Pudiendo utilizar ambos tipos de *backup* parciales en conjunto.

2.4.2 Mitigación

Con mitigación entendemos todas las medidas cuyo objetivo es el de, una vez se ha producido el ataque, detectarlo lo antes posible, para reducir al máximo los daños ocasionados. Para ello lo primero, será detectar cuando un *ransomware* se está ejecutando en el equipo.

Sabiendo que la principal arma de los *ransomware* es la encriptación de archivos, se pueden detectar tres patrones de comportamiento típicos [21]:

- El *ransomware* que abre un fichero, lo sobrescribe encriptando la información y lo cierra.

- El *ransomware* que lee un fichero, copia la información en un segundo archivo y lo encripta y por último elimina el archivo original sin preocuparse por borrarlo de forma segura.

- El *ransomware* que lee un fichero, copia su información a otro fichero, encripta este último fichero y finalmente se asegura de sobrescribir el archivo original para eliminar la información.

Por lo tanto, parece adecuado, en un sistema de monitorización, poner especial cuidado en los procesos que abren varios ficheros, los sobrescriben y borran otros, puesto que es probable que sean *ransomware*. Pero si nos centramos en esos parámetros es muy posible que detectemos otros programas que se encarguen de esas tareas de forma convencional, por ello es necesario refinar el funcionamiento del sistema:

Como primera aproximación se puede medir la entropía de un fichero, entendiendo entropía como la incertidumbre en una información [6]. Puesto que cuando un archivo es cifrado, aumenta considerablemente su entropía. Se puede mejorar el sistema, haciendo que cuando un archivo se esté sobrescribiendo, se comparare su entropía original con la resultante. Si esto diera como resultado un aumento, se puede suponer que el programa ha encriptado el fichero. Esta aproximación nos sirve para descartar programas que simplemente sobrescriban ficheros.

Otra forma de filtrar aún más es controlar los ficheros que se están sobrescribiendo, teniendo en cuenta las preferencias de los *ransomware* [22] de encriptar los archivos personales de los usuarios, que puedan ser más valiosos, se puede considerar centrarse en monitorizar los archivos que tengan extensiones doc, txt, jpeg, pdf etc, o al menos darlos más peso a la hora de detectar un *ransomware*.

También se podrían considerar otros parámetros, como el uso de la CPU y de la memoria, pero sin tener un filtro diferenciador claro, parece ser muy difícil conseguir diferenciar un *ransomware* de un programa corriente.

Una vez detectado el posible *ransomware*, es importante que hasta estar seguro de que ha sido completamente eliminado, tomar medidas de aislamiento del sistema, que impidan que el mismo se propague y envíe o reciba información, para ello se debería:

- Desmontar todas las unidades
- Desconectar todas las interfaces incluyendo las de red para impedir la conexión

En caso de no conseguir abortar el funcionamiento del mismo, apagar el sistema puede ayudar, no a limpiar el malware del sistema, pero sí a evitar que siga cifrando archivos y reducir el impacto.

2.4.3 Recuperación

Consideraremos dentro de este apartado todas las medidas llevadas a cabo con el objetivo de recuperar los archivos dañados, y para los que no existe un *backup* previo. En nuestro caso los archivos dañados son archivos que han sido cifrados por el *ransomware*. Es importante mencionar la existencia de herramientas públicas para descifrar los archivos cifrados por ciertos ransomware. Cabe destacar el proyecto nomoreransom [23], centrado en la lucha contra el *ransomware* y que cuenta con una sección de herramientas de descifrado para *ransomwares* específicos.

Teniendo en cuenta, que el principal efecto del ransomware es la encriptación, la mejor forma de atacarlo, es atacar a la propia encriptación que ha sido utilizada. Como se ha comentado en el apartado 2.2, los algoritmos de encriptación más extendidos en los *ransomware* son AES y RSA. Un ataque de fuerza bruta (probar todas las combinaciones de claves posibles hasta conseguir descifrar el texto) es altamente ineficiente contra este tipo de algoritmos, ya que calcular todas las posibles combinaciones con la capacidad computacional actual es prácticamente imposible [24][5]. Por lo que la única forma posible de revertir los efectos eficientemente es intentar captar la clave o claves utilizadas durante el cifrado:

Recuperación de la contraseña

En caso de que los atacantes quieran dar la posibilidad de que el usuario recupere sus archivos después de haber pagado el rescate, necesitan almacenar la contraseña en algún sitio, ya sea en algún tipo de fichero, en la cabecera de los mismos, o enviándola a través de internet a un servidor propio.

Una de las formas más comunes, es el envío de la misma a o desde un servidor *Command and Control* (C&C). El servidor, muchas veces alojado en la red Tor, es propiedad del atacante y envía o recibe uno o varios mensajes con la clave que se ha generado para el cifrado. Por lo que en determinados casos cualquier herramienta de *sniffing*, como Wireshark, nos permitirá monitorizar el tráfico de red e interceptar dicha clave [25].

En otros casos, se guarda una parte de la contraseña en la cabecera de los archivos encriptados. Por ejemplo en el caso de AES, se puede guardar el vector de inicialización seguido del tamaño del fichero en bytes [26]. En estos casos es mucho más compleja la recuperación de la contraseña completa.

Volcado de la memoria

Todo programa que se ejecute en un sistema tiene que utilizar la memoria del mismo para trabajar. Por lo que un ransomware que este encriptando tendrá necesariamente que alojar la clave de cifrado que esté usando en memoria. Llegados a este punto, un volcado de memoria podría conseguir recuperar la clave mediante técnicas forenses [27]. Es más, es posible que los datos de ejecución sigan en la memoria incluso cuando el programa ha terminado, si no se ha reiniciado el sistema.

Con técnicas de este tipo es incluso factible que fragmentos de los documentos sigan totalmente legibles en la memoria, aun habiendo sido encriptados por un ransomware [28], siendo fácilmente accesibles para su recuperación.

2.5 Impacto económico y social

La intención del siguiente apartado es la de conseguir plasmar la extensión real del ransomware y el impacto que tiene en nuestra sociedad. Para este objetivo se va aportar cifras sobre ataques y otros parámetros, basadas en varios estudios y en informes de compañías. No importa la estadística que observemos acerca del ransomware, todas tienen algo en común, están creciendo a un gran ritmo.

En la Figura 4 podemos comprobar, de acuerdo a un informe realizado por Symantec [29], como ha aumentado el número de familias de *ransomware*, siendo 2015 récord histórico. Según un informe de Trendmicro [30] durante los 5 primeros meses de 2016 se han descubierto 50 nuevas familias de *ransomware*, lo que nos hace suponer que durante este año se volverá a batir el récord.

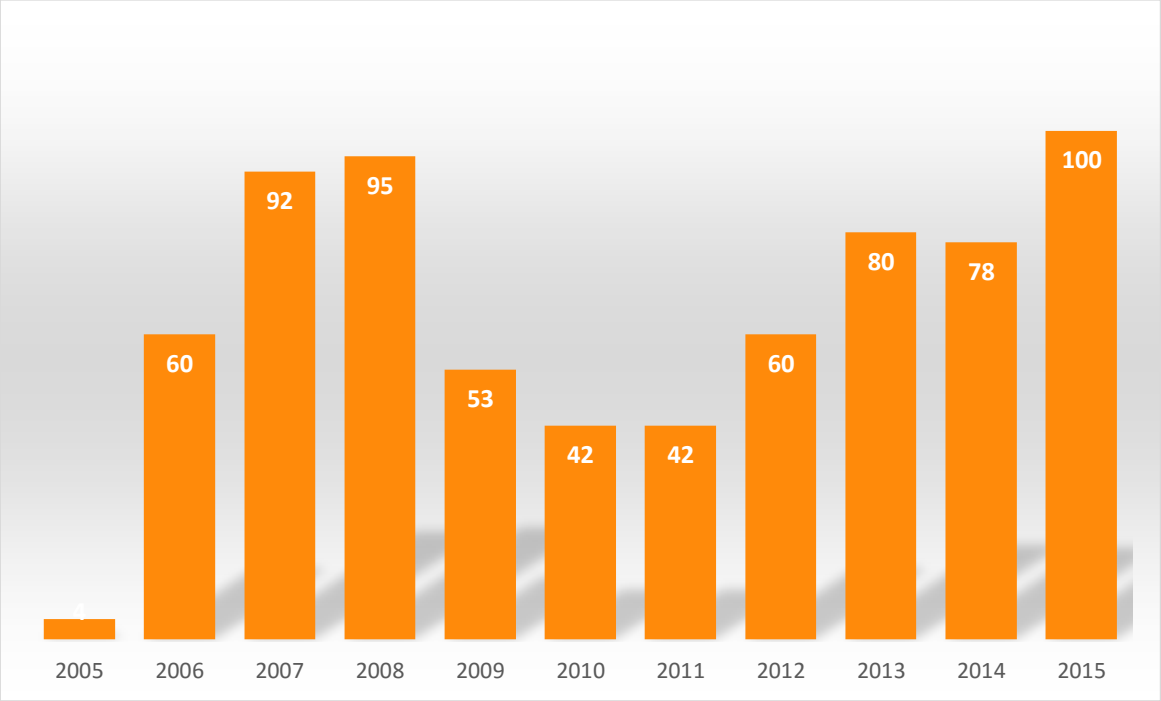

Figura 4. Nuevas familias *ransomware* descubiertas por año en el periodo 2005-2015. Imagen extraída de [29]**.**

El informe de Symantec [29] muestra que el número de infecciones mensuales promedio durante enero de 2015 y abril de 2016 fluctúa entre las 20.000 y las 35.000, cabe destacar un pico cercano a las 55.000 infecciones en marzo de 2016, posiblemente debido a la aparición del *ransomware* Locky.

Tradicionalmente, como se detalla en el apartado 2.3 historia del *ransomware*, el objetivo típico de los ataques, habían sido usuarios normales. En los últimos años está creciendo el % de organizaciones víctimas de *ransomware* frente a usuarios, llegando a ser de un 43% vs un 57% de usuarios durante el periodo comprendido entre enero de 2015 y abril de 2016 [29].

Este fenómeno puede explicarse debido al mayor valor que tienen los datos de las empresas frente a los de los usuarios, en contraposición con que los usuarios suelen tener peores medidas de seguridad en general.

En cuanto a la distribución de los ataques por sector [29], como podemos observar en la Figura 5, a simple vista resalta que ciertos sectores parecen ser mucho más proclives a ser atacadas, no parece haber una razón obvia, puede deberse a que ciertos sectores tienen más dependencia de uso de servicios en internet, por lo que son más propensos a recibir un ataque.

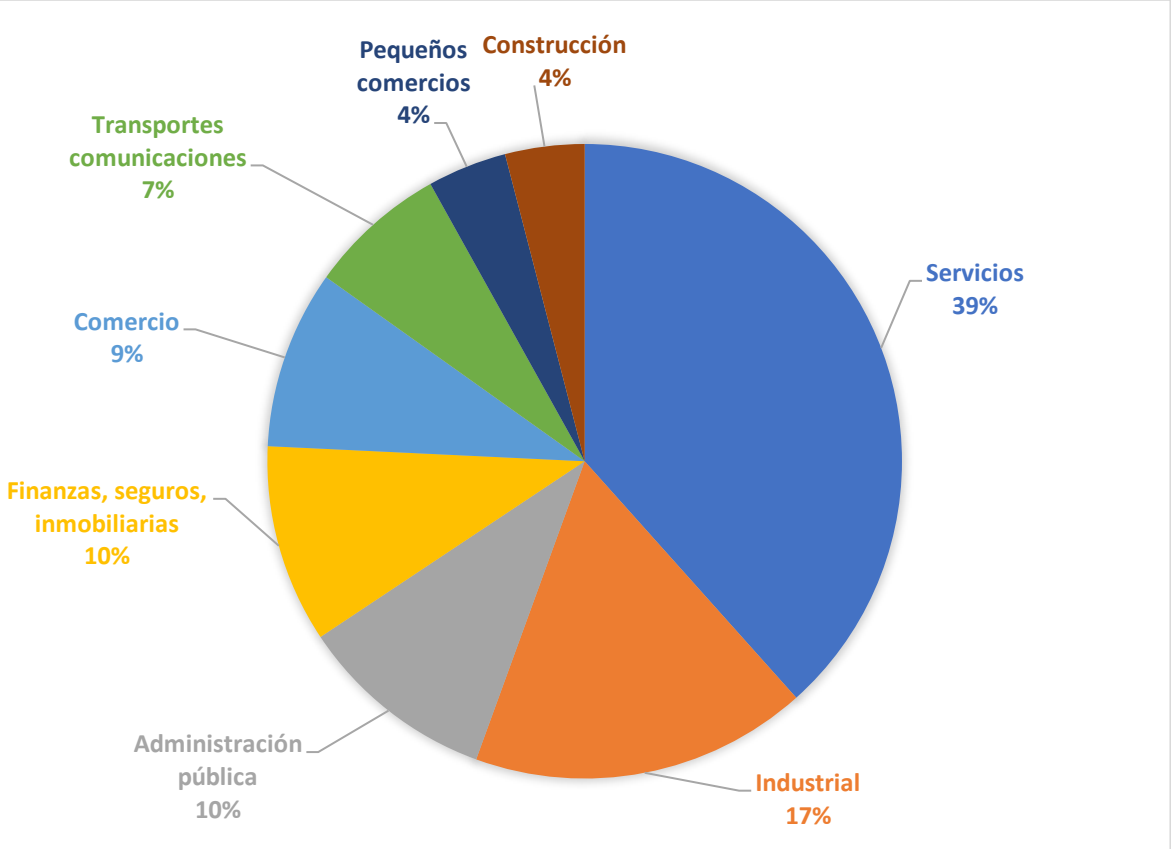

Figura 5. Objetivo de ataque *ransomware* por sector en el periodo de enero 2015-abril 2016. Imagen extraída de [29]**.**

Estimar el número de infecciones por *ransomware* es una tarea muy difícil, puesto que se reciben millones de ataques por todo el mundo. De acuerdo con un informe sobre la distribución de los ataques sobre organizaciones basado en una encuesta en Estados Unidos, Canadá, Alemania y Reino Unido en 2016 [18], casi la mitad de los encuestados (47%) afirmaban que su organización había sido objetivo de un ataque *ransomware* en los últimos 12 meses. Dato que nos revela la extensión del *ransomware* sobre las compañías. Este informe [18], también corrobora los datos revelados por el informe de Symantec [29], afirmando que los sectores más afectados, por orden decreciente, son sanidad (servicios), finanzas, industria y gobierno.

En cuanto al número de ataques, según el informe global de Sonicwall [31], el número de ataques ransomware en 2014 fue de 3.2 millones y en 2015 de 3.8 millones. Estos datos suponen un aumento del 19%, dato limitado en comparación con el increíble incremento durante 2016 hasta alcanzar los 638 millones, 168 veces más ataques que en el año anterior. Esta tendencia alcista parece confirmarse, Beazley [32] afirma que el número de incidencias *ransomware* que ha gestionado en 2016 con respecto a 2015 se ha cuadruplicado y estima que en 2017 este número volverá a doblarse.

Con respecto a las principales fuentes de infección en organizaciones [18], en la Figura 6 podemos comprobar que el correo, tanto con enlaces directos y adjuntos, supone un 59% de los ataques, si también contamos las páginas web, el 83% de los ataques conocidos provienen de internet.

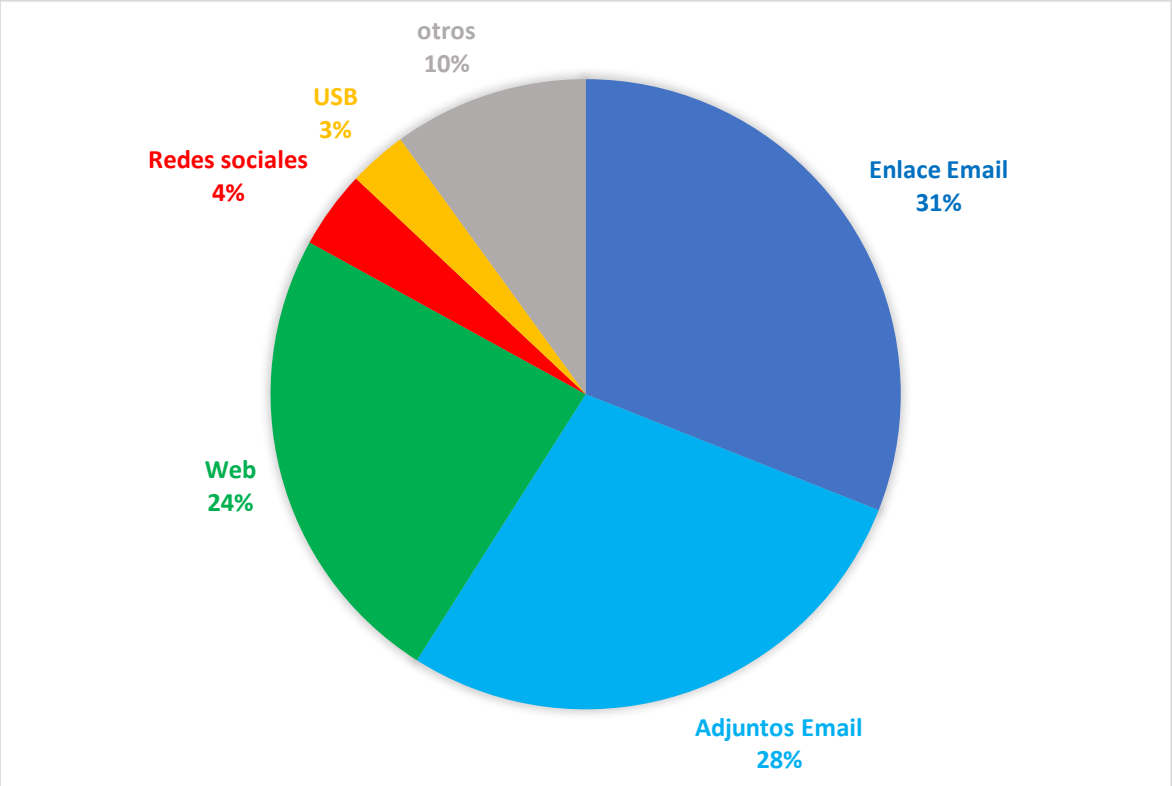

Figura 6. Fuentes de infección *ransomware* en organizaciones. Imagen extraída de [18]**.**

El en apartado económico, parece haber un consenso general por el que se acepta la estimación de que durante 2016, los *ransomware* en las organizaciones provocaron un coste aproximado de mil millones de dólares [33].

Si nos centramos en los rescates exigidos. Se puede apreciar una importante subida en el rescate promedio para recuperar los archivos desde el 2015 al 2016 [29]. Esto se plasma en la Figura 7:

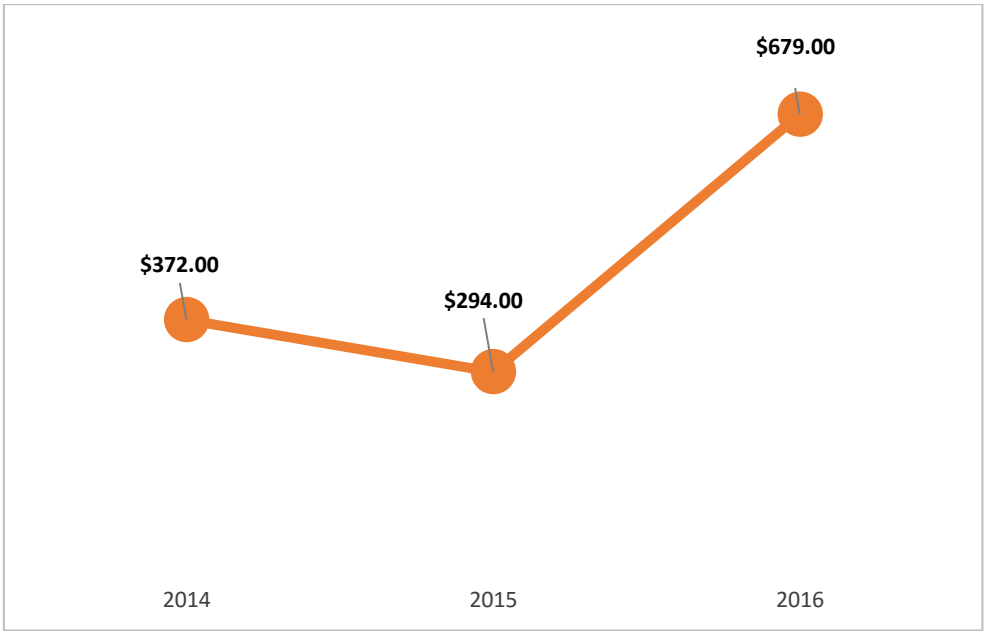

Figura 7. Rescate promedio por año en dólares en el periodo de 2014-2016. Imagen extraída de [29]**.**

Con respecto al impacto en organizaciones es importante destacar, que el rescate, en caso de ser pagado solo supone una parte del coste económico. También hay que considerar, el tiempo en el que la organización no ha podido funcionar correctamente debido al ataque, los costes legales y los costes derivados de la pérdida de los datos. Incluso si los datos se consiguen recuperar mediante el uso de un *backup* como se detalla en la subsección Backup dentro del apartado 2.4.2, esta recuperación tiene unos costes operacionales importantes [29].

También cabe destacar, la importante subida del valor del Bitcoin frente al dólar estadounidense, desde los 84$ por Bitcoin en sus comienzos en 2013, hasta los 2240$ que vale cada Bitcoin en la actualidad, lo que ha supuesto un aumento de aproximadamente 25 veces su valor [34]. Es sensato pensar que esta subida ha influido en la subida de precio de los rescates, ya que se ha comentado en la sección 2.3, los Bitcoin son el método de pago más extendido para abonar los rescates ransomware.

Para acabar con este apartado, se va a analizar la distribución por países que tiene el ransomware [29]. El porcentaje de ataques por países, en el periodo comprendido entre enero de 2015 y abril de 2016 es el siguiente:

Estados Unidos 29%, Canadá 16%, Australia 11%, India 9%, Japón 4%, Italia 4%, Reino Unido 4%, Alemania 2%, Holanda 2%, Malasia 2%.

A simple vista, nos podemos dar cuenta, de que los países objetivos son, países desarrollados, y con alta población. Se puede observar una correlación directa con el PIB, puesto que todos los países anteriormente citados, con excepción de Malasia, en el 2015 estuvieron en el top 17 del ranking mundial de PIB [35]. Es trivial pensar, que los objetivos de ataque de los criminales serán los usuarios u organizaciones, con mayor capacidad económica, puesto que será más probable que cedan al chantaje y terminen pagando, además de que tengan información de mayor valor.

También es importante mencionar, que el 69% de los países atacados son de habla inglesa [29]. Dato que podría explicarse si tenemos en cuenta Figura 6 , en la que se detallaba que la mayor parte de ataques, proviene del correo y la web, por lo que al compartir idioma, podrían ser objetivo de las mismas plantillas de ataques en correos e infecciones por navegación en las mismas páginas web, a diferencia con otros países, en las que se tendría que cambiar el idioma de toda la información con la que interactúa el usuario.

3 Diseño

En este apartado se va a proceder a realizar un análisis del sistema, detallando el catálogo de requisitos, tanto funcionales como no funcionales. Se va describir el diseño general de la aplicación, así como el de cada uno de sus módulos, con todas las dependencias propias, como pueden ser las bases de datos utilizadas.

Se ha decidido, dividir la aplicación en dos módulos distintos, debido a la gran diferencia de funcionalidad entre ambos. El módulo monitor, encargado de escanear el sistema en búsqueda de programas candidatos a ser *ransomware* y el módulo *backup*, encargado de gestionar los backup del usuario.

3.1 Catálogo de requisitos

Se va a proceder a desglosar los requisitos del sistema, primero describiendo los funcionales y después los no funcionales, diferenciando en cada caso entre cada módulo y entre los requisitos comunes a ambos.

3.1.1 Requisitos funcionales

Requisitos funcionales comunes

RF1: La instalación será automática

El sistema tendrá que ser capaz de autoinstalar todas las componentes necesarias, además de realizar los cambios necesarios en el sistema. Únicamente siendo necesario ejecutar un script por parte del usuario. Las bibliotecas necesarias, deberán haber sido instaladas por el usuario con anterioridad. Quedan fuera de este requisito, todos aquellos comandos y directrices que necesiten privilegios de administración, en tales casos se pedirá la contraseña al usuario.

RF2: La ejecución del sistema será autónoma

El usuario solo tendrá que ejecutar distintos ejecutables, sin necesidad de interacción con los mismos. Solo será necesario cambiar cierta información, en ficheros de configuración para alterar la ejecución con respecto a las preferencias del usuario. Quedan fuera de este requisito, todos aquellos comandos y directrices que necesiten privilegios de administración, en tales casos se pedirá la contraseña al usuario.

RF3: El sistema será capaz de iniciar y parar el servicio PostgreSQL

El sistema podrá iniciar y parar el servicio PostgreSQL cuando necesite utilizar el mismo para realizar las tareas pertinentes. Para ello será necesario que el usuario introduzca su contraseña de administrador.

RF4: El sistema será capaz de crear un base de datos

El sistema podrá crear una base de datos PostgreSQL, con los parámetros que crea pertinentes, como el nombre de la misma o el usuario que la ha creado.

RF5: El sistema será capaz de crear un usuario

El sistema podrá crear un usuario en PostgreSQL, con un nombre y una contraseña propios.

RF6: El sistema será capaz de crear una tabla en la base de datos

El sistema podrá crear una tabla en la base de datos PostgreSQL, con los campos, atributos y relaciones necesarias para su correcto funcionamiento.

RF7: El sistema deberá guardar los datos de acceso a la base de datos

El sistema deberá guardar toda la información relevante con respecto a las credenciales de acceso a la base de datos PostgreSQL que está gestionando. Como el nombre de la misma, el nombre del usuario y su contraseña, el nombre de las tablas o cualquier otro tipo de información que necesite para la correcta gestión de la misma.

RF8: Se podrá insertar información en una tabla

El sistema podrá insertar las filas con los campos que necesite, en una tabla perteneciente a una base de datos PostgreSQL.

RF9: Se podrá recuperar información de una tabla

El sistema podrá recuperar la información que necesite, contenida en los campos de una tabla perteneciente a una base de datos PostgreSQL, filtrándola por los campos que necesite.

RF10: Se podrá eliminar una fila de una tabla

El sistema podrá eliminar una fila de una tabla perteneciente a una base de datos PostgreSQL, filtrándola por cualquiera de sus campos.

Requisitos módulo monitor:

RF11: Se podrá obtener la entropía de cada uno de los ficheros

El monitor podrá calcular la entropía, en cualquier momento, de cada uno de los ficheros, obteniendo un valor entre 0 y 1.

RF12: Se podrá obtener un código hash de cada uno de los ficheros

El monitor podrá generar un código hash de cada uno de los ficheros, este código será basado en ssdeep [36].

RF13: Se podrán comparar dos códigos hash

El monitor podrá comparar dos códigos hash, pertenecientes a dos ficheros, para averiguar, como de parecido es cada uno al otro.

RF14: La información se almacenará de forma ordenada

La entropía y el código hash de cada fichero, será almacenada en una base de datos PostgreSQL, asegurando que esté disponible, en cualquier momento, para ser buscada por el nombre del fichero.

RF15: Se supervisará el directorio /home/ del usuario

Al comienzo, el monitor supervisará el directorio /home/ del usuario en búsqueda de programas que estén realizando operaciones de lectura y escritura sobre archivos pertenecientes al mismo.

RF16: Se deberá garantizar que el monitor escanee individualmente los procesos detectados

Cada uno de los procesos que realizan algún tipo de modificación cualquier fichero perteneciente al directorio, será supervisado individualmente, para comprobar su comportamiento.

RF17: Un proceso que sea considerado ransomware será eliminado por el monitor

El monitor, al detectar un programa como posible *ransomware*, lo eliminará de forma segura, desconectando todas las interfaces de red, bluetooth y desmontando todas las unidades externas en el proceso.

RF18: El monitor notificará al usuario cuando se ha eliminado un proceso

El sistema, notificará de un modo no directo, al usuario, esto es, con un archivo utilizado como log, cuando se ha detectada un proceso como posible ransomware y cuando se ha elimina.

Requisitos módulo backup:

RF19: Solo se hará un *backup* en presencia de un dispositivo usb

Como medida de seguridad, el módulo podrá realizar un *backup* del directorio que desee el usuario, siempre y cuando un usb esté conectado al sistema, el *backup* se guardará dentro del propio usb.

RF20: Se detectará cuando un usb esté conectado

El módulo podrá comprobar si hay algún usb conectado al sistema y en caso afirmativo, podrá obtener la ruta del mismo, para poder guardar el *backup* en la misma.

RF21: Los *backups* se realizarán automáticamente al arrancar el sistema

Al iniciar el sistema, el módulo comprobará automáticamente si existe un *backup* de la ruta establecida en un fichero de configuración. En caso de que no exista o en caso de que el *backup* haya sido hecho con una anterioridad superior al tiempo entre *backups* que especifica el usuario en el mismo fichero de configuración, el módulo intentará realizar una copia de seguridad automática.

RF22: Se guardará información relativa a los *backups*

El módulo, después de realizar un *backup*, o actualizar uno, dejará constancia en un log, además de escribir la información asociada, como la fecha, directorio del que se ha hecho el *backup* y ruta donde se ha guardado.

3.1.2 Requisitos no funcionales

RNF1: El sistema será capaz de escanear un solo directorio

En un futuro se planteará la posibilidad de escanear múltiples directorios al mismo tiempo.

RNF2: El sistema será desarrollado para Linux

El sistema será implementado en Linux, en concreto para la distribución Ubuntu y utilizará las bibliotecas y recursos propios del mismo, además de alguna extra que serán instaladas, no garantizando su funcionamiento en otros sistemas operativos o distribuciones.

RNF3: El sistema no utilizará interfaz gráfica

Todo el sistema podrá ser ejecutado mediante la shell de Linux, sin necesidad de que haya una interfaz gráfica o una interacción fuera de la shell por parte del usuario.

RNF4: El código estará comentado

Todas las funciones de los programas tendrán cabeceras para facilitar su entendimiento y mantenimiento.

RNF5: La aplicación está dirigida a un usuario con conocimientos Linux

Se presupone al usuario de la aplicación, conocimientos sobre sistemas Linux, modificación de archivos y funcionamiento de la *shell*.

RNF6: Se proporcionará un manual de usuario

La aplicación dispondrá de un manual de usuario, que explique las líneas generales de funcionamiento y ejecución, la instalación y como comprobar los log de información.

RNF7: El código será abierto y libre

El código se subirá a un repositorio de acceso público, para que pueda ser utilizado, compartido o modificado por cualquier persona interesada.

3.2 Diseño general

En este apartado se va a detallar las decisiones tomadas sobre el diseño y la arquitectura final del sistema, además de detallar cada uno de los módulos propios de la aplicación, los detalles de la implementación elegidos serán revelados en la siguiente sección. El sistema se ha diseñado para Linux, más en concreto Ubuntu, esto se ha decidido, puesto que no existen muchos programas, que incorporen la funcionalidad descrita. Además, prácticamente la totalidad de las herramientas y recursos necesarios son aportados por el propio sistema operativo, por lo que el uso de estos se vuelve intuitivo y eficaz.

En cuanto al lenguaje de programación, se ha utilizado C, tanto por su capacidad para hacer llamadas al sistema y por su buena integración con la terminal, que será utilizada desde el código de C con asiduidad. También, se utilizarán scripts *BASH* de Linux, para facilitar la ejecución de ciertas tareas críticas para el funcionamiento del sistema. Cumpliendo así uno de los principales objetivos del proyecto.

El sistema se ha dividido en dos módulos, ya que su funcionalidad está claramente diferenciada. Primero, el módulo *backup* que se encargará de realizar y gestionar toda la funcionalidad relacionada con los *backups*. Y el módulo monitor, cuya tarea será la de monitorizar los procesos. Primero monitorizará el directorio elegido en búsqueda de procesos sospechosos para más tarde monitorizar a estos mismos. Para finalmente, en caso de detectar un posible *ransomware* eliminarlo. A continuación, se va a describir en detalle el diseño de ambos módulos y sus miembros.

El plan de ejecución ha sido el siguiente: primero se ejecuta un script general de instalación que se encargue de realizar todos los cambios y generar los recursos necesarios, entre otras cosas este script creará la base de datos y generará los archivos de configuración necesarios para el correcto funcionamiento de la aplicación. El diseño de la base de datos será descrito en el apartado 3.2.3. Posteriormente ambos módulos se encontrarán operativos y el usuario podrá elegir cual utilizar.

3.2.1 Diseño del módulo backup

El planteamiento inicial de este módulo, era el de otorgar al usuario la capacidad de realizar un *backup*, de forma manual, del directorio que quisiera, en cualquier momento. Para finalmente salvarlo en el lugar que prefiriese. Al principio, con la instalación del sistema, se crea la base de datos y se guarda en un fichero las credenciales de acceso de la misma, para que posteriormente pueda ser usada por los programas que lo requieran.

La primera mejora que se pensó, fue la de automatizar el proceso, para que los *backups* se realizarán de forma automática, sin necesidad de que el usuario estuviera presente y que pudiera cambiar los parámetros del mismo, como la frecuencia en que se realizan y que directorios se guardan.

Para implementar esta solución, se decidió incorporar parte de la funcionalidad, en este caso el submódulo del sistema que comprueba los *backup* en el arranque de sistema. Para que cada vez que el sistema Linux se iniciara se ejecutara el proceso. Para incorporar esta funcionalidad se barajaron varias posibilidades, como Cron o crear un servicio de arranque. Pero finalmente, se decidió utilizar el fichero /etc/rc.local. Este fichero es ejecutado por el sistema después de cada arranque, una vez se han lanzado todos los demás servicios, por lo que nos proporciona toda la utilidad necesaria. Esto nos asegura que cualquier recurso del sistema que necesitemos estará disponible. Además su uso es muy sencillo, basta con añadir al propio fichero las líneas de código que necesitemos ejecutar. Después de la instalación rc.local, llamará a nuestro submódulo que hará lo siguiente:

Comprobará las preferencias del usuario, en cuanto a la ruta de la que se quiere realizar la copia de seguridad y comprobará en la base de datos si ya hay un *backup* previo de la misma. En caso de no haberlo, el sistema realizará el *backup* y creará la información relativa en la base de datos, después terminará la ejecución.

En caso de haber un *backup* previo, se comprobará la fecha en la que se realizó el mismo con la fecha actual y comparándolo con un umbral que designa el usuario, se decidirá hacer el *backup* o no. En caso de hacerlo se actualizará la información referente en la base de datos.

En todos los casos, en los que se realice el *backup*, se volcará la información referente al mismo, en un log, para que le usuario pueda seguir el historial de ejecuciones.

Después de un exhaustivo estudio del comportamiento de los *ransomware*, se decidió que salvar el *backup* en el mismo sistema de archivos que se quieren proteger, es altamente inseguro, ya que en caso de que el *ransomware* esté encriptando el disco, encriptaría también el *backup*, inutilizándolo. Por eso, se decidió establecer como norma, que los *backups*, solo se pudieran realizar en un usb. Por lo que en caso de que se vaya a realizar un *backup*, el módulo buscará los usb que haya conectados, y guardará el backup en los mismos.

En el anexo A se adjunta un esquema general del funcionamiento del módulo *backup*, para facilitar su entendimiento.

3.2.2 Diseño del módulo monitor

El módulo monitor por defecto se encarga del escaneado del directorio /home/ en búsqueda de procesos que realicen operaciones de entrada/salida. Antes de comenzar a escanear, mientras el script de instalación se ejecuta, se recopila la información referente a todos los archivos del mismo de la siguiente forma:

Recorre todos los archivos y subdirectorios del directorio recursivamente, y de cada uno de los archivos que encuentra, calcula su entropía y genera un código hash propio de cada uno. Este código hash es uno de tipo fuzzy hashing [36]. Esto se ha decidido ya que al comparar dos hashes de este tipo se obtiene un % que representa como de parecidos son los dos ficheros. Esto se realiza en base a lo explicado en el módulo 2.4.2, para posteriormente, cuando un proceso modifique los ficheros, comprobar los cambios de entropía y hash y decidir si se ha encriptado el archivo. Se ha elegido comparar ambos parámetros, porque la entropía detecta correctamente cuando un fichero se cifra y el fuzzy hashing cuando se modifica. Combinar ambos parámetros consigue una aproximación más eficaz, logrando evitar el mayor número de falsos positivos. Toda esta información se guardará en la base de datos.

Cuando el programa ha generado toda la información necesaria, el monitor se instala para ejecutarse cada vez que arranca el sistema. Esto se ha conseguido editando el fichero rc.local, como se detalla en la sección anterior. Se comienza cíclicamente a observar operaciones de escritura y lectura en el directorio. Cuando encuentra un proceso que ha realizado una de estas operaciones, se lanza un hilo con un monitor que se centra específicamente en el proceso. Este cada vez que el proceso realiza un cambio en el fichero, lo busca en la base de datos y en caso de tener la información, vuelve a generar la entropía y el código hash y los compara con los valores originales. En caso de detectar un determinado número de cambios significativos, se cataloga al proceso como *ransomware* candidato y se procede a su eliminación de forma segura.

Para realizar la eliminación de forma segura y contener al posible ransomware, se adoptan las siguientes medidas:

- Desconexión de todas las interfaces de red. Para prevenir posibles comunicaciones con servidores C&C o que se siga propagando la infección (ver apartado 2.4.3 recuperación de la contraseña).
- Desmontar todas las unidades. Para evitar que la infección se extienda y cifre otras unidades.
- Evitar otro tipo de conexiones, como puede ser las bluetooth
- Eliminación del proceso. Eliminar el proceso directamente, para que no se siga ejecutando.

3.2.3 Diseño de la base de datos

Se ha decidido utilizar como gestor de la base de datos PostgreSQL por sus propiedades [37]:

Código abierto, gratuita, facilidad de uso, existencia de una gran documentación referente, robustez, estabilidad y rapidez. Además, existen varias bibliotecas en C, para facilitar el acceso a la misma, lo que facilita mucho el uso de la misma. Teniendo en cuenta que las tablas que se van a usar en este proyecto son muy simples, PostgreSQL es una elección adecuada.

Como se ha comentado en las secciones anteriores, la base de datos será utilizada por los dos módulos, el de *backup* y el monitor, cada uno de ellos tendrá una tabla asociada, para guardar los datos asociados a su ejecución. A continuación, se muestran en forma de tabla los distintos campos de cada tabla con una descripción de cada uno.

En las siguientes tablas, vemos el esquema de la tabla propia de la base de datos que utiliza el módulo *backup*, la búsqueda de los *backups* se realiza mediante el campo Path, en la fecha se registra el día mes y año en la que se ha realizado el backup. El tipo de backup se diferencia entre los distintos tipos de *backups* (ver sección 2.4.1 backup), en esta primera implementación todos los *backups* serán de tipo completo.

Backup		
Tipo	Fecha	Path
Tipo de *backup*. Tipo varchar.	Fecha de realización del *backup* con el formato: año_mes_dia. Tipo varchar.	Ruta absoluta hasta el directorio del que se ha hecho el *backup*. Es la clave primaria. Tipo varchar.

Tabla 1. Esquema de la tabla *backup*

En la siguiente tabla, vemos el esquema de la tabla propia de la base de datos que utiliza el módulo *backup*, el Path o ruta del fichero, será la clave primaria y el campo de búsqueda. La entropía será un float, con un valor entre 1 y 0 y el código hash será una cadena de texto.

Monitor		
Path	Entropía	Hash
Ruta absoluta del archivo. Es la clave primaria. Tipo varchar	Representa el valor de la entropía del archivo. Tipo float	Código hash del archivo. Tipo varchar

Tabla 2. Esquema de la tabla monitor

4 Implementación, desarrollo y pruebas

En este apartado se van a describir los detalles de la implementación con respecto al diseño descrito en la sección anterior. Mencionando las librerías utilizadas, los procedimientos y explicando la elección de los mismos.

4.1 Librerías utilizadas

A continuación, se va mencionar las librerías que utiliza el sistema.

4.1.1 PostgreSQL

El sistema utiliza como gestor de base de datos y para aportar toda la funcionalidad relacionada con la base de datos a PostgreSQL [37].

4.1.2 Libpq

Biblioteca propia de PostgreSQL que permite gestionar una base de datos utilizando el Lenguaje C [38].

4.1.3 Binwalk

Herramienta que aporta herramientas para análisis, ingeniería inversa y la extracción de las imágenes *firmware* [39]. En este proyecto será utilizada para el cálculo de la entropía de los ficheros.

4.1.1 Fuzzy

Biblioteca que permite implementar en C las funciones necesarias para generar el código fuzzy hashing [40] de un archivo [41] .

4.2 Desarrollo

Después de un análisis primerizo, se comprobó que mucha de la funcionalidad necesaria, se podía incluir con la propia *terminal* de Ubuntu. Por ello, se buscó la forma más cómoda de utilizar sus comandos desde el código C.

Se decidió utilizar el comando popen, frente a otras alternativas como system(), ya que permite gestionar la salida del comando, recibiendo información necesaria para el funcionamiento de nuestro sistema.

Para facilitar la compilación de todos los submódulos y archivos .c, se han utilizado Makefiles. También se utilizan scripts se encargan de lanzar estos Makefiles y posteriormente ejecutar los ejecutables correspondientes. Además de los cambios necesarios en el sistema para el correcto funcionamiento del mismo. El script de instalación realizará los siguientes cambios:

Modificar el tipo de conexión, de PostgreSQL, a modo *trust*, para que cualquier usuario perteneciente al mismo, pueda conectarse, sin necesidad de ser usuario del sistema. Para ello se modifica el fichero, /etc/postgresql/*/main/pg_hba.conf, donde * es la versión de PostgreSQL. En el mismo se cambiarán las coincidencias de la palabra “peer” por “trust”.

Instalar la función comprobar backup y la función monitor para que cada vez que se arranque el sistema se ejecuten. Para conseguirlo, como se comenta en la sección de diseño, se modificará el fichero /etc/rc.local. En él se añadirá una llamada a un script que ejecute ambas funciones.

Se moverá toda la carpeta, que contiene al sistema, a la carpeta /etc/ por defecto, siendo posible que el usuario modifique esta localización. Se ha elegido esta localización por la restricción de acceso que tiene, asegurando así una seguridad básica.

Se creará la base de datos y las tablas relacionadas, generando el archivo de configuración con las credenciales para futuros usos.

4.2.1 Base de datos

Para la base de datos, se ha creado un submódulo, denominado pgsqldb, que es utilizado por los módulos monitor y *backup*. Su función es la de gestionar la base de datos. Para ello utilizará las funciones de la biblioteca Libpq o en algunos casos comandos directos de la terminal a través de popen.

El submódulo incorporará la siguiente funcionalidad:

Activar y desactivar el servicio PostgreSQL:

El submódulo podrá activar el servicio de PostgreSQL cuando necesite utilizarlo.

Crear base de datos:

Creación de un base de datos PostgreSQL, con el nombre y propietarios que se quieran usar. Se implementará mediante un comando de terminal.

Crear usuario:

Creación de un usuario en PostgreSQL, con el nombre y contraseña elegidos, esto se realizará mediante un comando de terminal.

Conectarse a una base de datos:

Se conectará a una base de datos PostgreSql mediante la función PQsetdblogin(), de la biblioteca citada, aportando el nombre de la misma y el usuario y contraseña para conectarse, se presuponen conexiones localhost.

Crear una tabla:

Se creará una tabla para una base de datos PostgreSql, eligiendo su nombre de la tabla y los campos que se serán parte de la misma, se utilizará el comando PQexec() de la biblioteca citada.

Insertar y recuperar de una tabla:

Se proporcionarán funciones para la inserción de tuplas en las tablas y la recuperación de las mismas. Para ello primero se deberá conectar con la base de datos PostgreSql. Se ejecutará esta función mediante el uso del comando PQexec() de la biblioteca citada.

4.2.2 Módulo backup

Las funcionalidades necesarias para realizar los *backup* serán implementadas en el submódulo *backup*. Estas se han implementado de la siguiente manera:

Crear *backup*

Para crear un *backup* se utilizará el comando tar, utilizando como argumentos el nombre del *backup* que se quiere crear y el directorio del que realizar la copia de seguridad.

Reconocer usb

Para detectar si hay un usb conectado y conseguir la ruta del mismo, primero se utilizó la concatenación de los siguientes comandos:

Lsblk buscando por aquellos dispositivos denominados usb, se consigue el dispositivo de bloque en el que está montado el usb. Si al lanzar el comando no obtenemos ningún dispositivo, asumimos que no hay usbs conectados al sistema. Después mediante el uso del comando df aplicado al dispositivo de bloque conseguimos la ruta absoluta del usb.

4.2.3 Módulo Monitor

El módulo monitor está compuesto por varios submódulos necesario para desarrollar la funcionalidad descrita en el apartado de diseño. Tiene varias funciones comunes a todos como:

Comprobar si una ruta pertenece a un directorio

Esto se realizará usando la función fdopendir() en C, la cual comprueba si la ruta pertenece a un directorio.

Descubrir un directorio recursivamente

Mediante una combinación del comando ls de la shell y la función anteriormente mencionada, conseguiremos una función que recopila todos los ficheros y subdirectorios de un directorio. Esto nos servirá para posteriormente calcular la entropía y el fuzzy hashing de cada uno de los ficheros de un directorio.

En cuanto a los submódulos son los siguiente:

Entropía

Encargado de calcular la entropía de un fichero dado. Esto se consigue utilizando las funciones de la herramienta binwalk [39].

Fuzzy Hashing

Su funcionamiento se basará en la utilización de la biblioteca Fuzzy, podrá:

Generar el código hash de un fichero

Se generará un código hash propio de cada fichero, mediante técnicas de ssdeep [41].

Comparar dos códigos hash

Se podrá comparar dos códigos hash, pertenecientes a dos ficheros distintos. Obteniendo un número entre uno y cien, que representa el porcentaje de igualdad entre ambos ficheros.

Monitorización

El submódulo estará encargado de la funcionalidad con respecto a la monitorización de directorios, ficheros y la detección de *ransomware*. Sus principales funciones son:

Monitorizar un directorio

Se utilizará el comando lsof [42] periódicamente, sobre el directorio que se quiere monitorizar. También se buscará solo por los descriptores de fichero de escritura y de ejecución. Esto nos devolverá, los procesos que estén realizando este tipo de operaciones en el directorio objetivo.

Monitorizar un proceso

Cada vez que la función anterior detecte un proceso, se lanzará un hilo que lo monitorice individualmente. Esto se realizará también con el comando lsof. Sin embargo, esta vez además de lanzarlo sobre el directorio objetivo, también se lanzará sobre el proceso. Una vez que se detecta que se ha abierto un archivo para escritura, se calcula su entropía y se genera su código fuzzy hashing como se ha explicado en el capítulo de diseño del módulo correspondiente. Después, se recoge de la base de datos, la entropía y el código hash para compararlos y se empieza acumular la diferencia de los cambios. Cuando el proceso llega a una diferencia significativa, se le considera *ransomware* y se le elimina con la siguiente función:

Eliminar procesos:

Se eliminará el proceso de forma segura empezando con el comando kill sobre su pid. Para conseguir esto, se desmontarán todas las unidades ajenas al sistema mediante el comando umount. Se desconectará el bluetooth parando el servicio y se desconectarán las interfaces de red con el comando ifdown.

4.3 Pruebas

A lo largo de la implementación se han ido realizando pruebas para comprobar el correcto funcionamiento del sistema. Para ello se ha probado toda la funcionalidad implementada, partiendo desde las pruebas más hasta llegar a la más simples a las más complejas.

Se han dividido las pruebas en dos tipos, las unitarias encargadas de comprobar el funcionamiento individual de cada módulo y las de integración que comprueban el correcto funcionamiento de todos los elementos en conjunto.

4.3.1 Pruebas unitarias

Para la funcionalidad de la base de datos, al comienzo se comprobó que todas las funciones (creación de base de datos, usuario tablas) se realizaban correctamente utilizando un gestor de base de datos. En este caso se usó pgAdmin. Después, se diseñó una clase de pruebas con la función assert, que probara la creación de los distintos elementos y a insertar y extraer información de la misma.

De la clase módulo se comprobó que se conseguía detectar distintos tipos de usb. También la correcta realización de los backup, esto se probó restaurando los datos guardados en los mismos.

En cuanto a la clase monitor, primero se comprobó los distintos cálculos de entropía y generación de código hash según se iban modificando y encriptando ficheros.
Para la comprobación del monitor, se crearon varios programas de prueba, encargados de modificar ficheros dentro del directorio target y se comprobó como el monitor iba analizando cada uno de ellos.

4.3.2 Pruebas de integración

El propósito de estas pruebas es comprobar la correcta disposición de todos los módulos para trabajar juntos y la coordinación entre los mismos.

La comprobación de la correcta instalación de los módulos, para lanzar los módulos *backup* y monitor al arrancar el sistema, se tuvo que realizar manualmente. Para ello, se sacó información de la ejecución en ficheros y se comprobó el fichero /var/log/syslog que contiene todas las ejecuciones en el sistema.

En la realización del *backup* se comprobó cómo se recuperaba correctamente la información de los *backups* de la base de datos para decidir si debía realizar o actualizar uno existente. Sobre esta parte se realizaron varias pruebas, cambiando la ruta objetivo y el número de días necesarios para actualizar los *backups*, comprobando manualmente si realizaban o no los *backups* en los dispositivos usb correctos.

5 Conclusiones y trabajo futuro

5.1 Conclusiones

Después del estudio y análisis realizado, se antoja necesaria la concienciación acerca de los peligros del ransomware que, a día de hoy, se puede considerar como una de las principales amenazas informáticas. Cualquier persona, organización o equipo puede verse afectada por *ransomware* en cualquier momento. Además de considerar el gran impacto económico que puede suponer para los usuarios afectados y sobre todo para las organizaciones, en especial para aquellas que no dispongan de las herramientas y medios necesarios para hacer frente a un ataque.

Por lo tanto, se considera imprescindible la utilización de medios, herramientas y *software* específicos para la protección de los sistemas. En base a lo que se explica en el apartado 2.4, se recomienda tomar medidas de los tres tipos, medida de prevención, detección y mitigación para estar lo más seguro posible. No obstante, cabe priorizar las medidas de prevención, particularmente los backups, por su gran efectividad con respecto al coste operacional de los mismos. Se recomienda a los usuarios realizar copias de seguridad periódicas, y en este contexto de necesidad, el presente proyecto aporta una herramienta eficaz y sencilla, que dispone de esta funcionalidad y permite la realización periódica de los *backups* de modo automático. En la atmósfera de las grandes organizaciones, se recomienda la utilización de sistemas de *backup* más complejas, eligiendo las políticas de respaldo que más se adecuen a las necesidades de la empresa (sección 2.4.3 *backups*) y teniendo en cuenta, que estas copias de seguridad deben ser revisadas, para comprobar su correcta funcionalidad (esto es, se debe comprobar la integridad de los *backups*). También se debe tener establecida una estrategia de recuperación, para que en el caso de que se necesiten restaurar los datos a partir de una de estas copias, se tengan claros los posibles costes, tanto operacionales como de recursos. También es importante recalcar que, como se detalla en la sección 2.5, la mayor fuente de infección es el correo y las páginas web. En este sentido, la concienciación y la instalación de filtros y protecciones será la forma más efectiva para reducir al máximo la posibilidad de infección frente al ransomware.

Al mismo tiempo se recomienda tener sistemas activos de monitorización para detectar lo antes posible si un ransomware se está ejecutando en el equipo y con ello conseguir reducir su impacto. Con respecto a esta necesidad, el sistema presentado en este TFG presenta una primera aproximación que permite aportar en parte esta funcionalidad, siendo necesarios medidas adicionales para la correcta protección de los equipos. Entre estas medidas está la utilización de otros softwares de protección específicos, tales como antivirus, que puedan facilitar la detección de los mismos. Parte de estas utilidades están presentes en las herramientas de la web de Jesse Kornblum. Las cuales sirven para la detección de modificaciones en archivos y la realización de un volcado de memoria [43].

La implementación de toda la funcionalidad descrita en el presente documento ha requerido del perfeccionamiento de la utilización de los scripts en Linux. Habiendo sido necesario estudiar y utilizar una gran cantidad de comandos referentes al sistema.

5.2 Trabajo futuro

Debido a la envergadura del proyecto, se ha realizado una aproximación, que aporta la mayor funcionalidad posible. Esta aproximación se puede perfeccionar en muchos aspectos, a continuación, se van a presentar de manera breve, las posibilidades que se han tenido en cuenta.

Ampliación del módulo *backup*

Elección por parte de los usuarios del tipo de backup que prefieren realizar. Comprobación de que los *backups* se realizan correctamente. Utilización de servicios en la nube para guardar los *backups*.

Mejora del módulo monitor

Supervisión de varios directorios concurrentemente. Detección de ransomware mediante la utilización de más parámetros, como puede ser, % de utilización de la CPU y el tipo de ficheros que se están sobrescribiendo.

Sofisticación del sistema de notificaciones

Añadiendo más posibilidades a la hora de notificar al usuario la información referente, como envío por correo o notificaciones en el móvil con servicios como Telegram.

Ejecución en una red completa

Ejecución en una red completa de ordenadores, gestionados por un administrador de red, que se encarga de recibir las notificaciones y administrar los *backup* y la monitorización de los equipos.

Funcionalidad de recuperación después de un ataque

Añadir alguna de las técnicas detalladas en el apartado 2.4.3, para que una vez se hayan cifrado los archivos, se intente descifrarlos. Podría implementarse un *sniffer* de red, que escanee el envío de paquetes en búsqueda de la contraseña o un módulo capaz de realizar un volcado de la memoria y un posterior análisis forense.

Interfaz

Diseño de una interfaz gráfica, que facilite la interacción con el usuario.

Análisis de datos avanzado

Envío de información a una base de datos común, con los datos de entropía, uso de la CPU y cualquier otro parámetro relevante, para mediante un análisis y comparativa mejorar las métricas de detección de *ransomware*.

Referencias

[1] N. de Tomas and A. Vazquez, “Ransomware: la bolsa o la vida (digital),” *dell.com*, vol. 1, p. 51.

[2] Symantec, “The evolution of ransomware,” p. 57, 2015.

[3] H. L. Kevin Savage, Peter Coogan, “The Evolution of ransomware,” USA, 2015.

[4] M. P. Sharma, S. Zawar, and S. B. Patil, “Ransomware Analysis : Internet of Things (Iot) Security Issues , Challenges and Open Problems Inthe Context of Worldwide Scenario of Security of Systems and Malware Attacks,” *Int. J. Innov. Res. n Sci. Eng.*, vol. 2, no. 3, pp. 177–184, 2016.

[5] G. Members, S. Jan, C. Gupta, and T. Saeed, “Public Key CryptoSystems & RSA Algorithm,” 2005.

[6] N. Smart *et al.*, “Cryptography: An Introduction (3rd Edition),” vol. 3, p. 436.

[7] V. R. Joan Daemen, “The Rijndael Block Cipher,” *csrc*, vol. 2, p. 47, 99AD.

[8] M. Prerna, A. Sachdeva, and P. Mahajan, “A Study of Encryption Algorithms AES, DES and RSA for Security,” *Glob. J. Comput. Sci. Technol. Netw.*

[9] Puodzius Cassius, “How encryption molded crypto-ransomware,” *welivesecurity*, 2016. [Online]. Available: https://www.welivesecurity.com/2016/09/13/how-encryption-molded-crypto-ransomware/. [Accessed: 22-May-2017].

[10] F. R. DeJesus Terrance, “The history of ransomware | CSO Online,” *CSO*, 2016. [Online]. Available: http://www.csoonline.com/article/3095956/data-breach/the-history-of-ransomware.html. [Accessed: 19-May-2017].

[11] “WHO | World Health Organization,” *WHO*, 2017.

[12] A. Young and Moti Yung, “Cryptovirology: extortion-based security threats and countermeasures,” in *Proceedings 1996 IEEE Symposium on Security and Privacy*, pp. 129–140.

[13] Dann Albright, “Which Websites Are Most Likely to Infect You with Malware?,” *MUO*, 2016. [Online]. Available: http://www.makeuseof.com/tag/websites-likely-infect-malware/. [Accessed: 15-Mar-2017].

[14] Marinho Renato, “Mamba: The new Full Disk Encryption Ransomware Family Member | Renato Marinho | Pulse | LinkedIn,” *linkedin*, 2016. [Online]. Available: https://www.linkedin.com/pulse/mamba-new-full-disk-encryption-ransomware-family-member-marinho. [Accessed: 19-May-2017].

[15] “Ransom.Wannacry | Symantec,” *Symantec*, 2017. [Online]. Available: https://www.symantec.com/security_response/writeup.jsp?docid=2017-051310-3522-99. [Accessed: 20-May-2017].

[16] Vencislav Krustev, “Wana Decrypt0r 2.0 - Decrypt Encrypted Files - How to, Technology and PC Security Forum | SensorsTechForum.com,” *sensortechforum*, 2017. [Online]. Available: http://sensorstechforum.com/wana-decrypt0r-decrypt-files-for-free/. [Accessed: 25-May-2017].

[17] Anderson Gray Ian, “10 Tips to Make Your Computer More Secure,” *iag.me*, 2016. [Online]. Available: https://iag.me/tech/10-tips-to-make-your-computer-more-secure/. [Accessed: 19-May-2017].

[18] Osterman, “Understanding the Depth of the Ransomware Problem in the United States,” *malwarebytes*, 2016. [Online]. Available: https://www.malwarebytes.com/surveys/ransomware/?aliId=13242065. [Accessed: 14-Jan-2017].

[19] T. Bednar *et al.*, “Oracle® Database Backup and Recovery Basics 10g Release 2 (10.2) An introduction to the basics of backup and recovery of Oracle databases, focusing on the use of Recovery Manager for common backup and recovery tasks. Oracle Database Backup and Recovery Basics, 10g Release 2 (10.2),” 2005.

[20] S. Nelson, *Pro Data Backup and Recovery*, 1st ed. Apress, 2011.

[21] A. Kharaz, S. Arshad, C. Mulliner, W. Robertson, E. Kirda, and A. Kharraz, “This paper is included in the Proceedings of the 25th USENIX Security Symposium UNVEIL: A Large-Scale, Automated Approach to Detecting Ransomware UNVEIL: A Large-Scale, Automated Approach to Detecting Ransomware.”

[22] Julian Bhardwaj, “Techniques in ransomware explained,” *Naked Security*, 2012. [Online]. Available: https://nakedsecurity.sophos.com/2012/09/14/new-technique-in-ransomware-explained/. [Accessed: 08-Jan-2017].

[23] “The no more ransom project.” [Online]. Available: https://www.nomoreransom.org/. [Accessed: 20-May-2017].

[24] Mohit Arora, “How secure is AES against brute force attacks?,” *Eetimes*, 2012. [Online]. Available: http://www.eetimes.com/document.asp?doc_id=1279619. [Accessed: 11-Feb-2017].

[25] Vencislav Krústev, “Utilice Wireshark para descifrar los archivos codificados por ransomware,” *Tech Forum*, 2016. [Online]. Available: http://sensorstechforum.com/es/use-wireshark-decrypt-ransomware-files/. [Accessed: 11-Mar-2017].

[26] E. C. & E. T. Andrea Allievi, “Threat Spotlight: TeslaCrypt – Decrypt It Yourself,” *Cisco*, 2016. [Online]. Available: https://blogs.cisco.com/security/talos/teslacrypt. [Accessed: 01-May-2017].

[27] Chris Hoffman, “Windows Memory Dumps: What Exactly Are They For?,” *Hot-To Geek*, 2014. [Online]. Available: https://www.howtogeek.com/196672/windows-memory-dumps-what-exactly-are-they-for/. [Accessed: 12-Mar-2017].

[28] K. Amari, K. A. Mil, and C. Cid, “Techniques and Tools for Recovering and Analyzing Data from Volatile Memory Techniques and Tools for Recovering and Analyzing Data from Volatile Memory GCFA Gold Certification Techniques and Tools for Recovering and Analyzing Data from Volatile Memory 3,” *SANS Inst.*, p. 61, 2009.

[29] Asim Rab *et al.*, “Ransomware and businesses,” *Symantec*, vol. 1, p. 30, 2016.

[30] “By The Numbers: Ransomware Rising - Security News - Trend Micro USA,” *Trendmicro*, 2016. [Online]. Available: https://www.trendmicro.com/vinfo/us/security/news/cybercrime-and-digital-threats/by-the-numbers-ransomware-rising. [Accessed: 15-Feb-2017].

[31] A. T. Report, “SonicWall Annual Threat Report,” *Sonicwall*, p. 23, 2016.

[32] “Beazley breach insights - January 2017,” *Beazley*, 2017. [Online]. Available: https://www.beazley.com/news/2017/beazley_breach_insights_january_2017.html. [Accessed: 10-Apr-2017].

[33] “The 2017 Endpoint Protection Ransomware Effectiveness Report,” *Knowbe4*, p. 9, 2017.

[34] “CoinDesk BPI,” *coindesk*. [Online]. Available: http://www.coindesk.com/price/. [Accessed: 23-May-2017].

[35] *World Economic Outlook Database*. International Monetary Fund, 2017.

[36] J. Kornblum, “Identifying Almost Identical Files Using Context Triggered Piecewise Hashing.”

[37] “Postgre SQL History and Advantages • ByteScout,” *ByteScout*, 2014. [Online]. Available: https://bytescout.com/blog/2014/10/postgre-sql-history-and-advantages.html. [Accessed: 10-Dec-2017].

[38] “PostgreSQL: Documentation: 9.1: libpq - C Library,” *PostgreSql*. [Online]. Available: https://www.postgresql.org/docs/9.1/static/libpq.html. [Accessed: 11-Jan-2017].

[39] Devttys0., “Binwalk,” 2015. [Online]. Available: https://github.com/devttys0/binwalk/wiki.

[40] F. Hashing and J. Kornblum, “Computer Forensics and Intrusion Analysis ManTech SMA,” *ManTech SMA*.

[41] H. G. Jesse KornBlum, “Fuzzy Hashing API: Main Page (3),” *ssdeep.sourceforge*, 2015. [Online]. Available: http://ssdeep.sourceforge.net/api/html/index.html. [Accessed: 20-Apr-2017].

[42] “lsof(8): open files - Linux man page,” *linux die*. [Online]. Available: https://linux.die.net/man/8/lsof. [Accessed: 26-May-2017].

[43] Jesse Kornblum, “Jesse Kornblum - Tools,” *jessekornblum.com*. [Online]. Available: http://jessekornblum.com/tools/. [Accessed: 29-Apr-2017].

Glosario

API	Application Programming Interface.
Bitcoin	Divisa electrónica descentralizada. Carece de un emisor central.
Bot	Programa informático que se encarga de imitar un comportamiento humano.
Backup	Copia de seguridad o proceso de realizar la copia de seguridad.
Cifrar	Acción de transformar una información por medio de distintos procedimientos para protegerla y evitar que sea legible a terceras personas.
Command and control	Servidores utilizados para controlar y mandar directivas a malwares.
Encriptar	Acción de transformar una información por medio de distintos procedimientos para protegerla y evitar que sea legible a terceras personas.
Entropía	Medida de la incertidumbre que existe en un mensaje.
Firewall	Programa informático que controla y filtra el acceso a la red.
Malware	Tipo de software malicioso.
Shell	Intérprete de comandos Linux con interfaz, que permite al usuario utilizar las herramientas del sistema.
Sniffing	Tráfico de red.
Tor	The Onion Router. Red utilizada para ocultar la identidad de sus usuarios, otorgándoles anonimato.
Troyano	Tipo de malware cuyo principal efecto es el de controlar el sistema de la víctima.
VPN	Virtual Private Network.

Anexos

A Esquemas del sistema

Se adjuntan dos esquemas del sistema. Tanto del módulo *backup*, como del módulo monitor para facilitar un entendimiento visual de su estructura.

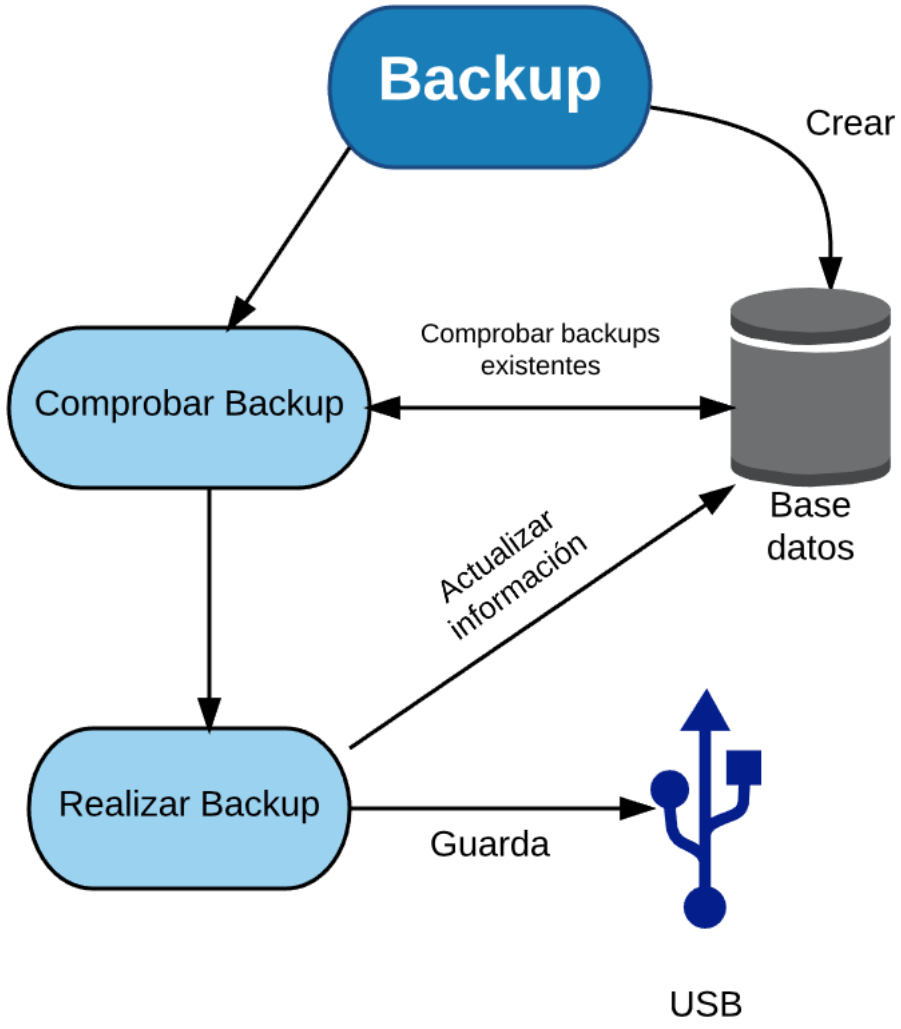

Figura 8. Esquema general del módulo *backup*.

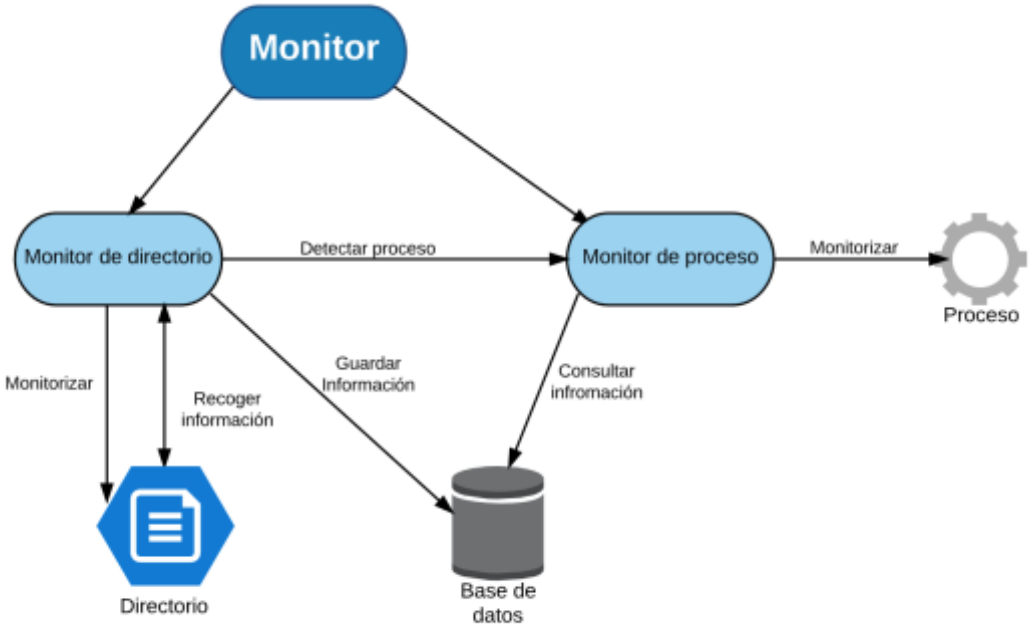

Figura 9. Esquema general del módulo monitor.

B Manual de uso

A continuación, se presenta un esquema del directorio del proyecto, que indica cómo está dividido en carpetas para facilitar su utilización.

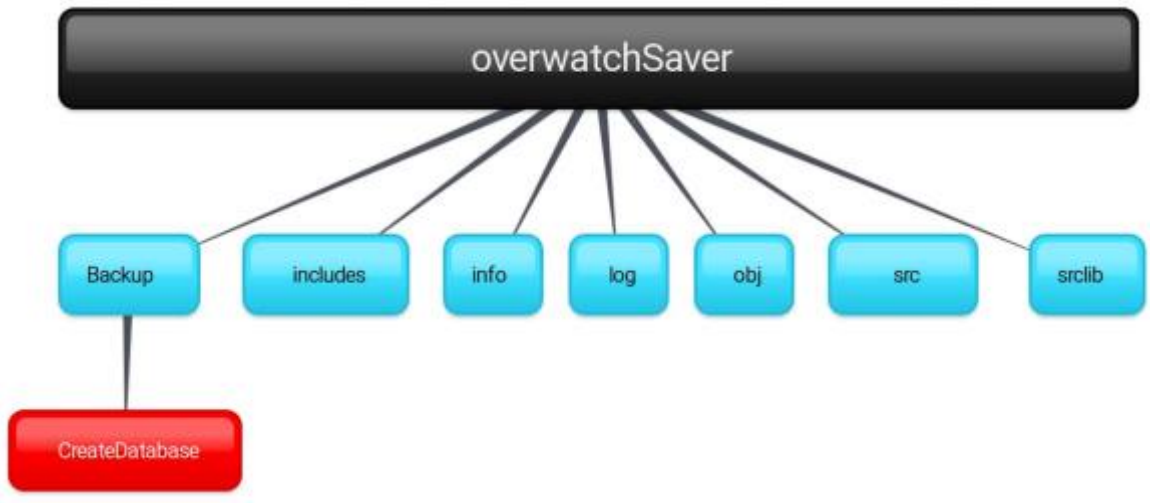

Figura 10. Esquema del directorio del proyecto.

En la carpeta raíz, overwacthSaver, encontraremos el script installingScript. El cual deberá ser ejecutado para instalar la aplicación en el sistema. Dentro del mismo se indica como ubicación para el proyecto el directorio /etc/ esto puede ser modificado libremente. Esto hará entre otras cosas que la aplicación se ejecuta cada vez que se arranca el sistema, modificando el fichero /etc./rc.local. En caso de querer desinstalarlo, es suficiente con comentar las líneas que se añaden.

En la carpeta info, se generarán un fichero denominado database.conf que contiene los credenciales de acceso a la base de datos. Este fichero no se debe modificar. En esta misma carpeta se encuentra otro archivo llamado conf.txt, en el mismo se pondrá la ruta objetivo para realizar los *backup* y un número. Este número representa los días mínimos que deben haber transcurrido desde la realización de un *backup* hasta actualizarlo.

C Repositorio del proyecto

Como se ha comentado en los requisitos del proyecto, el software generado será de código abierto y libre. Por ello se ha decidido compartirlo en repositorio de Github público. Al cual tendrá acceso cualquier persona interesada sin restricciones. Las cabeceras del código están comentadas para facilitar su entendimiento. A continuación, se adjunta el enlace al mismo:

github.com/jorge95ruiz/overwatchSaver