Programa de Doctorado en Tecnologías de la Información y la Comunicación de la Universidad de Granada

UNIVERSIDAD DE GRANADA

Cripto-*ransomware*: Análisis y detección temprana basada en el uso de archivos trampa

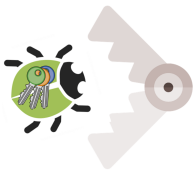
Autor
José Antonio Gómez Hernández

Director
Dr. Pedro García Teodoro

Escuela Técnica Superior de Ingenierías Informática y de Telecomunicación

—

Granada, junio de 2022



Cripto-*ransomware*: Análisis y detección temprana basada en el uso de archivos trampa

---

Sobre los sistemas operativos Linux, Windows y Android

Autor
José Antonio Gómez Hernández

Director
Dr. Pedro García Teodoro

Editor: Universidad de Granada. Tesis Doctorales
Autor: José Antonio Gómez Hernández
ISBN: 978-84-1117-461-9
URI: http://hdl.handle.net/10481/76803

Cripto-*ransomware*: Análisis y detección temprana basada en el uso de archivos trampa

José Antonio Gómez Hernández

Palabras clave: *ransomware*, detección temprana, archivos trampa, monitorización reactiva, Android, Windows, Linux

Resumen

La proliferación de amenazas en el ciberespacio, especialmente las producidas por cripto-*ransomware*, junto con la incesante penetración de las Tecnologías de la Información y la Comunicación en nuestro día a día, hacen que cada vez sean más necesarias herramientas que permitan a los usuarios una protección eficaz contra el *ransomware* y que a su vez sean sencillas de instalar y transparentes en cuanto a su funcionamiento.

Para establecer un método que cumpla estos criterios en diferentes plataformas (Linux, Windows y Android), se ha revisado una caracterización de este tipo de amenaza a través del modelo de ataque *Cyber Kill Chain*, que permite conocer todos los mecanismos de evasión de la seguridad así como el proceso de infección, cifrado y extorsión. Esto ha permitido establecer en nuestra propuesta el punto de defensa que ninguna muestra puede evadir, el cifrado.

Establecido el punto de defensa, analizamos las diferentes propuestas de defensa frente al *ransomware* que encontramos en la literatura. En nuestro caso optamos por una solución basada en archivos trampa, que permite determinar el comportamiento de un programa malicioso de forma rápida, con bajo coste de recursos y, sobre todo, que es capaz de detener la ejecución de la amenaza. Este método se ha implementado mediante la herramienta *R-Locker* sobre los sistemas Linux y Windows, y con *FileMonitor* en Android. Resaltar que dichas herramientas no requieren ningún tipo de entrenamiento, por lo que pueden detectar muestras de *ransomware* de día cero.

Se analizan en detalle los aspectos técnicos de dichas herramientas, cuyo código esta disponible públicamente, y se describen los pros y contras de las mismas.

Crypto-ransomware: Analysis and honeyfile-based early detection

José Antonio Gómez Hernández

Keywords: ransomware, detection, honeyfiles, reactive monitoring, Android, Windows, Linux

Abstract

The proliferation of threats in cyberspace, especially those produced by crypto-*ransomware*, together with the incessant penetration of Information and Communication Technologies in our daily lives, highlight the need of more tools that allow users an effective protection against *ransomware*, simple to install and transparent, in terms of how it works.

To establish an method that meets these criteria on different platforms (Linux, Windows and Android), a characterization of this type of threat has been reviewed through the attack model *Cyber Kill Chain*, that allows us to know all the security evasion mechanisms as well as the infection, encryption and extortion process. This has allowed to establish in our proposal the point of defense that no sample can avoid, the encryption.

Once the point of defense has been established, the different defense proposals against *ransomware* found in the literature have been reviewed and a solution based on trap files has been chosen here, which allows to determine the behavior of a malicious program very quickly, with low cost in terms of resource consumption and, in addition, that it is capable of stopping the execution of the threat. This approach has been implemented using the *R-Locker* tool on Linux and Windows systems, and *FileMonitor* on Android. Note that these tools do not require any type of training, so they can detect zero-day ransomware sample.

The technical aspects of these tools, whose code is publicly available, are analyzed in detail, and the pros and cons discussed.

Agradecimientos

Al finalizar esta memoria me gustaría mostrar mi agradecimiento más sincero a todos aquellos que me han ayudado y animado a llevar a cabo esta tarea.

Agradecer, sobre todo a mi tutor/director de Tesis, el Prof. Dr. Pedro García Teodoro, todo el tiempo y esfuerzo empleado en todas las etapas del proceso, desde los trabajos que la soportan hasta la elaboración de esta memoria, y por los buenos consejos que siempre recibo de él.

No puede faltar mi agradecimiento al grupo de investigación NESG (*Network Engineering & Security Group*) como colectivo, y a todos sus miembros a título individual (Pedro, Gabriel, Marga, Roberto y Rafael), su cooperación, apoyo y ánimo en todos estos años. Puedo decir que la decisión de incorporarme a él ha sido de las mejores tomadas en el ámbito profesional.

Sería injusto olvidar a mis compañeros/amigos del departamento y de la ETSIIT, que seguro les alegra saber que he alcanzado este objetivo.

Por supuesto no podría faltar mi familia: Encarni, José Antonio y Manuel, por su apoyo durante este larguísimo viaje, y a mis padres (Manuel y Mª Luisa) que con su sabiduría me permitieron iniciar este camino.

Glosario de acrónimos

AaaS	*Access as a Service*
ACO	*Ant Colony Optimization*
AD	*Active Directory*
ADS	*Alternate Data Stream*
AES	*Advanced Encryption Standard*
AML	*Adversarial Machine Learning*
AMSI	*Antimalware Scan Interface*
ANN	*Artificial Neuronal Network*
AONT	*All-Or-Nothing transform*
API	*Application Programming Interface*
APT	*Advanced Persistent Threat*
ART	*Android RunTime*
ATT&CK	*Adversarial Tactics, Techniques, and Common Knowledge*
BBN	*Bayesian Belief Network*
BiLSTM	*Bidirectional Long Short Term Memory*
BYOD	*Bring Your Own Device*
CFG	*Control Flow Graph*
CKC	*Cyber Kill Chain*
CNG	*Cryptography Next Generation*
CNN	*Convolutional Neuronal Network*
COBIT	*Control OBjectives for Information and related Technology*
CPS	*Control Process Systems*
CVE	*Common Vulnerabilities and Exposures*
CWE	*Code Weak Enumeration*
C&C	*Command and Control*
DAC	*Discretional Access Control*
DFR	*Digital Forensic Readiness*
DGA	*Domain Generation Algorithm*
DL	*Deep Learning*
DLL	*Dynamic Link Library*
DMARC	*Domain-based Message Authentication, Reporting & Conformance*
DoS	*Denial of Service*
DR	*Disaster Recovery*

DT	*Decision Tree*
DVM	*Dalvik Virtual Machine*
EDR	*Endpoint Detection Response*
EK	*Exploitation Kit*
EmRmR	*Enhanced maximun-Relevance and minimum-Redundancy*
exFAT	*Extensible File Allocation Table*
FA	*Factor Analysis*
FIFO	*First Input First Output*
FRP	*Fast Reverse Proxy*
FSM	*Finite State Machine*
FSRM	*File Server Resource Manager*
HIDS	*Host Intrusion Detection System*
HMM	*Hidden Markov Model*
HPC	*Hardware Performance Counter*
IA	*Inteligencia Artificial*
IAB	*Initial Access Broker*
IC	*Infraestructuras Críticas*
IDS	Intrusion Detection System
IoC	*Indicator of Compromise*
IoT	*Internet of Things*
IPC	*Inter-Process Communication*
IPFS	*InterPlanetary File System*
IPS	*Intrusion Prevention System*
IR	*Incident Response*
IRP	*Input/output Request Packet*
ISACA	*Information Systems Audit and Control Association*
K-NB	*K-Nearest Neighbors*
LBP	*Local Binary Pattern*
LR	*Logistic Regression*
LSA	*Latent Semantic Analysis*
LSASS	*Local Security Authority Subsystem Service*
LSTM	*Long-Short Term Memory*
MBR	*Master Boot Record*
MDM	*Mobile Device Management*
MI	*Mutual Information*
ML	*Machine Learning*
NAA	*Network-Assisted Approach*
NAS	*Network Attached Storage*
NB	*Naïve Bayes*
NIST	*National Institute of Standards and Technology*
NLP	*Natural Language Processing*
NN	*Nearest Neighbor*
NVMe	*Non-Volatile Memory express*
OFAC	*Office of Foreign Assets Control*

OoW	*Ocurrene of Word*
OTC	*Over The Counter*
OWA	*Outlook Web App*
OWASP	*Open Web Application Security Project*
PCA	*Principal Component Analysis*
PFM	*Programmable Forwarding Engines*
PIN	*Personal Indentification Number*
PMT	*Protection Motivation Theory*
RaaS	*Ransomware as a Service*
RANDEP	*RANsomware and DEPloyment*
RAP	*RAnsomware Protection*
RAT	*Remote Access Trojan*
RC4	*Rivest Cipher 4*
RCE	*Remote Code Execution*
RDP	*Remote Desktop Protocol*
RF	*Random Forest*
RLR	*Regularized Logistic Regression*
RNN	*Recurrent Neural Networt*
RPO	*Recovery Point Objective*
RSA	*Rivest, Shamir y Adleman*
RTO	*Recovery Time Objective*
SCADA	*Supervisory Control and Data Acquisition*
SDN	*Software Defined Network*
SDK	*Software Development Kit*
SDT	*Sistema de Distribución de Tráfico*
SGD	*Stochastic Gradient Descent*
SMB	*Server Message Block*
SVM	*Support Vector Machine*
TF-IDF	*Term Frequency-Inverse Document Frequency*
TI	*Tecnologías de la Información*
TTP	*Tactics, Techniques, Procedures*
TVD	*Truncated singular Value Descomposition*
VEH	*Vectored Exception Handling*
VM	*Virtual Machine*
VPN	*Virtual Private Network*
VSC	*Volume Shadow Copy*
W3RF	*Windows Registry and RAM Readiness Framework*
WMI	*Windows Management Instrumentation*
XGBoost	*eXtreme Gradient Boosting*
ZTNA	*Zero-Turst Network Access*

Índice general

Índice de figuras

Índice de tablas

Capítulo 1

Motivación y objetivos

Es más que evidente que las Tecnologías de la Información y la Comunicación constituye una parte prominente de nuestras vidas. Ya sea a nivel laboral, en nuestra relación con los demás o la parcela del ocio, los computadores, *tablets*, y otros tipos de dispositivos están presentes en nuestro día a día.

Esta penetración de la tecnología en todos los ámbitos de nuestra vida ha creado un nicho de explotación para el cibercrimen. Por tanto, es básica la seguridad de nuestros sistemas para garantizar la integridad y privacidad de los mismos en nuestro quehacer diario.

Las diez principales amenazas en 2002, según Embroker [1] son:

- La ingeniería social - La técnica de *hacking* denominada ingeniería social sigue siendo uno de los métodos de ataque más peligrosos ya que descansa en mayor medida en el error humano que en las vulnerabilidades técnicas. Los ataques de *phishing* o de suplantación de identidad siguen evolucionando con la incorporación de nuevas tendencias, tecnología y tácticas.

- Exposición de terceras partes - Los cibercrimanales pueden eludir los sistemas de seguridad atacando redes menos protegidas de terceras partes que tienen acceso privilegiado al objetivo principal.

- Errores de configuración - Incluso los sistemas de seguridad profesionales tienen errores en cómo se instala o gestiona el software, como se observa del hecho de que el 80 % de las pruebas de penetración externas encuentran una mala configuración explotable.

- Pobre higiene de seguridad - La *ciberhigiene*, es decir, los hábitos y prácticas regulares en el uso de la tecnología aún dejan mucho que

desear. Por ejemplo, en Estados Unidos casi el 60 % de las organizaciones dejan a la memoria humana la gestión de claves, el 42 % gestionan las claves con *post-it*, y solo el 37 % de las personas usan un doble factor de autenticación.

- Vulnerabilidades en la nube - Con el rápido crecimiento de la nube, las vulnerabilidades de la misma han ido creciendo. Es de esperar que en 2022 se asiente la arquitectura de confianza cero para reducir este aumento.

- Vulnerabilidades en dispositivos móviles - Desde la pandemia de 2019 ha habido un repunte del uso de dispositivos móviles en lo que respecta a monederos móviles o tecnologías de pago sin contacto. Además, muchas compañías han iniciado la implantación de políticas BYOD (*Bring Your Own Device*) dado el alto número de incidentes. Paradójicamente, los sistemas de gestión de dispositivos móviles (MDM) se han convertido en un objetivo para los cibercriminales ya que, al estar conectados a la red de dispositivos de este tipo, se ataca a todos los empleados simultáneamente.

- Internet de las cosas - El alto número de hogares con dispositivos inteligentes produjo en el primer semestre de 2021 un pico de 1.500 millones de brechas de seguridad. Esto, combinado con la mala ciberhigiene, produce que un dispositivo inteligente sea atacado en los cinco minutos siguientes a su conexión a Internet.

- *Ransomware* - Si bien el *ransomware* no es una amenaza nueva en sí, se ha convertido en la más cara en los últimos años. Este tipo de ataque tiene un coste para las empresas, debido pago del rescate y a la falta de ingresos mientras los equipos están inoperativos (que suele tener una media de 21 días), de una media de entre 5.000 y 200.000 dólares. Además, debemos contabilizar costes más difíciles de cuantificar como pérdida o daño a la reputación, responsabilidad legal, etc. El *ransomware* ha evolucionado en sofisticación, disponibilidad y conveniencia para los cibercriminales. El modelo de *Ransomware como servicio* (RaaS) permite que este tipo de ataque sea más asequible para los ciberdelincuentes, lo que pronostica un incremento de los mismos.

- Gestión pobre de los datos - La cantidad de datos creados por los consumidores se duplica cada cuatro años, pero la mitad de los mismos nunca se analiza o utiliza. Este excedente de datos genera confusión, lo que lo hace vulnerable a ataques y genera la posibilidad de brechas innecesarias de datos.

- Procedimientos pos-ataque inadecuados - Si bien es evidente que los agujeros de seguridad deben parchearse tras un ataque, debemos indi-

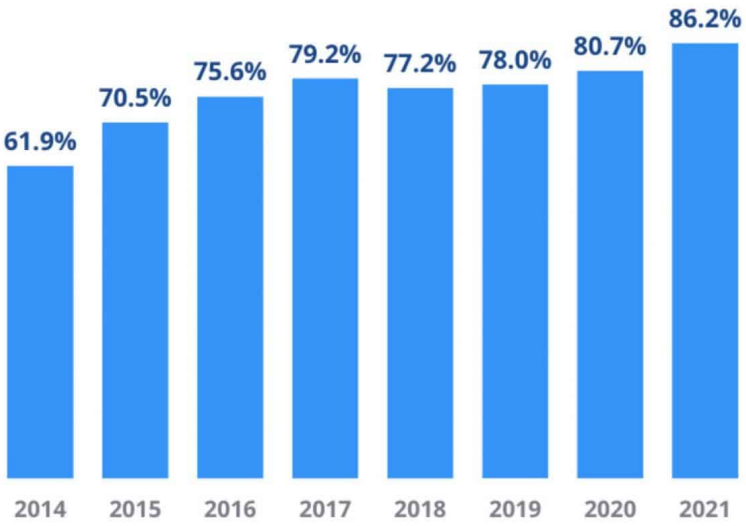


Figura 1.1: Porcentaje de organizaciones comprometidas por al menos un ataque con éxito [2].

car que en 2021 el 80 % de las víctimas de *ransomware* que pagaron el rescate reconocen haber sufrido otro poco después. De hecho, el 60 % de estos ataques se podrían haber evitado si se hubiesen aplicado los parches software correspondientes.

Hemos de resaltar que algunos de estos factores se han visto incrementados desde la pandemia de Covid-19, ya que las empresas han delegado en terceras partes algunas de sus funciones debido al teletrabajo. Así mismo, debido a trastornos socio-políticos y el estrés financiero continuado, ha aumentado la cantidad de errores por descuido cometidos en el trabajo.

Estos elementos tienen un efecto muy importante a nivel mundial. Las estadísticas indican que en 2021 el 86,2 % de las organizaciones se han visto comprometidas por al menos un ataque con éxito [2]. La Figura 1.1 recoge la evolución de esta cifra en el periodo 2014-2021. Desde el punto de vista económico, se estima que el coste del cibercrimen en 2022 es al menos de 1,2 mil billones de dólares [3]. También podemos ver su evolución en media durante el periodo 2013-2020, como se muestra en la Figura 1.2.

Como podemos observar, el *ransomware* es uno de los problemas de seguridad más relevantes desde hace tiempo, pero en los últimos años ha evolucionado en la selección de sus objetivos hacia organizaciones y empresas, donde el balance entre coste del ataque y las ganancias obtenidas es más rentable. Si lo comparamos con otro tipo de amenazas, según el informe de Imperva de 2021 [4], el *ransomware* es la segunda amenaza después del resto de tipología de *malware*, como muestra la Figura 1.3.

Para apreciar este impacto, el informe de Deloitte [5] de 2022 recoge que:

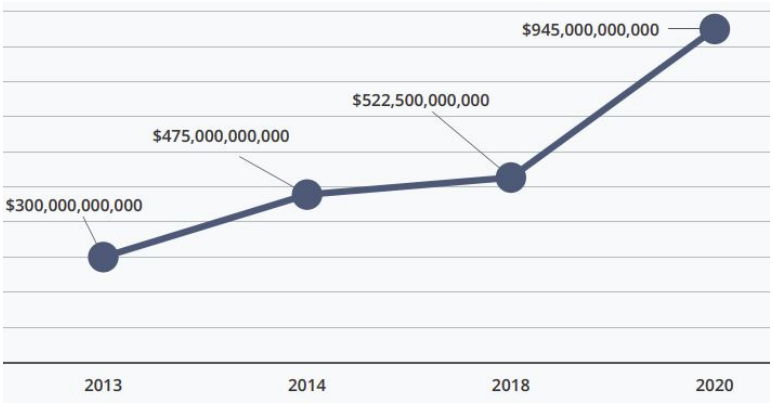


Figura 1.2: Coste medio estimado del cibercrimen en dólares [2].

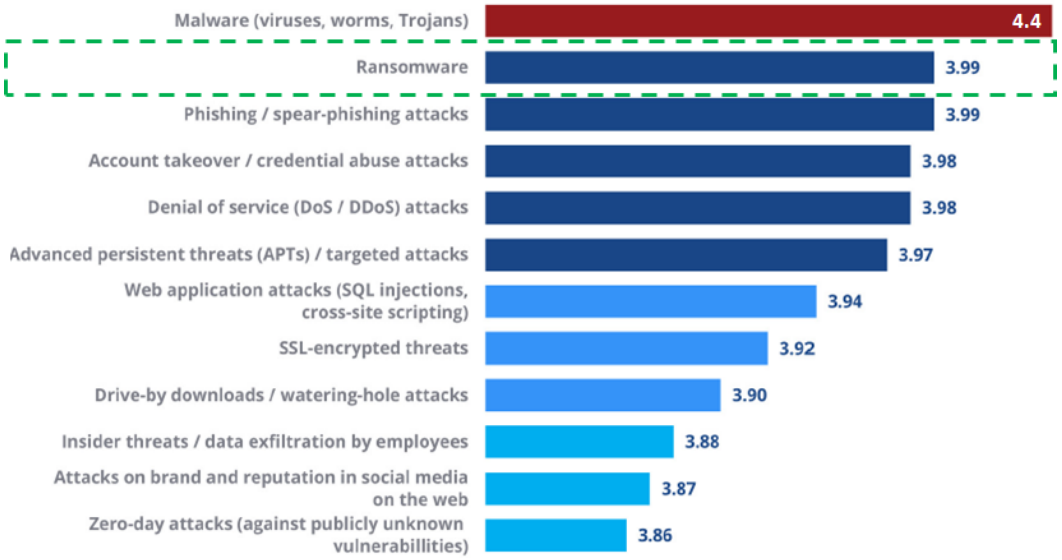


Figura 1.3: Percepción relativa de amenazas por tipo (escala de 1 a 5) [4].

se producen 4.000 ataques de *ransomware* diariamente; en media, trascurren 191 días hasta que una organización identifica la brecha; y que el número de muestras que ex-filtran datos ha crecido un 8,7 %. Desde el punto de vista económico, el pago de los recates ha supuesto a las víctimas 265 mil millones de dólares, dado que las cantidades demandadas se han visto incrementadas un 109 % respecto de 2019. Es más: un 42 % de empresas con ciber-seguro frente al *ransomware* no han podido recuperar todas sus pérdidas. Respecto al impacto en las propias empresas, indicar que: la media de tiempo de parada de los sistemas es de 19 días; el 92 % de las empresas que han pagado rescate no han recuperado todos sus datos; el 53 % indican que el mismo ha dañado a su marca; y el 26 % de las organizaciones informan de la necesidad de cierre temporal.

Reducir este problema requiere el avance de tres líneas de trabajo. La primera abordaría el problema antes de que se produzca, es decir, adoptar por parte las organizaciones e individuos una ciber-higiene correcta, que se debe afrontar desde una profunda concienciación de todos los sectores y usuarios. Segundo, desarrollar herramientas eficaces contra esta amenaza y que sean fáciles de gestionar e instalar por usuarios finales. Por último, también hay que adoptar medidas normativas y legales necesarias que dificulten los ataques de *ransomware*, rompan el retorno económico de la extorsión realizada y faciliten la persecución de los cibercriminales en cualquier parte del mundo.

Desde el punto de vista tecnológico, eje en que se desarrolla este trabajo, nuestro propósito es el diseño y puesta en marcha de una herramienta de uso fácil que evite que se produzca el daño real (cifrado de archivos) y permita eliminar la amenaza, en un amplio abanico de sistemas. Dado que la cuota de mercado de los sistemas operativos Android, Windows y Linux supone, a mayo de 2022, el 73,38 % (que se reparte al 43,23 %, 29,2 %, y 0,95 %, respectivamente) de las plataformas de computación [6], se aborda el desarrollo de un enfoque que permita solventar el problema en dichas plataformas dado que, aunque diferentes a *priori*, tienen muchos aspectos en común en cuanto a los mecanismos desplegados a través de su API.

1.1. Requisitos generales

El desarrollo propuesto en este trabajo tiene como partida una serie de requisitos autoimpuestos destinados a la creación de una herramienta anti-*ransomware* aceptable por usuarios finales, como son:

- R1 - *Efectividad*, de manera que las acciones dañinas del *ransomware* sobre el sistema sean nulas o mínimas.

- R2 - *Bajo consumo*, tanto desde el punto de vista computacional como del uso de memoria y almacenamiento, ya que de otra forma no sería

escalable.

- R3 - *Claridad*, esto es, que no requiera privilegios especiales para su instalación o ejecución, ya que en otro caso, no sería utilizable por los usuarios finales.

- R4 - *Transparencia*, desde la perspectiva de funcionamiento normal del entorno de ejecución, es decir, la solución no debe afectar al resto de aplicaciones y servicios.

- R5 - *Simplicidad*, de forma que no necesite procedimientos complejos de operación.

- R6 - *Multiplataforma*, debería tener un funcionamiento similar en las diferentes plataformas que puede utilizar un usuario para una mayor aceptación.

1.2. Objetivos y contribuciones

De lo indicado hasta el momento, está claro que los objetivos principales que se abordan en la presente Tesis son dos:

- O1 - Diseñar una propuesta de detección de cripto-*ransomware* que cumpla los requisitos R1-R6 y que permita establecer contramedidas eficaces una vez detectada la amenaza.

- O2 - Desarrollar una herramienta de acuerdo a O1 que pueda utilizarse en diferentes sistemas operativos (Linux, Windows y Android), pero sujeta a las características propias de cada plataforma para su implementación.

Evidentemente, para alcanzar los objetivos O1 y O2 será necesario establecer unos objetivos específicos previos, a saber:

- OE1 - Obtener un profundo conocimiento de la evolución y modo de operación del *ransomware* para establecer los mecanismos de seguridad que explota.

- OE1 - Realizar un estudio de la literatura especializada para ver qué soluciones se han dado hasta la fecha, sus pros y contras, para derivar una solución que cumpla con los objetivos principales O1 y O2.

Respecto de los expuesto, hemos de indicar que se han alcanzado los objetivos propuestos, tanto los generales (O1-O2) como los específicos (OE1-OE2), en base a:

- *Uso de archivos trampa* - Se ha conseguido diseñar y desplegar una solución basada en el empleo de archivos trampa efectivo para la detección del *ransomware* que tiene un impacto nulo o mínimo en el sistema de archivos protegido. El aspecto más novedoso de nuestra propuesta de archivos trampa es que, en contra de lo que se podría pensar, los archivos trampa son entidades activas (no archivos pasivos), que están constituidos por un FIFO y un proceso escritor. El FIFO es capaz de bloquear al proceso atacante, mientras que el proceso escritor actúa como monitor para la detección y permite lanzar las contramedidas para finalizar con la amenaza. Este enfoque ha mostrado su eficacia en la detección/finalización de las muestras de *ransomware*.

- *Construcción de herramientas* basadas en esta propuesta en varias plataformas - Hemos sido capaces de implementar la solución para los sistemas Linux, Windows y Android, si bien han sido necesarias modificaciones para adaptarlas a los mecanismos y servicios suministrados por las diferencias en la API que ofrece cada plataforma, en especial Android.

- *Estudio de la amenaza del ransomware* - Tras una valoración y cuantificación del problema que supone el *ransomware*, se ha abordado en profundidad un estudio de cómo funciona el mismo haciendo uso de un modelo de ataque, que permite aflorar todos las vías que los ciber-criminales utilizan para acabar infectando y bloqueando un sistema. Dado que el proceso de infección puede tomar diferentes caminos, queda claro que el único punto en común de esos caminos es la fase de cifrado; por ello, el trabajo está centrado en la detección de esta etapa mediante un mecanismo de engaño, archivos trampas, que permite alcanzar los objetivos propuestos.

- *Estudio de las propuestas de defensa frente al ransomware* - La memoria recoge una amplia revisión de las propuestas de defensa que permite comprender las líneas de trabajo existentes y poder comparar las propuestas desde diferentes aspectos como son los mecanismos usados, el coste de recursos necesarios, su complejidad, las tasas de detección, etc. Este estudio está enmarcado dentro de una taxonomía que facilita el trabajo y permite ubicar las diferentes propuestas.

A continuación y para finalizar el presente capítulo, describimos la estructura de la memoria con los detalles más relevantes de cada apartado.

1.3. Estructura de la tesis

El Capítulo 2 presenta el cripto-*ransomware* como el objeto de estudio entre los diferentes tipos existentes de *ransomware*. A continuación se

presenta una evolución histórica destinada a comprender cómo esta amenaza sigue creciendo en complejidad y alcance. En complejidad porque *(i)* la construcción tecnológica del mismo se hace más sofisticada en sus ataques con el objetivo de evadir las defensas de los sistemas, *(ii)* los mecanismos de operación han llegado a esquemas de una gran especialización entre los cibercriminales y *(iii)* los sistemas para llevar a cabo la extorsión se han diversificado. En alcance por cuanto que ya no se buscan víctimas aleatoriamente, sino que se seleccionan las organizaciones a atacar en base a factores como su volumen de negocio, número de sistemas potencialmente afectables, etc. con el objetivo de asegurar el pago del rescate, fin último de dicho tipo de ataque. En este capítulo se analizan con destalle las técnicas de ataque utilizadas por las diferentes variantes de *ransomware* para conocer cuáles deben ser y dónde deben ubicarse los puntos de defensa.

En el Capítulo 3 se aborda la descripción actual de los sistemas de defensa frente al *ransomware* propuestos en la literatura (prevención, detección y respuesta), si bien dedicaremos más atención a los de detección. Para analizar las propuestas de la literatura utilizamos una taxonomía de detección del *ransomware* basada en las fuentes de datos que se utilizan, ya sean provenientes del espacio de usuario o del *kernel*, y el procesamiento que se realiza sobre dichos datos, con las técnicas de aprendizaje máquina como actuales protagonistas en la literatura. Si bien existen mecanismos de detección para las diferentes etapas de un ataque, nuestro trabajo se centra en la fase en la que el *ransomware* inicia el cifrado de los archivos, que es la última antes de hacer efectivo el daño sobre el sistema, y por la que es seguro que todo programa de este tipo debe pasar. En concreto, utilizaremos la técnica de archivos trampa o *honeyfiles*, que creemos cumple mejor con los objetivos propuestos con anterioridad.

Establecido el punto de defensa seleccionado para el trabajo, el Capítulo 4, tras una revisión de la situación de la amenaza del *ransomware* en sistemas Linux, aborda el enfoque de detección basado en archivos trampa activos que permitan bloquear muestras de *ransomware* cuando intentan cifrar los archivos. El trabajo desarrollado se materializa en la implementación de la herramienta anti-*ransomware* R-Locker para la plataforma Linux. A diferencia de otras propuestas de archivos trampa, donde los archivos son entidades pasivas, nuestro enfoque se basa en simular un archivo mediante un FIFO que tiene en su extremo de escritura un proceso (por ello lo denominamos activo), que hace las veces de detector. En el capítulo se establecen cuáles son los requisitos establecidos para construir nuestro mecanismo de detección, que son básicamente la no necesidad de privilegios especiales para desplegar la herramienta, su sencillez y bajo consumo de recursos. Todo eso hace que nuestra herramienta sea fácil de instalar y utilizar, además de ser efectiva en su objetivo.

Una vez comprobada la bondad de la herramienta propuesta, se ve la posibilidad de migrarla a una plataforma más extendida en número de usuarios

como es el caso de Windows. Así, en el Capítulo 5 se describe cómo se ha realizado la re-implementación de R-Locker en este sistema operativo, que aunque soporta los mecanismos básicos precisos, su programación requiere cambios, ya que Windows suministra una semántica diferente para los mismos. También se han mejorado algunos aspectos de la herramienta como son la semi-automatización de la misma con la introducción de listas blancas/negras y la protección del sistema de archivos de forma completa.

Al objeto de cubrir los sistemas operativos mayoritarios, se aborda replicar la propuesta de R-Locker a Android, que es la plataforma de más amplio uso en móviles. Como se analiza en el Capítulo 6, los mecanismos que soportan R-Locker, FIFO y enlaces simbólicos, tienen un uso limitado (FIFO) o son inviables (caso de los enlaces simbólicos) en Android. Esta situación nos lleva a proponer un mecanismo basado en la monitorización del sistema. La monitorización activa presenta varios problemas, siendo el más significativo el consumo de recursos como la batería. Por ello, se propone una monitorización reactiva basada en eventos, mediante el mecanismo *FileObserver*, que permite delegar en el *kernel* la notificación de los eventos relevantes, liberando a la aplicación de realizar esta tarea.

En base a ello desarrollamos una aplicación, denominada *FileMonitor*, consistente en la monitorización de archivos trampa, o señuelos, distribuidos por el sistema de archivo para detectar los patrones de interacción entre el *ransomware* y los archivos en el proceso de cifrado de los mismos. Como en los casos anteriores, se mantiene el uso de listas blancas y negras. El principal problema de esta solución, comparada con R-Locker para Linux o Windows, es que no podemos bloquear la muestra detectada, por lo que el tiempo de respuesta debe ser tan corto como sea posible.

Finalizamos la memoria con el Capítulo 7, donde se exponen los resultados obtenidos durante la realización de trabajo, que de forma sintética son: *(i)* un conocimiento detallado de los mecanismos utilizados por esta amenaza basados en un modelo general de ataque y del modelo de explotación *ransomware as a service*, que muestra la necesidad de abordar la defensa no solo desde el punto de vista técnico, sino incluyendo también aspectos preventivos y normativos; *(ii)* la revisión de las propuestas de defensa en amplitud permite ver el panorama general y, con la ayuda de una taxonomía, localizar las propuestas basadas en archivos trampa activos; *(iii)* el diseño de enfoque de archivos trampa que permite desarrollar una herramienta en Linux, R-Locker, que cumple los requisitos establecidos; *(iv)* el desarrollo de la versión de la herramienta para Windows; y, ante la imposibilidad de utilizar los mecanismos base empleados en R-Locker en Android, *(v)* solución de detección en esta plataforma basada en un mecanismo de monitorización reactiva.

A modo de conclusión de este resumen del trabajo, hemos de señalar que las previsiones a corto plazo para esta amenaza no son especialmente buenas, y que es necesario seguir trabajando en fortalecer los tres pilares

básicos para atajar el problema: prevención, despliegue de herramientas, y legislación internacional.

Capítulo 2

Cripto-*ransomware*: Una visión general

En este capítulo definiremos qué es el *ransomware* y veremos una primera tipología basada en un criterio de severidad del ataque. A continuación, describiremos la evolución del mismo desde sus orígenes hasta la fecha. Estudiaremos cómo el aspecto económico ha controlado la evolución de este tipo de *malware*, en especial el modelo de negocio denominado *Ransomware-as-a-Service*, y cómo ha evolucionado hacia el cifrado como forma principal de amenaza y el uso de múltiples tipos de extorsión. Describiremos con detalle el proceso de operación del *ransomware* desde el punto de vista del atacante. Acabaremos el capítulo presentando algunos recursos para el estudio de este fenómeno.

2.1. Definición y tipología

Se denomina de forma genérica *ransomware* a cualquier tipo de *malware* (software malicioso) que restringe el acceso a los recursos del computador de su víctima y la extorsiona para que pueda recuperar el acceso a los mismos, si bien cada tipo puede variar en sus tácticas, técnicas y procedimientos (TTP). El término proviene de las palabras inglesas 'ransom' (rescate) y 'ware' (software). Si bien no está claro cuando surge exactamente [7], su origen parece estar en 2005, bien en el artículo de S. Schaibly [8] o el de J. Canavan [9]. Aunque términos más técnicos como criptovirus o extorsión criptoviral se hubiesen ajustado mejor, prevaleció el término con más gancho y mercadotecnia.

Además de la extorsión, este tipo de ataque puede causar actualmente caídas de los sistemas, pérdidas y brechas de datos, robo de propiedad intelectual y mala reputación de las compañías, que incluso puede extenderse

entre empresas, como ocurrió en el caso del ataque a Colonial Pipeline [10].

Si utilizamos el criterio de severidad del ataque, se establecen cuatro tipos básicos de *ransomware* [11, 12, 13]:

- *Bloqueo de dispositivos* - Bloquea funciones básicas del dispositivo de la víctima de forma que este no sea accesible para el usuario hasta que pague el rescate. Se utilizan diferentes mecanismos suministrados por el sistema pero no el cifrado, por lo que no suelen destruir archivos críticos. Por ejemplo, se puede denegar acceso al escritorio mientras el teclado y el ratón quedan parcialmente habilitados de forma que se puede interaccionar con la ventana donde se solicita el rescate. W32.Rasith o Android.Lockdroid.H se encuentran dentro de este tipo.

- *Cifrado de archivos* - Se caracteriza por cifrar archivos de valor para el usuario (documentos, imágenes, vídeos, etc.) de forma que estos no serán accesibles por la víctima hasta que pague el rescate solicitado. Después del pago, se le suministra la clave de descifrado de dichos archivos. En esta categoría se encuentran WannaCry, Petya, Ryuk, etc., que veremos a lo largo de este capítulo.

- *Malware de intimidación* (*Scareware*) - Este tipo utiliza tácticas para asustar a la presunta víctima informándole de que sus archivos han sido cifrados o bloqueados y solicitando el rescate, pero en realidad no se han modificado. Algunos ejemplos de este tipo son SpySheriff, Antivirus360 y DriveCleaner [14].

- *Exfiltración de datos* (*Leakware* o *Doxware*) - Recolecta información sensible de la máquina de la víctima y chantajea a esta para que pague el rescate. A diferencia del cripto-*ransomware*, la víctima puede seguir accediendo a sus datos. En la Sección 2.4 veremos cómo parte de esta funcionalidad se ha incorporado al *ransomware* actual de cifrado. Algunos ejemplos de este tipo son Ransoc y Chimera [15].

De todos estos tipos, la forma que prevalece actualmente es el cripto-*ransomware*, debido al éxito en alcanzar sus dos principales objetivos: supervivencia y reversibilidad [16]. La supervivencia asegura que el atacante mantiene el control sobre los recursos secuestrados de forma que, aunque el *malware* sea eliminado, el acceso a los mismos por parte de la víctima es irreversible. Por otro lado, el atacante debe tener un control reversible sobre el cifrado realizado de cara a mantener, en general, el incentivo en la víctima de que puede recuperar sus datos tras el pago. De aquí que el cifrado se haya convertido en la herramienta idónea para este tipo de ataque.

Esta forma de *ransomware* ha conseguido crear un negocio multimillonario [17]. Otros factores que han influido en esta hegemonía han sido la ausencia de factores de mitigación, como las copias de seguridad y de mecanismos de detección para todas las variantes. Razones por las cuales, la

víctima en muchos casos no tiene más remedio que pagar para recuperar la información. Este es el tipo en el que nos centramos a lo largo de este trabajo. Otros aspectos relacionados con el *ransomware* no cubiertos en este capítulo, como los económicos o sociológicos, se pueden encontrar en la monografía [18].

2.2. Evolución del *ransomware*

En este apartado vamos a describir la evolución histórica de *ransomware*, desde la aparición de la primera muestra hasta nuestros días. Esta descripción histórica está centrada en los ataques dirigidos a objetivos genéricos de individuos, equipos TI o de negocio de todo tipo de organizaciones. Este tipo de ataques son mucho más numerosos debido al alto número de víctimas potenciales y la generalidad de las herramientas de ataque.

No abordamos los ataques a sistemas de control industrial, que, por sus características, necesitan herramientas más específicas, y el número de potenciales objetivos es menor. Este menor número de víctimas puede verse compensado por el miedo de la víctima a parar los sistemas, afectar el funcionamiento de los equipos y posiblemente el daño a personas [19]. Es necesario tener presente que es una amenaza creciente desde su aparición en 2019 con Ekans [20]. Esta amenaza no solo afecta a sistemas de control SCADA (*Supervisory Control and Data Acquisition*) [21], sino que puede afectar a equipos como los usados en automoción [22, 23], sistemas eléctricos o de distribución de energía [24, 25], etc.

La Figura 2.1 da una visión general de dicha evolución en la que se muestran los hitos más relevantes, en texto azul en la parte superior de la línea temporal, que han dado al *ransomware* el poder dañino actual. Los momentos más destacables que se recogen son 1989, considerado su origen; 2005, donde aparece el *ransomware* moderno que incorpora cifrado asimétrico; 2013, donde se incorpora el pago en criptomonedas; 2015, considerado el año de explosión de esta amenaza; 2016, que pone el foco en las organizaciones; 2017, con *ransomware* patrocinados por gobiernos; 2019, que ve la incorporación de nuevas tácticas de extorsión; y 2021, que ve los inicios de ataques a infraestructuras críticas y cadenas de suministro. También la figura recoge, cuadros de color azul debajo de la línea de tiempo, ejemplos de las familias más relevantes por fecha de aparición. Son numerosos los trabajos donde se recoge la evolución del *ransomware*, entre ellos, [26, 27, 28, 29, 30].

Como indicamos en el párrafo anterior, el primer *ransomware* apareció en 1989, cuando el biólogo J.L. Popp envió 20.000 disquetes infectado a los asistentes al Congreso World Health Organitation's International (AIDS) en Estocolmo. Este virus, ahora conocido como troyano AIDS o PC Cyborg, estaba camuflado como cuestionario para determinar la probabilidad de infección del VHI. Una vez cargado en el computador, bloqueaba el acceso a

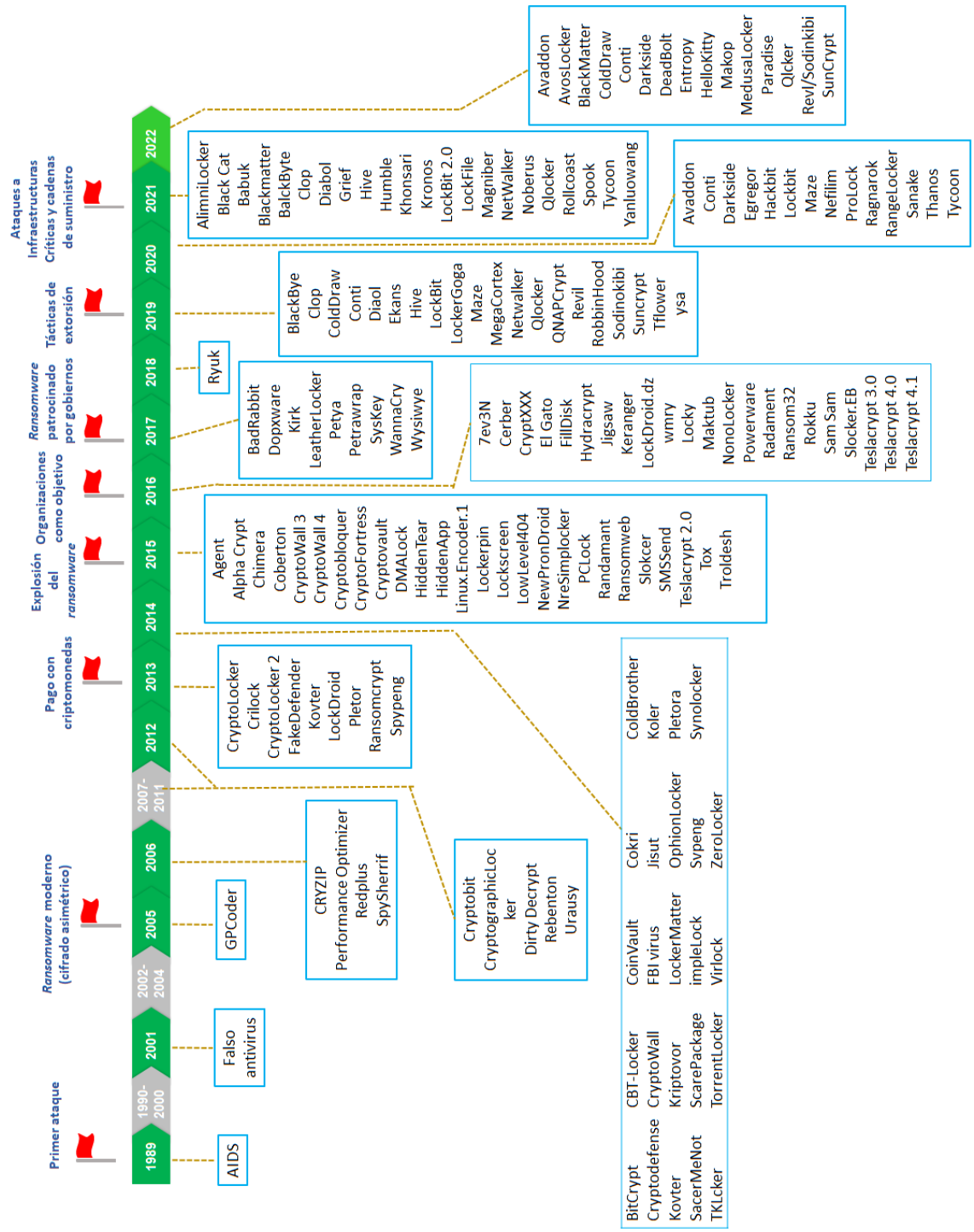


Figura 2.1: Línea temporal de los principales hitos en la evolución del *ransomware* y familias destacadas.

los archivos e informaba a sus víctimas de que debían enviar 189 dolares a un apartado de correos en Panamá para recuperar el acceso a sus archivos. En este caso, se utiliza un encriptador simétrico simple para bloquear el acceso a los archivos. El ataque no usó un método sofisticado, sino que tuvo éxito debido a la inconsciencia de las víctimas, algunas de las cuales perdieron años de trabajo.

Con el desarrollo de Internet, a finales de 2004 e inicios de 2005, GPCoder infectó, mediante un enlace a un sitio web malicioso con correos de *phishing*, diferentes sistemas utilizando un algoritmo de cifrado personalizado. Solicitaba el pago del rescate, 200 dólares, a través de Western Union o SMS premium. Afortunadamente, la clave de cifrado era débil y fácil de romper.

Con Archiveus en 2006, que cifraba solo la carpeta de 'Mis Documentos', los autores de *ransomware* se dieron cuenta de la importancia de utilizar un cifrado fuerte. Si bien fue la primera vez que se usó un código de cifrado RSA de 2.014 bits, el fallo de los autores fue no usar diferentes claves para bloquear los sistemas. Una vez descubierta su torpeza, este cayó en desuso.

Hasta este momento, el talón de Aquiles del *ransomware* eran las formas de pago usadas, ya que eran trazables. Por ello, las siguientes familias se centraron en las tarjetas de regalo para pagar el rescate. Este medio permitía que la víctima pagase el rescate simplemente acudiendo a una tienda, vendedor de videojuegos, o compañía de tarjetas de crédito. Esto es lo que hacía Locker, desde 2009 en Rusia y a partir de 2010 en el resto de mundo, y también WinLock. Locker se instalaba al visitar un sitio web malicioso o comprometido y estaba escrito generalmente en JavaScript. Su objetivo era bloquear el acceso a la funcionalidad del dispositivo con un *popup* que indicaba cómo pagar el rescate mediante una tarjeta regalo o monedero. Esta variante también se extendió a los móviles mediante la descarga de aplicaciones fuera de las fuentes oficiales. Esto se ha vuelto a ver durante la pandemia del Covid-19, donde los cibercriminales desarrollaron un rastreador Covid-19 que se tornó en *ransomware* de bloqueo [31]. La mayoría de bloqueadores se hacían pasar por algún cuerpo de seguridad del estado e indicaban a la víctima que había cometido una ilegalidad por la que debía pagar una sanción. Esta técnica tuvo poco éxito pues los archivos de la víctima no se veían comprometidos y era posible saltar la pantalla de bloqueo.

La unión de los *ransomware* de bloqueo y las criptomonedas dio lugar a la aparición de Reventon, 2012, que fue el primero en utilizar el modelo de negocio *Ransomware-as-a-Service* (RaaS - que estudiaremos en la Sección 2.3) y el pago en Bitcoin. Con él se inició la capacidad de infectar a las víctimas de forma masiva. Reventon mostraba también un mensaje fraudulento haciéndose pasar por algún cuerpo y fuerza de seguridad del estado dependiendo del país y acusándolas de cometer un acto delictivo por el que podían ser encarceladas si no pagaban la extorsión. El surgimiento de las criptomonedas, en 2009, revolucionó el negocio del *ransomware* al permitir

un pago de forma fácil y anónima.

Hasta mediados de la década de 2010, el *ransomware* estaba más centrado en PC dada la popularidad del mercado de Microsoft que ofrecía un número elevado de víctimas. A pesar de ello, se inició una ampliación de las plataformas objetivo: móviles, Linux y Mac. Concretamente, en 2012 SimpleLocker se convierte en el primer *ransomware* en cifrar archivos en la tarjeta SD de dispositivo Android, abriendo así un nuevo nicho de víctimas y ataques.

Otro hito importante es 2013, con la aparición de CryptoLocker. Este tenía la doble forma de pago a través de Bitcoin o de tarjetas monedero. Además, usaba una clave RSA de 2.048 bits. Se propagaba como un adjunto de un correo electrónico de aspecto inocuo. Ha sido una de las variantes que ha logrado recaudar una cantidad importante de dinero, 27 millones de dólares en dos meses y se estima que afectó a unas 234.000 víctimas. La lucha contra él fue un ejemplo de colaboración entre fuerzas de seguridad y compañías de seguridad privadas, que culminó con la identificación del responsable, que nunca fue detenido.

En 2015, LockerPin, que también tenía como objetivos dispositivos Android, tenía como fin bloquear completamente el acceso del usuario cambiando el PIN del dispositivo, en lugar de cifrar archivos. En este año se libera Linux.Encoder.1, el primer *ransomware* para Linux.

Desde el punto de vista de la defensa frente a esta amenaza, en 2015 se publica el código fuente de Hidden Tear por un grupo de investigadores turcos, con la intención de que los equipos de seguridad conociesen su funcionamiento [32]. Lamentablemente, esto también sirvió para que los atacantes realizasen mejoras y lanzasen nuevos ataques. En 2020, algunas nuevas variantes de *ransomware* contenían trazas de este código como Chaos Ransomware Builders [33], un software con interfaz gráfica para crear *ransomware* de acuerdo a opciones determinadas. Por ejemplo, estas similitudes aparecen en su versión V3 tanto en el código para generar las claves como en la forma de realizar el cifrado AES, que son prácticamente idénticos. Algo parecido ocurre con Bagli.

El cambio de evolución culmina con variantes capaces de atacar sistemas Windows, Linux y Mac sin códigos diferenciados para cada plataforma. Ransom32, que apareció en 2016, es una variante del modelo RaaS desarrollada en JavaScript, y que permite operar en la mayoría de las plataformas.

En la siguiente etapa se incrementa la sofisticación de las técnicas de ataque, además de expandirse a nivel global. En 2016, Petya fue la primera variante en sobre-escribir el *Master Boot Record* (MBR) en lugar de cifrar archivos individuales, bloqueando el acceso al disco de una forma más rápida que otras técnicas. Este mismo año aparece Locky, que enviaba hasta 500.000 correos *phishing* por día para su propagación [7]. Otras familias que también aparecieron este año fueron TeslaCrypt, Jigsaw y Cerber. Todas ellas tienen en común que fueron utilizadas para ataques automatizados contra

una única máquina y la entrega se hacía mediante correos *phishing*, kits de explotación o anuncios maliciosos en sitios web. Hubo tantas variantes que algunos designaron 2016 como el 'año del *ransomware*', si bien solo fue el inicio de su auge.

Otra de estas familias fue SamSam (que toma su nombre de un pueblo del noroeste de Irán) que, en lugar de usar un kit de explotación o suplantación de identidad, explotaba las vulnerabilidades en JBoss (Servidor abierto de aplicaciones Java) y buscaba servidores RDP (*Remote Desktop Protocol*) expuestos para lanzar ataques de contraseña de fuerza bruta para obtener acceso. En lugar de instalarse en una máquina, como hacen las familias contemporáneas, utilizaba diferentes herramientas y *exploits* para expandirse por la red de la víctima e instalarse en todas las máquinas posibles. Estuvo operativo hasta 2018, fecha en la cual el Departamento de Justicia de Estados Unidos acusó a sus autores y se detuvieron los ataques.

Pocos meses después, Zcryptor, que combina características de un gusano, crea un ataque denominado *cryptoworm* o *ransomworm*, que es especialmente dañino ya que puede inutilizar un sistema completo al duplicarse discretamente a través de la red.

WannaCry, 2017, se caracteriza por ser uno de los ataques más grandes de la historia en cuanto a máquinas y sectores empresariales afectados. Utilizaba el *exploit* EternalBlue [34] que aprovechaba vulnerabilidades del protocolo *Server Message Block* (SMB) de Microsoft. Este pedía un rescate de 300 dolares en Bitcoins pero, como la clave de cifrado no estaba disponible, el millar de víctimas que pagaron no pudieron recuperar sus archivos. A finales de 2017, Estados Unidos y Reino Unido atribuyeron este *ransomware* a Corea del Norte. Dos meses después de WannaCry, se produjo el ataque de NotPetya (sucesor de Petya), que además de cifrar archivos hacía lo mismo con el MBR, lo que suponía que aún disponiendo de la clave de desencriptado la víctima no podía recuperar los archivos. Este se distribuyó mediante una versión troyanizada del software de actualizaciones M.E.Doc, necesario para cualquier empresa que quiera hacer negocios en Ucrania. Los atacantes comprometieron el servidor de actualizaciones de dicho software para insertar dicho programa malicioso. Pocos meses después se atribuyó el ataque a Rusia, según Estados Unidos, Canadá y Australia.

Este periodo en la evolución se caracteriza también por el desarrollo de nuevas variantes de *ransomware* existente en lugar de desarrollar nuevas familias. Por ejemplo, en 2017 aparece Goldeneye, que es una variante de Petya, y hermanastro de WannaCry, que es más peligroso al solventar problemas de cifrado de sus predecesores.

El efecto de Wannacry obligó a las empresas a reforzar sus defensas. En 2018, el desarrollo de *ransomware* parecía haberse ralentizado a medida que los mineros de criptomonedas superaban al *ransomware* en términos de detección y disminuían las nuevas familias de *ransomware*. Sin embargo, pronto se supo que esta tendencia solo significó un importante punto de

inflexión para el *ransomware*.

Si bien desde 2016 existía *ransomware* que usaba el modelo RaaS, como Stampado, Goliath y Locky, los operadores solo suministraban el ejecutable, por lo que toda la operativa recaía en el afiliado que lo adquiría. Este aspecto cambió con GandCrab (2018), que, mediante la creación de ofertas llave en mano, ofrecía un portal para los afiliados que les permitía seguir el ataque y gestionar el pago. Este mismo año apareció Ryuk, que estableció un nuevo estándar al ser el primero en operar como un ataque dirigido. Usaba Tribot y PowrShell Empire para propagarse e instalarse. Además, usaba PowerShell y el instrumental de gestión de Windows (WMI) para realizar movimientos laterales.

En 2019 aparecen nuevos ataques dirigidos que serán la norma en 2020. A partir de este momento aparecen dos nuevos aspectos que hacen más peligroso y destructivo al *ransomware*: la doble extorsión y establecer a las organizaciones como objetivo en lugar de a los individuos. Dado que la víctima podría no pagar el rescate, la doble extorsión cifra y roba los archivos de la víctima, de forma que si esta no paga se la puede amenazar con hacer públicos los datos o venderlos en el mercado negro. Ejemplo de ello es Maze, 2019, que introduce la revolucionaria novedad de lanzar un sitio de filtraciones y, por tanto, considerar además el *ransomware* como un ataque de robo de datos (doble extorsión). Su sucesor Egregor (2020) incorpora un servicio de soporte a las víctimas para proteger sus sistemas, si estas pagaban. Por otro lado, para maximizar el beneficio, los atacantes seleccionan víctimas en grandes organizaciones bien conocidas donde la cuantía del rescate puede ser mayor, lo que no supone que los ataques individualizados desaparezcan. En la Figura 2.2 [35] se pueden apreciar las familias de *ransomware* que utilizan la doble extorsión en el periodo comprendido entre noviembre de 2019 y octubre de 2020.

La pandemia del Covid-19 ha sido un acontecimiento que ha disparado la explosión de la doble extorsión y del RaaS. Durante este periodo aumentó el número de ataques a hospitales, organizaciones gubernamentales y universidades, con un 72 % de nuevas muestras y 77 nuevas campañas durante los primeros meses [36]. Los agentes de ataque aprovecharon el acontecimiento para ejecutar más ataques y más rápidos (acortando el periodo entre la infección y la activación), reclutar colaboradores para maximizar el impacto y ofrecer RaaS en la *Dark Web* [37]. Este incremento fue generalizado para todo tipo de *malware*. En el caso del *ransomware*, podemos ver en la Figura 2.3 [38] cómo la media y mediana de los pagos realizados por este ataque se dispara a partir del primer trimestre de 2020. En este sentido, hemos de destacar campañas que aprovecharon el tema del coronavirus, como Ransomware-GVZ [39], NetWalker y CoViper. Una de las razones de la proliferación es el aumento del tele-trabajo, que permitió explotar a mayor nivel las vulnerabilidades del protocolo de escritorio remoto. También, indicar que, en media, la cantidad solicitada en los rescates en estos ataque se

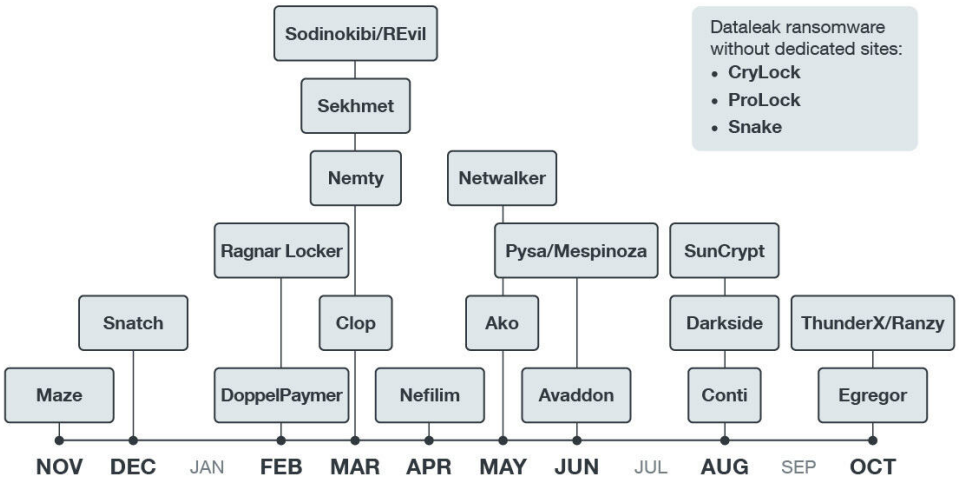


Figura 2.2: Familias de *ransomware* que usan doble extorsión aparecidas entre nov-2019 y oct-2020 [35].

incrementó en un 60 %, y que gran parte de los casos se pagaba por la dependencia del funcionamiento de los sistemas por parte de las organizaciones [40].

Frente al ataque a sistemas generales de computación, en 2019 apareció QLocker, cuyo objetivo eran sistemas de almacenamiento conectados a la red (NAS) explotando la vulnerabilidad CVE-2021-28799 (véase Sección 2.4.3 relativa a credenciales embebidas en el código). Este *ransomware* ha vuelto a actuar a principios de 2022 [41].

Conti, que aparece en 2019, es uno de los *ransomware* más prolijos, con más de 450 víctimas conocidas y está considerado pariente de Ryuk (2018-2021), ya que ambos están operados por el mismo subgrupo Wizard Spider, y reutiliza código del mismo. Se caracteriza por ser especialmente duro, ya que durante la pandemia persiguió organizaciones de salud a la espera de que se vieran obligadas a pagar dada la situación sanitaria. Después del ataque al *Health Service Executive* de Irlanda, el grupo se vio obligado a entregar la clave de descifrado por temor a represalias del gobierno. En 2020, el troyano Trickbot comenzó a usarse junto con Emotet y Ryuk, y finalmente, Trickbot fue absorbido por Conti [42]. Conti es multihebrado, lo que hace que su ejecución sea más rápida que otras familias.

Lockbit, que apareció en 2019, seguiría en 2020 reportando más de 9.000 incidentes. Dada la gran cantidad de afiliados, es difícil de establecer cómo opera pues unos utilizan campañas de suplantación de identidad para obtener acceso mientras que otros usan servidores RDP expuestos, o incluso explotan vulnerabilidades de VPN u otras infraestructuras de la nube como SonicWall, Microsoft SharePoint, etc. (como veremos en el Apartado 2.4.3).

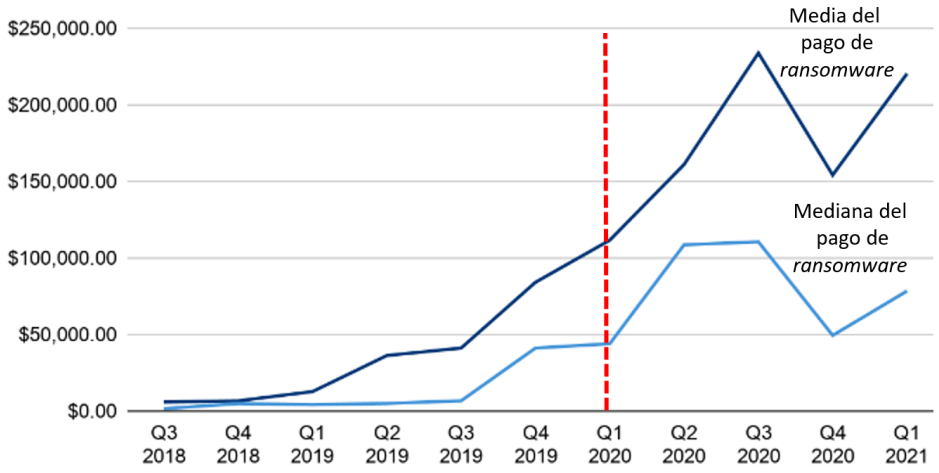


Figura 2.3: Evolución de la media y mediana de las cantidades pagadas como rescate entre 2018-2021 (modificada de [38]).

Después de la desaparición de REvil, reapareció como LockBit 2.0, con la esperanza de captar a los afiliados de REvil. Una de las características del mismo es que automatiza el proceso de despliegue para los afiliados, que solo tienen que tomar el control del Directorio Activo (AD) y ejecutar un *script*. El resto lo hace el programa.

En mayo de 2021, la variante RaaS de REvil se utilizó para realizar uno de los ataque más grandes de la historia, donde los atacantes demandaron al proveedor de servicios gestionados Kaseya la cantidad de 70 millones de dólares por desbloquear más de 1 millón de dispositivos. En diciembre de este mismo año aparece Konsari, el primero en explotar la reciente notificada vulnerabilidad Log4Shell (CVE-2021-4428).

Grief (2021) fue el sucesor de DoppelPaymer. Se desplegaba en un entorno ya comprometido por Dridex y los actores de ataque realizan la post-explotación usando Cobal Strike. Esta familia está ofuscada y emplea técnicas anti-análisis que incluyen *hashing* API, manipulación de los vectores de excepción (VEH), técnica Puerta del Cielo [43] y cifrado de datos relevantes mediante RC4. Grief se ejecuta con parámetros específicos calculados en base al entorno de la víctima y falla si estos no se suministran o son incorrectos. Además desactiva Windows Defender y borra las copias sombra con `vssadmin` y `Diskshadow` [44].

Otro hito importante en la evolución del *ransomware* es la aparición en 2021 de los ataques a Infraestructuras Críticas (IC). El grupo Sabbath que opera varios *ransomware*, entre ellos Rollcoast, se dio a conocer con la exposición de datos y la extorsión de varios distritos escolares de Estados Unidos utilizando redes sociales como Reddit y Twitter. Desde julio comenzó

a utilizar Themida para empaquetar las muestras del *ransomware* y prevenir la detección. Está diseñado para ejecutarse en memoria y comprueba el idioma del sistema (tiene una lista de exclusión de 40 lenguas). Se han detectado similitudes con el *ransomware* Tycoon.

En el reciente trabajo sobre ataques a IC [45], se muestra un cambio en este tipo de incidentes y toma como punto de inflexión la pandemia derivada del Covid-19. Se observa un cambio en el tipo de organizaciones afectadas, donde se desconcentran los ataques en organizaciones gubernamentales (pasando del 35,9 % al 16,6 %) para distribuirse entre todos los sectores, pasando al segundo lugar los relacionados con salud. El estudio también refleja cómo el pago del rescate, dada la presión y rastreo de Bitcoin, se ha desplazado al uso de medios convencionales frente a los realizados en criptomonedas.

Otro aspecto importante de 2021 es la publicación por parte de uno de los desarrolladores del *ransomware* Babuk (o Babyk), de 17 años y enfermo avanzado de cáncer, en un foro de *hackers* el código fuente completo [46]. El grupo que opera dicha familia anunció también un cambio en la forma de funcionar según el cual no iban a cifrar los datos sino solo robarlos. Si, tras la notificación de robo, la víctima no se pone en contacto con ellos, publicarán los datos [47].

En 2022, los actores de ataque UNC2596 (conocido como Cuba) operan el *ransomware* ColdDraw Cuba, cuyo objetivo ha incluido proveedores de servicios públicos, agencias gubernamentales y organizaciones que apoyan a entidades sin ánimo de lucro y de atención sanitaria. Se sigue el sistema de cambio de marca, como ocurre con Entropía, que tiene muchas similitudes con el *malware* de propósito general Dridex, o el de Sabbath con Arcane.

A modo de resumen, indicar que desde la aparición de la primera muestra de *ransomware* hasta la fecha, el crecimiento de ataques ha sido enorme. Entre los factores que han influido en ello podemos encontrar los mecanismos de pago seudo-anónimos [48], las redes anónimas [49], *ransomware-as-a-service* [50, 51] y *botnets* [52], que hace que los operadores de estos sistemas funcionen con impunidad.

Destacar que en el primer semestre de 2021 podíamos encontrar 130 familias activas agrupadas en 30.000 clústeres de *malware* (se denomina *clúster* a una agrupación de muestras de comportamiento similar). La más activa correspondía a GandCrab, con 6.000 clústeres, seguida de Babuk, Cerber, Matsnu, Congur, Locky, TeslaCrypt, Rkor y Reveon, como puede verse en la Figura 2.4 [53].

Un dato interesante relativo a la evolución es ver cuán frescas son las muestras utilizadas en los ataques. VirusTotal, en su informe de 2021, muestra cómo hay una correlación entre los ejemplares ya examinados en la plataforma y los de primera aparición (ver Figura 2.5). Esto podría indicar que los atacantes preparan nuevas muestras para la mayoría de sus ataques, salvo el pico observado en el primer cuatrimestre de 2021, donde la actividad parece mostrar que se están reutilizando ejemplares previos. Estos datos ex-

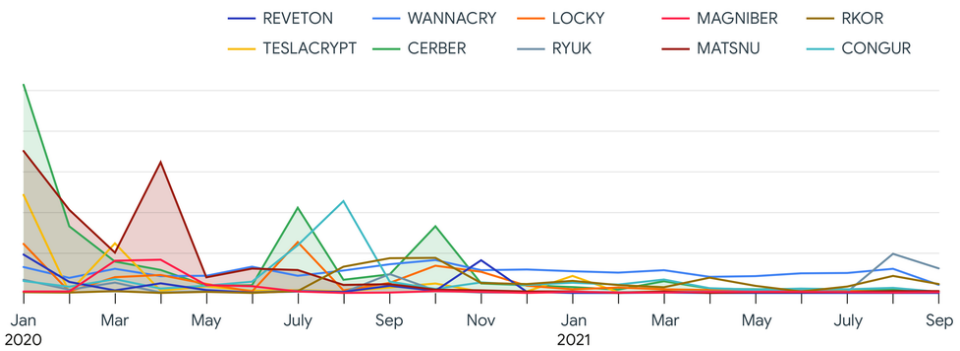


Figura 2.4: Actividad de las 10 familias más activas de *ransomware* [53].

perimentales concuerdan con el análisis realizado en [54], que establece una fórmula para predecir la probabilidad de nuevos ataques y el total de los mismos. La probabilidad de un nuevo ataque se obtiene de multiplicar por 100 el cociente entre nuevos ataques de *ransomware* y nuevos ataques de *malware*. La probabilidad del total de ataques *ransomware* se define de la misma manera respecto del total de ataques. Los datos analizados entre 2016 y 2020 indican que es mayor la probabilidad de un ataque de *ransomware* ya visto que uno nuevo.

El estudio de familias de *ransomware* muestra que, si bien los vectores de ataque del *ransomware* suelen coincidir con los del *malware* general [55, 56], de acuerdo con [57] se puede ver una tendencia reciente a utilizar algunos de ellos de forma prominente:

- Ataque a la cadena de suministro (*Supply Chain Attack*) - En este ataque se trata de extender el radio de afectación del *ransomware* a toda la cadena de suministro de software mediante la inserción de código maligno dentro de un componente confiable de dicha cadena [58, 59].
- Uso del modelo RaaS para lanzar y mantener una campaña de ataque [12].
- Ataque de sistemas sin parchear - Si bien aparecen nuevos *ransomware* que explotan vulnerabilidades de día cero (es decir, vulnerabilidades recién descubiertas y no parcheadas), se siguen explotando vulnerabilidades conocidas en sistemas no parcheados.
- *Phishing* - Aunque no son la causa principal, los correos de *phishing* son frecuentes en los ataques de *ransomware*.

Es muy probable que esta amenaza siga evolucionado en un futuro próximo de diferentes maneras. Algunas predicciones en la evolución y lucha contra esta amenaza son:

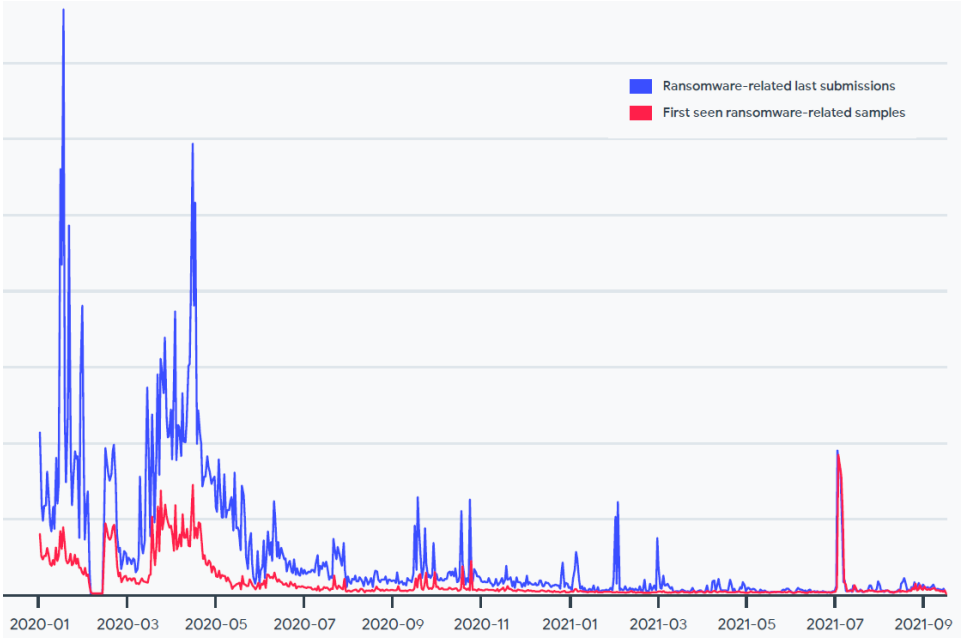


Figura 2.5: Distribución temporal de muestras de *ransomware* nuevas y ya analizadas por VirusTotal [53].

- Los gobiernos se involucrarán más especialmente en el desarrollo de legislación contra el pago de la extorsión de cara a cortar la realimentación de los ataques [60].

- Se prevén nuevos modelos de extorsión que incluyen múltiples niveles de extorsión, personalización, evolución hacia nuevos bienes protegidos como IoT [61]. Respecto a la múltiple extorsión, indicar que según un informe de la consultora Unit 42 [62], algunas familias de *ransomware* han evolucionado desde la doble extorsión (cifrado y robo de datos) hacia una cuádruple extorsión: cifrado (con objeto de que se pague el recate), robo de datos (los datos se ex-filtran de forma que, si no se paga el rescate, se hacen públicos o se venden), denegación de servicio (DoS sobre servidores web públicos de la víctima) [63], acoso (se contacta con clientes, empleados, socios y medios, para notificar el ataque) u otros más insólitos como el ocurrido a Nvidia donde se solicitaba por el grupo Lapsus$ el código fuente y *firmware* de las tarjetas de vídeo del citado fabricante [64]. Es más, en enero de 2022, el grupo de Suncrypt ha llegado incluso a llamar por teléfono a la víctima para presionarla y que realice el pago [65].

- Incremento del cifrado intermitente, donde se cifra solo una parte de

cada archivo, haciéndolo parecer corrupto para evadir los mecanismos de detección [66].

Como los *ransomware* actuales utilizan criptomonedas como forma de pago del rescate de cara a mantener el anonimato del atacante, existen algunos trabajos, como [67], que permiten en sistemas de pago seudo-anónimos, como es Bitcoin, identificar o diferenciar los actores que interviene en transacciones de *ransomware* de las de otros tipos en base a algoritmos de agrupación y métodos de aprendizaje máquina (ML) de grafos supervisados. En una línea similar, estudiando diez familias de *ransomware* que utilizan Bitcoin, [48] permite recolectar, identificar y analizar las direcciones de los cibercriminales, utilizando técnicas de agrupación e información obtenida de la *Blockchain*. El estudio incluye el cifrado, el proceso de infección y la ejecución de dichas familias.

A modo de resumen de la situación actual, el último informe de ciberinteligencia de S21Sec [68] que abarca la segunda mitad de 2021, permite tener una visión de conjunto. En concreto, como se muestra en la Figura 2.6 [68], solo tres grupos de *ransomware* fueron responsables del mayor porcentaje de víctimas: Lockbit, Conti y Pisa. Del mismo informe, se puede ver en la Figura 2.7 la afectación por sectores económicos. Es de especial relevancia el incremento de los ataques con ex-filtración de datos, que ha sido del 82 % en 2021 respecto al año anterior, como puede verse en la Figura 2.8 [69]. En cuanto a los países principalmente afectados, encontramos a Estados Unidos (757), Reino Unido (85), Canadá (82), Alemania (78) y España (32).

Como hemos visto, el *ransomware* no solo es peligroso, sino una forma de crimen basada en criptomonedas siempre dinámica y cambiante. Una tendencia que se está observando es el creciente uso de Monero debido a la mayor anonimato que ofrece esta criptomoneda. Esto provocará el cambio de las tácticas de investigación.

Para finalizar el apartado y ver un ejemplo de la gravedad del problema, sírvase indicar cómo en los cinco primeros meses de este año las extorsiones por *ransomware* han recaudado más de 15 millones de dólares. Entre las familias más activas encontramos a Karakurt, BlackMatter, y Conti. Los datos se han obtenido de [70], que publica periódicamente (semanas, meses y años) su evolución.

Respecto a la plataforma objetivo, indicar que según los datos ofrecidos por Purplesec [71], en 2021, la plataforma preferida fue Windows con un 66 % de los ataques, seguida por MacOS con un 7 %, Android con el 5 % y finalmente iOS con el 3 %.

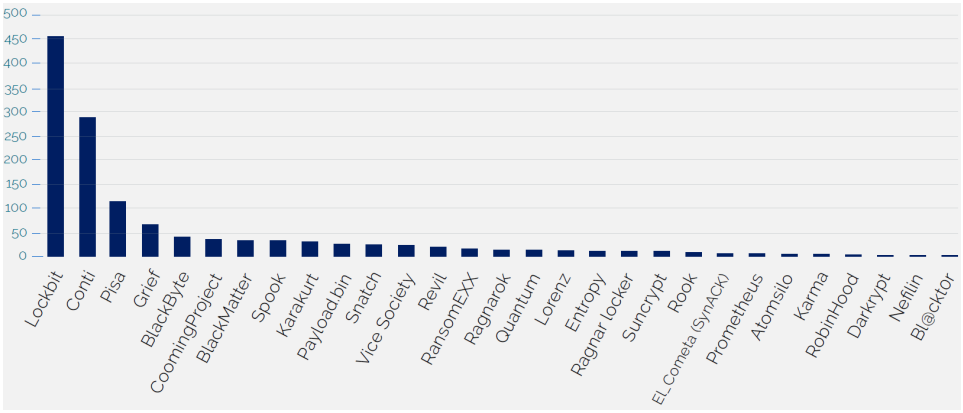


Figura 2.6: Número de ataques por grupos de *ransomware* en el segundo semestre de 2021 [68].

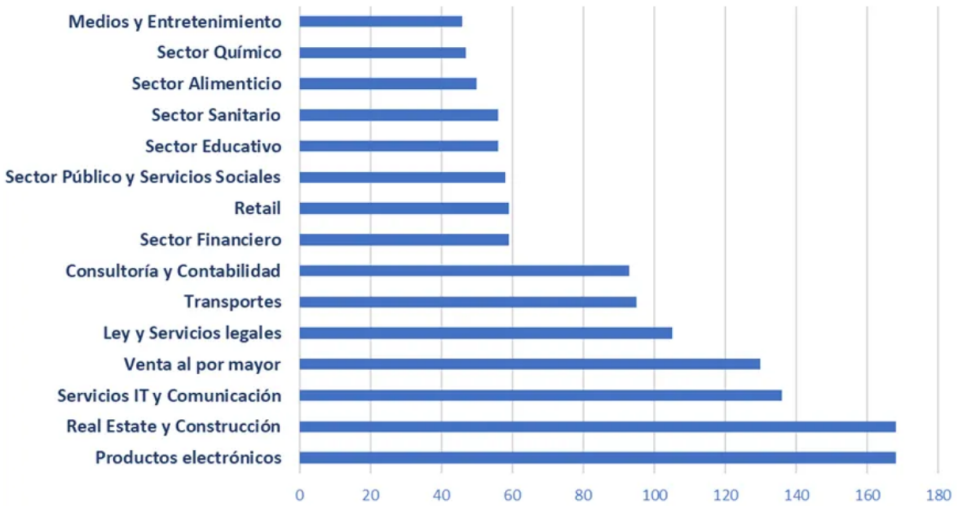


Figura 2.7: Afectación de *ransomware* por sectores en el segundo semestre de 2021 [68].

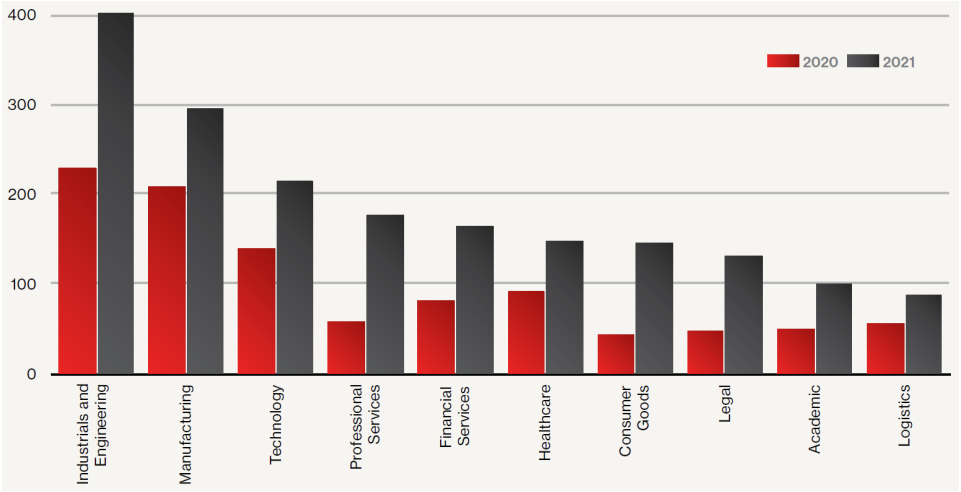


Figura 2.8: Número de ataques de *ransomware* por sectores con robo de datos 2020-2021 (modificada de [69]).

2.3. Ecosistemas RaaS, criptomonedas y extorsión

Como hemos visto, RaaS se aplica al modelo de negocio ilegal donde se venden/alquilan los elementos para el ataque de *ransomware* a compradores, denominados afiliados, para que lo exploten [72, 73]. Este modelo ha supuesto un enorme auge en la proliferación de este tipo de *malware*, dado que es fácil de usar por una amplia variedad de actores incluidos los que no tienen muchos conocimientos técnicos.

El modelo RaaS se basa en el modelo *Software-as-a-Service*, en el que puede accederse al software en línea mediante suscripción. Sin embargo, este modelo tiene su propia evolución completamente funcional e independiente que prospera en la clandestinidad con actores clave como los operadores, que desarrollan/venden *ransomware* y que suelen estar generalmente organizados en grupos con roles concretos como líder, desarrolladores y operadores de infraestructura y sistema. En este modelo, algunas funciones o herramientas son a su vez compradas o subcontratadas, como por ejemplo, el Acceso como Servicio (AaaS), que proporciona varios medios de acceso a víctimas específicas, o equipos potentes de pruebas de penetración, que a su vez pueden participar como afiliados del RaaS. Los afiliados pueden operar de forma independiente o como miembros de los grupos organizados.

Más concretamente, los operadores se encargan de reclutar afiliados en foros, dar acceso a los mismos a los paquetes para construir el *ransomware*, crear un panel de orden y control para que el afiliado mantenga el seguimiento del paquete, establecer un portal de pago para la víctima, asistir a los afiliados en las negociaciones con la víctima y administrar un sitio dedicado

de ex-filtración. Por su parte, los afiliados pagan por el servicio y se ocupan de las víctimas objetivo: efectuar la demanda del rescate, configurar los mensajes posteriores al ataque, comprometer los bienes de la víctima, intentar maximizar la infección mediante técnicas que usan los recursos del sistema (*living-off-the-land*) [74], ejecutar el *ransomware*, comunicarse con la víctima mediante portales u otros medios, y gestionar las claves de descifrado.

Con la mercantilización del *ransomware* aparecen nuevos actores. De parte de la víctima, surge el negociador, encargado de tratar con los actores de ataque y gestionar el pago, especialmente si es cuantioso, caso en el que negocia la cantidad. Algunos trabajos abordan la supresión de esta figura y proponen sustituirla por contratos inteligentes [75]. Por el lado del atacante, el Agente de Acceso Inicial (IAB), encargado de buscar vulnerabilidades en Internet. Algunos de ellos están especializados en la reutilización de credenciales robadas; otros intentan obtenerlas mediante fuerza bruta o ataques de diccionario; pero también los hay especializados en explotar diferentes vulnerabilidades. Su papel en el ataque es ser el punto de entrada y vender el acceso a los actores de ataque. Estos se pueden encontrar en los foros clandestinos con el eufemismo de 'pentester'. También, del lado del atacante, encontramos a quienes realizan el lavado del dinero, encargados de blanquear cientos de millones de dólares. Por último, existe la figura de Agente Contable (OTC), que suelen ser individuos o compañías que tienen gran cantidad de criptomonedas y que, a cambio de una comisión, permiten al atacante cambiar la recaudación de criptomoneda a metálico.

El esquema de funcionamiento puede verse en la Figura 2.9, donde las etapas del modelo de explotación del RaaS son:

1. El desarrollador del *ransomware* crea un *exploit* personalizado que es licenciado/compartido con un afiliado.

2. El afiliado actualiza el sitio de hospedaje con el código del *exploit*.

3. El afiliado establece un objetivo y vector de ataque para entregar el *exploit* a la víctima.

4. La víctima accede al sitio web donde se aloja el mecanismo de explotación; por ejemplo, pinchando en un enlace de correo malicioso.

5. El *ransomware* se descarga del sitio web y se ejecuta en el computador víctima.

6. El *ransomware* conecta con el servidor de control y se descarga la llave para el cifrado.

7. Este cifra los archivos de la víctima pero, además, puede identificar objetivos adicionales en su red, modificar la configuración para asegurarse la persistencia, destruir las copias de seguridad y cubrir su rastro.

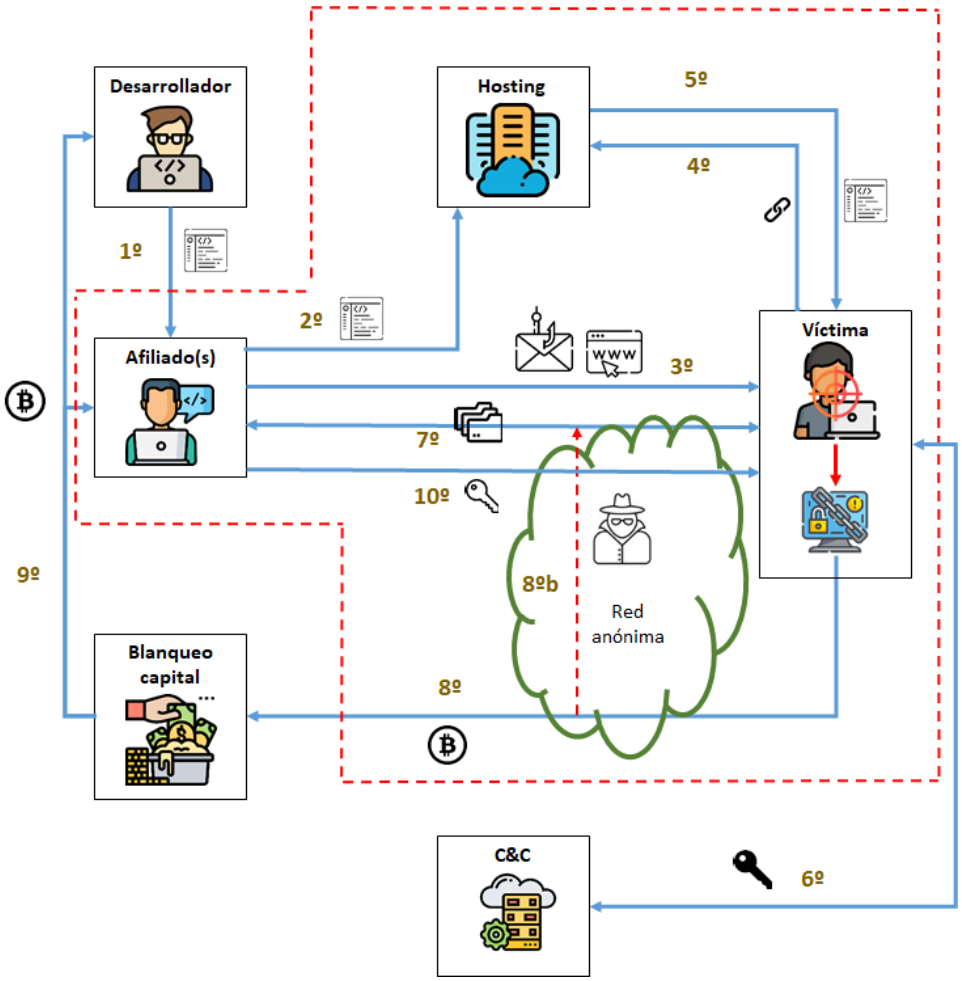


Figura 2.9: Etapas del modelo de negocio *Ransomware-as-a-Service*.

8. Se muestra el mensaje de extorsión a la víctima y se le indica dónde y cómo pagar, normalmente por medios anónimos. En [76] podemos ver los detalles del modelo de negociación de Conti.

9. Otro ciberdelincuente blanqueará el dinero moviéndolo a través de múltiples transferencias para dificultar la identidad del afiliado y del desarrollador.

10. El afiliado puede enviar la clave de descifrado a las víctima una vez que ha recibido el pago. También podría no enviarle la clave o exigir nuevas demandas.

El funcionamiento de un ataque de *ransomware* que no utilice el modelo RaaS es la parte de la Figura 2.9 que está en el área delimitada por la

línea roja discontinua (pasos 2º a 8ºb). El esquema descrito en la figura está simplificado, para tener una visión más realista del modelo. En [77] podemos ver el modelo de afiliación de EVil Corp.

El modelo de suscripción puede tener cuatro formas [78]: pago mensual, programa de cooperación (además de la suscripción hay un modelo de compartición de beneficios), pago por licencias de un uso, o solo reparto de beneficios. Independientemente del modelo, algunas empresas de RaaS lo implementan de forma fácil: solo hay que ir a la DarkWeb, crear una cuenta e iniciar una sesión, elegir el modelo, realizar el pago y finalmente distribuir el *malware* y esperar.

El producto adquirido puede incluir el código del *ransomware*, las claves de cifrado y descifrado, correos para el *phishing* de inicio del ataque, y documentación y soporte. También puede incluir facturación, seguimiento, actualizaciones, previsiones de gastos e ingresos.

Para el desarrollador, este modelo, además de rentable, le da más seguridad pues hace que sea mucho más complejo el proceso para su identificación. Para los afiliados, reduce los costes de desarrollo además de seguir dándoles beneficios ya que los rescates son cada vez más cuantiosos. Estas razones provocan que se desarrollen nuevas familias de *ransomware* debido a las ganancias generadas. Por esta razón, algunos gobiernos están legislando para penalizar el pago del rescate.

Desde el punto de vista tecnológico, las herramientas de construcción de *ransomware* han facilitado el desarrollo de este modelo. Un ejemplo de ello es Thanos [79], que permite a los operadores la posibilidad de crear clientes de *ransomware* con diferentes opciones, como muestra la Figura 2.10 [79]. Esta herramienta configura automáticamente algunas opciones pero el operador debe ajustar otras como la dirección de Bitcoins para el pago, o la nota de extorsión. Entre las 43 opciones de configuración tiene algunas como: ayuda para evitar la interfaz anti-*malware* de Windows (AMSI), deshabilitar el uso de gestor de tareas, establecer la ayuda a las funciones criptográficas, deshabilitar Windows Defender, vaciar la papelera de reciclaje, subir datos a un servidor FTP, establecer el mecanismo de propagación por la red, etc. Configuradas las opciones, Thanos genera un ejecutable .NET. Este ejecutable se puede ofuscar con una de las dos opciones que suministra: una versión rota de herramienta SmartAssemble de Redgate, o crea un instalador Inno Setup embebido en el cliente. Ejemplos de *ransomware* construido con esta herramienta son Hakbit (2020) y Narumi (2022). Otro ejemplo de constructor es *Chaos Ransomware Builder* [80], basado en el *ransomware* abierto Hidden Tear y cuyo ejecutable está disponible en el Foro Vx-Underground [81].

Como se indicaba antes, algunos autores exploran la posibilidad de que nuevas tecnologías como los contratos inteligentes, junto con propuestas como el Sistema de Archivos Interplanetario (IPFS), podrían usarse para el lanzamiento de campañas RaaS que se sirvan de ellas para ocultar su identidad y permanecer un mayor tiempo fuera de línea durante el ataque [51].

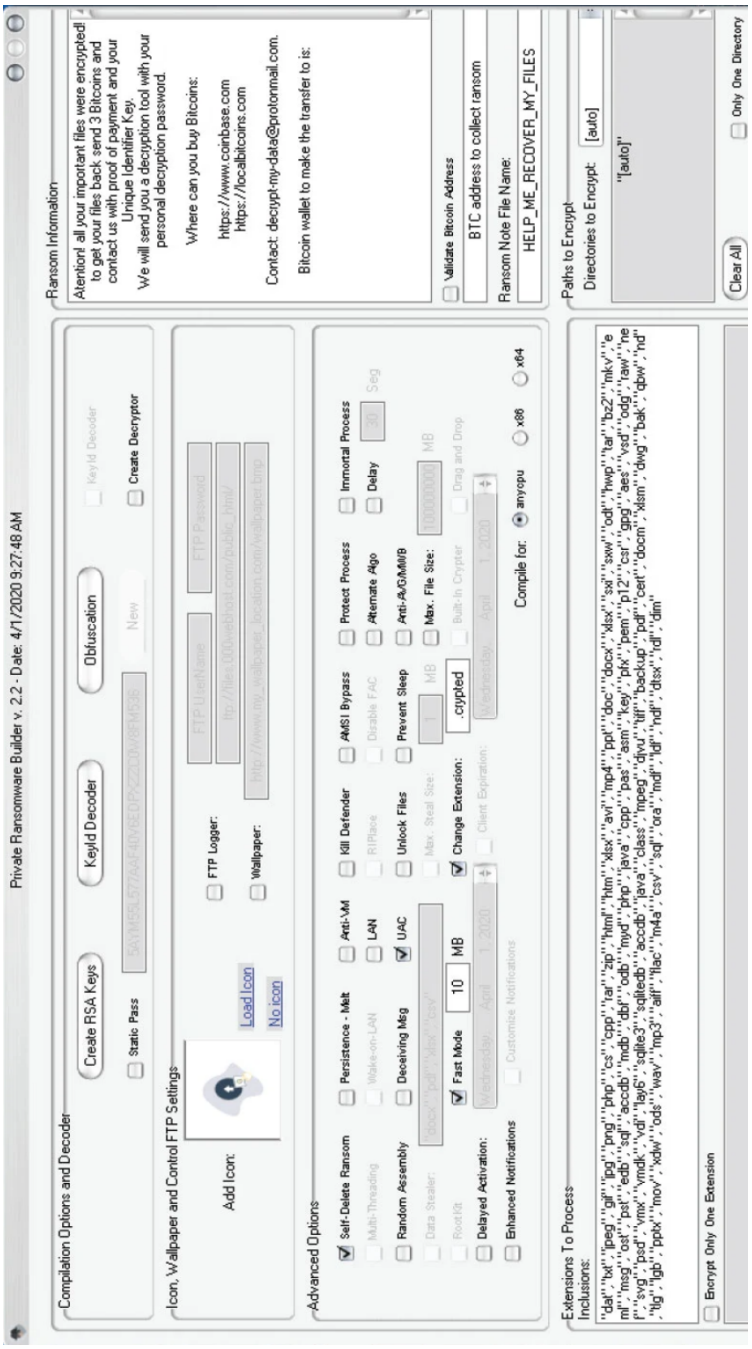


Figura 2.10: Ventana de configuración del constructor de *ransomware* Thanos [79].

Los datos de 2021 [82] muestran para los *ransomware* operados por grupos de amenazas persistentes avanzadas (APT) un crecimiento de un 7 % desde el bienio 2018-20. Estos ataques están ligados a países como China (7), Rusia (12), EEUU (2), Corea del Note (2), Ucrania (1) e Iran (1), además de 14 APTs disponibles para contratar.

Las implicaciones negativas de este modelo para las empresas son [83]:

- Aumento del tiempo en el que los equipos están fuera de servicio. Esta inaccesibilidad tiene un impacto negativo en los clientes en cuanto a pérdida de credibilidad y disponibilidad.
- Muchos clientes entienden que compartir datos con las compañías es un mal necesario, pero cuando estas son víctimas de un ataque los datos pasan a manos del atacante, produciendo una pérdida de confianza y de credibilidad en las políticas de seguridad y, por tanto, de pérdidas empresariales.
- El ataque significa que la seguridad de los sistemas no es suficientemente robusta y se puede incurrir en faltas de cumplimiento normativo que pueden acarrear sanciones.
- Se pueden perder datos irreemplazables o críticos para la continuidad empresarial, salvo que tengamos copias de seguridad actualizadas.
- Los costes del pago de rescate son altos y puede ser múltiples y no solo afectan al balance final de cuentas, sino que muestran a terceras partes la debilidad de nuestra seguridad.

Desde el punto de vista monetario, el informe de Chainalysis [84] pone de manifiesto los costes económicos de las extorsiones realizadas por el *ransomware*. La Figura 2.11 muestra la evolución de los pagos en diferentes criptomonedas a direcciones relacionadas con el *ransomware*. Como se puede observar, la cantidad recaudada en 2020 fue de 692 millones de dólares y en 2021 de 602 millones. También se conoce el pago distribuido por familias, donde se muestra como Conti (un ejemplo del modelo RaaS) es la familia que ha recaudado una mayor cantidad: unos 180 millones (Figura 2.12).

Otro dato interesante es la esperanza de vida de los ataques de *ransomware*. En [84] encontramos cómo se ha ido acortando esta vida útil en los últimos años. Por ejemplo, en 2021 el número de días de actividad se redujo a un mes, frente al año que encontrábamos en 2019 (Figura 2.13). Una de las razones principales de esta reducción obedecería al cambio de marca, es decir, los atacantes intentan crear la ilusión de que varios ataques pertenecen a organizaciones criminales distintas, configurando sitios de pago y otras infraestructuras diferentes para cada víctima pero compartiendo similitudes de código. Por ejemplo, la organización rusa Evil Corp ha realizado múltiples ataques con cambio de marca incluyendo Doppelpaymer, Bitpaymer,

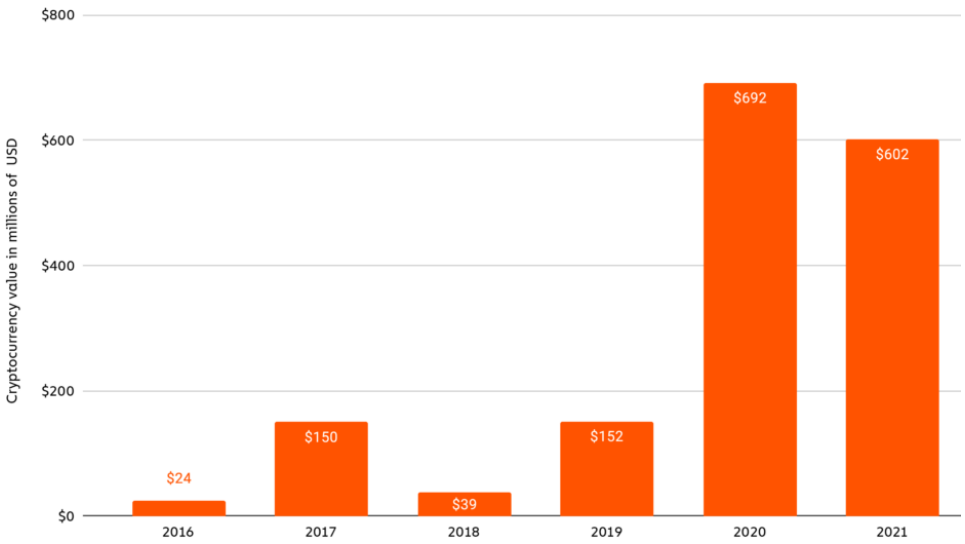


Figura 2.11: Pagos totales en criptomonedas recibidos en direcciones de *ransomware* entre 2016 y 2021 [84].

WastedLocker, Hades, Phoenix Cryptolocker (realizó un único ataque que recaudó 40 millones de dólares), Grief (similar a Doppelpaymer pero solicita pago en Monero), Macaw (utiliza un método de negociación diferente) y PayloadBIN. Estas relaciones son visibles analizando las transacciones de criptomonedas. Para las clases de *ransomware* citadas podemos ver en la Figura 2.14 como casi todas ellas realizan el proceso de lavado de dinero a través de las mismas carteras intermedias, y como estas carteras intermedias lo mueven a los mismos destinos. El estudio de los movimientos en criptomonedas relacionados con el *ransomware* puede llevar a las autoridades a comprender el fenómeno en profundidad [85].

Este esfuerzo para realizar un cambio de marca puede estar justificado de cara a evadir las sanciones. Concretamente, Evil Corp, cuyo líder está relacionado con el gobierno ruso, está sancionado por Estados Unidos desde diciembre de 2019. En 2020, la Oficina de Control de Activos Extranjeros del Tesoro del EE.UU. (OFAC) reiteró la notificación de que quienes pagasen un rescate de *ransomware* podrían enfrentarse a una multa. Esto puso al grupo en una situación comprometida ya que muchas víctimas eran reacias a pagar. El cambio de marca propuesto por el grupo estaría pensado para engañar a las víctimas para pagar antes de que los investigadores pudiesen descubrir el riesgo de sanciones. De hecho, este cambio parece haberle funcionado, ya que ha recaudado más de 85 millones de dólares.

Por supuesto, este cambio de marca lo han realizado otros grupos. El grupo detrás de DarkSide lanzó un ataque similar denominado BlackMatter

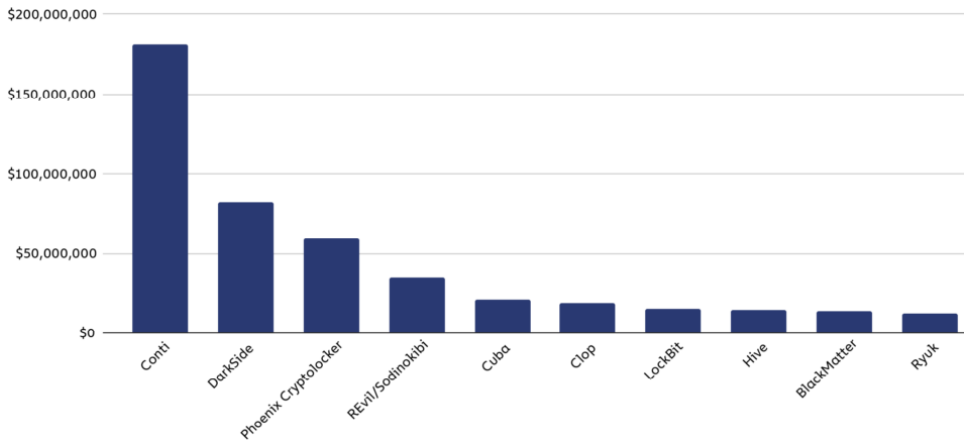


Figura 2.12: Recaudación de las 10 familias de *ransomware* más activas en 2021 [84].

después de causar el ataque a Colonial Pipeline y el FBI incautara la mayor parte del rescate. Este cambio tenía como objetivo rebajar la presión de los cuerpos de seguridad. En enero de 2022, un *ransomware* denominado Blue Cat (o Alphv), que parece ser un cambio de marca de Dark Side, atacó a las compañías de gas y petroleo Mabanaft y Oiltanking GmbH Group.

Si bien muchos ataques de *ransomware* tienen una motivación económica, algunos casos muestran más una motivación geopolítica orientada hacia el engaño, espionaje, daño a la reputación o la interrupción de las operaciones de un gobierno enemigo. En estos casos, las familias de *ransomware* no contienen mecanismos para recolectar dinero o permiten que las víctimas recuperen sus archivos. Un ejemplo de ello es el ataque descrito por el equipo de respuesta a incidentes informáticos de Ucrania (CERT-UA), que informó de la disrupción de varias agencias gubernamentales. Otra situación similar es la causada por la familia basada en NotPetya, que no contenía un mecanismo viable de pago y que afectó a varias organizaciones ucranianas. Otro ataque, conocido como WhisperGate [86], estaba diseñado para que las víctimas no pudiesen recuperar sus archivos, y en él se utilizó una variante conocida como DEV-0586 (un hermanastro de WhiteBlackCrypt). Como dato curioso, su hermanastro WhiteBlackCrypt que atacó organizaciones rusas, parece, dadas las similitudes entre ellos, que fue un intento de hacerlo pasar por ucraniano. Todos los casos parecen tener el claro objetivo geopolítico de disrupción de servicios por parte de Rusia hacia Ucrania. Pero estos no son los únicos países en litigio. Analistas de ciber-seguridad de Crowdstrike y Microsoft concluyen que también lo han utilizado Irán, contra EEUU e Israel, China y Corea del Norte.

Muy recientemente, muestras de *ransomware* con similitudes con el an-

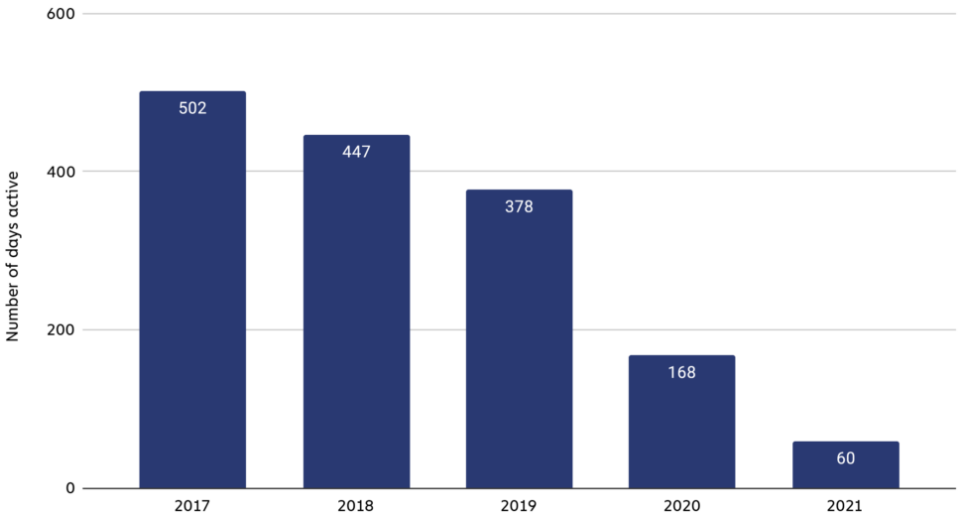


Figura 2.13: Evolución anual de la vida útil media del *ransomware* contabilizada en días [84].

teriormente usado WhisperGate [87], han sido utilizadas como señuelo en el ataque lanzado por Rusia contra Ucrania mediante el *malware* HermeticWiper (KillDisk.NCV) [88, 89]. Dichas muestras muestran una nota de rescate que incluye dos direcciones de correo para contactar con la organización, y un mensaje político "The only thing that we learn from new elections is we learned nothing from the old!", como muestra la Figura 2.15 [87].

Recientemente, un investigador ucraniano ha liberado el código fuente, chats, paneles, etc. de Conti poco después de la invasión de su país por Rusia [90] desde la cuenta de Twitter @contiLeak, con más de 309 archivos y 60.000 mensajes internos. El código esta accesible en la web de Vx-Underground [91].

Como hemos podido ver, el *ransomware* es una tapadera útil para la negación estratégica y el engaño contra Estados enemigos de bajo coste y que permite la negación plausible, al ser fácilmente atribuible a grupos de ciber-criminales o a otros Estados. En todo caso, estos ataques suelen dejar pistas en la *Blockchain*.

2.4. Modelo de ataque para cripto-*ransomware*

Para una comprensión efectiva de las técnicas de defensa frente al *ransomware*, vamos a caracterizar primero el funcionamiento de los *ransomware* de cifrado. Para ello, vamos a establecer una taxonomía basada en el modelo de defensa denominado *Cyber Kill Chain* (CKC), en línea con los trabajos [12, 92, 93, 94]. El modelo CKC deriva del propuesto *Intrusion Kill Chain*

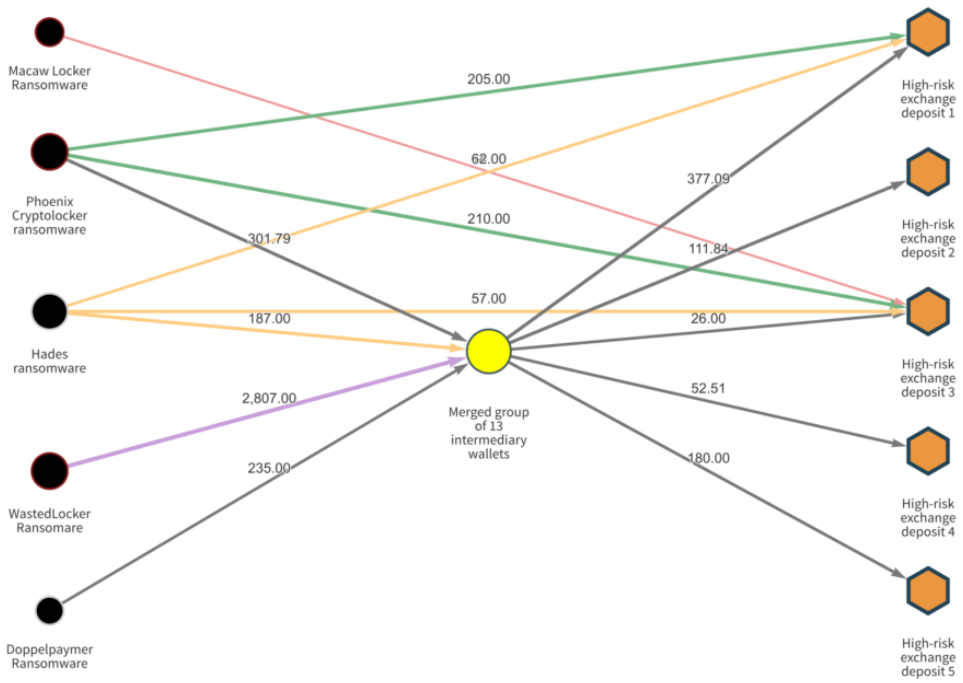


Figura 2.14: Movimientos de criptomonedas entre carteras de clases *ransomware* del grupo Evil Corp [84].

por Lockheed Martin [95], pero suministra información de grano fino de cada etapa que el atacante debe completar antes de alcanzar su objetivo. La caracterización de cada etapa permite entender qué evidencias necesitamos para defendernos del ataque. El modelo se ha usado por el sector industrial para modelar ataques de intrusión [96] y también para obtener inteligencia sobre tácticas, técnicas y procedimientos (TTPs) del atacante y atribución del ataque [97]. Otros trabajos previos que también abordan los métodos de infección, las cargas dañinas y sus tácticas son [98, 99, 100, 101].

El modelo CKC establece siete etapas para que el atacante alcance su objetivo: 1) Reconocimiento, 2) Armamento (*Weaponization*), 3) Entrega, 4) Explotación, 5) Instalación, 6) Orden y control, y 7) Acción sobre el objetivo. Los principales elementos de cada etapa son:

1. *Reconocimiento* - El atacante intenta recoger tanta información como sea posible sobre sus objetivos (listas de correo, RRSS, vulnerabilidades de los sistemas o servicios, etc.) para decidir las herramientas a usar para llevar a cabo un ataque robusto.

2. *Armamento* - El atacante arma sus cargas maliciosas de forma que sobrepasen o evadan los controles de seguridad del objetivo [102].

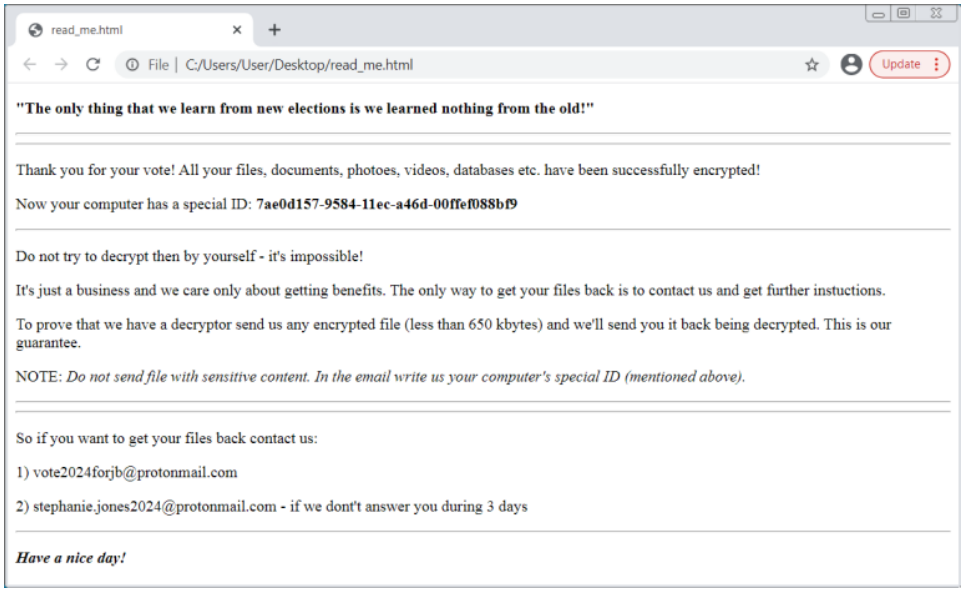


Figura 2.15: Nota de extorsión mostrada por el *ransomware* utilizado como señuelo [87].

3. *Entrega* - El atacante debe encontrar al menos un componente del vector de ataque para hacer llegar la carga al objetivo, por ejemplo, un correo electrónico, una memoria extraible, etc.

4. *Explotación* - La carga maliciosa explota una vulnerabilidad y ejecuta su programa binario con el mínimo privilegio requerido. El desarrollo de *Crimeware-as-a-Service* [103] simplifica la labor del atacante en esta etapa.

5. *Instalación* - El atacante intenta acceder a nuevos nodos de la red e instalar herramientas administrativas, como Troyanos de Acceso Remoto (RAT) o puertas traseras [104].

6. *Orden y control* (C&C o C2) - El atacante toma control sobre el objetivo mediante un canal de comunicación para dar órdenes al *malware* o ex-filtrar datos del sistema.

7. *Acciones sobre el objetivo* - Se realizan sobre el equipo infectado los objetivos propios del ataque; por ejemplo, acceder a datos y ex-filtarlos, denegar el acceso del legítimo propietario a los datos o equipo, etc.

En base a este modelo, la taxonomía de las características de amenaza propuesta por [92] aparece resumida en la Figura 2.16, donde se ha excluido la fase de Reconocimiento ya que no es exclusiva en el caso de ataques

ransomware [105] y existen multitud de formas de reconocimiento dada la naturaleza del ciberespacio, que además pueden ser combinadas de diferentes formas para realizar un ataque, ya sea de la misma naturaleza o diferente tipo.

2.4.1. Etapa de armamento

Los *ransomware* basado en *scripts* cifran los datos mediante órdenes embebidas en sus guiones. Por lo general, estos *scripts* se eliminan al finalizar el cifrado y el código de funcionamiento del *malware* solo reside en memoria. Por ello, se suelen denominar 'sin archivo (*fileless*)' o 'malware-solo-en-memoria' [106]. Dado que no necesitan instalación, les es fácil sobrepasar controles a nivel de anfitrión e infectar a usuarios privilegiados. La mayoría de ellos están escritos en JavaScript, PHP, PowerShell y Python.

Con el aumento de soluciones anti-*malware* basadas en web, en esta etapa se realiza una diversificación de las cargas de entrega en la que se suelen embeber dentro de una carga de aspecto benigno, tales como archivos de procesadores de texto o vídeos, para evadir la detección.

Algunas familias de *ransomware* tienen un patrón de interacción con los archivos similar (lo abren, lo leen y lo cifran), por lo que son fácilmente detectables. Para evitarlo, los atacantes intentan tanto diversificar los patrones de acceso a archivos y su interacción con el sistema de archivos, como diversificar el proceso de cifrado ofreciendo diferentes esquemas para manipular los archivos a cifrar, como veremos en el Apartado 2.4.8.

Otro ejemplo de esta diversificación en el modo cifrado lo encontramos en LockFile [66], donde el *ransomware* realiza por una parte un cifrado intermitente (es decir, cifra bloques cada cierto tamaño de *bytes*, para sesgar el análisis estadístico y hacer más rápido el cifrado) y por otro lado interacciona con los archivos a través de la interfaz de archivos, proyectado en memoria en lugar de a través de la API convencional de archivos.

Respecto a los métodos de cifrado, algunas familias de *ransomware* suelen usar cifrado estándar mientras que otras aplican uno personalizado. Las primeras suelen usar primitivas del sistema operativo o API de cifrado, y pueden dividirse en tres categorías según su tipo: cifrado asimétrico, cifrado simétrico y cifrado híbrido (que detallaremos en el Apartado 2.4.8). Si bien el uso de cifrado estándar suministrado es cómodo, cuando se utilizan muchas API con un gran volumen de datos a cifrar se suelen requerir privilegios de administrador, lo que no es siempre el caso. Una herramienta de detección podría limitar de forma fácil el uso de la API de cifrado, por lo que a veces se usa uno personalizado. Por tanto, la diversificación de los métodos de cifrado puede considerarse una técnica de evasión de las herramientas de detección basadas en una cripto-API estándar [107].

Las técnicas de evasión permiten a los programas maliciosos sobrepasar los controles de seguridad y se consideran en esta categoría ya que aumen-

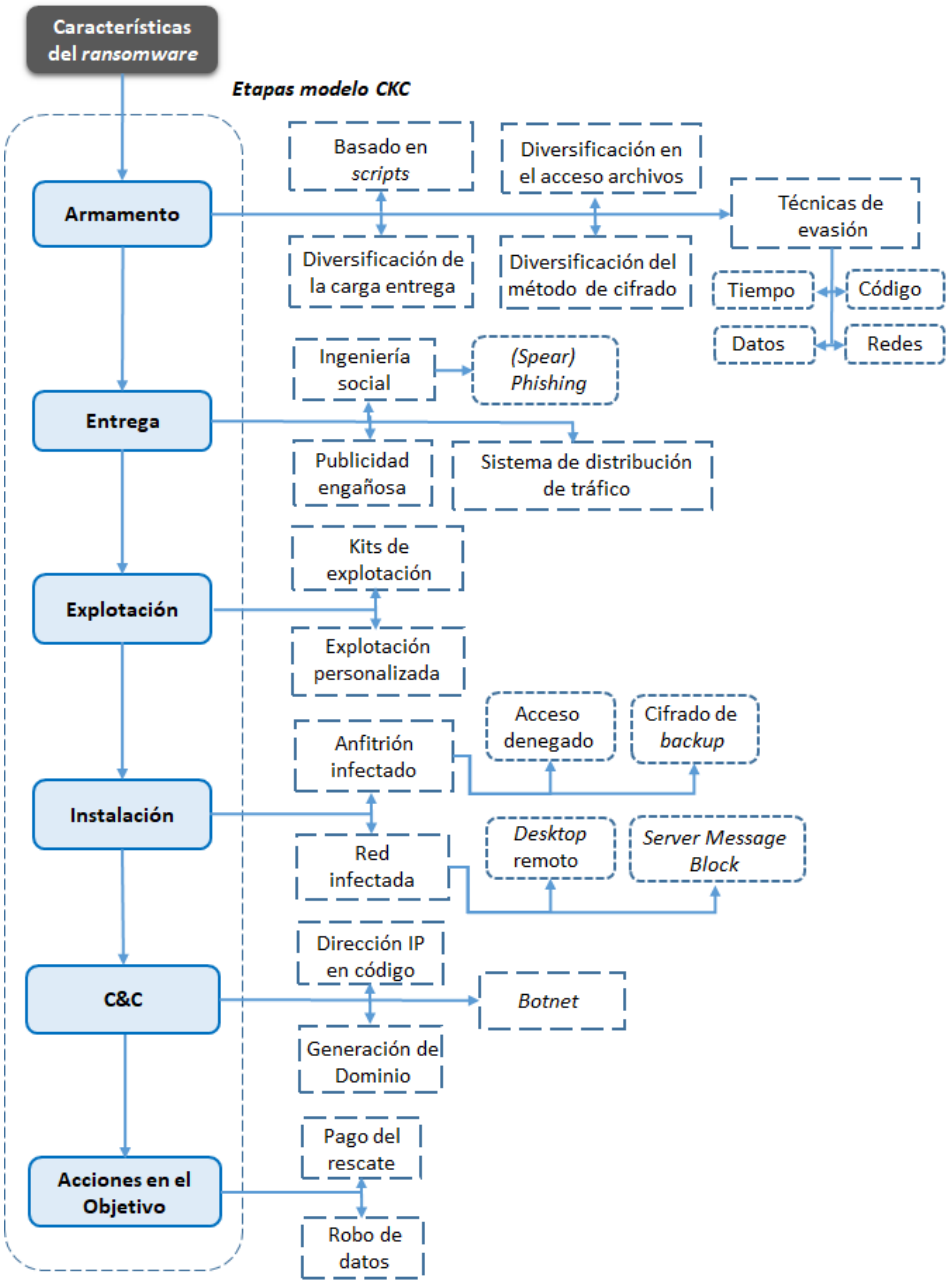


Figura 2.16: Taxonomía de *ransomware* basada en el modelo CKC (modificado de [92]).

tan la capacidad ofensiva de dichos programas [108, 109, 110]. Las técnicas de evasión más utilizadas por el *ransomware* pueden agruparse en cuatro categorías: *(i)* basadas en temporizador, *(ii)* evasión de datos, *(iii)* evasión de código, y *(iv)* evasión de red. El trabajo [111] analiza la posibilidad de la utilización por parte de los constructures de *malware* de técnicas de evasión específicas para escapar de los métodos que utilizan *Machine Learning*.

Las técnicas de evasión basadas en temporizador ejecutan el programa malicioso en un tiempo/fecha específicos, o en otros casos miden lo que tarda la ejecución del código, lo que les ayuda a determinar si están siendo depuradas. Las técnicas de evasión las podemos dividir en *(i)* ejecución retrasada, y *(ii)* ejecución basada en eventos. En el primer caso, ciertas familias de *ransomware* retrasan el proceso de cifrado durante minutos o incluso días ya que muchas herramientas, como veremos en breve, se basan en el comportamiento de la fase de cifrado para su detección. En el segundo caso, las muestras de *ransomware* permanecen durmientes a la espera de algún evento del sistema (*p.ej.*, acceso de administrador o reinicio) que active el ataque.

Las técnicas de evasión de datos abordan la eliminación de las evidencias de la actividad maliciosa con el objeto de dificultar su detección. La autodestrucción tiene como objetivo borrar las trazas de la infección en la máquina no solo para dificultar la detección, sino también la posible aplicación de técnicas de análisis forense. Las técnicas de anti-volcado tienen como finalidad dificultar la ingeniería inversa del código compilado de las muestras procedente de un volcado de memoria o incluso de obtener las claves de cifrado; por ejemplo, mediante el borrado de las cabeceras del ejecutable o cargando el mismo de forma dispersa en diferentes partes de memoria (técnica denominada *bytes robados*). El mecanismo de flujos de datos alternativos (*Alternate Data Streams* - ADS) es un mecanismo introducido en algunos sistemas de archivos, como Windows o McOS, que permite añadir atributos extra a un archivo. En el caso de programas maliciosos se ha utilizado para añadir el ejecutable del mismo en un flujo de datos alternativo distinto del principal, que es el que contiene el archivo legítimo. Por último, también es posible el borrado de identificadores de zona, que, como su nombre indica, lo que realiza es un borrado del Identificador de Zona, esto es, un ADS especial para archivos en una partición de disco NTFS que especifica el origen de un archivo [112].

El análisis del armamento usado por las familias de *ransomware* en los últimos dos años [82] revela que 109 vulnerabilidades son explotadas para la ejecución remota de código (RCE), de las cuales 23 permiten escalar privilegios y 13 provocar denegación de servicio (DoS), mientras que 40 son capaces de explotar vulnerabilidades/debilidades de aplicaciones web (hablaremos con más detalle de ellas en el Apartado 2.4.3).

El análisis de las secuencias de llamadas a la API realizadas en la etapa previa al ataque, denominadas actividades *paranoia* [113], permite clasificar

muestras de *ransomware* constituyendo así su huella identificativa. En el repositorio de GitHub previamente indicado en el trabajo se indican los 188 artefactos (funciones de la API) que se invocan y que permiten analizar su contexto mediante el uso inyección de las DLL para su análisis dinámico.

- **Técnicas de evasión de código**

Aquí encontramos cuatro tipos de técnicas muy utilizadas por el *malware* en general. La técnica de evasión del depurador donde, usando por ejemplo el marco ANTI [114], el *ransomware* detecta si ha sido ligado/inyectado a un depurador y, si es así, evita la ejecución del programa de la muestra o mata al depurador antes de ejecutarse.

Las técnicas anti-desensamblado permiten bien inyectar código muerto/basura o bien ofuscar la carga dañina o cifrar el código, con el objeto de dificultar el desensamblado de la muestra en el proceso de ingeniería inversa que realizará el analista de *malware*. En el empleo de las técnicas de ofuscación se suelen emplear los empaquetadores (*packers*) o los cifradores (*crypters*) [115], ya que su aplicación reduce la detección por parte de los antivirus entre un 86,71 % y un 50,88 % [116].

Las técnicas anti-*sanboxing* permiten que una muestra no se ejecute y permanezca durmiente si detecta que se está ejecutando en una máquina virtual o *sandbox*, que suelen utilizarse por los analistas para examinar dichas muestras. Estas técnicas anti-*sandbox* pueden estar basadas en tiempo, donde se mira el registro *Time Stamp Counter* -TSC- para contar los ciclos de CPU y tratar de determinar si hay diferencias entre la ejecución normal y virtualizada, o basadas en artefactos, donde se detecta la virtualización mirando la dirección MAC de la máquina anfitriona en el registro, o bien comprobando los procesos que se ejecutan en la máquina anfitriona.

La técnicas de polimorfismo y metamorfismo introducen cambios pequeños o transitorios en las muestras para evadir la detección basada en firmas. El comportamiento polimórfico es la capacidad de auto-mutación mediante el cifrado produciendo multitud de firmas para el mismo *malware*, por ejemplo, cambiando la clave de cifrado en cada ejecución del programa. El comportamiento metamórfico se basa en la continua reprogramación en cada iteración/distribución de ejecución al objeto de cambiar su firma.

- **Técnicas de evasión de red**

Los sistemas de detección/prevención de intrusiones (IDS/IPS) se suelen basar en firmas (patrones conocidos de tráfico de ataques) o detección de anomalías (comparación con el tráfico normal) para detectar los programas maliciosos [117]. La detección basada en firmas es eficiente en la detección de ataques conocidos, pero no en los desconocidos. Por contra, los basados en anomalías tiene tasas altas de falsos positivos pero son capaces de detectar ataques no vistos antes.

El *ransomware* utiliza diferentes técnicas para engañar a estos mecanismos de detección [118]. Entre ellas están el cifrado de tráfico de red, la utilización de anonimizadores de tráfico, el sombreado de dominios (*Domain Shadowing*), y el flujo rápido (*Fast Flux*). El cifrado el tráfico permite que las comunicaciones entre la víctima y el C&C no puedan analizarse. La anonimización de tráfico, por ejemplo con el uso de TOR, cifran el tráfico y evaden la detección del origen del mismo. El sombreado de dominios, basado en el robo de credenciales de dominios legítimamente registrados, permite la creación de gran número de sub-dominios que se hacen corresponder con los servidores maliciosos de forma que se van rotando estos con un servidor para evadir las defensas. Esta técnica es especialmente utilizada en familias que utilizan kits de explotación. Por último, en el flujo rápido, la dirección IP asociada con un dominio, o registro DNS asignado al dominio, rota entre una lista de direcciones IP para protegerse de la detección mediante listas negras de IP ligadas a una campaña específica de *ransomware*.

2.4.2. Etapa de entrega

Las técnicas para encontrar una forma de acceder al sistema se basan generalmente en la ingeniería social, en la publicidad maliciosa o los sistemas de distribución de tráfico [119, 120].

De forma general, recibe el nombre de ingeniería social el conjunto de técnicas mediante las cuales el atacante hace que la víctima confíe en él y realice acciones en su beneficio, como por ejemplo pinchar en una URL o suministrarle información sensible como credenciales. Este sistema es uno de los más utilizado en la entrega tanto de *malware* como de *ransomware* [121]. En cuando al *ransomware*, dos de los métodos más frecuentes son el *phishing* general o el *spear-phishing*.

El *phishing* intenta convencer a la víctima para que le dé información sensible, como claves, número de tarjetas de crédito, etc. Esta técnica, que va en aumento, en cuanto a su número y sofisticación [122], puede realizarse mediante correos electrónicos de *spam*, mensajería instantánea, llamadas telefónicas, etc. Otras veces, intentan que la víctima visite un sitio web infectado o descargar un adjunto maliciosos del correo.

Frente al *phishing* general, el *spear-phishing* está destinado a objetivos específicos de usuarios que tienen acceso a datos sensibles dentro de una organización, que son localizados generalmente mediante una fase de reconocimiento muy exhaustiva. Las formas más comunes para la distribución del *ransomware* son adjuntar el mismo a un correo, enviar enlace a un sitio web infectado, y usar un enlace a un sitio de hospedaje en la nube.

En el método de la publicidad maliciosa, los operadores del *ransomware* lanzan una campaña publicitaria en sitios web legítimos que redirigen a la víctima a un dominio propiedad del atacante donde se descarga el *malware* o un kit de explotación para encontrar las vulnerabilidades del sistema de

Medio de Entrega	**% de uso**	**Protocolo**	**% de uso**
Correo *spam*	60	SMTP	45
RDP	21	IMAP	26.5
Troyano	20	Web-browsing	22.3
Explotación de vulnerabilidad	15	POP3	3.8
Botnet / Descargador	11	FTP	2.3
Publicidad maliciosa	8	Otros	3.3
Endpoint	7		
Servidor externo a la empresa	7		
Medios extraibles	6		
Medios sociales	5		
Personal interno	3		
Administrador desleal	2		
SMS	1		
Esquemas de afiliados	1		

Tabla 2.1: Porcentaje de uso de diferentes medios de entrega (las dos columnas a la izquierda) y protocolos empleados (las dos columnas a la derecha) por el *ransomware*.

la víctima para su instalación.

En lugar de entregar el *ransomware* vía un sitio web infectado o pagar por uno, en el modelo de sistema de distribución de tráfico (SDT) se paga por redirigir el tráfico de un sitio web legítimo (por ejemplo, sitio de contenidos para adultos, videojuegos, ...) al sitio malicioso del atacante mediante el SDT.

El informe [123] de 2021 muestra los porcentajes de las diferentes formas de entrega, que se recogen en las dos primeras columnas de la Tabla 2.1. En las últimas dos columnas se detallan los protocolos utilizados [124].

Respecto a los artefactos concretos utilizados para su distribución en los últimos dos años podemos citar por orden de importancia Emotet, Zbot, Dridex, Gozi y Danabot [53].

2.4.3. Etapa de explotación

Tras la entrega de la carga maliciosa, el *ransomware* necesita un medio para ejecutarse. Este puede ser el uso de un kit de explotación o bien una explotación dirigida.

Un kit de explotación (EK) es una caja de herramientas de *hacking* que se utiliza para escanear vulnerabilidades de la máquina objetivo y explotarlas de cara a ejecutar el programa malicioso [125, 126, 127]. El atacante puede redirigir a la víctima hacia un dominio malicioso donde el kit de explotación escanea y explota las vulnerabilidades. En este sentido, está creciendo en el mercado negro la tendencia *exploit-as-a-service* para la compra de este ser-

vicio y posterior dotación del *ransomware* que deseemos como carga. Entre los kits más conocidos tenemos [128]: Angler, que explota vulnerabilidades de Adobe Flash y Microsoft Silverlight; Neutrino [129] y Magnitude, cuyo objetivo es Adobe Flash; Rig y Silverlight, con objetivo MicrosoftLight; Nuclear, que también tiene como destino Adobe Flash; Blackhole, que explota vulnerabilidades de Adobe Reader, Adobe Flash y Java; Fallout, que afecta a Microsoft Windows [130] y Adobe Flash; EternalBlue, que afecta a Windows SMB Server [131]; o Spelevo, que aprovecha vulnerabilidades de Internet Explorer y Adobe Flash [132].

El informe de 2022 de [82] indica cómo el uso de vulnerabilidades no parcheadas explotadas por el *ransomware* ha crecido un 29 % entre 2020 y 2021, con un total de 288 CVE (*Common Vulnerabilities and Exposures*) para este último año, de las cuales, un cuarto (65) han sido tendencia en Internet. Esto subraya desde el punto de vista defensivo la necesidad de priorizar y solventar dichas vulnerabilidades. Otro aspecto relevante del citado informe es que solo el 92,7 % de las vulnerabilidades fueron detectadas por los escáneres más conocidos (Nessus, Qualys y Nexpose), y que 21 de ellas no fueron detectadas por ninguno de los citados escáneres. El informe de S21sec de 2021 [68] recoge las vulnerabilidades más explotadas por los creadores de *ransomware* y que se recogen en la Tabla 2.2 (se han resaltado en negrita las vulnerabilidades de día cero).

Para un mayor efecto en aspectos que van desde obtener acceso inicial hasta conseguir mayor penetración en la red, se puede observar el uso de vulnerabilidades encadenadas. Por ejemplo, se puede observar que el ataque PetitPotam [133] (CVE-2021-36942) encadena las vulnerabilidades de Microsoft Exchange, conocidas como ProxyShell (CVE-2021-34473, CVE-2021-34523 y CVE-2021-31207), o el conjunto de las cuatro vulnerabilidades de Microsoft conocidas como Exchage ProxyLogon (CVE-2021-26855) que encadena a CVE-2021-26857, CVE-2021-26858 y CVE-2021-27065.

Tabla 2.2: Lista de vulnerabilidades recientes utilizadas por el *ransomware* agrupadas por producto comercial (las vulnerabilidades de día cero están marcadas en negrita).

Producto	**Vulnerabilidad**	**Descripción**
Pulse Secure VPN	CVE-2021-22893	Omisión de autenticación expuesta por Windows File Share Browser y Pulse Secure Collaboration
	CVE-2020-8260	Un atacante autenticado podría ejecutar código arbitrario usando una extracción gzip no controlada

(Continúa en la página siguiente ...)

(... viene de la página anterior)

Producto	Vulnerabilidad	Descripción
	CVE-2020-8243	Un atacante autenticado podría cargar una plantilla personalizada para ejecutar código arbitrario
	CVE-2019-11539	Un atacante autenticado puede inyectar código y ejecutar órdenes en la interfaz web
	CVE-2019-11510	Un atacante remoto no autenticado puede enviar una URI para explotar una vulnerabilidad de lectura de archivos arbitraria
Citrix	CVE-2020-8196	Divulgación de información limitada para usuarios poco privilegiados
	CVE-2020-8195	Divulgación de información limitada para usuarios poco privilegiados
	CVE-2019-19781	Permite un salto de directorio
	CVE-2019-11634	Control de acceso incorrecto
Microsoft Exchange	CVE-2021-34523	Elevación de privilegios
	CVE-2021-34473	Ejecución remota de código
	CVE-2021-31207	Omisión de la Característica de Seguridad de Microsoft Exchange Server
	CVE-2021-26855	Ejecución de código remota
Fortinet	CVE-2020-12812	Autenticación inapropiada en SSL VPN
	CVE-2019-5591	Permite a un atacante no autenticado en la misma subred interceptar información confidencial haciéndose pasar por el servidor LDAP
	CVE-2018-13379	Limitación inadecuada de un nombre de ruta a un directorio restringido bajo el portal web SSL VPN, que permite a un atacante no autenticado descargar archivos del sistema mediante solicitudes HTTP
SonicWall	**CVE-2021-20016**	Permite a un atacante remoto no autenticado realizar consultas SQL a información de sesión como claves y nombres de usuario

(Continúa en la página siguiente ...)

(... viene de la página anterior)

Producto	**Vulnerabilidad**	**Descripción**
	CVE-2020-5135	Desbordamiento del búfer que permite a un atacante remoto causar una Denegación de Servicio (DoS) y ejecutar potencialmente código arbitrario mediante el envío de una petición maliciosa al *firewall*
	CVE-2019-7481	Permite a atacante no autenticado ganar acceso de lectura a recursos no autorizados
F5	CVE-2021-22986	Ejecución remota de órdenes no autenticada en la interfaz REST de iControl
	CVE-2020-5902	Ejecución remota de código en páginas no reveladas en la intefaz TMUI
Palo Alto	CVE-2020-2021	*Bypass* en la autenticación SAML
	CVE-2019-1579	Ejecución remota de código de atacante no autorizado
QNAP	CVE-2020-2021	*Bypass* en la autenticación SAML
	CVE-2019-1579	Ejecución remota de código de atacante no autorizado
	CVE-2019-28799	Autorización inadecuada que permite acceso a dispositivos NAS
Sophos	CVE-2020-12271	Inyección SQL que permite ex-filtrar nombres de usuarios y *hashes* de claves
SharePoint	CVE-2019-0604	Ejecución remota de código en SharePoint
Microsoft Windows	CVE-2019-0708	Ejecución remota de código en RDS
	CVE-2020-1472	Elevación de privilegios cuando el atacante establece conexión de canal seguro Netlogon
	CVE-2021-31166	Ejecución remota de código en la pila HTTP
	CVE-2021-36942	Suplantación de identidad en Windows LSA
Microsoft Office	CVE-2017-0199	Un atacante remoto puede ejecutar código arbitrario a través de documentos manipulados

(Continúa en la página siguiente ...)

(... viene de la página anterior)

Producto	**Vulnerabilidad**	**Descripción**
	CVE-2017-11882	Un atacante puede ejecutar código arbitrario en el contexto de un usuario al no gestionar correctamente objetos en memoria
	CVE-2021-40444	Ejecución de código remoto en Microsoft NSHTML
vCenter	CVE-2021-21985	Ejecución de código remoto por falta de comprobación de entrada en *plugin* Virtual SAN Heakth Check
Accelion	CVE-2021-27101	Inyección SQL mediante encabezado de *Host* diseñado en petición de archivo *document_root.html*
	CVE-2021-27104	Ejecución de órdenes del sistema operativo (SO) mediante petición POST diseñada para varios *endpoints* de administración
	CVE-2021-27102	Ejecución de órdenes del SO mediante llamada al servicio web local
	CVE-2021-27103	Vulnerabilidad tipo SSRF mediante petición POST diseñada para el archivo *wmProgressstat.html*
FileZen	CVE-2021-20655	Permite a atacante remoto con privilegios de administrador ejecutar órdenes arbitrarias del SO
Atlassian	CVE-2021-26084	Inyección OGNL que permite a un atacante ejecutar código arbitrario en un *Confluence Server* o *Data Center*
Log4j	**CVE-2021-44228**	Permite la ejecución remota de código mediante la validación incorrecta de una entrada al procesar solicitudes LDAP
	CVE-2021-45046	Permite a un atacante realizar un ataque de Denegación de Servicio
	CVE-2021-45105	Permite a un atacante realizar una Denegación de Servicio
Kaseya	**CVE-2021-30116**	Permite la divulgación de credenciales
ProxyLogon	**CVE-2021-44228**	Permite la ejecución remota de código (RCE) mediante la validación incorrecta de una entrada al procesar solicitudes LDAP

(Continúa en la página siguiente ...)

(... viene de la página anterior)

Producto	Vulnerabilidad	Descripción
	CVE-2021-45046	Permite a un atacante realizar un ataque de Denegación de Servicio
	CVE-2021-45105	Permite a un atacante realizar una Denegación de servicio

(Fin de la tabla)

Además de las vulnerabilidades, también se explotan debilidades del código catalogadas en la lista CWE (*Code Weak Enumeration*) o el proyecto OWASP (*Open Web Application Security Project*) y que se muestran en la Tabla 2.3.

Mientras que los EK están pensados para aplicarlos masivamente, la explotación dirigida se enfoca hacia una máquina concreta. Este tipo de ataques está creciendo dado que los operadores del *ransomware* buscan víctimas para la extorsión del alto perfil dentro de una organización y obtener así más beneficios [134].

El informe de VirusTotal de 2021 [53] recoge los tipos de muestras detectadas entre 2020 y 2021. Como se puede observar (Figura 2.17): *i)* el 95 % de los archivos detectados eran ejecutables o bibliotecas dinámicas (DLL) de Windows; *ii)* un 2 % estaba basado en Android; y *iii)* resaltar un conjunto de *ransomware* EvilQuest para plataformas OSX con un amplio número de muestras detectadas por vez primera. Por otra parte, a la vista del modelo CKC, se pueden ver pocas muestras asociadas con *exploits*, lo que se puede deber a que usualmente se entregan usando ingeniería social y/o goteadores (*dropper* - programa pequeño para instalar *malware*). De hecho, solo el 5 % de las muestras analizadas están asociadas con *exploits* de Windows y relacionadas con escalado de privilegios, revelación de información de SMB o ejecución remota.

2.4.4. Etapa de instalación

Tras la etapa de explotación, el *ransomware* instala la carga binaria maliciosa. En esta fase, algunas familias se conectan a un C&C, otras funcionan como un gusano que se propaga por la red local. Por ello, la fase de instalación se puede dividir en dos sub-etapas: *(i)* instalación sobre el sistema infectado, e *(ii)* instalación en la red objetivo.

En la sub-etapa de instalación en el anfitrión infectado el binario desplegado cifra los archivos, y cualquier copia de seguridad detectada también es cifrada. Para hacer inaccesibles los archivos, algunas familias solo cifran los archivos específicos dependiendo de alguna propiedad (extensión, tamaño, etc.), otras archivan estos en un repositorio protegido por clave durante la instalación, pero en todos los casos la máquina infectada sigue operativa para poder pagar el rescate. Existen algunas familias más intrusivas que cifran

CWE	Categoría	Descripción
20	OWASP Top 25	Validación inadecuada de entradas
22	OWASP Top 25	Limitación inadecuada de nombre de camino a directorios restringidos (*Path traversal*)
77	MITRE Top 25 OWAP Top 25	Inyección de órdenes: neutralización inadecuada de elementos especiales en una orden
119	MITRE Top 25	Restricción inadecuada de operaciones dentro de los límites de un búfer en memoria
190	MITRE Top 25 OWAP Top 25	Desbordamiento de enteros o *Wraparound* que puede introducir debilidades en la gestión de recursos o lógica de control
264	OWASP Top 25	Permisos, privilegios y controles de acceso
290	OWASP Top 10	Saltar autenticación mediante *spoofing*, que puede provocar revelación de información, DoS o pérdida de reputación
330	OWASP Top 10	Atacante puede predecir valores generados por falta aleatoriedad de los mismos, que pueden usarse para impersonar a otros usuarios o acceder a información sensible
522	OWASP Top 10 MITRE Top 25	Credenciales insuficientemente protegidas, que puede provocar que se acceda a ellas en la red o almacenamiento
787	MITRE Top10	Escritura fuera de límites
798	OWASP Top 10	Credenciales embebidas en el código, que puede provocar que se acceda a ellas y hacer *login* o comprometer a las aplicaciones
829	OWASP Top 10 MITRE Top 25	Inclusión de funcionalidad desde controles no confiables, que pueden comprometer a aplicaciones seguras que se comunican con un componente vulnerable
862	OWASP Top 10 MITRE Top 25	Autorización perdida puede provocar revelación de información, sobrepasar los controles de protección o el acceso no autorizado

Tabla 2.3: Lista de debilidades software explotadas por el *ransomware* en 2020 y 2021.

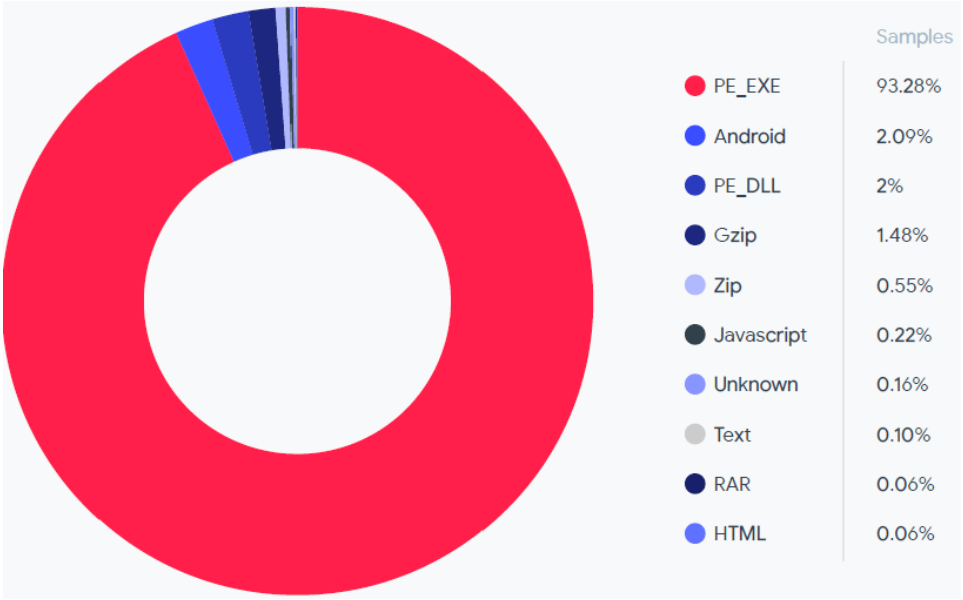


Figura 2.17: Tipos de muestras de *ransomware* [53].

el disco duro completo y dejan el sistema inservible; por ejemplo, las familias conocidas por Boot Locker que cifran el MBR.

Como se indicaba anteriormente, algunas familias de *ransomware* infectan las copias de seguridad, unidades extraibles o unidades montadas de forma remota/nube, y eliminan los archivos de restauración y borran todas las VSC (*Volume Shadow Copies*). Otras deshabilitan el modo de arranque seguro (*Windows Startup Repair*) y cambian la política del estado de arranque.

En cuanto a las instalación en la red infectada, algunas familias intentan infectar todos los nodos y discos de la red moviéndose lateralmente (mediante el uso de Mimikatz o Cobaltstrike, guiones realizados en AutoIT o PowerShell, múltiples RAT -Phorpiex, Smokeloader, Nanocore y Ponystealer [53]) y comportándose como gusanos. En este sentido, tres son los métodos principales basados en aprovechar vulnerabilidades bien del Protocolo Remote Desktop (RDP) [135], del Protocolo Server Message Block (SMB) [136], o del protocolo Virtual Private Network (VPN) [137].

2.4.5. Etapa de orden y control

La comunicación del *ransomware* con el servidor C&C es básica para gestionar las claves de cifrado (especialmente cuando se usa cifrado asimétrico, como veremos en breve) y gestionar el pago del rescate. Si bien hay diferencias entre familias, se pueden identificar dos fases: *(a)* Conexión con el C&C

antes iniciar el cifrado al objeto de recibir las claves, y *(b)* comunicación tras el cifrado para recibir el pago.

Entre las técnicas usadas para encontrar la dirección de servidor de orden y control tenemos: dirección IP embebida en el código, uso de algoritmos de generación de dominios, o uso de *botnets* existentes. Las dos primeras las discutiremos en la Sección 3.3 dedicada a los mecanismos de detección. En cuanto al uso de *botnets*, indicar que los atacantes aprovechan estas (redes de robots o máquinas comprometidas controladas por el C&C) para realizar diferentes actividades maliciosas: robo de información, *phishing* y propagación de *malware* [138].

En relación a la ex-filtración de datos, las familias más recientes suelen utilizar alguna de las siguientes técnicas:

- Envío de datos a sitios de fugas, como es el caso de Avaddon, Ako, Conti, Darkside, DoppelPaymer, Egregor, Everest, Lockbit, Light, Maze, Mespinoza, MountLocker, Nefilim, Nemty, Netwalker, Pay2Key, Ragnarok, RagnarLocker, RansomeEXX, REvil, Sekhmet, Snatch o Suncrypt.
- Envío/publicación de datos a/de foros *underground*, como hacen Avaddon, Ako, Darkside, Egregor, Kupidon, Maze, Nemty, REvil, Sekhmet, Suncrypt.
- Publicación de los datos ex-filtrados a Twitter, como es el caso en DoppelPaymer, Maze, RagnarLocker, Snatch.
- Datos vendidos o subastados, como ocurre en DoppelPaymer, Maze, REvil.

2.4.6. Etapa de acciones sobre el objetivo

Una vez que el *ransomware* ha cifrado los archivos, muestra un mensaje en la máquina de la víctima que notifica la infección y suministra directrices para pagar el rescate, que es el fin último de este tipo de *ransomware*. Si bien este es el procedimiento general, algunas familias también roban credenciales o los archivos de la víctima (como acabamos de mencionar). En los últimos incidentes también se están produciendo varios niveles adicionales de extorsión. El más frecuente, la doble extorsión, produce una nueva amenaza al notificar a la víctima que, de no pagar, filtrará el contenido de la información robada previamente. Se ha llegado incluso a cuatro niveles de extorsión, como indicábamos en la Sección 2.2. También se puede hacer que la máquina quede inutilizada si se reinicia.

Mientras que las primeras versiones de *ransomware* utilizaban sistemas de pago convencionales de transferencia de dinero (contactos SMS, Paypal, etc.), las más recientes utilizan criptomonedas. Además, algunas de ellas

suelen suministrar mecanismos para ayudar a la víctima en el pago, que pueden ir desde portales de pago hasta portales de descifrado.

Si bien la mayoría de las familias suelen mantener la promesa de descifrar los archivos tras el pago para mantener la reputación de los atacantes, ha habido algunos casos donde no se ha producido. Para que el pago sea rápido, hay familias como JIGSAW que, además de incrementar la cantidad a pagar de forma exponencial, van borrando archivos. Otro ejemplo es French Locker, que borra un archivo de forma permanente cada diez minutos hasta que se realice el pago.

Recientemente, algunos autores han propuesto un sistema de clasificación de la virulencia del *ransomware* en cinco niveles crecientes teniendo en cuenta el daño producido: CAT1 a CAT5 [139]. El algoritmo propuesto realiza la clasificación basándose en si el sistema de cifrado es simétrico o de clave única, si borra los volúmenes sombra, y si sobreescribe y borra los archivos originales. Un ejemplo de sistemas en CAT1 sería AnonPop; en CAT2 estaría Bad Rabit; en CAT3, Jigsaw, Linux.Encoder, AIDS; en CAT4, CryptoDefense, CryptoLocker, CryptoWall, Locky y TeslaCrypt; y en CAT5, encontraríamos a Cerber, Erebus, Petya, SamSam, VenusLocker, WannaCry y ZCryptor.

Como ejemplo, la Figura 2.18 [140] muestra la expansión de un ataque por todo un dominio en un plazo de 3 días (en otros casos, como Nefilim, puede llevar meses [141]). En este ejemplo y de manera abreviada, el ataque se produce con un Acceso Inicial producido por un *script* PowerShell explotando la vulnerabilidad ProxyShell de Exchange Server, que permite la creación de varios *web shell* para descargar en el servidor algunos archivos y ejecutarlos.

En la etapa de Ejecución, un *script* permite crear dos directorios y mover a ellos algunos archivos. Primero borra el planificador de tareas, CacheTask, y lo sustituye por un *script* (*CacheTask.bat*) que utiliza un *Fast Reverse Proxy* (FRP) para exponer el servidor detrás del NAT o cortafuegos, es decir, a Internet. El archivo de configuración del FRP crea un *proxy* http en el puerto 10151/tcp. Por otro lado, se descarga y ejecuta un programa, *plink.exe*, para crear un túnel ssh a la dirección 148[.]251[.]71[.]182 para alcanzar el puerto RDP del sistema Exchange. Tras ello, tiene acceso RDP al servidor.

En la fase de Persistencia en la primera víctima, debido a que la explotación realizada con PowerShell le da privilegios `NT AUTHORITY SYSTEM`, habilita la *DefaultAccount* mediante una orden usando un *web shell*, tras lo cual ajusta la clave para esta cuenta por defecto y añade esta a los grupos `Administrators` y `Remote Desktop Users`. También realiza un volcado de *LSASS* (*Local Security Authority Subsystem Service*), que le da acceso a las credenciales, y roba una cuenta del dominio de administrador que se usa en el movimiento lateral.

Si bien no se utilizan técnicas avanzadas de evasión de defensas, los ata-

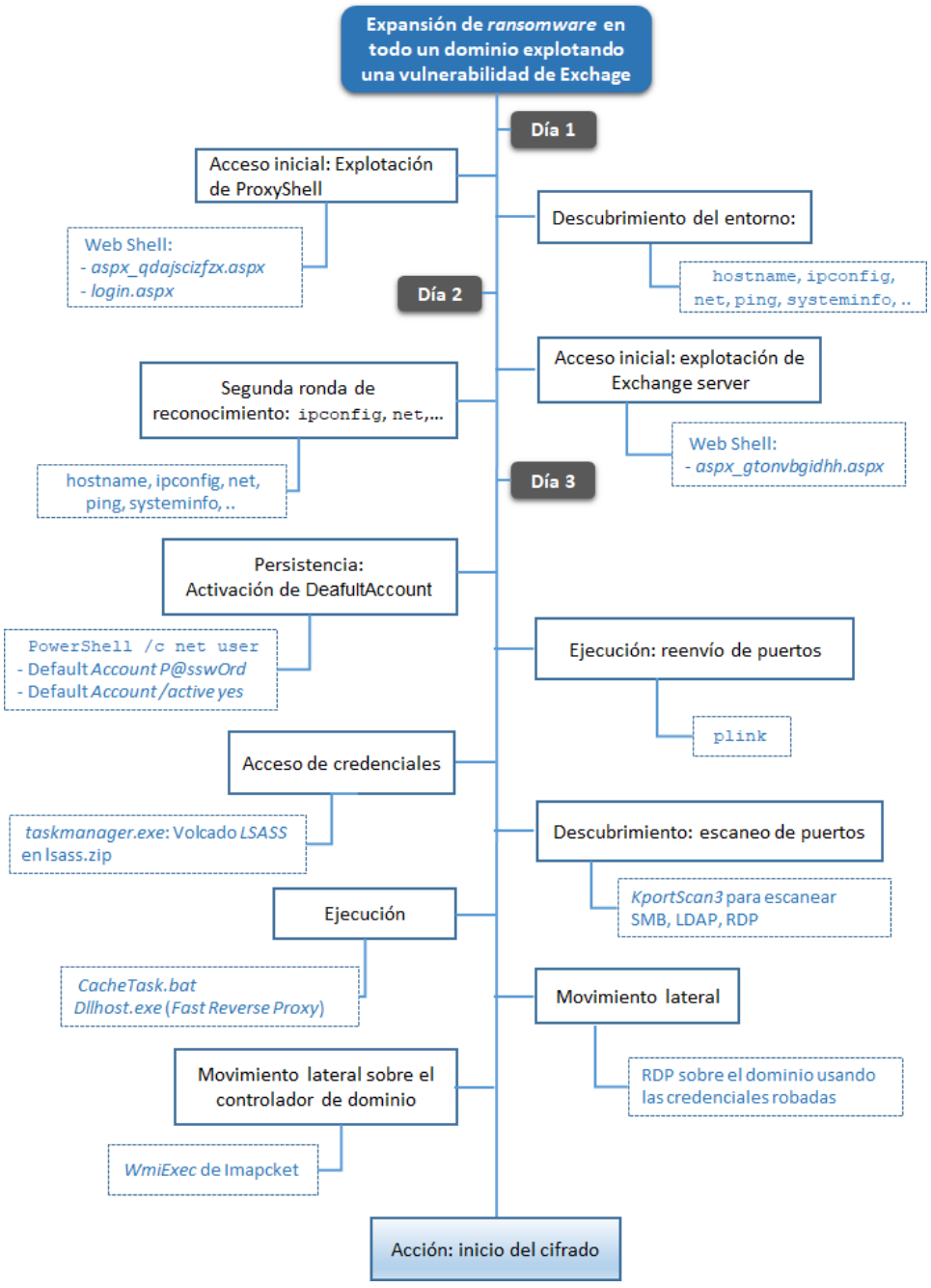


Figura 2.18: Línea temporal de acciones de un *ransomware* hasta alcanzar todo un dominio (modificada de [140]).

cantes enmascaran muchas de sus herramientas. En este sentido, crean el *web shell login.aspx* en la misma carpeta que la página legítima de *login* de OWA (*Outlook Web App*), renombra FRP a *dllhost.exe* para pasar inadvertidos y, por último, el planificador de tareas tiene el mismo nombre de camino que el original.

Uno de los *web shell* utiliza numerosas órdenes convencionales del servidor para escanear la red y, aprovechándose de RDP, realiza el movimiento lateral a otros servidores utilizando la credencial de administrador robada.

Este ataque no utiliza C&C; en su lugar, utiliza el túnel ssh creado. Solo se ex-filtra el archivo *LSASS* y, para ello, se comprime el mismo con el nombre *lsass.zip*.

Para realizar las Acciones, el actor de amenaza utiliza BitLocker y un encriptador de fuentes abiertas, DiskCryptor. En los servidores se utiliza un *script*, *setup.bat*, mientras que en los equipos individuales se emplea una aplicación GUI y *dcrypt.exe* (DiskCryptor), ambos ejecutándose después de acceder al sistema autentificarse mediante RDP en cada equipo. Como se puede observar, en este caso se utilizan herramientas convencionales para el cifrado en lugar de las habituales en un modelo RaaS.

Para finalizar el apartado del modelo de ataque, la Figura 2.19 muestra el ciclo de vida de un incidente con las etapas que acabamos de analizar y las principales técnicas utilizadas en cada una de ellas. Para ilustrar las etapas y técnicas comentadas, la Tabla 2.4 muestra un amplio número de familias de *ransomware* y las técnicas que utilizan en cada etapa del modelo CKC.

2.4.7. Recursos y análisis de muestras

A pesar de la descripción realizada, debemos tener claro que los actores de amenazas plantean de forma continua nuevos métodos de ataque, de forma que no podemos saber qué vectores de ataque se utilizarán en un futuro próximo. Para disminuir esta incertidumbre, la *inteligencia de ciberamenazas* [142] suministra información procesable y temprana sobre las acciones de dichos actores de forma que podamos reforzar las defensas. En este sentido, diferentes organizaciones ofrecen este tipo de información, tal como se muestra en la Tabla 2.5. Un ejemplo es el método propuesto en [143], donde se muestra la correlación entre las búsquedas realizadas por usuarios buscando soporte para el *ransomware* con los ataques analizando los registros de los buscadores web. Otro enfoque diferente es la propuesta de NER [144], orientada a analizar temas relacionados con *ransomware* mediante fuentes externas como foros y dominios de *malware* y utilizando minería de textos.

Otro aspecto relativo al estudio de muestras de *ransomware* de más bajo nivel es la preparación del entorno de análisis. En [145] se describe un entorno amigable basado en la *sandbox* Cuckoo para la identificación de *malware*. La obtención de los indicadores de compromiso (IoC) de una muestra pueden obtenerse con la herramienta @DisCo [146] en conjunción

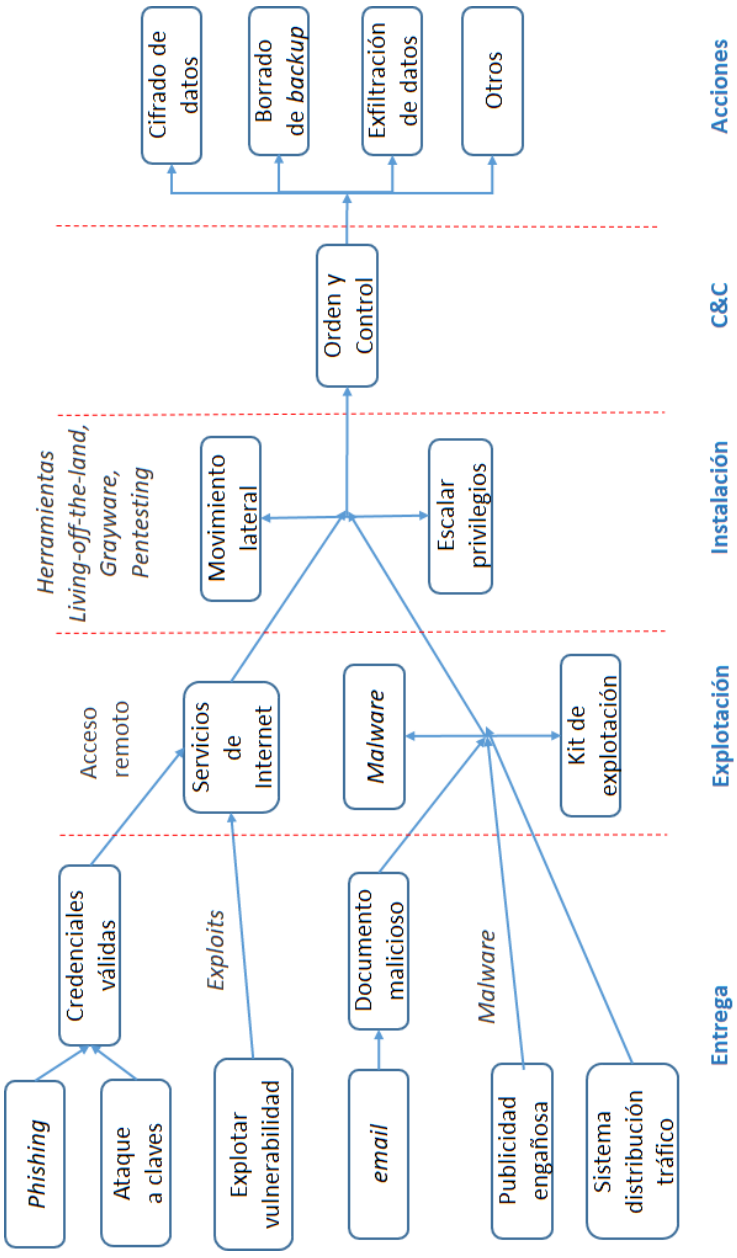


Figura 2.19: Ciclo de vida de un incidente de *ransomware*.

con algoritmos Fuzzy basados en firmas (SSDEEP y TLSH), tal como se propone en [147]. Un trabajo similar es el marco de trabajo descrito en [148], que se centra en las llamadas a la API y, mediante análisis estático y dinámico de diferentes muestras, pretende caracterizar el comportamiento malicioso para el desarrollo de técnicas de detección.

Basado en el análisis forense y la minería de datos, el trabajo [149] extrae componentes del código (ingeniería inversa combinando un enfoque estático y dinámico) de las muestras, que luego son correlacionados con minería de datos al objeto de crear reglas de identificación únicas de cada familia.

Respecto a la identificación de amenazas, un aspecto relevante es el triaje de muestras. Las firmas criptográficas presentan el problema de que dos muestras casi idénticas tienen firmas diferentes. En forense digital, se suelen utilizar métodos de firmas difusas o firmas con reglas YARA. El uso de estos métodos permite a los investigadores determinar si una muestra es maliciosa o benigna, para poder filtrarla adecuadamente. El principal problema que se plantea es la exactitud del filtrado. Este es el objeto del estudio en [150],donde se analizan tres métodos difusos: IMPASH, SSDEEP y SDHASH. El método SDHASH es el que da mejores resultados y estos son comparables a los de YARA. En un segundo trabajo, [151], se extiende el método para la identificación de nuevas muestras y de los actores de amenaza.

2.4.8. Proceso de cifrado de archivos

Como hemos comentado con anterioridad, una de las formas más eficaces desde el punto de vista de alcanzar la extorsión es el cripto-*ransomware*, especialmente cuando usa un cifrado fuerte [152], ya que mientras en las otras variantes puede ser fácil revertir o evadir el ataque, en esta modalidad suele ser irreversible.

Ahora bien, dentro de los cripto-*ransomware*, encontramos diferentes esquemas de realizar el cifrado [153, 154]: cifrado simétrico, cifrado asimétrico y cifrado híbrido. En un enfoque de cifrado simétrico (una única clave para cifrar y descifrar), como la clave de cifrado debe estar almacenada en la propia muestra, hace que sea vulnerable a la ingeniería inversa de la misma y permitiría recuperar la clave. En el cifrado asimétrico (una clave para cifrar y otra diferente para descifrar), el principal inconveniente es que es más lento de realizar que el simétrico y, por tanto, más costoso de aplicar a archivos voluminosos [155]. Este hecho favorecería la detección del ataque antes de finalizar el proceso de cifrado.

La encriptación híbrida usa una combinación de los esquemas simétrico y asimétrico que la hace mucho más efectiva y eficiente.

Tabla 2.4: Familias de *ransomware* y técnicas de ataque.

Familia de *ransomware*	Etapas de *Cyber Kill Chain*																								
	Armamento									Entrega				*Exploit*		Instalación					C&C			Acción
	G	Ce	A	En	Técnicas evasión					IS		M	DT	Kit	E	*Host*			Red		H	GD	B	
					T	D	C	N		P	S					DA	Ba	DR	BM					
Avaddon [156, 157]			√	AES-256 RSA-2048		√	√				√				√		√							I
BlackByte [158, 159]			√	AES		√	√				√				√		√							I
BlackCat [160, 161]		√	√	AES		√	√	√		√			√		√				√					O
Cerber [162] [163, 164]	√	√		RSA RC4	√	√	√			√	√	√		√		√							√	
Crypto-Locker [165]	√		√	AES RSA			√			√		√			√	√						√		X
CryptoWall [166, 167, 168]		√	√	AES		√	√	√						√										
CTBLocker [169, 170, 171]		√	√	AES RSA		√		√		√		√	√	√			√				√		√	
Diavol [172, 173]		√	√	RSA				√			√			√			√	√			√			X

(Continúa en la página siguiente ...)

Familia de *ransomware*	Etapas de *Cyber Kill Chain*																						
	Armamento									Entrega				*Exploit*		Instalación					C&C			Acción
	G	Ce	A	En	Técnicas evasión					IS		M	DT	Kit	E	*Host*		Red			H	GD	B	
					T	D	C	N		P	S					DA	Ba	DR	BM					
DMA-Locker [174, 175, 176]	√		√	AES RSA	√									√			√	√			√			
Ekans [177, 20]	√			AES-256 RSA-2048			√	√			√	√					√	√						I
Entropy [178, 179]		√	√	RC4		√	√	√		√				√				√					√	
FAKBEN [180, 181]		√	√	AES	√					√				√				√			√	√		
GlobeImposter [182, 183, 184]			√	RSA				√		√		√	√	√			√	√						
Hive [185]	√		√	RSA		√				√			√		√	√								
Khonsari [186, 187]		√	√	AES-128 CBC											√			√				√		
Locky [188] [108] [189, 190]	√	√	√	AES CTR-ECB		√	√	√		√			√	√			√				√	√	√	

(Continúa en la página siguiente ...)

Familia de *ransomware*	Etapas de *Cyber Kill Chain*																					
	Armamento								Entrega				*Exploit*		Instalación				C&C			Acción
					Técnicas evasión				IS		M	DT			*Host*		Red					
	G	Ce	A	En	T	D	C	N	P	S			Kit	E	DA	Ba	DR	BM	H	GD	B	
Megacortex (v2) [191]	√		√	AES-256 RSA-4096 Salsa20			√	√		√			√			√			√			
PadCrypt [192]		√	√	RSA AES		√		√		√				√	√					√		
PayCrypt [193]	√	√	√	RSA	√			√		√	√		√		√						√	
(Not)Petya [194, 195, 196]				AES-128	√	√		√	√					√	√			√				W
REvil [197]		√		RC4		√	√	√					√	√								X
Ruyk [198, 199]				AES			√		√				√			√	√			√		X
Sabbath [200]		√	√	AES			√	√			√		√			√					√	X
Sage [201] [202, 203, 204]	√		√	CE ChaCha20 ECC		√	√	√	√			√	√		√	√						
Sodinokibi [205]	√	√		ECDH AES Salsa20	√	√		√	√			√		√		√	√					IX

(Continúa en la página siguiente ...)

Familia de *ransomware*	Etapas de *Cyber Kill Chain*																							
	Armamento									Entrega			*Exploit*		Instalación						C&C			Acción
	G	Ce	A	En	Técnicas evasión					IS		M	DT	Kit	E	*Host*			Red		H	GD	B	
					T	D	C	N		P	S					DA	Ba	DR	BM					
TeslaCrypt [206, 207, 208]	√		√	AES-256 RC4 ECB-CBC		√	√	√		√			√											
TFLower [209, 210]		√		RSA		√	√				√					√			√			√	√	
TorrentLocker [211]	√	√	√	AES CTB-CBC	√		√	√		√	√					√	√				√	√		
WannaCry [212, 213]			√	AES RSA		√	√										√		√					I

Abreviaturas: A: Patrón acceso archivos; B: *Botnet*; Ba: Cifrado *backup* BM: Bloque Mensaje Servidor; C: Código; Ce: Carga de entrega; CE: Curva elíptica; D: Datos; DA: Denegación Acceso; DR: *Desktop* Remoto; DT: Sistema Distribución Tráfico; En: Método de encriptación; E: Explotación personalizada; Ex: Exfiltación de datos; G: Basada en guiones; GD: Generación Dominios; H: Dirección IP en código; I: Inhibe recuperación del sistema; IS: Ingeniería social; M: Publicidad maliciosa; N: Red; P: *Phishing*; S: *Spear phishing*; O: DoS; T: Basado en tiempo; X: Exfiltración; W: *Wiper*.

(Fin de la tabla)

Fuente	**Comentario**	**Enlace web**
Malware Traffic Analysis	Monitoriza tráfico de red malicioso	`https://www.malware-traffic-analysis.net/2019/index.html`
APT Groups and Operations	Información de grupos APT	`https://apt.threattracking.com`
Automated Indicator Sharing	Intercambia información entre el gobierno federal y empresas	`https://www.cisa.gov/ais`
Threatfeeds	Información libre y abierta de inteligencia de amenazas	`https://threatfeeds.io/`
FireHOL	Información relacionada con ataques en línea (IP maliciosas, *malware*, *botnets*, ..	`https://iplists.firehol.org/`
IBM K-Force Exchange	Listado de los últimos riesgos de seguridad mundial	`https://exchange.xforce.ibmcloud.com/`
Malware trackers	Mapas mundiales con distribución geográfica de ciberataques	`https://www.thewindowsclub.com/malware-trer-maps`
SANS Internet Storm Center	Información sobre ciber-amenazas	`https://isc.sans.edu/diaryarchive.html`
Botvrij	IoC de fuentes abiertas	`https://www.botvrij.eu/`
Feodo Tracker	Comparte servidores C&C asociados con Dridex, Emotet, TrickBot, QakBot y BazarLoader	`https://feodotracker.abuse.ch/`
Cyner Cure	Colección de indicadores de seguridad provenientes de *honeypots* y *honeynets*	`https://www.cybercure.ai/`
CIRWAs	*Dataset* de ataque a infraestructuras críticas de 2019-2022	`https://sites.temple.edu/care/ci-rw-attacks/`
ID Ransomware	Identifica *ransomware* por la nota de rescate	`https://id-ransomware.malwarehunterteam.com/`
Spin	*Ransomware tracker*	`https://spin.ai/resources/ransomware-tracker/`

Tabla 2.5: Recursos de ciber-inteligencia para el seguimiento de *ransomware*.

Dentro del esquema de cifrado híbrido podemos encontrar cuatro tipos que son utilizados por varias muestras de *ransomware* de acuerdo con [214]. En el cifrado tipo A, Figura 2.20(a), vemos que el atacante genera un par de claves pública y privada para la víctima ($K_{V_{pub}}$, $K_{V_{priv}}$) y embebe la clave pública en el ejecutable malicioso. Cuando el programa malicioso se ejecuta en la víctima, una clave simétrica K_i se genera por cada archivo f_i para su cifrado, tras lo cual K_i se cifra con la clave pública $K_{V_{pub}}$ y se añade al final del archivo cifrado.

En el segundo esquema, B, 2.20(b), el programa malicioso se distribuye con una clave del atacante K_A. A diferencia del anterior, el par de claves pública y privada se generan en el equipo infectado. La clave privada $K_{V_{priv}}$ se cifra con la clave K_A y se devuelve al atacante y se elimina de memoria. El proceso de cifrado es el mismo que en A.

El esquema de cifrado C, 2.20(c), es similar a A, con la diferencia de que se genera una clave general simétrica K, en lugar de una por archivo.

En el esquema D, conocido como modelo de tres niveles y recogido en la 2.20(d), el *ransomware* contiene la clave pública $K_{A_{pub}}$ del atacante. Después se generan las claves de la víctima ($K_{V_{pub}}$,$K_{V_{priv}}$) y la clave $K_{A_{priv}}$ es cifrada con la clave $K_{A_{pub}}$. El proceso de cifrado de archivos es el mismo de los tipos A y B, donde la clave simétrica es personalizada para cada archivo. Este esquema es el que el usó, por ejemplo, WannaCry y se considera el más seguro, ya que las víctimas no pueden intercambiar las claves. El trabajo [182] muestra la preferencia por modelos basados en generación de claves locales frente a otros medios, debido a la transparencia y seguridad del uso de las API criptográficas del sistema operativo y OpenSSL.

La Figura 2.21 muestra con detalle un ejemplo de los pasos seguidos en un esquema híbrido, donde las claves pública y privada son generadas por el atacante (tipos A, C y D), indicando las acciones en cada etapa del ataque: pre-ataque, ataque en sí y post-ataque. Estas etapas son:

1. Previo al ataque, se establece un servidor de control y órdenes, que generará mediante criptografía asimétrica un par de claves pública y privada. Como hemos indicado, este par de claves suele ser único para cada infección, al objeto de que a una víctima no le sean útiles las claves de otra.

2. En la fase de ataque, la muestra de *ransomware* crea una clave simétrica aleatoria que será utilizada para el cifrado. Para ello, suele hacer uso de la API de cifrado del sistema operativo anfitrión.

3. Tras cifrar los archivos seleccionados con la clave simétrica, la muestra se conecta a su C&C para obtener la clave pública generada previamente.

4. Con esta clave pública se cifra la clave simétrica que está en texto

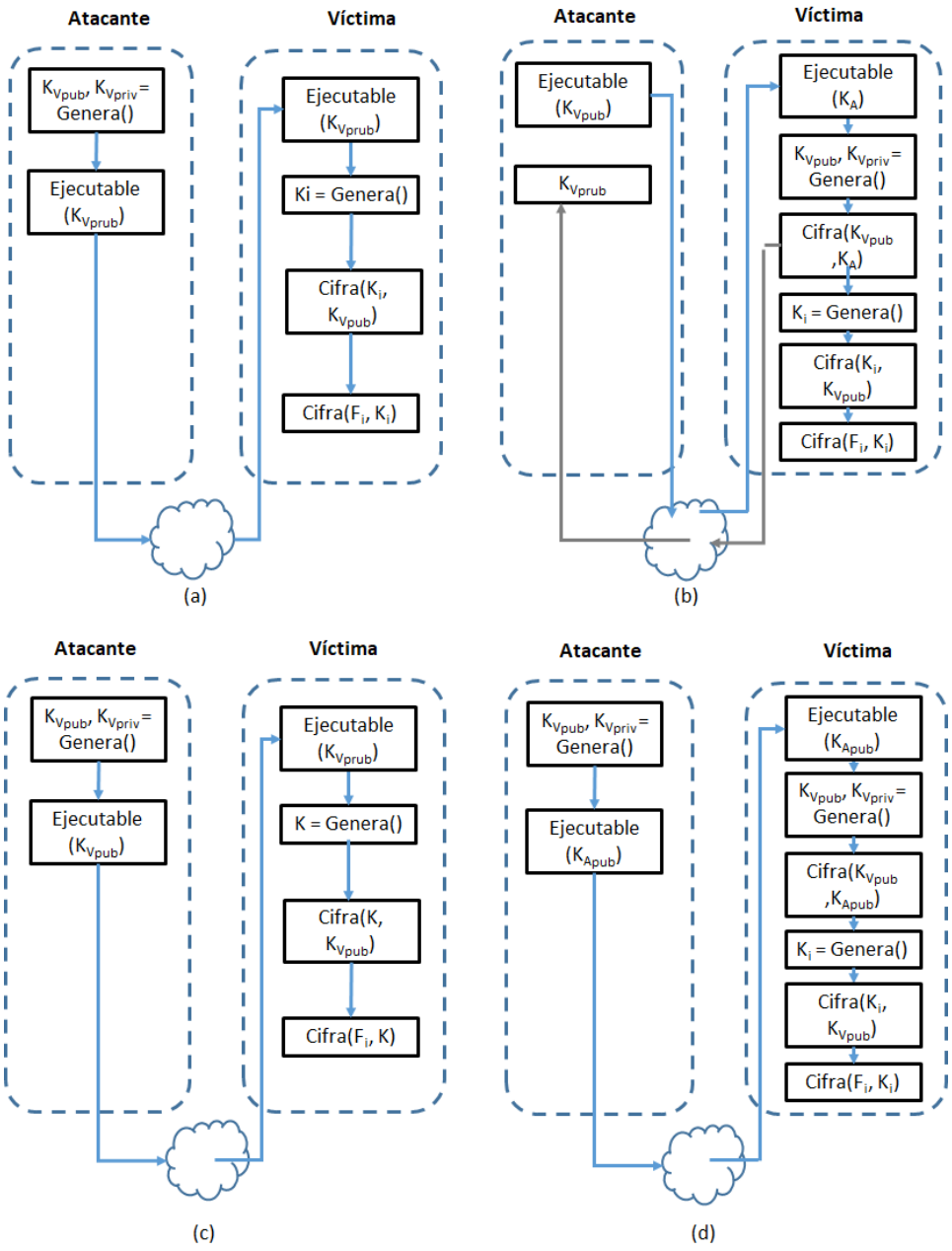


Figura 2.20: Esquemas de cifrado usados por el *ransomware*: (a) tipo A, (b) tipo B, (c) tipo C, y (d) tipo D.

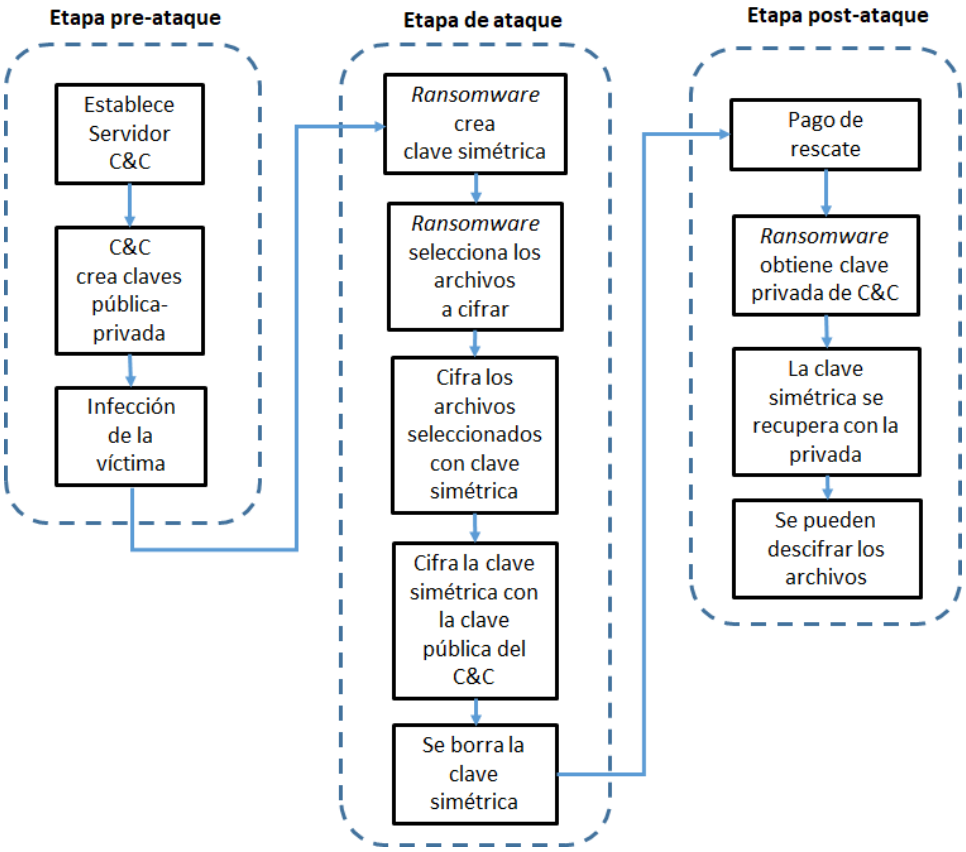


Figura 2.21: Pasos seguidos por un *ransomware* que usa el enfoque híbrido de cifrado.

plano y posteriormente se borra de forma que sea irrecuperable del código del *ransomware*.

5. En la fase posterior al ataque, si la víctima paga el rescate, el *ransomware* recupera del C&C la clave privada con la que descifra la clave simétrica con la que se cifraron los archivos.

6. Para finalizar, se descifran los archivos con la clave simétrica recuperada.

Como resumen, la Tabla 2.6 recoge, respecto a las muestras analizadas en [214], la API utilizada en dichas muestras y las funciones que utilizan. Los algoritmos de cifrado de diferentes familias se ha mostrado con anterioridad en la Tabla 2.4, en la columna 'En(criptación)'. Otro estudio sobre generación de claves en la plataforma .NET podemos encontrarla en [215].

Familia	**Tipo**	**API**	**Función de generación claves (S: simétrica, A: asimétrica)**
Ryuk	A	Windows CryptoAPI	S: `CrypGenKey` A: Embebida
Dharma	A	axTLS embebido SSL	S: `get_random_NZ` A: Embebida
LockBit	A	Windows CryptoAPI + conjunto instrucciones AES-NI + Rijndael optimizado	S: `CryptGenRandom` A: Embebida
SamSam	A	.NET System.Security. Cryptography	S: `RNGCryptoService` Provider A: Embebida
GandCrab	B	Windows CryptoAPI	S: `CryptoGenRandom` A: CryptGenKey
Clop	A	Windows CryptoAPI	S:`CryptGenKey` A: Embebida
Katyusha	A/C	OpenSSL	S: `rand()` A: Embebida
Santch	A	Paquete OpenPGP Go	S: `rand()` A: Embebida
Phobos	A/C	Windows CryptoAPI + implementación syslinux RSA	S: `CryptGenRandom` A: Embebida
Nemty	C/D	Windows CryptoAPI + implementación personalizada AES	S: rand() A: `CryptGenKey`

Tabla 2.6: API y funciones usadas para la generación de claves en diversas familias de *ransomware*.

Al objeto de ver la efectividad del cifrado utilizado por el *ransomware*, analizamos el trabajo [216], donde se estudian 78 herramientas de descifrado desarrolladas por 11 compañías de seguridad frente a 61 muestras de *ransomware*. Los resultados muestran que prácticamente la mitad de dichas herramientas fallan al intentar recuperar los datos cifrados.

Algunos proyectos ofrecen recursos para luchar contra este tipo de ataque. En Estados Unidos encontramos Stop RansomWare [217] y en Europa No More Ransomware [218], que ofrecen cientos de herramientas de descifrado y un detector, Crypto Sheriff, que permite subir archivos cifrados para identificar el tipo de *ransomware*. Si bien este detector puede reconocer muchos tipos, algunos no son reconocidos, como Alcatraz, Avest, BigBobRoss, GandCrab, GetCrypt, Globe v3, Gomasom, InsaneCrypt, Noobcrypt, SpartCrypt y TeslaCryptV4. En parte, esto se debe a que algunas muestras evitan identificar sus nombres o imitan a otras para no ser identificados. Por tanto, revertir los efectos pasa por ser capaces de identificar las muestras. Otros servicios para la identificación de *ransomware* son ID Ransomware [219] o VirusTotal [220]. En [221] podemos ver los diferentes descriptores de que disponen cada una de las plataformas, además de algunos comerciales.

Otro aspecto del funcionamiento del *ransomware*, que a *priori* podría ser útil para el desarrollo de herramientas de detección, es el proceso de selección de archivos a cifrar dentro del equipo de la víctima. El trabajo [222] muestra que no hay realmente un patrón a la hora de seleccionar los archivos objetivo. En cuanto al recorrido del árbol de directorios, algunas muestras exploran en profundidad (primero se recorre en profundidad una rama el árbol de directorios, luego las siguientes ramas) el sistema de archivos mientras que otras lo hacen en anchura (se explora primero el primer nivel de directorios del árbol y luego se va bajando de nivel). En cuanto a la selección de archivos objetivo dentro de una carpeta, podemos encontrar todas las posibilidades: selección por orden alfabético, inverso al alfabético, o aleatorio. La Tabla 2.7 muestra cómo realizan esta selección algunas muestras. Por tanto, para la construcción de una herramienta de detección debemos tener presente que debemos monitorizar todos los objetos del sistema de archivos, ya que el *ransomware* puede comenzar por cualquier punto del sistema de archivos.

De cara a la detección, también son importantes las diferentes formas en las que las muestras de *ransomware* pueden manipular los archivos individualmente en el proceso de cifrado. Estas se recogen en la Figura 2.22. En la opción (a) de la figura, el *ransomware* sobreescribe el archivo cifrado (*p.ej.*, CrypoLocker o CryptoWall). En el esquema (b), tras leer el archivo, el *ransomware* crea una versión cifrada del archivo y posteriormente borra el original (*p.ej.*, FileCoder). Por último, la (c) es similar a la (b) pero esta vez se sobreescribe el archivo original con su versión cifrada (*p.ej.*, CryptoVault).

Para finalizar el apartado y en relación con la selección de archivos por parte de *ransomware*, indicar que cada familia tiene por lo general su propia lista de extensiones de nombres de archivos que son candidatos a ser cifrados.

Ransomware	Exploración	Orden de archivos	Orden de carpetas
GandCrab	Primero profundidad	Alfabético	Alfabético
TelsaCrypt	Primero profundidad	Alfabético	Alfabético
CryptXXX	Primero profundidad	Alfabético	Alfabético inverso
Osiris	Aleatorio	Primero extensión, luego alfabético	Aleatorio
Sage2.2	Aleatorio	Archivos individuales	Aleatorio

Tabla 2.7: Orden de exploración del árbol de directorios y de archivos dentro de una carpeta para diversas muestras de *ransomware* [222]

Estas listas suelen contener extensiones de archivos de datos generados por diferentes aplicaciones de uso común, como pueden ser aplicaciones ofimáticas, de imágenes y vídeos, bases de datos, etc. La Tabla 2.8 muestra una relación de estas extensiones de archivos para algunas de las familias de *ransomware* [223].

2.5. Aspectos regulatorios

Si bien hasta aquí hemos abordado el estudio del *ransomware* desde un punto de vista técnico, es importante comentar que no es la única vía para luchar contra esta amenaza. Desde el punto de vista normativo, los legisladores deben de seguir avanzando para dificultar cada vez más esta actividad delictiva.

A nivel español, el *ransomware* encajaría dentro de una conducta de daños o sabotaje contemplada en el artículo 264 de Código Penal si afecta a datos, programas o documentos ajenos, o en el Artículo 264.2 en lo que se refiere al funcionamiento del sistema informático con agravante, en el caso de realizarse al amparo de una organización criminal. Conductas delictivas similares se recogen en la Directiva Europea 2016/1148 de julio de 2016 (también conocida como Directiva NIS), relativa a las medidas destinadas a garantizar un elevado nivel común de seguridad de las redes y sistemas de información en la Unión, que se transpuso en España mediante el Real Decreto-Ley 12/2018, de 7 de septiembre, de seguridad de las redes y sistemas de información.

En el caso que nos ocupa, el problema radica principalmente en la dificultad de identificar a los autores de la extorsión dado su uso de sistemas anonimizados. Por ello, algunos países ha desarrollado normas para intentar atajar o minorar el problema. En este sentido, EEUU, a través del *Joint Statement of the Ministers and Representatives from the Counter Ransom-*

Ransomware	**Extensiones de nombres de archivos víctima**
TeslaCrypt Alpha Crypt	.sql, .mp4, .7z, .rar, .m4a, .wma, .avi, .wmv, .csv, .d3dbsp, .zip, .sie, .sum, .ibank, .t13, .t12, .qdf, .gdb, .tax, .pkpass, .bc6, .bc7, .bkp, .qic, .bkf, .sidn, .sidd, .mddata, .itl, .itdb, .icxs, .hvpl, .hplg, .hkdb, .mdbackup, .syncdb, .gho, .cas, .svg, .map, .wmo,.itm, .sb, .fos, .mov, .vdf, .ztmp, .sis, .sid, .ncf, .menu, .layout, .dmp, .blob, .esm, .vcf, .vtf, .dazip, .fpk, .mlx, .kf, .iwd, .vpk, .tor, .psk, .rim, .w3x, .fsh, .ntl, .arch00, .lvl, .snx, .cfr, .ff, .vpp_pc, .lrf, .m2, .mcmeta, .vfs0, .mpqge, .kdb, .db0, .dba, .rofl, .hkx, .bar, .upk, .das, .iwi, .litemod, .asset, .forge, .ltx, .bsa, .apk, .re4, .sav, .lbf, .slm, .bik, .epk, .rgss3a, .pak, .big, .wallet, .wotreplay, .xxx, .desc, .py, .m3u, .flv, .js, .css, .rb, .png, .jpeg, .txt, .p7c, .p7b, .p12, .pfx, .pem, .crt, .cer, .der, .x3f, .srw, .pef, .ptx, .r3d, .rw2, .rwl, .raw, .raf, .orf, .nrw, .mrwref, .mef, .erf, .kdc, .dcr, .cr2, .crw, .bay, .sr2, .srf, .arw, .3fr, .dng, .jpe, .jpg, .cdr, .indd, .ai, .eps, .pdf, .pdd, .psd, .dbf, .mdf, .wb2, .rtf, .wpd, .dxg, .xf, .dwg, .pst, .accdb, .mdb, .pptm, .pptx,.ppt, .xlk, .xlsb, .xlsm, .xlsx, .xls, .wps, .docm, .docx, .doc, .odb, .odc, .odm, .odp, .ods, .odt
CoinVault	.odt, .ods, .odp, .odm, .odc, .odb, .doc, .docx, .docm, .wps, .xls, .xlsx, .xlsm, .xlsb, .xlk, .ppt, .pptx, .pptm, .mdb, .accdb, .pst, .dwg, .dxf, .dxg, .wpd, .rtf, .wb2, .mdf, .dbf, .psd, .pdd, .pdf, .eps, .ai, .indd, .cdr, .dng, .3fr, .arw, .srf, .sr2, .mp3, .bay, .crw, .cr2, .dcr, .kdc, .erf, .mef, .mrw, .nef, .nrw, .orf, .raf, .raw, .rwl, .rw2, .r3d, .ptx, .pef, .srw, .x3f, .der, .cer, .crt, .pem, .pfx, .p12, .p7b, .p7c, .jpg, .png, .jfif, .jpeg, .gif,.bmp, .exif, .txt
Locker	3fr, .accdb, .ai, .arw, .bay, .cdr, .cer, .cr2, .crt, .crw, .dbf, .dcr, .der, .dng, .doc, .docm, .docx, .dwg, .dxf, .dxg, .eps, .erf, .indd, .jpe, .jpg, .kdc, .mdb, .mdf, .mef, .mrw, .nef, .nrw, .odb, .odm, .odp, .ods, .odt, .orf, .p12, .p7b, .p7c, .pdd, .pef, .pem, .pfx, .ppt, .pptm, .pptx, .psd, .pst, .ptx, .r3d, .raf, .raw, .rtf, .rw2, .rwl, .srf, .srw, .wb2, .wpd, .wps, .xlk, .xls, .xlsb, .xlsm, .xlsx
CryptoLocker	3fr, .accdb, .ai, .arw, .bay, .cdr, .cer, .cr2, .crt, .crw, .dbf, .dcr, .der, .dng, .doc, .docm, .docx, .dwg, .dxf, .dxg, .eps, .erf, .indd, .jpe, .jpg, .kdc, .mdb, .mdf, .mef, .mrw, .nef, .nrw, .odb, .odm, .odp, .ods, .odt, .orf, .p12, .p7b, .p7c, .pdd, .pef, .pem, .pfx, .ppt, .pptm, .pptx, .psd, .pst, .ptx, .r3d, .raf, .raw, .rtf, .rw2, .rwl, .srf, .srw, .wb2, .wpd, .wps, .xlk, .xls, .xlsb, .xlsm, .xlsx
TorrentLocker (Crypt0L0cker)	.wb2, .psd, .p7c, .p7b, .p12, .pfx, .pem, .crt, .cer, .der, .pl, .py, .lua, .css, .js, .asp, .php, .incpas, .asm, .hpp, .h, .cpp, .c, .7z, .zip, .rar, .drf, .blend, .apj, .3ds, .dwg, .sda, .ps, .pat, .fxg, .fhd, .fh, .dxb, .drw, .design, .ddrw, .ddoc, .dcs,.csl, .csh, .cpi, .cgm, .cdx, .cdrw, .cdr6, .cdr5, .cdr4, .cdr3,.cdr, .awg, .ait, .ai, .agd1, .ycbcra, .x3f, .stx, .st8, .st7, .st6, .st5, .st4, .srw, .srf, .sr2, .sd1, .sd0, .rwz, .rwl, .rw2, .raw, .raf, .ra2, .ptx, .pef, .pcd, .orf, .nwb, .nrw, .nop, .nef, .ndd, .mrw, .mos, .mfw, .mef, .mdc, .kdc, .kc2, .iiq, .gry, .grey, .gray, .fpx, .fff, .exf, .erf, .dng, .dcr, .dc2, .crw, .craw, .cr2, .cmt, .cib, .ce2, .ce1, .arw, .3pr, .3fr, .mpg, .jpeg, .jpg, .mdb, .sqlitedb, .sqlite3, .sqlite, .sql, .sdf, .sav, .sas7bdat, .s3db, .rdb, .psafe3, .nyf, .nx2, .nx1, .nsh, .nsg, .nsf, .nsd, .ns4,.ns3, .ns2, .myd, .kpdx, .kdbx, .idx, .ibz, .ibd, .fdb, .erbsql, .db3,.dbf, .db-journal, .db, .cls, .bdb, .al, .adb, .backupdb, .bik, .backup, .bak, .bkp, .moneywell, .mmw, .ibank, .hbk, .ffd, .dgc, .ddd, .dac, .cfp, .cdf, .bpw, .bgt, .acr, .ac2, .ab4, .djvu, .pdf, .sxm, .odf, .std, .sxd, .otg, .sti,.sxi, .otp, .odg, .odp, .stc, .sxc, .ots, .ods, .sxg, .stw, .sxw, .odm, .oth, .ott, .odt, .odb, .csv, .rtf, .accdr, .accdt, .accde, .accdb, .sldm, .sldx, .ppsm, .ppsx, .ppam, .potm, .potx, .pptm, .pptx, .pps,.pot, .ppt, .xlw, .xll, .xlam, .xla, .xlsb, .xltm, .xltx, .xlsm, .xlsx, .xlm, .xlt, .xls, .xml, .dotm, .dotx, .docm, .docx, .dot, .doc, .txt

Tabla 2.8: Extensiones de archivos víctima para varias familia de *ransomware*.

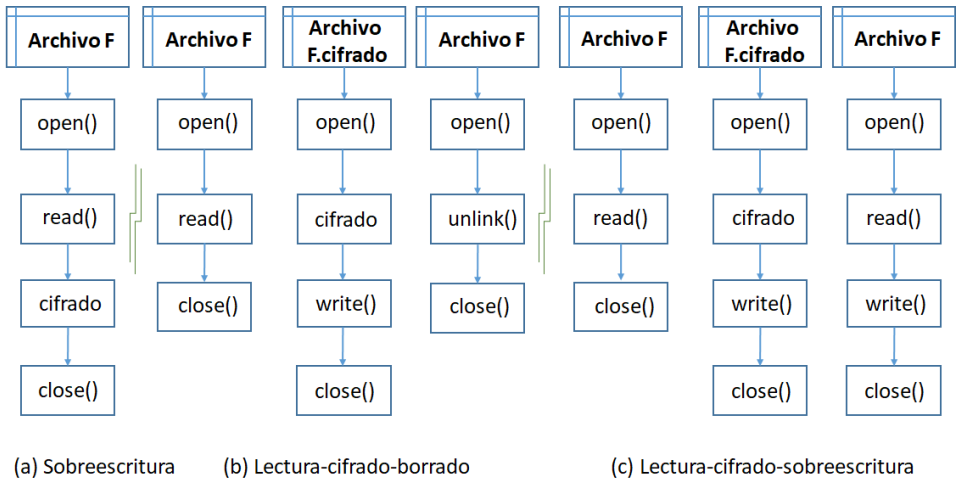


Figura 2.22: Posibles modos de interacción del *ransomware* con los archivos a cifrar.

ware Initiative Meeting [224], establece las bases para aumentar la resilencia, endurecer el sistema finaciero frente al lavado de dinero, disrupción de los ecosistemas RaaS para asegurar la ley, y tomar las medidas diplomáticas necesarias para que los Estados adopten las medidas necesarias contra grupos que operen en sus territorios. También se toman medidas respecto a la legalidad de pagar el rescate [225]. Otro ejemplo reciente es Países Bajos, que ha regulado el uso de inteligencia o fuerza armada para responder a estos ataques [226].

Algunas líneas normativas que se han puesto en marcha o se están preparando en algunos países son, entre otras [227]: *(i)* informar del incidente, bien por el afectado u operador en el caso de infraestructuras críticas; *(ii)* notificar el pago por parte de las víctimas; o *(iii)* reconsiderar el aseguramiento de este tipo de incidentes.

Un aspecto importante es la colaboración entre Estados. El Departamento de Justicia de EEUU creó en abril de 2021 un grupo de trabajo para coordinar, a nivel del gobierno federal, los esfuerzos en la disrupción de los ataques de *ransomware* y su persecución, que van desde detener los servidores utilizados para propagar *ransomware* hasta incautar las ganancias ilícitas de estas empresas criminales. No obstante, esto no es directamente extrapolable a nivel internacional. Los grupos criminales tras el *ransomware* operan generalmente de forma remota, descentralizada y sus miembros tienen responsabilidades separadas. Además, algunas familias, como REvil, han sido cuidadosamente elaboradas para no dejar rastros en sus muestras del país de procedencia del ataque. Por ello, es difícil realizar la atribución de un ataque. Todo ello, hace que la cooperación internacional y la diplo-

macia retrasen la persecución del delito. Esto se ve agravado en ocasiones, ya que cuando se localiza a los responsables de un ataque, el proceso de extradicción puede llevar años [228].

2.6. Conclusiones

En los últimos años están en auge los ataques de *ransomware*, especialmente con la incorporación de tendencias que se han mostrado eficientes en cuanto a resultados. Nos estamos refiriendo a los ataques a la cadena de suministros, la doble extorsión, RaaS, ataques a los sistemas sin parchear, y el *phishing*. En lo que va de año son más de 15 millones de dólares los recaudados por los atacantes [70]. El futuro inmediato nos depara un crecimiento del número de formas de extorsión y una mayor personalización, y un incremento del cifrado intermitente [229].

Así, es de esperar que los ataques de *ransomware* sigan creciendo en volumen y sofisticación. Los atacantes están capitalizando los avances tecnológicos, la adopción de la nube y los entornos de trabajo híbridos mediante el lanzamiento de campañas de *ransomware* dirigidas y persistentes operadas por humanos (RaaS). Por ello, tanto individuos como organizaciones deben seguir mejorando sus defensas para ir doblegando la curva creciente de ataques. Para ello, se necesita seguir mejorando en planes de continuidad de negocio, recuperación de desatres y, por supuesto, herramientas. En este último caso, centrarse especialmente en herramientas predictivas que eviten en última instancia que se cifren los archivos.

Para finalizar el capítulo, sírvase indicar algunas de las previsiones para los próximos años [230]:

1. A partir de 2022 se prevé un crecimiento de este tipo de ataques sobre dispositivos IoT.

2. Para 2024, el 30 % de las organizaciones adoptarán modelos de Acceso a Red de Confianza Cero (ZTNA), para mitigar diferentes amenazas entre las que se encuentra el *ransomware*.

3. Para 2025, el riesgo en ciberseguridad será un factor primordial para evaluar nuevas oportunidades de negocio por parte del 60 % de las organizaciones, incluidos inversores y capitales de riesgo.

4. Para 2025, el 30 % de los estados promulgarán legislación para regular los pagos y las negociaciones en ataques de *ransomware*.

5. Para este mismo año, el 70 % de los CEO invertirán en una cultura de resiliencia en sus organizaciones.

En resumen, se prevé que los ataques de *ransomware* se vayan mitigando en las próximas décadas en tanto en cuanto las herramientas mejoren, los

usuarios sean más consciente de la amenaza y los gobiernos den pasos para que se cumplan las sanciones [231].

En relación a este aspecto, son necesarios cambios en la regulación en la lucha frente al *ransomware* con el objetivo de definir un camino claro para alcanzar mejores políticas de seguridad. En este sentido, también es necesaria la cooperación de los gobiernos con las grandes corporaciones con iniciativas para mejorar la seguridad, como por ejemplo la autenticación de correo electrónico DMARC (*Domain-based Message Authentication, Reporting & Conformance*), pero también para interrumpir las infraestructuras criminales sobre las que operan los ciberdelincuentes. También será necesario ayudar a las organizaciones a desarrollar la capacidad de ciberseguridad necesaria para prevenir, detectar y responder a la amenaza de *ransomware* mediante la adopción de principios y medidas comentadas a lo largo de este capítulo [232].

Capítulo 3

Defensas frente al cripto-*ransomware*: Una revisión

Una vez que hemos estudiado los detalles relativos a las características, construcción y comportamiento del *ransomware* en el capítulo anterior, en este tema abordaremos la investigación y el desarrollo de los mecanismos y técnicas para mitigar/prevenir, detectar y recuperarse de la amenaza. Abordamos con mayor detalle las propuestas de detección que han centrado principalmente la investigación en esta última década, en especial las de detección temprana que persiguen evitar que el *ransomware* cifre con éxito los archivos del equipo víctima.

3.1. Defensa frente al *ransomware*

Inicialmente y de cara a su definición, establecimos una clasificación basada en la severidad del ataque al objeto de abordar las tipologías genéricas del *ransomware* (Sección 2.1). En este apartado abordamos una taxonomía basada en otras características que permita encajar la situación actual de las técnicas anti-*ransomware*, conocer sus fuentes de datos y/o requisitos del sistema, y también compararlas en términos de exactitud y uso de recursos [233] [234] [13].

Dada la amplitud de técnicas, especialmente de detección, vamos a abordar por separado la prevención/mitigación, la detección y la recuperación.

Inicialmente, mostramos una taxonomía de técnicas o mecanismos relacionados con la mitigación o prevención de los ataques de *ransomware*. La Figura 3.1 muestra los cuatro medios básicos de prevención de este tipo de ataques, y que detallaremos en la siguiente sección.

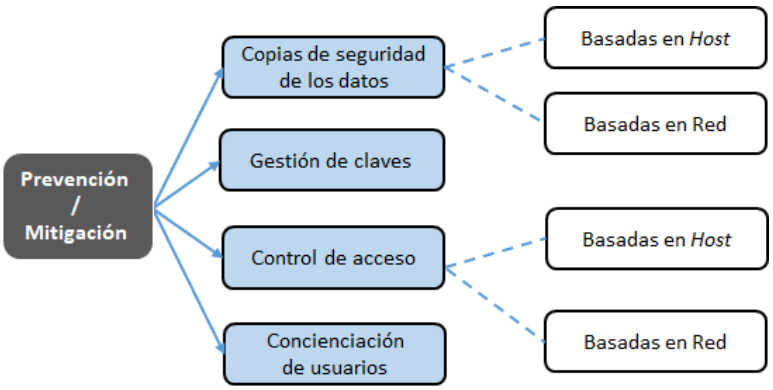


Figura 3.1: Taxonomía general de *ransomware* relativa a la prevención.

Respecto a una taxonomía relativa la detección, indicar que diversos autores han propuesto diferentes clasificaciones en parte tanto a la gran variedad de técnicas y procedimientos desarrolladas al efecto, como en cuanto a la etapa/fase del ataque donde se centran. En [11] se propone una taxonomía basada en tres factores: la severidad, el sistema operativo y el objetivo (individuos u organizaciones). En cambio, [235] presenta una clasificación de alto nivel que divide el *ransomware* en dos clases, *ransomware* criptográfico frente a no criptográfico. El primero es a, su vez, dividido en *ransomware* de cripto-sistemas de clave privada, de clave pública y sistemas híbridos de cifrado.

Una clasificación más detallada se presenta en [13], donde las técnicas de detección se separan en función de las fuentes de datos que utilizan, y las técnicas de *Machine Learning*. De una forma similar, [234] propone una clasificación basada en la fuente de datos utilizada, separando además entre datos de origen local (estáticos o dinámicos) de los procedentes de la red. Una versión de grano fino, con bastantes similitudes a las dos anteriores, la encontramos en [233]. En ella se introduce un criterio adicional que es el procesamiento realizado sobre las fuentes de datos, para incluir todas las técnicas de *hashing*, puntuación, estadísticas, entropía, etc. En este trabajo utilizaremos una taxonomía que unifica las propuestas en [13] y [233].

Otra clasificación diferente la encontramos en [236], donde la detección se divide en centrada en datos y centrada en procesos. Las centradas en datos tratan de seguir la pista de los recursos afectados más que las operaciones maliciosas causantes del ataque. Aquí encontraremos técnicas como las basadas en la entropía, la similitud, los archivos trampa o la monitorización de archivos intercambiados en la red. Por contra, las centradas en los procesos estudian la ejecución de los procesos correspondientes a programas maliciosos para detectar actividades sospechosas. Dentro de esta última clase se realiza una subdivisión adicional: detección basada en eventos y basada en

técnicas de ML. Dentro de las técnicas de ML, se discrimina entre técnicas de detección retardadas (el proceso observado se ejecuta completamente y por tanto el daño ya se ha producido) y las de detección temprana (se identifica o predice el comportamiento dañino antes de que se produzca el cifrado). El problema de esta clasificación parece indicar que solo dentro de estas técnicas se puede realizar detección temprana, cosa que no es cierta como veremos en breve.

En [237] encontramos, por una parte, una clasificación de las características del *ransomware* donde se separa por tipo de objetivo (víctima y plataforma) y etapas de un modelo de ataque (método de infección, comunicaciones C&C, acciones maliciosas), similar al que realizamos con anterioridad pero con un número menor de etapas. Por otro lado, se clasifican las técnicas de defensa en base a los métodos de análisis (estático o dinámico), los de detección, los de recuperación y otros (aquellos que no encajan en las categorías previas). Respecto a la clasificación de análisis, hace una revisión de las técnicas desde el punto de vista del análisis estático y dinámico. Además, todas las clases se separan por el tipo de plataforma a la que están destinadas: PC/*workstations*, dispositivos móviles e IoT/CPS (*Internet of Things / Control Process Systems*). En esta clase se analizan las características estructurales (estáticas) y conductuales (dinámicas), que en taxonomías previas están incluidas en las fuentes de datos utilizadas (datos estáticos o dinámicos). Respecto a la clase de detección, esta se subdivide en 8 subclases en función de la metodología empleada: basadas en listas negras, basadas en reglas, estáticas, métodos formales, computación bio-inspirada, teoría de la información, ML e híbridas. El problema de esta taxonomía es que, dada la naturaleza compleja de muchas técnicas de detección, los autores se ven avocados a separar el nivel inferior a abrir múltiples ramas redundantes para abordar toda la casuística.

La taxonomía propuesta por [52] se basa en su primer nivel en separar estas técnicas en cuando a la naturaleza del análisis (estático, dinámico e híbrido), y en su segundo nivel en las características y en la forma de extraerlas, que se utilizan en la detección. El problema es que esta clasificación esta más centrada en el análisis que en la detección propiamente dicha. Por último, el trabajo [238], si bien no es estrictamente una taxonomía, organiza los mecanismos de detección mediante una clasificación basada en las fuentes de datos utilizadas y el tipo de análisis realizado.

Como comentario general y antes de analizar en detalle las propuestas, hay que indicar que las estrategias de mitigación/prevención, destinadas a detener totalmente o reducir el daño realizado por el *ransomware*, suelen ser genéricas [239]. Las de detección son las que tienen mayor volumen de investigación y se centran en características concretas de este tipo de *malware*. Para las de recuperación (destinadas a revertir las acciones realizadas) encontraremos tanto soluciones genéricas como específicas, y en muchas casos guardan relación con las de prevención.

Muy recientemente, el NIST (*National Institute of Standards and Technology*) ha publicado una guía sobre la gestión de riesgos del *ransomware* donde se presenta el denominado Perfil del *Ransomware* [240], destinado a alinear la prevención/mitigación del *ransomware* (requisitos, objetivos, apetito de riesgo y recursos de las organizaciones) con los elementos del Marco NIST de Ciberseguridad (identificación, protección, detección, respuesta y recuperación). En España, el CCN-CERT publicó la Guía IA-11/18 [241], que recoge medidas se seguridad frente a esta amenaza, tanto preventivas como reactivas y de recuperación.

3.2. Mitigar/prevenir el cripto-*ransomware*

Tanto las organizaciones como los usuarios individuales deben tomar diferentes medidas generales para mitigar esta amenaza ya que no existe un único mecanismo contra ella. Por ello, se necesita un enfoque multinivel donde podemos establecer diferentes mecanismos a diferentes niveles para proteger los activos de nuestros sistemas.

Entre las recomendaciones para la mitigación/prevención debemos utilizar una defensa activa con la combinación de medidas entre las que se encuentran [242, 243, 244, 245, 246, 247, 248, 249, 250]:

- Utilizar programas anti-*malware* en todos los equipos para que analicen automáticamente correos electrónicos y almacenamiento extraible.
- Parchear regularmente tanto el software como el *firmware* ya que a menudo el *ransomware* ataca sobre vulnerabilidades conocidas [251].
- Segmentar la red para reducir el número de sistemas alcanzables. Estudios como [252] muestran cómo efectivamente el alcance de la propagación del *ransomware* se puede contener ajustando la configuración de la red. De forma más general y debido a los problemas que presenta el modelo de perímetro de seguridad, se puede adoptar el Modelo *Zero Trust* [253]. En este modelo no hay perímetro confiable, y las reglas a seguir son: *(i)* nunca confiar, *(ii)* siempre verificar, y *(iii)* asegurar el principio de menor privilegio. En una red de confianza cero, los elementos principales son la autenticación de usuarios/aplicaciones, la autenticación de dispositivos, y una puntuación de la confianza de cada elemento, que garanticen la autenticación de cada flujo en la red.
- Monitorizar continuamente los servicios de directorio y cualquier almacenamiento primario de usuarios de cara a detectar indicadores de compromiso (IoC) o ataques activos.
- Bloquear acceso a recursos web potencialmente peligrosos como servidores de nombres, direcciones IP, puertos y protocolos que se han identificado como maliciosos o que son sospechosos.

- Utilizar solo aplicaciones autorizadas mediante el empleo de listas permitidas o listas blancas.
- Usar cuando sea posible cuentas de usuario estándares frente a privilegiadas.
- Establecer una política *Bring Your Own Device* (BYOD) [254] para el uso de dispositivos personales en el entorno organizacional.
- Evitar el uso de aplicaciones personales en los equipos de trabajo como clientes de correo, redes sociales, etc.
- Educar y concienciar a los empleados y/o usuarios de los sistema TIC sobre los riesgos de la ingeniería social, ya que algunas infecciones pueden materializarse simplemente con el hecho de clicar en enlaces maliciosos. Además, también habría que vigilar dichos comportamientos.
- Asignar, gestionar y verificar periódicamente las credenciales de autorización para todos los bienes y servicios de la organización.
- Asegurar un estricto control de acceso que evite que el *ransomware* acceda al sistema de archivos.
- Realizar copias de seguridad de forma confiable y regularmente de aquellos datos críticos. Especialmente copias de seguridad fuera del sitio (*off-site backup*), es decir, copias de seguridad en un servidor remoto o medio extraible tal como se concluye en [255].
- Protegerse frente a los diferentes mecanismos de ex-filtración de datos identificados por el marco ATT&CK de MITRE [256].

La mayoría de los puntos indicados son generales para mantener de forma global un sistema seguro. De cara a la prevención del *ransomware*, los tres últimos puntos tienen una especial importancia, por lo que los discutiremos con mayor detalle en los epígrafes control de acceso, copias de seguridad y concienciación de usuarios. Posteriormente, analizaremos algunos modelos específicos destinados a mitigar/prevenir este tipo de ataques, teniendo claro que no hay una única contramedida eficiente [257].

El diseño de contramedidas relativas a la defensa frente al *ransomware* puede abordarse desde la aplicación del estándar NIST 800-53 (Controles de Seguridad y Privacidad para Organizaciones y Sistemas de Información Federal) y el Modelo de Madurez de Seguridad de la Información (COBIT de ISACA), que establece cuatro etapas: identificación de vulnerabilidades, evaluación de vulnerabilidades, proponer contramedidas, implementarlas y evaluarlas. Esto permite a las organizaciones medir su estado actual de ciberseguridad y conocer las medidas específicas contra el cripto-*ransomware* y su priorización [258].

Por supuesto, un elemento importante en la prevención es la elaboración de un plan de respuesta a incidentes y de recuperación de desastres, que si bien no tiene el objetivo único de luchar contra el *ransomware*, sí se debe contemplar [259]. Una importante decisión que se debe tomar es si pagar o no el rescate cuando se produzca el incidente, al objeto de tomar una decisión sin la presión del pánico. Otro aspecto relacionado a tener presente es cómo se va a realizar el pago; por ejemplo, si se va a cubrir mediante aseguradora o se va a disponer de un cartera de criptomonedas. También es útil conocer la disponibilidad de la víctima a pagar el rescate para establecer la política adecuada. Por ejemplo, en lo que afecta a individuos se muestra una mayor disponibilidad a pagar entre jóvenes y mujeres cuyos datos son especialmente fotos y que no realizan copias de seguridad con asiduidad [260].

Una pequeña recesión relativa a si pagar o no el rescate. La respuesta corta es no, pero la larga es más complicada. Lo primero que se debe plantear es si es la única opción. Si lo es, hay que tener presentes riesgos éticos, técnicos y legales. Desde el punto de vista ético, pagar el rescate supone estar financiando una organización criminal, que tendrá más fondos para mejorar los ataque y motivos para seguir explotando este tipo de ataques. Desde el punto de vista técnico, hemos de indicar que las estadísticas recogen que el 80 % de las víctimas de un ataque vuelven a ser atacadas, bien porque son un objetivo fácil bien porque se sabe que pagan. Desde el punto de vista legal, numerosos países contemplan sanciones legales contra quieres pagan por la extorsión o pagan a través de entidades sancionadas [261]. Como muestra el trabajo [262], el compromiso de no pago, bien mediante sanciones bien eliminado los seguros cibernéticos, reduciría la incidencia de éxito de este tipo de ataques.

Una vez que se ha decidido pagar, se necesita un negociador, y hacer caso a sus recomendaciones. Un negociador puede saber qué grupos de *ransomware* suministran llaves de descifrado rotas y que, por tanto, no funcionan [263]. Además, el negociador debe conocer los grupos que están sancionados, como son Evil Corp, responsable de Dridex (que desarrolló Hades y PayloadBIN, para evitar las sanciones), WastedLocker, Grief (que indicaban que si se utilizaba un negociador se destruirían los archivos), o DroppelPaymer. También está sancionado SUEX, un operador de cambio de criptomonedas. Si bien algunos trabajos han abordado este tema desde la teoría de juegos [264, 265], o cómo se deber realizar una negociación ética [266, 267], la opinión mayoritaria es no pagar.

Con el pago no se acaba el problema. La recuperación es un proceso complejo. Para comenzar, las herramientas que se suministran por los atacantes no suelen ser muy buenas o pueden estar infectadas por *malware*. El siguiente paso es decidir si restaurar los archivos en el sistema actual o reemplazar el sistema antes de reponer los archivos, de cara a evitar posibles artefactos remanentes del *ransomware*. La solución más aceptada es construir un nuevo

sistema y restaurar los archivos, lo que lleva tiempo y es más caro. Esto hace que, a veces, los costes de recuperación después de pagar se dupliquen [268]. Por último, la organización debería actualizar los sistemas de seguridad con nuevas tecnologías y personal para no volver a ser atacados.

En la lucha preventiva contra el *ransomware*, la recuperación del desastre (DR) deberá incluir: *(i)* definir claramente los objetivos de recuperación; *(ii)* establecer unos objetivos realistas tanto de punto de recuperación (RPO) como de tiempo de recuperación (RTO); *(iii)* un plan para probar los objetivos y realizar los ajustes del plan basándose en los resultados; *(iv)* saber cuándo es el momento de pedir ayuda externa; y *(v)* comprender cuándo será necesario pagar el rescate. Una experiencia de recuperación de un ataque de este tipo se puede encontrar en [269].

En la respuesta a incidentes (RI), el equipo de respuesta debería conocer: *(i)* cuál es la familia a la que pertenece el *ransomware*; *(ii)* cuál fue el vector de acceso inicial; *(iii)* cuánto tiempo lleva el grupo del *ransomware* en la red; y *(iv)* qué archivos se han ex-filtrado. Por supuesto, estos elementos deben tener en consideración los aspectos legales y éticos necesarios [270].

3.2.1. Control de acceso

La utilización de mecanismos de control de acceso son básicos para evitar que el *ransomware* escale privilegios una vez que accede a los archivos con las credenciales de un usuario [271]. Se recomienda implementar los principios de menor privilegio y separación de funciones mediante un control de acceso basado en roles que restrinja en la medida de lo posible el acceso a datos en el sistema de archivos y que, de forma rutinaria, audite los permisos y roles.

La propuesta *AntiBotics* [272] hace uso conjunto de autenticación biométrica y humana, como Captchas, para prevenir el borrado o modificación de datos al imponer el control de acceso de forma periódica mediante retos de identificación. Los permisos de acceso se asignan mediante una regla específica del administrador y la realimentación de los retos presentados cuando se intentan borrar o modificar datos. Las muestras de *ransomware* actuales no son capaces de evadir este mecanismo pero podría hacerse si se inyecta código en un proceso legítimo.

Las soluciones basadas en listas blancas parecen una solución prometedora y efectiva de cara a la prevención [273]. En este sentido, la propuesta [274] se basa en la creación de listas blancas de programas por tipo de archivo, es decir, una lista de programas a los que les está permitido acceder a los archivos. Este mecanismos bloquearía efectivamente a cualquier programa que intentase acceder a una archivo y no esté en la lista, sea un *ransomware* conocido o no.

También se aborda la posibilidad de diferir las decisiones de control de acceso al sistema de archivos cuando sea necesario para poder observar las consecuencias de las peticiones de acceso y poder deshacer los cambios, si es

necesario [275]. La experimentación realizada con numerosas muestras pone de manifiesto la viabilidad de la propuesta.

Un enfoque diferente es el denominado *Moving Target Defense* (MTD) [276], donde se cambian las extensiones de los archivos de manera continua y aleatoria como medida preventiva. De esta forma, el ataque no surte efecto dado que los objetivos están cambiando constantemente. Se ha mostrado efectivo, detectando 141 ataque de los 143 lanzados, a un coste relativamente bajo (de menos de 1s para renombrar 1.000 archivos).

Sophos ha desarrollado la herramienta de mitigación Intercept X, que es capaz de eliminar familias APT de día cero. Dicha herramienta utiliza el análisis conductal para evitar que el *ransomware* escriba en el registro [277]. Dicha herramienta utiliza técnicas de prevención de *exploits* para bloquearlos antes de que puedan escribir en el registro. Entre sus características se encuentra la herramienta Crypto Guard, que puede recuperar archivos cifrados.

Por otro lado, Microsoft ha liberado dos productos denominados *Defender for Endpoint* y *Defender for Identity* de cara a proteger los sistemas del *ransomware*. El primero de ellos permite el acceso a las carpetas solo por parte de las aplicaciones de confianza [278], supervisando los eventos sobre las mismas. El segundo está destinado a analizar los eventos del directorio activo local para identificar, detectar e investigar amenazas avanzadas, identidades puestas en peligro y acciones malintencionadas dirigidas a la organización por parte de usuarios internos, en especial las de movimientos laterales por compromiso de credenciales [279].

Ya que hay muestras de *ransomware* que utilizan los generadores de número seudoaleatorios, en [280] se propone considerar el generador de números seudoaleatorios como un recurso crítico y controlar el acceso a su API para detener a las aplicaciones no autorizadas. En la experimentación realizada, esta estrategia tiene un 97 % de eficiencia.

Otra línea de trabajos son las propuestas que descansan en el *firmware*. KEY-SSD [281] propone un mecanismo para que las aplicaciones no autorizadas no sean capaces de leer los datos de un archivo incluso si ha sobrepasado la protección del sistema de archivos, basado en denegar las peticiones de bloques si no se conoce la clave registrada en disco. Para la prevención de ataques como Petya dirigidos al MBR, [282] propone una arquitectura basada en hardware para proteger y controlar el acceso al proceso de pre-arranque y sus datos.

Respecto al control de acceso a la red, varias propuestas utilizan *Software Defined Network* (SDN) para evitar la propagación por la red de forma que la muestra de *ransomware* no alcance nuevos equipos de la misma. En [167] se utilizan listas negras dinámicas de C&C para que el *ransomware* no pueda obtener las claves de cifrado. En [283] encontramos una idea similar pero centrada en la detección de anomalías en el tráfico de los puertos 139 y 445, utilizados por SMB.

3.2.2. Copias de seguridad

Las copias de seguridad son un mecanismo genérico básico de cualquier sistema informático de cara a la recuperación de la información frente a cualquier posible riesgo. Disponiendo de dichas copias, frente a una amenaza *ransomware* solo se verían afectados los datos creados/modificados tras la última copia de seguridad. Por tanto, es necesario establecer un equilibrio entre la cantidad de datos a salvaguardar, la frecuencia de las copias y el tiempo que se mantienen. Además, frente al *ransomware* también debemos asegurarnos que estas copias de seguridad no sean accesibles desde el equipo afectado por el ataque.

La propuesta *FlashGuard* [284] no descansa en el software para hacer la copia, sino que hace uso del hecho de que el *firmware* de los discos de estado sólido (SSD) no sobreescriben los bloques de almacenamiento sino que los copian en bloques menos usados. Por ello, los autores han modificado el *firmware* del disco para recolectar la basura más lentamente con objeto de que se puedan recuperar bloques antiguos en caso de ataque. Las pruebas realizadas muestran que se recuperan los datos encriptados con una sobrecarga pequeña.

En base al estudio del proceso de ataque del *ransomware*, el funcionamiento de las copias de seguridad y de la habilidad de estas para abordar los ataques, en [285] se realizan mejoras para la evaluación de riesgos de seguridad frente ataques de este tipo y se presenta una herramienta que permite analizar los sistemas de copia de seguridad.

Amoeba [286] es un sistema de copias de seguridad y restauración autónomo para SSD que contiene un acelerador hardware para detectar la infección de páginas (unidades constitutivas de un bloque de NAND *flash*), además de un mecanismo de grano fino de control de copias para minimizar la sobrecarga de las mismas. Las pruebas realizadas en el simulador SSD de Microsoft muestran una mejor eficiencia que la propuesta *FlashGuard*.

La aplicación denominada Safe Zone [287] mantiene en un único archivo, área segura, todos los archivos comprimidos del usuario; es decir, funciona como copia de seguridad. Este archivo se mantiene en un modo de escritura sin parada, por lo que el sistema operativo evita que pueda modificarse por otra fuente. Esta aplicación mantiene un registro *File Watcher* que recopila todos los eventos de la zona segura. Si se produce un ataque de *ransomware*, puede recuperar la última copia de seguridad de la zona segura.

Respecto a propuestas de modificaciones del sistema operativo, *Redemption* [288] mantiene un búfer transparente de operaciones de E/S sobre el almacenamiento permanente de cara a monitorizar los patrones de dichas solicitudes por parte de las aplicaciones en busca de firmas de *ransomware*. Cuando se observa un patrón de solicitud que puede indicar actividad de *ransomware*, se termina el correspondiente proceso peticionario y se restauran los datos. Las pruebas realizadas con familias actuales de *ransomware*

demuestran que no se pierden datos a coste de introducir una pequeña sobrecarga del 2,6 % usando cargas realistas.

Recientemente, [289] propone un esquema de copias de seguridad denominado RAP (*RAnsomware Protection*), cuyo objetivo es cubrir las deficiencias de los sistemas tradicionales (más enfocados a la eficiencia de recuperación de datos), centrándose en los ataques a la confidencialidad y DoS. Utiliza una optimización todo-o-nada (AONT) y establece un sistema para ajustar la carga de las copias y su recuperación a través de un canal seguro mediante el uso de la tecnología *Blockchain*.

Otra solución propuesta por Dell es el sistema EMC [290], que duplica todas las operaciones para escribir o añadir a un servidor. Una se realiza en local (se mantiene en el sistema de producción), la otra en la red (sitio de recuperación). Se utiliza una ventana de tiempo para medir la tasa de de-duplicación de un tamaño de datos arbitrario. Si se supera cierto umbral en la tasa, se entiende que hay un ataque de *ransomware*. En este caso, se pueden suprimir las operaciones de escritura/añadidura pendientes de realizar en el sitio remoto, para su protección.

Algunas propuestas van un paso más allá y proponen la creación de un sistema de archivos específico, como Model Core [291]. Este es un sistema de archivos descentralizado para protegerse del *ransomware*. El sistema puede comprobar en tiempo-real las peticiones de los clientes y ver si un archivo está, o no, infectado o dañado. La integridad de los datos se realiza monitorizando las estructuras de metadatos del archivo ejecutable.

3.2.3. Concienciación de usuarios

Como indicábamos en el Apartado 2.2, gran parte de los vectores de ataque del *ransomware* son los mismos del *malware* en general; concretamente, la ingeniería social y el *phishing*. En este sentido, es de especial importancia la concienciación y educación de los usuarios de cara a afrontar este tipo de amenazas. También es posible utilizar un paso intermedio mediante el uso de *malware*, concretamente un troyano bancario para su instalación, como es el caso de Conti, que recientemente ha utilizado IcedID como paso previo para *ransomware* [292].

El estudio [293] está destinado a destacar los vectores relevantes de esta amenaza e identificar los aspectos de concienciación y educación que se pueden aplicar a mitigar la ingeniería social de los ataques de *ransomware*. Este concluye que las soluciones basadas en software (gamificación, simulación, etc.) son adecuadas para alcanzar el objetivo. En [294] encontramos otro estudio similar pero centrado en el *spear phishing*.

Aplicando la teoría de motivación de la protección (PMT), los autores de [295] han investigado la motivación del usuario de sistema de computación para adoptar medidas contra el *ransomware*. El estudio experimental indica que factores que afectan a la protección están mediados por el miedo,

como son la severidad y el cómo se percibe la vulnerabilidad de una amenaza. Destaca la autoeficacia como un factor importante para afrontarlo. Las recompensas inadaptadas y los costes de respuesta tienen una influencia negativa en la motivación de la protección.

En este sentido, [296] indica las precauciones que los empleados de una organización deben tener frente a este tipo de ataque, que incluyen: *(i)* instalar antivirus/anti-*malware* en todos lo computadores y dispositivos móviles de la organización, *(ii)* poner claves fuertes y únicas para los equipos personales y de trabajo, *(iii)* realizar copias de seguridad regularmente en dispositivos externos, *(iv)* nunca abrir adjuntos de correos sospechosos, y *(v)* usar tecnologías *mirror shielding* como forma de protección de respaldo de datos.

El análisis de las TTP de diferentes familias de *ransomware* se ha usado para crear un modelo predictivo de las características de todas las familias denominado RANDEP (*RANsomware and DEPloyment*) [297]. Este modelo permite determinar cuándo los usuarios son conscientes del despliegue del *ransomware* en su sistema.

Más concretamente, analizando cómo los individuos pueden evitar caer en ataques de *phishing*, [285] propone varias recomendaciones: *(i)* segmentar los usuarios de una organización en base a factores como familiaridad con el *phishing* y el nivel de impacto en sus trabajos, y *(ii)* desarrollar una formación específica para cada grupo que puede incluir ejemplos de la vida real, de la seriedad del problema y el daño causado, etc. En esta línea, se puede incluir la realización ejercicios de mesa que ayuden a identificar brechas potenciales y garantizar la corrección de los procesos de mitigación y recuperación de amenazas [298].

3.2.4. Metodologías diversas de mitigación/prevención

Un medio para abordar tanto la prevención como la recuperación de un ataque de *ransomware* es que seamos capaces de obtener las claves de cifrado mediante técnicas de *sniffing* de tráfico, tal como se propone en [299]. Esta propuesta diseña un módulo de captura y análisis de tráfico de red entre la muestra y el servidor C&C para poder extraer las claves. Una vez obtenidas, aunque el cifrado tenga éxito, podemos recuperar los archivos mediante esas claves.

En otra línea de trabajo, algunos autores establecen cómo la expansión de *ransomware* que utiliza criptomonedas puede detectarse analizando las tasas de estas y la congestión de la *Blockchain* asociada [300]. Si las tasas suben, proponen cerrar los mecanismos de seguridad del sistema para evitar que el *malware* entre en el sistema [301].

Dado que el *phishing* es un vector de ataque muy frecuente, algunos investigadores proponen el análisis basado en ML destinado a determinar si una URL es maliciosa o benigna, a partir de las características protocolo,

dominio y ruta [302].

Otra opción es un esquema de prevención multinivel como el propuesto en [303, 304] y basado en el despliegue de software anti-*malware* en las máquinas individuales, la configuración adecuada de los cortafuegos, el filtrado de DNS/Web, la seguridad del email, las copias de seguridad y entrenamiento del personal. De esta forma, se intenta detener la infección pero, si esta prospera, la defensa multicapa da una opción de recuperar los datos.

El uso de redes definidas por software (SDN) puede ser utilizado para analizar el proceso de cifrado de una muestra de *ransomware* para mitigar la infección del sistema [168].

Algunas revisiones de la literatura recientes sobre trabajos de prevención y mitigación son [305, 306], donde se recogen muchas de las propuestas indicadas previamente.

Respecto al uso de metodologías específicamente de mitigación, la unión del marco de preparación digital forense (DFR) para la recolección proactiva de artefactos, la información del registro de Windows y el marco de preparación RAM (W3RF) permiten capturar artefactos procedentes del sistema o la red. Todos los artefactos obtenidos son almacenados en una base de datos a través de un canal con su propio mecanismo de autenticación para el posterior análisis forense que permitiría recuperar las claves de cifrado y, por tanto, recuperarse del incidente [307].

3.3. Predicción y detección del cripto-*ransomware*

Antes de iniciar el apartado, hemos de distinguir entre los términos detección y predicción. Por detección entendemos la fase en la que, mediante diferentes técnicas, tenemos conocimiento del ataque durante su transcurso o con posterioridad. Por predicción, o detección temprana, entendemos la fase que permite aplicar medidas para evitar el ataque; en el caso de *ransomware*, el cifrado de los datos. Si bien debemos aclarar que en muchos casos ambos términos se suelen utilizar de forma intercambiable, son preferibles los métodos de predicción al objeto de que no se produzca el cifrado de nuestros datos, lo que haría innecesaria la recuperación.

Existen numerosos trabajos dedicados a la detección de *ransomware*, es decir, a localizar al programa malicioso y una vez localizado poder detenerlo y eliminarlo [308]. Como indicábamos con anterioridad, para analizar dichas propuestas vamos a considerar: *(i)* la fuente de información que utilizan, y *(ii)* el procesamientos que realizan, tal como se muestra en la taxonomía de la Figura 3.2 (realizada a partir de [233] y [309]).

3.3.1. Fuentes de datos

En este apartado estudiaremos de forma separada las técnicas usadas en función del origen de los datos.

- **Información de sistema**

Para poder realizar la detección, muchas técnicas necesitan previamente recoger información del sistema operativo, las aplicaciones o el *hardware*. Dichos datos pueden provenir del espacio del *kernel*, del espacio de usuario, o una mezcla de ambos. Los datos provenientes del espacio *kernel* son valiosos pues pueden indicarnos cuál es el estado interno del mismo pero, por contra, tendremos que tener privilegios de *root* para obtenerlos. Otro problema, no menos importante, es que el desarrollo de un módulo o controlador para el *kernel* es una tarea compleja y el mínimo error puede provocar el fallo general del sistema (*kernel panic*). La instalación de un controlador *kernel* puede ser factible en plataformas con Windows o Linux, pero es inviable en dispositivos móviles, ya que *rootear/jailbreak* un sistema móvil eliminaría gran parte de los mecanismo de seguridad del dispositivo y sería contraproducente. Los datos de usuario son también valiosos y más fáciles de obtener pero no cuentan toda la historia del estado del sistema.

En este sentido, obtener datos de ambos espacios reflejará de forma más precisa el estado del sistema y las aplicaciones que se ejecutan en él. Veremos este último enfoque cuando hablemos de la aplicación AMon [310] para Android en la Sección 6.3.

A partir de la observación realizada en [311], que muestra cómo el *ransomware* tiende a modificar y añadir muchos valores al Registro de Windows, este trabajo propone la detección mediante la monitorización continua del Registro, junto con la actividad del sistema de archivos. En [312] se analizan los archivos de registros, *logs*, y se extraen varias características relevantes para la actividad del *malware* en general y del *ransomware* en particular, especialmente cuando los archivos de registro contienen sobre todo eventos benignos. Además, la solución propuesta es resiliente al polimorfismo.

Para tener una visión completa del sistema de archivos en plataformas Windows es frecuente implementar *Windows Filesystem Minifilter Driver* [313], que permite un acceso sin restricciones a las peticiones realizadas de paquetes de E/S (IRP). Estos paquetes IRP contienen información como el tipo de solicitud y el contenido del búfer de usuario. Este mecanismo se usa en sistemas como UNVEIL [314], Redemption [288], ShieldFS [315], *Data Aware Defense* [316] y la propuesta en [317].

Si no nos importa perder algo de flexibilidad a costa de simplicidad cuando monitorizamos eventos *kernel*, podemos usar *Fibratus* [318], que es una herramienta escrita en Python que permite capturar información del *kernel* como E/S del sistema de archivos, tráfico de red y actividad del Registro.

Propuestas como [233] utilizan la correlación entre los *bytes* escritos en un archivo, que deben tener un valor bajo para los archivos cifrados, o también la distancia de edición de los nombres absolutos de archivos.

Entre las fuentes de datos del *kernel* encontramos la red, el registro, el

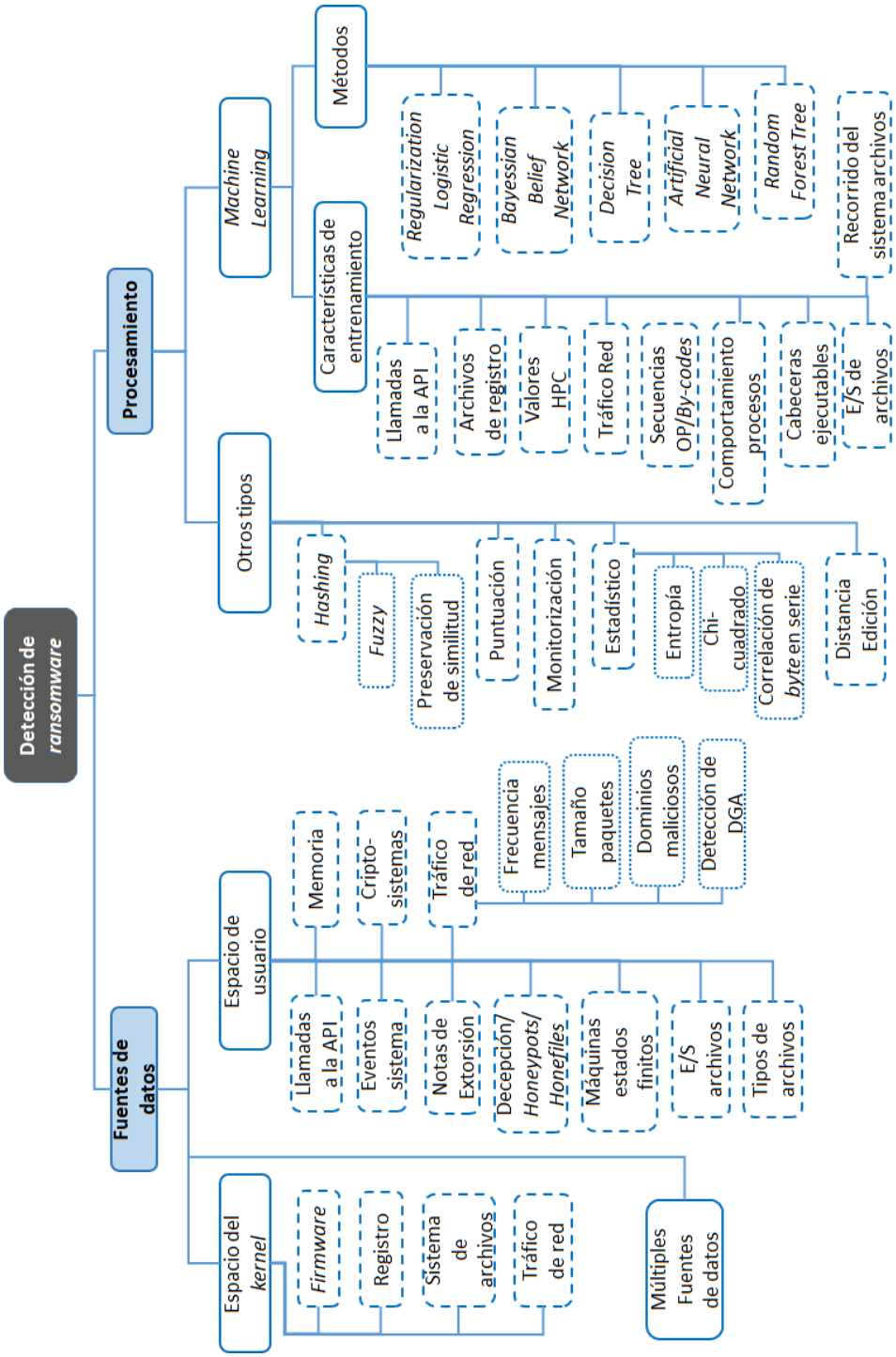


Figura 3.2: Taxonomía de mecanismos de detección de *ransomware*.

firmware, y eventos del sistema de archivos. La monitorización de eventos de red puede mostrar conexiones con servidores *Command & Control* (C&C), paquetes de red que pueden exfiltrar datos, como claves de cifrado, y los archivos de *logs* pueden indicar comportamientos diferente de la actividad de base. Trabajos como [235] y [319] detectan *ransomware* que utiliza algoritmos de generación de dominios (DGA) mediante la monitorización de tráfico DNS.

Un trabajo más reciente en [320] propone la construcción del controlador Ransomware SSD (RSSD), que permite reconocer tres tipos de ataques: 1) ataques de recolección de basura, que explota la capacidad del almacenamiento y mantiene la escritura de datos para activar el recolector de basura, forzando así al disco a liberar los datos retenidos; 2) ataques de sincronización, que reducen intencionadamente el ritmo de cifrado de datos para escapar a los mecanismos de detección basados en patrones de E/S; y 3) ataques de recortes, que utiliza la orden *trim* disponible en SSD para borrar datos físicamente. Para ello, RSDD implementa un registro asistido por hardware que retiene de forma conservadora las copias antiguas de datos y realiza las peticiones de almacenamiento con una pequeña sobrecarga. También emplea el protocolo *Non-Volatile Memory Express* (NVMe) aislado por hardware a través de Ethernet para expandir la capacidad de almacenamiento local mediante la descarga transparente de los registros de servidores o la nube de una manera segura. Además, permite el análisis posterior al ataque construyendo una cadena de evidencias confiable de las operaciones de almacenamiento para asistir a la investigación de dicho ataque.

El sistema ShiledFS, citado con anterioridad, permite recuperar primitivas de cifrado de la memoria de los procesos así como material relacionado con claves. Otra forma de recolectar información del espacio de usuario es la usada por PayBreak y UShallNotPass [280], que utilizan ganchos para interceptar llamadas a las bibliotecas criptográficas.

- **Nota de extorsión**

Una vez que el *ransomware* ha cifrado los archivos de la víctima, muestra una nota de extorsión que indica al usuario lo ocurrido y cómo debe proceder para pagar el rescate. Esta nota suele guardarse como un archivo de texto o mostrarse en pantalla. El análisis tanto estático como dinámico realizado en [321] muestra las características del *ransomware* NEFILIM. Este se ejecuta con privilegios de administrador dado que la nota se escribe en el directorio raíz de la máquina, ya que si no fuese así se escribiría en la carpeta *AppData* del usuario. El archivo de la nota, denominado *NEFILIM-DECRYPT.txt*, se crea y escribe con las llamadas `CreateFileW()` y `WriteFileW()`, respectivamente. Cuando esta nota se escribe en pantalla, tanto la captura de la misma como el texto, son analizados para determinar la presencia del *ransomware*, como veremos a continuación.

El trabajo RanDroid [322] detecta el *ransomware* empotrado en aplicaciones de Android comparando la similitud estructural de un conjunto de imágenes recogidas de la aplicación inspeccionada frente a un conjunto de imágenes de extorsión extraídas de otras variantes del *ransomware*. Para ello, mediante análisis estático, RanDorid descompila la apk y extrae las imágenes de la carpeta de recursos y los archivos de diseño XML. Para el análisis dinámico, se interactúa con la aplicación, sin necesidad de instrumentación, con un generador de entradas de prueba guiado por la interfaz de usuario con el objetivo de disparar eventos de la app, capturar las actividades y recolectar imágenes adicionales. Este marco fue probado con 300 aplicaciones (100 con *ransomware* y 200 benignas) alcanzando un 91 % de tasa de aciertos.

UNVEIL [314] tiene en parte también como objetivo detectar la pantalla de *ransomware* de bloqueo. En su caso, se apoya en la característica de que esta pantalla ocupa casi o todo el monitor. Esta herramienta toma varias instantáneas de la pantalla antes y después de la ejecución de la muestra, que se analizan y comparan con métodos de análisis de imágenes para determinar si una parte importante de la pantalla ha cambiado entre capturas. Se evaluó frente a 148.223 muestras, alcanzando un 96,3 % de tasa de detección y sin falsos positivos.

La captura de la ventanoa o *pop-up* que muestra la nota de rescate es utilizada por la herramienta forense propuesta en [323]. Esta captura se pasa por un reconocedor óptico de caracteres para recuperar el mensaje y la dirección de pago. Ademas, la herramienta permite realizar un volcado de memoria para ser analizada mediante técnicas forenses.

Un trabajo más reciente en [324] propone el estudio de los archivos relacionados con el *ransomware* (nota de rescate y extensión dada a los archivos) para su identificación. Este trabajo propone realmente dos métodos: el primero utiliza Análisis Semántico Latente (LSA) para ver las similitudes entre los archivos; el segundo utiliza ML para clasificar los archivos en benignos o maliciosos.

- **Archivos y sistemas de archivos**

Otras propuestas están basadas en la caracterización de los patrones de acceso a archivos y sistemas de archivo por parte del *ransomware*. Sentinel [325] está destinado tanto a la detección como a la recuperación en base a los patrones de acceso a archivos, utilizando los mecanismos del propio sistema de archivos como los *logs* de auditoría, los controladores de filtro (Windows), la pila de sistemas de archivos (Unix) y eventos ligeros (Linux e IBM Spectrum Scale). Peeler [326] combina la caracterización basada en el acceso a archivos junto con patrones de creación de procesos. La herramienta ARW [327] combina el análisis del ciclo de vida de los archivos con el uso de su contenido para detectar los ataques basados en criptografía (sin falsos

positivos). Además, presenta un marco de trabajo para la monitorización de archivos suficientemente genérico como para permitir la detección de nuevas muestras.

Existen varias métricas que pueden usarse para detectar cambios significativos en archivos como los que produce el *ransomware* cuando los cifra. En la literatura podemos encontrar diferentes métricas: tipo de archivo, diferencias de archivo o similaridad, operaciones de Entrada/Salida a disco, o patrones en el recorrido del sistema de archivos. Vamos a estudiar algunas propuestas de cada una de ellas.

El tipo de archivo o extensión suele ser modificado por el *ransomware* cuando cifra. Estos cambios se usan como características para determinar su presencia [328]. El sistema de detección establecido en [329] monitoriza cambios de numerosos archivos creados con la misma extensión o archivos con más de una extensión.

Frente a la similitud de los cambios que realiza un programa benigno a la hora de modificar/extender un archivo, el *ransomware* produce cambios no similares al cifrar un archivo. Midiendo la similitud de dos versiones de un archivo podemos detectar si una ha sido afectada por el *ransomware*, tal como se hace en [330] al medirla con la función `sdhash`, que produce una salida de 0 a 100 respecto a la confianza de similitud entre dos archivos (un archivo respecto a su versión encriptada da un valor próximo a 0). Esta función también se usa en [328] con el mismo propósito.

El *ransomware* utiliza las mismas operaciones de E/S que los usuarios para acceder a los archivos. Por ejemplo, utiliza `read()` para leer un archivo y `write()` para escribirlo, etc. La operación de escritura para almacenar el archivo cifrado puede realizarla sobre un nuevo archivo o sobrescribir el original (como vimos en la Sección 2.4.8). En el primer caso, el *ransomware* debe borrar el archivo original. En este sentido, [331] desarrolla un sistema para la detección en discos SSD que aprende las características de comportamiento del *ransomware* a partir de la información de las cabeceras de las operaciones de E/S que se realizan sobre bloques de datos y que incluyen la dirección de los bloques, el tipo de operación -lectura o escritura-, y el tamaño de los datos. En [332] se desarrolla un sistema que genera una probabilidad de *ransomware* comparando la actividad actual de E/S con un histórico. Si esta actividad supera un umbral, el sistema toma acciones para mitigar la actividad del *ransomware*. Por otro lado, el sistema propuesto en [314] extrae características de las peticiones de E/S durante la ejecución de una muestra, que se comparan con firmas de patrones de acceso de E/S para determinar la existencia del *malware*.

- **Máquina de estados finitos**

El comportamiento del *ransomware* se adapta bien al modelo matemático de máquina de estados finitos (FSM), que puede usarse para representar el estado del sistema y seguir la pista a los cambios. El comportamiento del *ransomware*, que difiere de un programa benigno, puede detectarse asociando acciones de éste con eventos del sistema que producen transiciones entre estados, y si se alcanzan ciertos estados estamos ante la presencia de un programa maligno del tipo en estudio. La monitorización de los cambios de estado que ocurren en un computador en términos de utilización y movimiento lateral de recursos pueden ayudar a detectarlo.

La propuesta [329] establece una FSM de ocho estados donde los cambios representan: cambios en la entropía, cambios en el estado de retención de las aplicaciones (se producen cuando un proceso se añade al registro un valor o abre un directorio), movimiento lateral (comprueba la existencia de archivos con doble extensión -por ejemplo, .pdf.exe), y recursos del sistema, que busca procesos que cambian ajustes de restauración del sistema o detienen multitud de procesos en poco tiempo. El sistema se considera bajo un ataque de *ransomware* si se alcanza alguno de estos estados finales. Este método detectó un 98,1 % de muestras sin falsos positivos de entre un conjunto de 475 muestras malignas y 1.500 benignas. El problema del método es que no es capaz de detectar *ransomware* de bloqueo o muestras que usan técnicas de ofuscación de código y desempaquetado incremental, como por ejemplo NotPetya.

Para la detección de *ransomware* cuyo objetivo sean servidores de bases de datos, DIMAQS (*Dynamic Identification of Malicious Query Secuences*), se utilizan redes de Petri coloreadas como clasificador para detectar consultas maliciosas [333].

- **Técnicas de decepción. *Honeypots* y *honeyfiles***

Es manifiesta la importancia de las técnicas de engaño en la ciberseguridad [334] en cuanto que suponen un enfoque más activo en la defensa de los sistemas. Estas técnicas se pueden usar en todos las elementos de un sistema y para combatir contra todas las etapas de un ataque, desde la red hasta los datos.

A nivel de red, un *honeypot*, tarro de miel, es un sistema informático diseñado y configurado como un señuelo para atraer ciberataques. En esta línea, RansomTracer [335] es un sistema diseñado para recolectar pistas sobre el atacante, utilizando un entorno de engaño monitorizado. Este entorno de red está configurado para atrapar, rastrear y desalentar ataques al protocolo RDP, simulando un entorno actual de usuario. El entorno está configurado para monitorizar los accesos remotos, el portapapeles, los discos montados y los archivos ejecutables.

Por otro lado, los *honeyfiles* son archivos que funcionan como trampas para el *ransomware*. De esta forma, si este archivo es atacado, entonces se detecta y notifica su presencia. Estos archivos trampa son fáciles de establecer y mantener; sin embargo, no hay garantía que el ataque los alcance. El trabajo [336] propone la creación de un gran archivo ficticio para ser monitorizado. El proceso de cifrado de este archivo por el *ransomware* conllevará tiempo, lo que permite la detección y protección del resto de archivos (cambiando los atributos de los archivos restantes y una lista de archivos infectados y otra de no infectados). Se utilizan filtros de notificación para observar los cambios en algunos metadatos de los archivos de la carpeta monitorizada (nombre, fecha último acceso, última escritura, seguridad y tamaño). El sistema no impide que algunos archivos se vean afectados, pues el archivo trampa se genera cuando se detectan cambios en otros archivos.

Podemos considerar que una forma más ligera de los archivos trampa son los archivos señuelo. En [337] se utilizan archivos señuelo repartidos por el sistema de archivos (especialmente en carpetas no de uso común) y que sirven para contabilizar el número de veces que una hebra de un programa pasa por él. Normalizado este contador por el número total de carpetas señuelo, si se supera cierto umbral, se considera *ransomware*. Esta idea ya fue utiliza por [338], donde simplemente se monitorizaban los cambios de nombre y los archivos señuelo para detectar la presencia de *ransomware*.

El sistema UNVEIL [314] genera un entorno virtual con el objetivo de atraer a los atacantes y limitar los daños antes de ser detectados con archivos trampa. Esta solución detecta el 96,3 % de las muestras y no tiene falsos positivos. El análisis del comportamiento de un programa analizando las secuencias de llamadas a la API (mediante la inyección de código en las DLL) para la exploración del sistema de archivos junto con un archivo señuelo son utilizadas por POSTER [339] para la detección temprana de *ransomware*.

Otro trabajo en esta línea es [340], que investiga la creación y monitorización de las actividades de un *ransomware* utilizando la técnica de carpetas que actúan como *honeypots*. El trabajo estudia dos métodos para su implementación: el *File Screening Service* del *File Server Resource Manager* (FSRM) de Microsoft y el EventSentry para manipular los registros de seguridad de Windows. El trabajo desarrolla una respuesta por etapas para los ataques y establece unos valores umbrales para cuando sea necesario activarla. El resultado del trabajo determinó que los archivos testigos tenían un valor limitado, ya que no permitían influir en el *ransomware* para acceder a las carpetas monitorizadas.

RWGuard [328] utiliza archivos señuelo (archivos que no deberían escribirse) para la detección mediante su monitorización junto con el seguimiento del comportamiento de los procesos respecto de sus peticiones E/S (IRP) y cambios en los archivos (operaciones de creación, borrado y escritura), en busca de comportamientos maliciosos. Los resultados de la monitorización alimentan un modelo de ML, como comentaremos en el apartado correspon-

diente.

RansomWall [341] es una defensa multi-capa que incorpora archivos trampa como forma de protección frente a cripto-*ransomware*. Cuando se sospecha de un proceso malicioso en la capa trampa, los archivos modificados se copian hasta determinar si es maligno o benigno en otras capas. El porcentaje de exactitud para esta propuesta es del 98,25 % y no genera falsos positivos. Un reto para la propuesta es la existencia de *ransomware* que tiene una actividad limitada en cuanto a la interacción con el sistema de archivos.

SentryFS [342] es un sistema de archivos especializado que distribuye estratégicamente archivos trampa por todo el sistema de archivos. Estas trampas se generan usando procesamiento de lenguaje natural (NLP) y tanto su contenido como los metadatos se actualizan constantemente para parecer más atractivos para aquellas muestras de *ransomware* más selectivo. Además, para ayudar con la gestión de los archivos trampa, SentryFS se conecta con un servicio web anti-*ransomware* para descargar la información más reciente sobre nuevas estrategias de *ransomware*. Además, como contingencia, aprovecha la clonación de archivos para evitar la escritura directa de estos en el caso de que un *ransomware* pase desapercibido. De este modo, cifraría los clones en lugar de los archivos reales. Con posterioridad, un agente de IA asigna una puntuación de sospecha a la actividad de escritura para que los usuarios puedan aprobar o descartar los cambios.

- **Tráfico de red**

La interceptación del tráfico de red para su posterior análisis puede servir para detectar las comunicaciones entre el *ransomware* y su servidor C&C, que son diferentes si se comparan con comunicaciones en condiciones normales. Entre las características que pueden determinar el comportamiento anómalo tenemos el tamaño de los paquetes, la frecuencia de los mensajes, dominios maliciosos y algoritmos de generación de dominios, entre otros. Vamos a repasar estas cuatro, si bien hay otras muchas, tal como se recoge en [343].

El tamaño de los paquetes intercambiados suele ser largo cuando estos contienen una clave o instrucciones de cifrado. En [343] este atributo se usa para alimentar un sistema basado en supervisión para la detección de este tipo de *malware*. El análisis del tamaño de los mensajes a partir de las cabeceras de HTTP de mensajes entre muestras de CryptoLocker y Locky con sus respectivos C&C [344], permitió construir un detector de anomalías en base a esta característica.

Los repuntes en la frecuencia de los mensajes pueden usarse para detectar *ransomware*. El trabajo [345] puso de manifiesto cómo el *ransomware* Locky tiene un alto número de mensajes POST de HTTP comparado con un tráfico normal, así como numerosos paquetes RST y ACK utilizados para

terminar anormalmente las conexiones maliciosas de TCP. En base a estas características, se llevó a cabo un detector de intrusiones multi-clasificador. En un sentido similar, [343] utiliza el número de paquetes RST, ACK, y duplicados ACK de TCP junto con el número de sesiones de comunicación como características para su modelo supervisado.

En otra línea, [344] propone una red definida por software cuyo objetivo es cortar la comunicación del *ransomware* con el *dominio malicioso* que actúa de C&C usando listas negras dinámicas de servidores *proxy* para bloquear la comunicación entre el computador víctima y el servidor C&C. De esta forma, el tráfico DNS es remitido a un controlador que comprueba si un dominio está en la lista negra, en cuyo caso se descarta el mensaje y se bloquea el tráfico desde el equipo. En una línea más amplia se encuentra el trabajo [346], donde se propone un enfoque defensivo basado en el paradigma de redes auto-organizadas y tecnologías emergentes como SDN, virtualización de funciones de red y la computación en la nube.

Dado que las direcciones de dominio almacenadas en el código son susceptibles de ser detectadas, algunos *ransomware* utilizan algoritmos de generación de dominios para generar nombres que puedan usarse como punto de cita con sus servidores de control. Algunas propuestas están encaminadas a detectar el uso de estos algoritmos para bloquearlos [347] [348].

El algoritmo REDFISH puede detectar las acciones del *ransomware* y evitarla en documentos compartidos mediante la monitorización pasiva del tráfico de red entre los equipos y los volúmenes compartidos de red [349]. Aplica tres valores umbrales sobre el número de archivos borrados, el intervalo entre eventos de borrado y la velocidad media de lectura/escritura. Si bien puede detectar muestras maliciosas en 20s, no ocurre así con *ransomware* más sofisticados que puedan, por ejemplo, evadir el mecanismo de detección mediante un *shell* inverso.

- **Múltiples fuentes de datos**

En este grupo de trabajos se aborda la detección a partir de diferentes fuentes de datos tanto de los sistemas como de la red. En esta línea, [350] establece un entorno con herramientas de fuentes abiertas que divide a los equipos en cliente y servidor. En el servidor se construye un *Endpoint Detection Response* (EDR), compuesto de las herramientas File Finder de Google Rapid Reponse (GRR), Osquery y el HIDS OSSEC. En los clientes se instala la versión de fuentes abiertas del EDR antes de que se produzca el ataque. El ataque se detecta analizando los cambios en los nombres o metadatos de los archivos.

El Enfoque Asistido por Red (NAA) [351] es una aplicación que realiza la detección tanto a nivel de anfitrión como de red, estando más orientada a la detección de cripto-gusanos. Para la detección de anfitrión utiliza como fuentes de datos la entropía, la frecuencia de lecturas/escrituras, y patrones

de lectura/escritura, para determinar si su comportamiento es anómalo o no. La detección de red recoge información de seguridad de los equipos de la red y determina cuándo una máquina está infectada. Si muchas máquinas muestran el mismo comportamiento anómalo, están infectadas. Si solo lo muestran unas pocas, puede que estén mal diagnosticadas. Esta decisión se realiza utilizando un mecanismo ACO (*Ant Colony Optimization*).

3.3.2. Procesamiento

Una vez realizada la recogida de datos, algunas técnicas necesitan llevar a cabo un procesamiento dependiente de la fuente de los mismos. Este procesamiento es necesario antes de que la herramienta de detección sea capaz de realizar el proceso de recuperación. El paso puede ir desde la monitorización a la extracción de características para aplicarlas a técnicas de ML. Una técnica utilizada de forma general por los vendedores de anti-*malware* es la de *hashing* [352], empleada para identificar y clasificar las muestras. También se puede combinar con otras técnicas como las firmas difusas y la entropía para detectar muestras de las cuales no se conoce la firma [353]. Si bien resulta útil, es de limitada aplicación dada la naturaleza imitadora del *ransomware* y del RaaS. Como ejemplo, indicar cómo PayBreak utiliza una firma de función difusa de 32 *bytes* para identificar el uso de bibliotecas criptográficas enlazadas estáticamente, o CryptoDrop [330], un *hash* de preservación de similitud para cuantificar la diferencia entre un archivo y su posible versión cifrada.

Otras propuestas se basan en asignar puntuaciones que cuantifican el comportamiento malicioso de un proceso, es decir, se monitorizan y puntúan las ocurrencias de determinados indicadores del comportamiento del *ransomware* hasta que el valor alcanza un umbral establecido. Esto ocurre con Redemption o CryptoDrop. También lo encontramos en el marco de trabajo propuesto en [354], el cual utiliza reglas YARA [355] para caracterizar el comportamiento de una muestra basado en las funciones de la API, palabras clave relacionadas con la extorsión, firma criptográfica de funciones de cifrado y extensiones de nombres de archivos. Cuando una regla alcanza cierto umbral de puntuación, la muestra se clasifica como *ransomware*. Este esquema puede detectar muestras desconocidas.

El uso de tests estadísticos se basa en que un cripto-*ransomware* debería escribir los datos cifrados de una manera aleatoria y esta aleatoriedad se puede medir con dichos tests. Hay muchos trabajos que hacen uso de la entropía para detectar *ransomware*; por ejemplo, [356] [357]. Dado que la entropía por si sola no permite diferenciar un archivo cifrado de uno comprimido [358] o puede evadirse [111], es necesario complementar el análisis. En [359] se propone utilizar la entropía pero solo aplicada para fragmentos de la cabecera de los archivos. Los experimentos realizados de fragmentos de los 192 primeros *bytes* de 80.000 archivos dan una exactitud del 99,96 %.

Esta característica sirve para alimentar un algoritmo *Support Vector Machine* (SVM) para discriminar ambos tipos de archivos [360]. Otra forma es usarla junto con archivos señuelo en caso de que el *ransomware* utilice cifrado personalizado tal como se propone en [361]. Los métodos basados en la entropía pueden evadirse como demuestra [350], donde se usan algoritmos de codificación base64 que permiten neutralizar las herramientas de detección basadas en esta técnica.

La entropía de un archivo mide la aleatoriedad del mismo. Los archivos cifrados y comprimidos tiene una entropía mayor si se comparan con archivos de texto plano. Por tanto, calculando la entropía de un archivo en varios instantes podremos determinar si se ha infectado. En [330] se calcula la entropía mediante la fórmula de Shannon y se utiliza como característica para su detección, al igual que en [328]. En cambio, [362] aplica *Machine Learning* para clasificar archivos infectados en base al análisis de su entropía. En esta línea, EntropySA y DistSA utilizan la frecuencia de *bytes* y la variación de la frecuencia de *bytes*, respectivamente, junto con técnicas ML para reducir el sobrecoste de otros métodos de detección basados en entropía [363]. Otros trabajos basados en el análisis de la entropía ya comentados son [286, 314, 315].

Otro enfoque diferente utilizado entre otros factores para reducir el coste computacional de calcular la entropía es el uso de la visualización. En [364] se usa una técnica diferente de procesamiento para describir el *ransomware* como imágenes y transformar el problema en uno de clasificación de imágenes tratadas con métodos de redes neuronales.

Como hemos analizado, muchos mecanismos de detección tratan de determinar el comportamiento del *ransomware* para su correcto funcionamiento. En este sentido y de cara al desarrollo de nuevos mecanismos, hay que tener presente que pequeñas variaciones en el comportamiento del *ransomware* pueden hacer que este pase por un programa benigno. Por ejemplo, una encriptación parcial, aunque solo sea del 3 al 5 %, es suficiente para secuestrar los archivos [365].

3.3.3. Detección con técnicas de aprendizaje máquina

Tanto las datos extraídos directamente de las fuentes de datos como los procesados por diferentes técnicas pueden ser usados para entrenar y probar algoritmos de aprendizaje máquina. En la actualidad existen numerosos trabajos que utilizan métodos de aprendizaje máquina destinados a clasificar los programas como malignos o benignos. Estos modelos, con un conjunto de entrenamiento suficiente, pueden determinar con bastante grado de exactitud un ataque de esta naturaleza. Además, en muchos casos son capaces de detectarlo antes de que se produzca el cifrado. Sin embargo, encontrar el modelo adecuado suele ser complicado y se puede producir sesgos o sobreajustes si no se toman las medidas oportunas [366].

Existen varias métricas de evaluación de la eficiencia de un modelo ML. El valor de estas métricas está presente en la matriz de confusión del modelo. La definición de estos parámetros es la siguiente:

- *Positivo cierto* (TP) - Valor real positivo, y predicho positivo.
- *Falso Positivo* (FP) - Valor real negativo, pero predicho positivo.
- *Falso Negativo* (FN) - valor real positivo, pero predicho negativo.
- *Negativo cierto* (TN) - Valor real negativo y predicho negativo.
- *Exactitud* - Razón del número de pruebas de muestra correctamente clasificadas:

$$Exactitud = \frac{TP + TN}{TotalMuestrasPrueba} \tag{3.1}$$

- *Precisión* - Razón de positivos ciertos respecto de positivos reales:

$$Exactitud = \frac{TP}{TP + FP} \tag{3.2}$$

- *Exhaustividad* - Razón entre los positivos ciertos respecto de los resultados predichos:

$$Exactitud = \frac{TP}{TP + FN} \tag{3.3}$$

Hay algunos retos sobre los sistemas de detección basados en ML. La limitación más común es el coste de entrenamiento. El entrenamiento con grandes conjuntos de datos y muchas características exige costes de entrenamiento muy altos. Para reducir este coste se suele reducir la dimensionalidad. Otro aspecto en este tipo de soluciones es el modelo aprendizaje máquina adversario (AML), que significa que un atacante puede usar tácticas en nuevas variantes de muestras para evitar el detector [367]. Este aspecto se puede solventar utilizando clasificadores híbridos, bien mezclando características estáticas y dinámicas, bien utilizando varios algoritmos de clasificación (*ensemble malware detector*) [368]. Otra propuesta es la de [369], que analiza la resiliencia en integridad de los algoritmos ML desde el punto de vista de la seguridad.

Lo que distingue a unos modelos de otros es el algoritmo clasificador que se aplica y las características utilizadas por este para su entrenamiento [370]. A continuación describimos buena parte de los trabajos en base a las características seleccionadas, y en la Tabla 3.1 se recogen los enfoques usados. Recientes revisiones de los trabajos en este campo son [368, 371, 372, 373, 374, 375].

Tabla 3.1: Enfoques de detección basados en *Machine Learning*.

Nombre	**Algoritmo(s) clasificador(es)**	**Característica(s)**	**Cita**
-	Random Forest	*Bytes* en bruto	[376]
-	SVM, Ramdom Forest	Cadenas, llamadas sistema/API	[377]
DNAact-Ran	Regresión lineal	Frecuencia k-mer	[378]
RansomWall	Logistic Regression, SVM, ANN, Random Forest, Gradient Tree Boosting	Llamadas sistema/API	[341]
ShieldFS	Random Forest	Archivos *logs*	[315]
	Naïve Bayes, Logistic Regression, Decision trees, Random Forest	Archivos *logs*	[328]
-	KNN, Linear regression, Logistic Regression, Decision trees, SVM, ANN	E/S archivos	[362]
PEDA	Random Forest	Llamadas sistema/API	[379, 380]
EldeRan	Logistic Regression, SVM, Naïve Bayes	Llamadas sistema/API, claves Registro, E/S archivos, cadenas	[381]
-	SVM	Llamadas sistema / API	[382]
-	SVM	Llamadas sistema / API	[383]
DPBD-FE	Logistic Regression, LDA, KNN, CART, Naïve Bayes, Decision trees, Random Forest, KNN, Boosting, ANN	Llamadas sistema/API	[384]
DRDT	CNN	Llamada sistema/API	[385]
-	ANN	Archivos *logs*	[386]
-	Random Forest, Logistic Regression, Naïve Bayes, SGD, KNN, SVM	Llamadas sistema/API	[387]

(Continúa en la página siguiente)

(Viene de la página anterior)

Nombre	**Algoritmo(s) clasificador(es)**	**Característica(s)**	**Cita**
-	Linear Regression, Decision trees	Llamadas sistema/API	[388]
-	Decision trees, Random Forest, Naïve Bayes, Bayesian networks, Logistic Regression, LogiBoost, Bagging, AdaBoost	Llamadas sistema/API, volcado de memoria volátil	[389]
iBagging/ ESRS	Linear Regression	Llamadas sistema/API	[390]
RAPPER	ANN (LSTM)	HPC	[391]
-	Prueba de concepto	Tráfico de red	[392]
-	Random Forest, Bayesian Networks, SVM	Trafico de red	[345]
-	Naïve Bayes, Decision tree, Random Forest	Tráfico de red	[343]
-	KNN, ANN, SVM, Random Forest	Consumo energía de CPU	[393]
-	Random Forest	Tráfico de red	[394]
-	CNN	Códigos de operación	[395]
-	SVM	Secuencias códigos operación/*byte*	[396]
-	CNN	Cabeceras de ejecutables PE	[397]
-	Naïve Bayes, Logistic Regression, SVM, Random Forest, Decision trees	Llamadas a DLL, Código operación/byte	[398]
-	Logistic Regression, SVM, Random Forest, Decision trees	Llamadas a DLL, secuencias códigos operación/*byte*	[399]
DRTHIS	LSTM, CNN	Secuencia de eventos	[400]
-	KNN, SVM, ANN	Tráfico de red	[347]
-	k-means Clustering	Tráfico de red	[167]
-	SVM, Naïve Bayes	Tráfico de red	[401]
-	ANN, KNN	Tráfico de red	[402]
DRDT	TextCNN	Llamadas sistema	[385]
-	RF	Llamadas DDL, recursos SO	[403]
-	RF, OoW	Llamadas DDL	[404]

(Continúa en la página siguiente)

(Viene de la página anterior)

Nombre	**Algoritmo(s) clasificador(es)**	**Característica(s)**	**Cita**
-	NB, KNN, LR, RF, SGD, SVM	Llamadas al sistema	[405]
AIRaD	AI	Llamadas en DLL	[406]
DeepRan	BiLSTM, FC	*Logs*	[407]
RanStop	LSTM	Eventos hardware	[408]

(Fin de la tabla)

Las llamadas al sistema/API permiten solicitar servicios al sistema operativo o aplicaciones/bibliotecas y se utilizan por el *ransomware* para comunicarse con el C&C, mantener/escalar privilegios, enumerar/leer/escribir archivos, y otras funciones. Tanto los malignos como los benignos utilizan dichas llamadas, si bien tienen patrones de invocación diferentes. De acuerdo a los resultados de [405], estas llamadas son las característica más relevantes para determinar si una muestra es *ransomware* o no, tras analizar 30.976 características mediante los algoritmos de clasificación *Naive Bayes* (NB), *K-Nearest Neighbors* (KNN), *Logistic Regression* (LR), *Random Forest* (RF), *Stochastic Gradient Descent* (SGD) y *Support Vector Machine* (SVM).

El análisis de las llamadas a funciones se puede combinar con características de las DLL y los niveles de ensamblado, para suministrar un análisis de comportamiento y correlación a las técnicas de inteligencia artificial (IA). En esta línea, en [406] se realiza una propuesta de análisis híbrido (métodos estáticos y dinámicos) para el estudio de las cadenas de comportamiento utilizando IA; es decir, del estudio de secuencias de invocaciones de cierta funcionalidad dentro de una DLL.

TextCNN(DRDT) [385] es un detector dinámico de *ransomware* proveniente del campo del procesamiento de lenguaje natural, que analiza las secuencias de llamadas a la API para determinar la naturaleza maligna o no de un programa. Así mismo, el trabajo [403] intenta caracterizar el comportamiento de los procesos a partir de las DDL invocadas, recursos utilizados (RAM, CPU, disco, conexiones de red, etc.) y archivos abiertos; y utilizando diferentes técnicas de ML concluyen los mejores resultados con *Random Forest*. El algoritmo PEDA (*Pre-encryption Detection Algorithm)* [379] puede detectar casi todos los cripto-*ransomware* en la etapa previa al cifrado. PEDA es un algoritmo híbrido que primero examina un binario sospechoso de forma estática mediante la comparación se sumas de comprobación y luego monitoriza las llamadas a la API de cifrado previas a la encriptación. La única limitación e PEDA es la dependencia de la API de Windows.

Las llamadas a la API se utilizan como características para construir clasificadores SVM [382], clasificadores LSTM (*Long-Short Term Memory*) [409], clasificadores RNN (*Recurrent Neural Network*) [410], clasificadores

Restrited Boltzmann Machine [411]. N-gramas de las llamadas a la API se utilizan para construir varios clasificadores basados en ML [412]. Por otro lado, [312] genera grafos de flujos de llamadas (CFG) para entrenar diferentes clasificadores y [413] construye un clasificador SVM utilizando valores la correlación de Pearson de las llamadas a la API pertenecientes a diferentes grupos.

Algunos estudios se centran en encontrar las llamadas más significativas, como proponer un nuevo método de filtraje en la selección de características para encontrar los N-gramas de llamadas API más apropiadas [414]. Además, comprueba el rendimiento de varios clasificadores ML, como hace [390]. El trabajo [415] propone un enfoque de detección no basado en firmas cuyo objetivo también es eliminar aquellas secuencias de llamadas no efectivas para usar técnicas de aprendizaje máquina supervisado. En este caso se propone un filtro *Enhanced Maximum-Relevance and Minimum-Redundancy* (EmRmR) para eliminar las características que introducen ruido y poder obtener el comportamiento del *ransomware*.

Otra propuesta se centra en las llamadas a la API criptográfica para realizar un modelo adaptativo pre-encriptación para la detección temprana del *ransomware* [416]. La propuesta está pensada para abordar el problema de deriva de la población *ransomware*. Utiliza una técnica similar a [384] para determinar la fase de pre-cifrado para extraer las características de las muestras que se usarán para la clasificación posterior.

Partiendo de las llamadas a la API como características e incorporando los enfoques de detección de anomalías y conductales, en [417] se propone un mecanismo de detección temprana. En primer lugar usa un conjunto de clasificadores basados en comportamiento y, en segundo, un estimador basado en anomalías. Las decisiones de ambos tipos de estimadores se combinan mediante una técnica de fusión. Este método puede detectar ataques de día cero con una alta tasa de aciertos y baja tasa de falsos positivos. Basándose en la detección de anomalías del comportamiento, [418] propone un modelo de clasificación conjunto que utiliza los algoritmos *Random Forest*, *Decision Tree* y *K-Nearest neighbor*, que, junto a técnicas de votación tanto *soft* como *hard*, producen un mejor resultado.

Basándose en las actividades paranoia vistas en la Sección 2.4.1, surge la propuesta de un mecanismo de identificación de muestras de *ransomware* de cara a atribuirle la familia a la que pertenece, y poder así aplicar las medidas de mitigación correspondientes para ese ataque [404]. Los autores proponen una técnica enraizada en Procesamiento de Lenguaje Natural (NLP) y Ocurrencia de Palabras (OoW) para analizar las llamadas a la API de evasión generadas por el *ransomware*, a la vez que ajustan varios algoritmos de ML y DL (*Deep Learning*) para realizar la clasificación. Se concluye que las técnicas de *Random Forest* y OoW producen la exactitud de clasificación óptima (94,92 %).

Los archivos de *logs*, como vimos con anterioridad, contienen informa-

ción proveniente de diversas fuentes y se pueden usar en detección junto con técnicas ML. Algunas muestras, como Wannacry y Petya, puede detectarse analizando los registros del DNS o de NetBIOS que contienen paquetes de solicitudes E/S, con parámetros tales como el tipo de operación, la dirección, el tamaño de los datos a leer/escribir, y que pueden utilizarse como características en ML [392]. En la propuesta [419], el registro de eventos sirve mediante minería de datos para extraer las características de un proceso modelo. Estas características son las utilizadas en la fase de clasificación.

El detector DeepRan [407] trata de detectar la actividad anormal de un gran volumen de equipos en una red recogiendo los *logs* de los servidores. La normalidad se modela utilizando el método *Attention-based Bidirectional Long Short Term Memory* (BiLSTM) con una capa totalmente conectada.

Respecto a las E/S de archivos, un *ransomware* realiza por lo general muchas más operaciones de lectura/escritura que un programa benigno, dado que debe leer un archivo antes de cifrarlo y escribir su versión cifrada. Por ello, pueden usarse como métricas de un ataque estimadores como el número de operaciones de lectura/escritura, la entropía media de operaciones de escritura sobre un archivo, el número de operaciones realizadas por tipo de extensión de archivo, o el número total de archivos accedidos, como ocurre con ShieldFS [315] o EldeRan [381]. Por otro lado, las modificaciones en el *firmware* de un disco SSD permiten detectar patrones en las cabeceras de las solicitudes de E/S y poder así retrasar las escrituras basándose en las propiedades de las memorias NAND [331], que ofrecen además un medio de recuperación.

Podemos alimentar un modelo con valores de series temporales de los *Hardware Performance Counter* (HPC) para determinar la integridad estática y dinámica de los programas para establecer si ha habido modificaciones, como ocurre en RAPPER [391]. RanStop [408] propone un esquema apoyado por hardware donde se utilizan eventos hardware para entrenar una red neuronal recurrente usando el modelo LSTM.

Ya comentamos en el punto anterior los principales elementos utilizados relativos al tráfico de red. En este sentido, los modelos de aprendizaje máquina ayudan a distinguir entre tráfico normal y anómalo. Los nombres de dominios pueden usarse como característica para detectar paquetes entrantes/salientes transmitidos de/hacia dominios maliciosos [347].

Las secuencias de *opcodes/bytecodes* o códigos de operación tiene una rica información del contexto y semántica, que suministra una instantánea del comportamiento del programa analizado. Esta información, que se puede extraer mediante análisis dinámico, puede alimentar un modelo para predecir el comportamiento malicioso o benigno de un proceso. Estos datos, que se utilizan para generalmente para detección de *malware*, ofrecen diferentes aplicaciones en Android [420]. Otro trabajo en esta línea es [396], que utiliza la densidad de *opcodes* para construir un clasificador basado en SVM. El trabajo [421] los utiliza para varias instrucciones (de procesamiento de datos,

aritmético-lógicas y de control de flujo) con el objetivo de construir una Cadena Oculta de Markov (HMM). Los basados en el uso de N-gramas de *opcodes* para construir una red neuronal profunda [395], o para construir varios clasificadores ML [422].

Un enfoque diferente plantea utilizar la irregularidad en la textura de una imagen para detectar inyecciones anómalas de operaciones en los *bytecodes* [423]. Como extractor de las características utiliza una modificación del algoritmo *Local Binary Pattern* (LBP) y como clasificador una *Convolutional Neuronal Network* (CNN).

Las secuencias de acciones de los procesos cuando estos se ejecutan provocan eventos diferenciables entre procesos malignos y benignos. DRTHIS (*Deep Ransomware Threat Hunting and Intelligence System*) [400] ejecuta durante 10 segundos un programa para extraer la secuencia textual de eventos que genera (etapa denominada *caza de amenazas*). Estos eventos se codifican numéricamente y se clasifican utilizando dos técnicas de aprendizaje profundo: *Long Short-Term Memory* (LSTM) y *Convolutional Neuronal Network* (CNN). Estas técnicas son capaces de detectar amenazas de día cero.

Otras características que se han usado son, por ejemplo, la extracción de *bytes* brutos mediante análisis estático [376], o relacionadas con dominios web (por ejemplo, la longitud del nombre del dominio, el número de días que este está registrado) [424] o con DNS (como el número de errores en nombres o nombres sin sentido) [345]. También se puede utilizar el consumo de CPU u otros recursos hardware, el comportamiento de los procesos, frecuencias de subcadenas k-mer, memoria volátil, y el Registro de Windows [393] [389][381] [425].

Algunas propuestas utilizan técnicas forenses junto con ML para la detección. En [426] se extraen características de las muestras de un volcado forense de memoria (descriptores de procesos, información de DLL, etc.) para realizar una selección mediante tres métodos: Chi-cuadrado, matriz de correlación con mapa de calor y Lasso. Con ellas se alimenta un clasificador *eXtreme Gradient Boosting* (XGBoost). Los autores tienen publicado el conjunto de datos en el *Harvard Dataverse Repository* [427]. También se pueden utilizan los privilegios de acceso de los procesos a diferentes regiones de memoria (lectura, escritura, ejecución o copia, obtenidos mediante un volcado de memoria de una *sandbox*) para determinar el comportamiento de un ejecutable para estimar rápidamente sus intensiones antes de provocar daños serios [428].

Recientemente, en [429] se realiza una propuesta donde para la detección se analizan de forma estática metadatos de los archivos ejecutables de Windows, como pueden ser: el número mágico del archivo, el tamaño del código, los datos inicializados, la imagen, el subsistema que se necesita para ejecutar dicha imagen, las características de las DLL, la dirección del punto de entrada del código, las direcciones virtuales de cada sección de programa,

el tamaño virtual del programa y de los datos, y la tabla de importación de direcciones. Por ejemplo, algunas muestras de la familia Petya utilizan la biblioteca *wininet.dll*. Los autores encontraron similitudes entre 727 muestras de 50 familias activas de *ranomware*. El método propuesto utiliza primero PCA (*Principal Component Analysis*) para reducir las características, y a continuación el algoritmo K-Means para descubrir agrupaciones. Los experimentos reflejan un mejor comportamiento del algoritmo *Local Outlier Factor* (LoF) que el *One-Class SVM* y el *Isolation Forest*.

Los programa ejecutables como fuente de características son la base de [430]. Las características relevantes se extraen visualizando el conjunto de datos mediante un mapa de calor en donde se eliminan los atributos con máxima correlación. Tras la prueba de varios algoritmos, *Random Forest* se muestra como el que ofrece mejor exactitud.

El marco de trabajo *Zeroshot Learning framework* (ZSL) [431] está destinado a la detección de ataques de *ransomware* de día cero en tiempo de ejecución, inspirado en los mecanismos del cerebro para identificar nuevos conceptos. El marco parte de un amplia variedad de secuencias de llamadas a la API que pueden considerarse maliciosas (acceso a archivos, acceso al registro, descargas de archivos, búsquedas de extensiones de archivos y nota de extorsión) y se divide en dos etapas. La primera etapa de Aprendizaje de Atributos basado en Auto-codificador de Contracción Profunda (DCAE-ZSL) está destinada a la extracción de las características fundamentales de *ransomware*, conocido o no. Una segunda etapa de inferencia se basa en un Conjunto de Votación Heterogénea (DCAE-ZSL-HVE) para la toma de decisión final. La propuesta muestra una eficiencia razonable y una mejora respecto a las anteriores en la detección de ataques de día cero, reduciendo los falsos negativos.

Otra herramienta que utiliza diferentes fuentes de datos es la propuesta en el trabajo [432] y que permite la detección temprana. Para facilitar el esfuerzo forense, la herramienta permite visualizar las características discriminatorias del *malware* y sus correlaciones. Las pruebas realizadas con varias muestras indican que el enfoque TF-IDF (*Term Frequency–Inverse Document Frequency*) obtiene mejores resultados en la discriminación de características. También se usan conjuntamente los *opcodes* y las DLL de binarios para construir un clasificador RF [398].

La propuesta [433] utiliza hasta diez características como son la tasa de cifrado de archivos, las operaciones de escritura, el uso de CPU, el borrado de copias sombra, los cambios en el registro, el renombrado de archivos, el incremento del tamaño de archivos, entre otras. Por su parte, [434] utiliza los cambios en el registro, la actividad del sistema de archivos y las DLL.

En lugar de solo usar características estáticas o dinámicas, algunas propuestas combinan ambas para la detección. En este caso, los clasificadores utilizados son: redes neuronales artificiales y SVM [435], modelos de Markov y RF [436], Naive Bayes y DT [373], SVM [437], regresión logística [381],

y varios clasificadores en [438, 439, 440, 441]. Si bien las cadenas de caracteres son las características estáticas más usadas, las llamadas al sistema, las operaciones sobre archivos y sistema de archivos, las claves de registro, las extensiones de archivos y descargas procesadas, son las características dinámicas más utilizadas. Algunos estudios utilizan técnicas específicas para seleccionar las mejores características para los clasificadores: [440] que utiliza Información Mutua (MI) y *Particle Swarm Optimization*; [441] que utiliza MI, PCA (*Principal Component Analysis*) y N-gramas; o [437] que incluye algoritmos de optimización Grey Wolf.

BigRC-EML [442] también utiliza características estáticas y dinámicas relativas a funciones de la API y valores del Registro de Windows seleccionadas mediante el uso de PCA. Estas características se procesan utilizando un método conjunto que produce el resultado por votación mayoritaria de los clasificadores SVM, RF, KNN, NN y XGBost.

La forma de recorrer el árbol del sistema de archivos en la fase de selección de archivos sirve a [337] para implementar un mecanismo de detección temprana basado en grafo donde el recorrido empleado por el *ransomware* se compara con otros patrones, y mediante la similitud de matrices mediante varias técnicas de ML se puede clasificar dentro de una familia de *ransomware*.

Un par de técnicas de detección, que podemos denominar híbridas, realizan un procesamiento ML junto con archivos señuelo para detección de procesos tipo *ransomware* en [328]. Además de archivos señuelo, utilizan monitorización tanto de procesos basada en ML como de cambios en archivos, ganchos a la API de cifrado, y clasificación de archivos. La monitorización de procesos entrena un clasificador ML utilizando las solicitudes de operaciones sobre archivos (apertura, cierre, creación, lectura y escritura) y el número de archivos temporales creados. La monitorización de archivos compara la similitud, entropía, tipo y tamaño de archivos, antes y después de los cambios. El módulo de ganchos a funciones de criptográficas intenta obtener las claves de cifrado de los procesos maliciosos. Por otro lado, [443] propone un sistema de dos capas que combina técnicas basadas en ML y en reglas. El clasificador ML intenta detectar el *ransomware* mediante las llamadas a la API, las operaciones del Registro y las DLL. El sistema basado en reglas monitoriza los cambios en las firmas y la entropía de los archivos.

El análisis del tráfico de red permite alimentar diferentes modelos de ML. Entre ellos encontramos [444], que, centrado en el protocolo TCP, utiliza una selección características de tráfico para calificar familias de *ransomware* mediante los modelos clasificación multi-clase. La experimentación indica que los árboles de decisión dan los mejores resultados, con un 99,83 % de exactitud. Además, concluye que si bien los resultados, con y sin selección de características, tienen resultados similares, la selección de características necesita menos cantidad de RAM y reduce el tiempo de detección. Entre las características más importantes estarían el tiempo delta, la longitud de la

trama, la longitud IP, la dirección IP de destino y origen, la longitud TCP, la secuencia TCP, la longitud de la cabecera TCP, y el viaje de ida-vuelta inicial TCP. La propuesta de [401] es capaz de detectar los cambios en el tráfico de red cuando se está ejecutando el *ransomware*. Este patrón alimenta un clasificador probabilístico supervisado para extraer las características de la muestra en ejecución. Necesita de intervención humana para establecer el *dataset* que alimentará al modelo ML basado en reglas.

NetConverse [445] construye un clasificador DT (*Decision Tree*) usando el tipo de protocolo, direcciones IP, número de paquetes y duración de la comunicación para la detección del *ransomware*. Con el objeto de detectar *ransomware* en el tráfico web cifrado, [446] utiliza 28 características incluidas las de conexión (flujo, paquetes y carga maliciosa), las de SSL (tasas de flujos SSL-TLS, entre otras) y de certificados (validez, edad, etc) para construir clasificadores RF, SVM y LR.

En relación al hardware de red, *Programmable Forwarding Engines* (PFM) [394] monitoriza el tráfico de red entre un computador infectado y el C&C. En la fase de monitorización, se extrae la desviación estándar de la longitud de los paquetes y su número en *bytes* del tráfico saliente y entrante, la longitud media de la ráfaga entrante, el tiempo medio entre llegadas en la entrante y la ratio de paquete entrantes y salientes, para construir un sistema de detección utilizando un clasificador RF.

Basándose en el concepto de deriva, [447] propone un mecanismo de selección de características independiente de los algoritmos de clasificación subyacentes denominado FeSA (*Feature Selection Architecture*), que se muestra robusto y mantiene altas tasas de detección.

La detección de *ransomware* en entornos Big Data tiene problemas en términos de procesamiento computacional o velocidad de detección. Estos se pueden solventar en parte con un método de reducción de la dimensionalidad para mejorar la eficiencia. El trabajo [448], tras analizar los métodos de reducción *Principal Components Analysis* (PCA), *Factor Analysis* (FA) y *Truncated Singular Value Descomposition* (TSVD), concluye que PCA no solo es el método más rápido y significativo, con una media de detección de 34,33s, sino el que también tiene mayor exactitud, precisión y exhaustividad.

En lugar de establecer a priori las características a analizar, [449] realiza una selección de las más relevantes en base a un valor establecido de la varianza del factor de inflacción (VIF) que establece las 12 características para alimentar a los clasificadores: tamaño de las cabeceras opcionales, número de versión del enlazador, dirección de punto de entrada, alineamiento de sección, número menor de versión del sistema operativo, tamaño de las cabeceras, tamaño de la reserva de pila, indicadores de carga, entropía máxima y mínima de las secciones, tamaño máximo físico de sección, tamaño virtual mínimo de la sección y entropía mínima de los recursos. Tras la experimentación con múltiples clasificadores, se establece que RF arroja mayor exactitud.

Dada la importancia actual de predecir la infección antes de que los recursos queden inaccesibles, indicaremos algunas propuestas de ML o DL que permiten la detección temprana: [379, 380, 390, 392, 408, 407, 414, 432].

3.4. Acciones tras la detección

Una vez detectado el *ransomware* hay que establecer las acciones a realizar, o formas de reaccionar, con el objetivo de detener el proceso.

Una propuesta sería detener o matar al proceso malicioso, tal como hace el sistema como *Data Aware Defense* [316] cuando no cabe duda de la naturaleza del proceso. Otra acción posible sería ponerlo bajo "vigilancia", es decir, monitorizarlo para obtener más indicadores de su naturaleza maliciosa. Si bien esto suministra un mayor conocimiento del mismo, también es cierto que gasta más recursos del sistema. RAPPER utiliza este mecanismo reduciendo el consumo de recursos [391].

Por último, también se suele incluir algún tipo de notificación al usuario para asegurarse que la decisión tomada por la herramienta es adecuada según el contexto. Por ejemplo, si un usuario suele cifrar los datos y esta acción es malinterpretada por la herramienta, es bueno que se le notifique para que corrija la acción del anti-*ransomware*.

Si bien es necesario aclarar que la recuperación no implica que necesariamente se mitigue todo el daño causado en tanto en cuanto debemos considerar el coste económico y reputacional de la organización.

Las propuestas más comunes intentan obtener las claves de cifrado para poder restaurar los archivos cifrados. Otras lo abordan desde diferentes enfoques, como pasamos a ver.

- **Gestión de claves**

La gestión de claves trata sobre la recuperación de la clave de cifrado que fue usada para encriptar los archivos y que permite desencriptarlos sin pagar el rescate. En algunas muestras esto es directo, pues la clave está incrustada en el ejecutable, por lo que es relativamente directa su recuperación. En los modelos híbridos este proceso es más complejo debido a que las claves solo están disponibles en texto plano mientras se están cifrando los archivos.

Algunas muestras de *ransomware* utilizan técnicas de generación de claves pobres realizando llamadas a bibliotecas comunes, como muestra [215] tras la descompilación de 8 muestras de diferentes variantes de *ransomware* para .NET. Este conocimiento se puede usar para almacenar copia de la clave de cifrado simétrico.

En otra línea, [450] pone de manifiesto que algunas muestras usan la biblioteca CNG (*Cryptography Next Generation* [451]) del sistema Windows, lo que permite desarrollar un gancho a dicha API (ver por ejemplo [452]) y almacenar la clave de cifrado. La experimentación de una muestra de

ransomware sintética mostró que es posible obtener dicha clave en el 100 % de los casos. El principal problema de la misma es la suposición del uso de esta biblioteca específica y no otra.

Para las muestras de *ransomware* que utilizan para el cifrado clave de sesión simétrica, el sistema *Paybreak* [453] desarrolla una solución de copia de clave que descansa en firmas. Implementa un enfoque de clave depósito que almacena la clave de sesión en una caja fuerte. Este mecanismo recupera todos los archivos cifrados con firmas de encriptación conocidas.

Bastantes muestras de *ransomware* descansan sobre cifrado AES, lo que llevó a [454, 455] a desarrollar un ataque de canal lateral a la gestión de claves del *ransomware* para extraer las claves de la memoria RAM durante el proceso de cifrado, con un 100 % de éxito sin importar la API utilizada por el *malware*.

- **Otras técnicas**

Otra técnica propuesta para las familias más comunes de *ransomware* como pueden ser CryptoWall, CTB-Locker (*Curve-Tor-Bitcoin Locker*), Locky, o TeslaCrypt, es el renombrado de las herramientas nativas para la gestión de copias sombra para que la muestra que ha infectado el sistema no pueda encontrar y borrar los puntos de restauración realizados antes de la infección [171].

La herramienta SH-VARR permite la recuperación específica de documentos XML basada en la idea de usar enlaces para mantener versiones distribuidas de un documento mientras se aplican mecanismos de control de acceso para proteger las diferentes versiones, evitando así que se cifren o borren [456].

3.5. Resumen de herramientas de detección y recuperación

Esta sección aborda el análisis de las herramientas anti-*ransomware* operativas vistas desde el punto de vista de los elementos que cubren las carencias que tienen y la exactitud de las mismas según indican sus autores.

La Tabla 3.2 muestra una visión general de las herramientas existentes y sus elementos característicos. Esta evidencia la existencia de una preferencia por la monitorización, por ejemplo, del sistema de archivos, y de la detección frente a la recuperación. También podemos ver cómo algunos enfoques prometedores, como los de puntuación, no han recibido mucha atención.

Respecto a la comparativa en términos de exactitud, a partir de los datos suministrados se muestra una alta tasa de detección en las herramientas de monitorización del sistema de archivos. Por ejemplo, Redemption da una tasa de detección del 100 % con una tasa de falsos positivos del 0,5 %, analizadas 1.174 muestras de 29 familias. En este sentido, para algunos autores

[233], de cara a la detección es preferible maximizar la tasa de aciertos positivos que minimizar la de falsos positivos, ya que desde el punto de vista de daños causados, un falso positivo (proceso benigno clasificado de forma incorrecta) puede ser molesto pero un falso negativo (muestra no detectada) puede ser catastrófico. Esto no quiere decir que no se debe tener en consideración la tasa de falsos positivos, ya que si es muy elevada un usuario puede plantearse dejar de usar la herramienta que la produce.

La propuesta del marco de trabajo FARFEL [457] tiene como fin evaluar el comportamiento en tiempo-real de los detectores de *ransomware* que no descansa en muestras conocidas, y especialmente enfocado a ver la capacidad de detectar muestras polimórficas o nuevos patrones de ataque. Este marco descansa en 21 variaciones de comportamientos fundamentales (descubrimiento y selección de datos, método de lectura, mecanismo de cifrado, estrategia de escritura de datos,) que se pueden seleccionar para formar 1.536 ataques. Este marco se ha usado para evaluar 7 diferentes herramientas, comerciales y académicas, y determinar brechas en la cobertura del comportamiento básico. Algunas herramientas tienen huecos, mientras que otras ni siquiera detectan ninguna muestra.

Otro aspecto a considerar, que en algunos casos se deja de lado, es minimizar la sobrecarga del sistema donde se ejecuta la herramienta. Un ejemplo es *Data Aware Defense*, que tiene una sobrecarga de varios órdenes de magnitud por debajo de otras propuestas. Una cuestión importante para poder hacer una comparativa real entre propuestas es la de utilizar herramientas de terceras partes para medir esta sobrecarga, como CrystalDiskMark [458], Geekbench 5 [459] o PCMark 10 [460].

3.5.1. Bases de muestras de *ransomware*

Un elemento importante para el desarrollo de herramientas de detección es la disponibilidad de un amplio conjunto de muestras de *ransomware* para realizar una adecuada experimentación.

Una metodología típica para la evaluación de las herramientas de detección de *ransomware* pasa por la adquisición de las muestras, ejecutarlas, evitando muestras que no funcionen, y analizar los resultados.

	Detección																	Recuperación							
	Fuente		*Machine Learning*					Procesamiento					Acciones					Fuentes				Proc		Acción	
Nombre	*Kernel*	Usuario	RLR	DT	RF	ANN	BBN	H	S	M	E	ED	Su	K	B	L	Nu	Bk	Es	Ch	API	K	DD	Re	D
Connection Monitor		N								√	√						√								
CryptoDrop	SF							√	√	√	√			√	√		√								
Data Aware Defense	SF									√	√				√										
EldeRun	SF R	CA	√							√															
Honeypot		HT								√				√		√	√								
RAPPER	CA				√				√			√				√									
Redemption	SF								√	√	√			√			√	√	√						
R-Killer	SF	N				√				√				√			√								
R-Locker		HT								√				√											
Pay-Break																					√	√			√
SSD-Insider	A			√												√	√		√	√			√	√	
USNP(*)		CS								√				√		√	√								
Unveil	SF	No								√	√			√		√									
2entFOX	R/SF	API CS Ev					√				√														

Abreviaciones: B: Bloquear; Bk: Copia de seguridad; Ch: Caché; CS: Cripto-sistema; D: Descifrado; DD: Detección retrasada; E: Estadístico; ED: Distancia; Es: Interceptación de E/S; Ev: Eventos; H: Firmas; HT: Trampa; K: Obtener claves; Ke: Kernel; L: Aislamiento; M: Monitorización; K: Matar; N: Red; No: Nota extorsión; Nu: Notificar usuario; R: Registro; Re: Restauración; S: Puntuación; SF: Sistema de archivos; Su: Vigilancia; Us: Usuario. // (*) USNP: USahllNotPass

Tabla 3.2: Caracterización de herramientas anti-*ransomware*.

La adquisición de muestras se realiza generalmente mediante repositorios, como por ejemplo, VirusTotal o VirusShare. Estos conjuntos de muestras pueden tener etiquetadas las mismas según su comportamiento, familia y firmas. Estas muestras se suelen descargar en masa. Como algunos trabajos indican [315, 330, 461], estas muestras no necesariamente deben funcionar debido a que pueden estar diseñadas para ejecutarse una vez, tener fecha de expiración, requerir un servidor C&C que está fuera de servicio, detectar su ejecución en una máquina virtual, requerir opciones para la invocación, o están mal etiquetadas (por ejemplo, no son *ransomware*).

La forma general para detectar muestras que no funcionan es obtener las firmas (*hashes*) de los archivos del sistema experimental antes y después del experimento. Si las firmas no coinciden, la muestra es activa [330]. En muchos de los trabajos estudiados la tasa de fallo de las muestras analizadas está entre el 12 % y el 67 %, como muestra la Tabla 3.3, que recoge los resultados de evaluación de varios trabajos revisados en la Sección 3.3.

Una vez localizadas las muestras activas, se realiza la experimentación con las mismas. Las muestras se agrupan en familias, donde una familia describe una categoría de programas maliciosos con un patrón de comportamiento similar [462]. Algunas muestras dentro de una familia pueden tener ligeras variaciones sobre el comportamiento general pero no suficientemente significativas como para extraerlas de la familia. Este hecho puede generar un sesgo en los resultados si algunas familias tienen superposición en el comportamiento o la relación muestra-familia es alta. Si la relación es alta, los sistemas se entrenan con muchas muestras que pueden exhibir un comportamiento similar. Las pruebas en modelos construidos de este corpus de *ransomware* tendrán una alta precisión, ya que el conjunto de prueba exhibirá comportamientos similares a los del conjunto de entrenamiento.

La comparación entre trabajos es complicada en ocasiones debido a dos cuestiones. Primero, en muchos de ellos se realiza un preprocesamiento de las muestras y este *dataset* propio no es compartido, lo que impide la reproducibilidad. Segundo, incluso disponiendo de dichas muestras no hay garantías de que estas sigan funcionando.

3.6. Conclusiones

En el capítulo hemos abordado las propuestas de defensa frente al *ransomware*. Para su consecución hemos adoptado una taxonomía que nos permita organizar dichas propuestas. Además, para clarificar el estudio hemos separado los trabajos en base a cuando son aplicables. Así, abordamos primero la prevención/mitigación, destinadas a establecer soluciones antes de que se produzca el ataca. Sin bien hay propuestas específicas para luchar contra el *ransomware*, la mayoría son recomendaciones general para asegurar los sistemas y redes. Se han presentado trabajos en relación con el control

de acceso, las copias de seguridad y la concienciación de los usuarios.

Hemos seguido con una revisión de los mecanismos de detección, que hemos clasificado en base al origen de los datos que recolectan y al procesamiento que se realiza de los mismos. Respecto los múltiples orígenes de los datos (información del sistema, archivos, tráfico de red, notas de extorsión, etc), resaltar los desarrollos realizados mediante honeyfiles, dado que es el marco referencia desde el que se afronta trabajo que se describe en el resto de la memoria. En cuando al tipo de procesamiento realizado, se muestra cómo las técnicas de ML son prolijamente utilizadas en la detección, pero no olvidamos otras como las estadística, la basada en entropía, monitorización, difusas, etc.

Desde el punto de vista de la detección, seguimos teniendo algunos problemas pese a la existencia de múltiples herramientas. Algunas de las razones radican en la falta de caracterización de las propiedades fundamentales del *ransomware*, la identificación de evaluaciones problemáticas normalmente en muestras que se solapan en comportamiento, lo que requiere un análisis de los posibles comportamientos y no solo los que se han observado. Y, por último, el desarrollo de un marco de trabajo de pruebas fiable que permita detectar todos los posibles ataques [457].

En tercer lugar y correspondiendo a la etapa de recuperación tras un ataque, revisamos especialmente los trabajos destinados a la recuperación de claves de cifrado utilizadas por el *ransomware* para deshacer el proceso.

Trabajo	**#Muestras**	**#Muestras activas**	**TP**	**#Muestras activas por familia**	**Fuente(s) o Método(s) de recolección**
Ahmadian [319]	NC	27	100	NC	NC
Ahmed [414]	1.254	673	98,8	48,1	`virusshare.com`, `virustotal.com`, araña a repositorios, foros
Berrueta [463]	70	NC	NC	NC	PCAP repository `http://dataset.tlm.unavarra.es/ransomware/`
Cabaj [344]	NC	787	98	395,5	`malwr.com`, `ransomtracker.abuse.ch`
Continella [315]	688	383	100	34,8	`virustotal.com`
Hasan [438]	1.283	360	97	17,1	`virusshare.com`
ISOT [464]	669				`https://www.uvic.ca/ecs/ece/isot/datasets/botnet-ransomware/`
Kharaz [314]	3.156	2.121	96,3	176,8	`minotauranalysis.com`,`malwaretips.com`, `malwareblacklist.com`, `malware.dontneedcoffee.com`
Kharraz [288]	9.432	1.174	100	40,5	`minotauranalysis.com`, `malwaretips.com`, `malwareblacklist.com`, `malware.dontneedcoffee.com`
Kharraz [461]	3.921	1.359	NC	90,6	Anibus [465], araña para repositorios púbicos, foros de seguridad
Kolondeker [453]	71	107	79,4	37,5	`virustotal.com`, `malc0de5.com`, `vxault.siri-urz.net`
Mbol [466]	NC	54	100	2,8	NC
Morato [349]	NC	54	100	2,8	NC
RISSP [467]	540	NC	NC	NC	`https://github.com/rissgrouphub/ransomwaredataset2016`
Scaife [330]	2.663	492	100	35,1	`virustotal.com`
Sgandurra [381]	1.450	582	96,3	52,9	`virustotal.com`
Temple [468]	1.156				Incidentes a IC (2013-2021) `https://sites.temple.edu/care/ci-rw-attacks/`
Tseng [469]	NC	155	NC	6,7	NC
Vinayakumar [470]	NC	755	100	107,9	`offensivecomputing.net`, `contagiodump.blogspot.in`, `malwr.com`, `github.com/ytisf/theZoo`, `virustotal.com`, `virusshare.com`

Tabla 3.3: Colecciones de muestras de *ransomware* encontradas en la literatura sobre detección (modificada de [457]).

Capítulo 4

R-Locker: detección temprana de *ransomware* en Linux

Como se ha visto con anterioridad, el *ransomware* es una pandemia hoy en día y se prevé que lo seguirá siendo en los próximos años. Si bien existen numerosas propuestas para combatirlo en diferentes etapas no existen tantas que sean lo suficientemente efectivas para construir una herramienta fácilmente despegable en los sistemas de red.

Centrándonos en la plataforma Linux, este capítulo abre con la descripción de la situación actual de esta amenaza para estos entornos y seguiremos con diferentes propuestas centradas en dicha plataforma, finalizando la descripción detallando la herramienta que hemos desarrollado, denominada R-Locker [471], para bloquear *ransomware* en Linux. Su nombre proviene de, e intenta ser, un juego de palabras sobre el *ransomware* y sus efectos: *ransomware locker*.

La propuesta de R-Locker es un nuevo enfoque no solo pensado en una detección temprana, sino que pretende frustrar la acciones del *ransomware*. Esta propuesta se basa en el despliegue de un conjunto de archivos trampa por el sistema de archivos del equipo a proteger. Estos archivos trampa tienen la propiedad de que el propio sistema operativo bloquea cualquier proceso que intente leer su contenido cuando se encuentran vacíos (no se ha escrito en ellos). Además, para frustrar las acciones del *ransomware*, el mecanismo es capaz de activar automáticamente las contramedidas para detener el cifrado. Es más, no necesita entrenamiento o conocimiento previo, por lo que el mecanismo es capaz de detener muestras desconocidas, es decir, ataques de *ransomware* de día cero.

Como prueba de concepto, se ha desarrollado su implementación para

plataformas Linux con el objetivo adicional de traspasarla a otros sistemas operativos. Así, en los siguientes capítulos, abordaremos por una parte la solución implementada de este enfoque para plataformas Windows y, seguidamente, las alternativas propuestas para Android. Esta solución muestra excelentes resultados tanto desde la perspectiva de la exactitud como desde el punto de vista de la complejidad; todo ello con un consumo de recursos muy bajo. Además, no requiere el uso de privilegios especiales para su instalación y no afecta al funcionamiento normal del sistema donde se instala.

4.1. *Ransomware* en plataformas Linux

Antes de presentar R-Locker, en este apartado veremos algunas familias de *ransomware* desarrolladas bien específicamente para Linux o bien variantes de Windows que se han adaptado a este sistema operativo. Esto permitirá ver que, si bien son poco numerosas en general, están creciendo en los últimos dos o tres años, especialmente las destinadas a atacar máquinas virtuales.

El primer troyano de cifrado dirigido contra sistemas Linux fue Linux.Encoder, conocido también con los nombres ELF/Filecoder.A o TRojan.Linux.RAnsom.A. Fue descubierto en 2015 y tuvo varias versiones destinadas a solventar diferentes errores de sus predecesores, pero todas ellas eran vulnerables al ataque de recuperación de claves [472].

Un par de años después, 2017, se descubrió la variante KillDisk para Linux, que incrementó la familia de *malware* para la limpieza de discos del mismo nombre, que se solía usar en ciber-espionaje o ciber-sabotaje [473]. A diferencia de su versión Windows, en Linux no utilizaba C&C a través de la API de Telegram. También el cifrado era diferente, ya que utilizaba DES-triple aplicado a bloques de archivos de 4.096 *bytes* y cada archivo se cifraba con un conjunto de claves de 64 bits diferentes. Además, reescribía el sector de arranque y utilizaba un cargador `Grub` para mostrar la pantalla de extorsión. Se utilizó por el grupo BlackEnergy como señuelo para borrar los equipos y las evidencias de sus ataques.

También en 2017 fue detectado Erebus, que infectó 153 servidores y unos 3.400 sitios web Linux de la compañía de hospedaje coreana Nayana [474]. Este ataque aprovechó la vulnerabilidad conocida como Dirty Cow [475] para acceder como administrador a los equipos, junto con la vulnerabilidad CVE-2017-5638 de Apache. A diferencia de otros *ransomware*, Erebus eleva a otro nivel la superposición de capas de cifrado, de forma que cada archivo encriptado tiene formato compuesto: una cabecera con el nombre original de archivo cifrado con RSA-2048, la clave RSA-2048 cifrada AES, la clave RC4 cifrada RSA-2048, y los datos cifrados con RC4. Mientras que cada archivo cifrado tiene sus claves RC4 y AES, la clave pública RSA-2048 es compartida. Estas claves RSA-2048 se generan localmente, pero la clave

```
Usage: %s [OPTION]... -i '/path/to/crypt'
Recursively crypts files in a path or by extention.

Mandatory arguments to long options are mandatory for short options too.
  -i, --indir        path to crypt
  -m, --minfile      minimal size of a crypted file, no less than 4096
  -r, --remove       self remove this file after work
  -l, --log          prints the log to the console
  -n, --nolog        do not print the log to the file /tmp/locker.log
  -d, --daemonize    runs a program as Unix daemon
  -w, --wholefile    encrypts whole file
  -b, --beginfile    encrypts first N bytes
  -e, --extentions   encrypts files by extentions
  -o, --nostop       prevent to stop working VM
  -p, --wipe         wipe free space
  -s, --spot         upper bound limitation value of spot in Mb
```

Figura 4.1: Interfaz de la orden para ejecutar LockBit [477].

privada se cifra con AES y otra clave generada aleatoriamente. El análisis resultante indica que el descifrado no es posible sin tener las claves RSA. Si bien, como suele ser general, se veían afectados archivos típicos (documentos de Office, bases de datos y multimedia), esta versión estaba codificada para alcanzar y cifrar equipos específicos; a saber, servidores web y los datos que estos contenían. Esto se vio corroborado por el hecho de que buscaba archivos en caminos específicos */var/www* de un servidor web o *ibdata* de la base de datos MySQL.

Recientemente, en 2021, se detectó cómo LockBit 2.0 afectó a servidores VMware ESXi y vSphere, que si bien no es exactamente Linux sí comparte muchas características, entre las que encontramos el poder ejecutar archivos en formato ELF64 [476]. Pero esto no es nuevo; muchas variantes de *ransomware* utilizan encriptadores de Linux, generalmente para poder atacar máquinas virtuales. Es el caso de las operaciones de HelloKitty (o sus variantes DeathRansom y Fivehands), BlackMatter, REvil, AvosLocker, Hive, Babuk, RansomExx/Defray, Mespinoza y GoGoogle. Cabe resaltar que el ataque a este tipo de servidores que soportan máquinas virtuales tiene un alto impacto en las compañías.

En el caso de LockBit, este suministra una interfaz de línea de comandos que permite establecer las características del cifrado, como son el nombre de camino a seleccionar, el tamaño mínimo de los archivos seleccionables, si se cifra el archivo o solo los primeros *bytes* especificados, si detener la máquina virtual o borrar el espacio libre, tal como muestra la Figura 4.1 [477].

Pero lo más destacable de esta familia, e igual que ocurría con HelloKitty [478], la posible detección del estado de las máquinas virtuales y detenerlas de forma que no queden corruptas mientras se cifran completamente con una única orden, ya sea en modo soft, hard o forzado, ya que en el código

Figura 4.2: Nota de extorsión de LockBit 2.0 en servidores VMWare ESXi [477].

se ha descubierto el uso de la orden `esxcli`:

```
esxcli vm process kill -t=[soft | hard | force] -w=%d
```

La Figura 4.2 ilustra la nota de rescate que muestra LockBit en servidores ESXi. Los operadores de Lockbit v2.0 suelen amenazar a sus víctimas con el hecho de que, si no cumplen sus demandas, publicarán los datos robados en su sitio de filtraciones. La Figura 4.3 muestra la página principal del sitio con los ataques en marcha.

El *ransomware* RansomExx, también conocido como Defray777, provocó en 2020 bastantes daños ya que atacó a las redes del gobierno de Brasil, al Departamento de Transportes de Texas (TxDOT), Konica Minolta, IPG Photonics y Tyler Technologies. Según el informe de Kaspersky [480], este

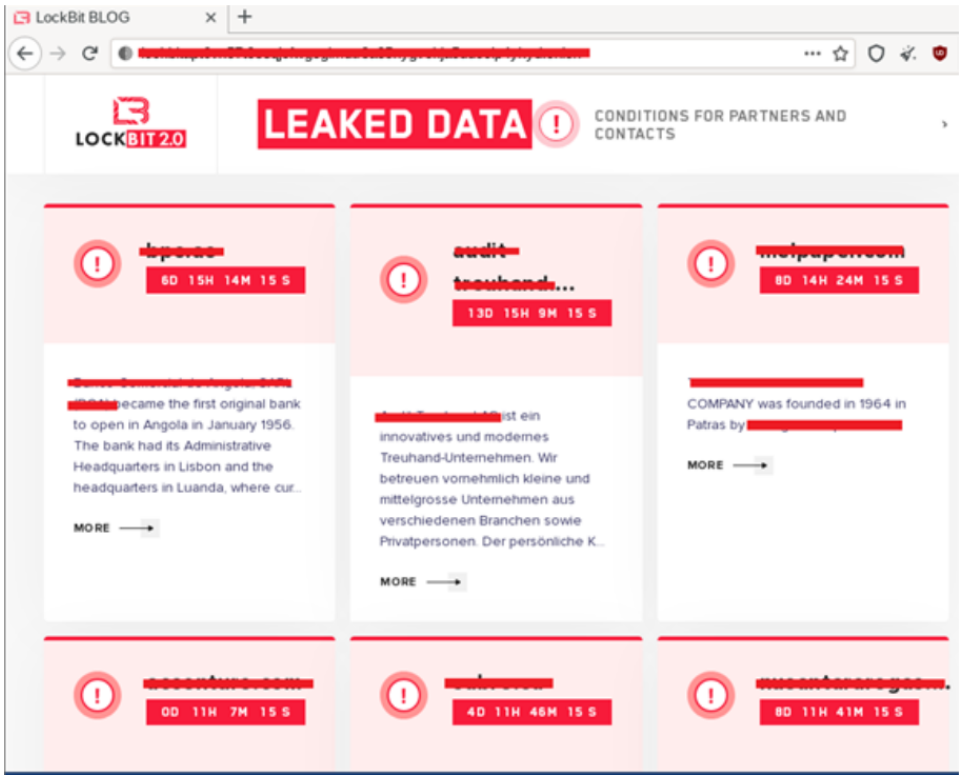


Figura 4.3: Sitio de filtración de datos de LockBit en la red TOR [479].

ransomware tiene bastantes similitudes con el troyano del mismo nombre de Windows (en el código, la nota de extorsión y el enfoque de la extorsión), pero ahora entregado como un ejecutable ELF de nombre *svc-new*. Embebidas en este ejecutable se encuentran la clave de cifrado (RSA-4096), la nota de extorsión y la extensión que recibirán los archivos cifrados. Un fragmento de código del mismo puede verse en la Figura 4.4 [480], donde se muestra en seudo-código el procedimiento de cifrado, y los nombres de variables guardadas junto a la información de depuración; elemento este último poco frecuente en este tipo de *malware*. Esta familia, como suele ocurrir con otras para Linux, no suelen tener mecanismos complejos como comunicarse con un C&C, terminar procesos, evadir el análisis inverso, etc.

En 2021 hace su aparición la versión de Linux del *ransomware* SFile aparecido en año anterior para Windows y operado por el grupo Escal [481]. Presenta algunas diferencias respecto a su versión de Windows, como que es capaz de cifrar archivos dentro de un rango de fechas (lo que permitiría, por ejemplo, cifrar solo archivos recientes no incluidos en una copia de seguridad), o utilizar el nombre de la víctima como parte de los nombres de archivos.

```
if ( a1 )
{
  v10 = strlen(s) + 277;
  v2 = alloca(16 * ((v10 + 23LL) / 0x10uLL));
  dest = (16 * ((&s + 7) >> 4));
  if ( dest )
  {
    strcpy(dest, s);
    strcat(dest, ".[redacted]");
    v3 = rand();
    sprintf(src, "-%08x", v3);
    strcat(dest, src);
    if ( fsize(dest) == -1 )
    {
      stream = fopen64(s, "r+");
      if ( stream )
      {
        v9 = fsize(s);
        if ( v9 )
        {
          if ( v9 > 15 )
          {
            mbedtls_aes_init(aes_ctx);
            pthread_mutex_lock(&csPreData);
            qmemcpy(ptr, &g_RansomHeader, 0x200uLL);
            mbedtls_aes_setkey_enc(aes_ctx, &g_KeyAES, 256LL);
            pthread_mutex_unlock(&csPreData);
            if ( !fseek(stream, 0LL, SEEK_END)
              && fwrite(ptr, 1uLL, 0x200uLL, stream)
              && !fseek(stream, -512 - v9, 1)
              && ProcessFileHandleWithLogic(stream, aes_ctx, a2, v9, CryptOneBlock) )
            {
              v13 = 1;
            }
```

Figura 4.4: Fragmento de código de RansomExx [480].

Los autores de TellYouThePass, descubierto en 2019, reescribieron en 2021 este *ransomware* en Golang para atacar tanto sistemas Windows como Linux [482]. Ambas versiones comparten más de 85 % del código. Este código es la carga de una herramienta de explotación de la vulnerabilidad conocida como Log4Shell, que permite la ejecución remota de código. Utiliza los paquetes de Go para claves RSA, entre los que encontramos `crypto_x509_MarshalPKCS1PublicKey`, `crypto_x509_MarshalPKCS1PrivateKey`, `encoding_pem_EncodeToMemory` y `crypto_rsa_GenerateMultiPrimeKey`. A diferencia de los comentados anteriormente, antes del cifrado, esta variante *ransomware* sí trata de finalizar diferentes procesos, si tiene permisos de *root*, relacionados con clientes de correo, aplicaciones de bases de datos, servidores web y editores de documentos. Tras ello, pasa a cifrar los directorios por orden alfabético A-Z, y cifra los archivos con extensiones de archivos multimedia y de los tipos de documentos más comunes. En el caso de Linux, se excluyen los directorios del sistema: */bin*, */boot*, */sbin*, */tmp*, */etc*, */lib*, */proc*, */dev*, */sys*, */usr/include* y */usr/java*.

Otro objetivo frecuente de algunas versiones de *ransomware* para Linux son los dispositivos NAS. Las razones principales para ello son básicamente dos [483]: almacenan datos de valor para sus usuarios y el nivel de concienciación de su seguridad es bajo. Entre las familias de *ransomware* que atacan estos sistemas encontramos:

- QLock - Explota la vulnerabilidad CVE-2021-28799 del software Hybrid Backup Sync, y cifra los archivos usando la utilidad 7-Zip con una clave generada como combinación de información del hardware (el número de serie del dispositivo) con un par de claves público-privadas. Además de cifrar los archivos, este borra las instantáneas (*snap*) del dispositivo.

- REvil - También denominado Revix (posiblemente de REvil for Linux), es un *ransomware* de post-intrusión, es decir, los actores lo ejecutan manualmente una vez que han accedido al sistema. Su configuración se establece mediante un archivo JSON que contiene, entre otros parámetros: una clave de 64 bits, el texto de la nota de extorsión codificado en base64, el nombre de la nota y la extensión que se les dará a los archivos cifrados. Después del cifrado, la nota contiene una clave única por víctima.

- eCh0raix - Se conoce también como QNAPCrypt ya que originalmente tenía como objetivos solo dispositivos NAS de QNAP, pero se ha ampliado a dispositivos de Synology. El grupo que lo opera muestra un conocimiento exhaustivo del funcionamiento de estos dispositivos, como evidencia el hecho de excluir del cifrado archivos y directorios que son cruciales para el funcionamiento del mismo y su interfaz web. Por ejemplo, se excluyen las rutas */usr/syno* y *qbox*, exclusivos de Synology y QNAP, respectivamente. A diferencia de otras familias, esta utiliza un C&C, situado en TOR, para descargar las claves de cifrado y una cartera de criptomonedas para el pago del rescate. Si no puede comunicarse con su C&C a través de un servidor *proxy* SOCKS5, no cifrará el dispositivo. Además, para evitar el doble cifrado, verifica si el archivo de la nota de extorsión (*README_FOR_DECRYPT.txt*) está presente y, si es así, aborta su ejecución. Este es un buen ejemplo de ataque *ransomware* que puede detenerse si se bloquea el conjunto de indicadores de compromiso (IoC) o si las defensas de red están activas.

- DarkSide - Si bien parece que no se conocen ataques a este tipo de dispositivos por parte de esta familia, las muestras analizadas revelan que el código de DarkSide esta preparado para atacar a dispositivos NAS, simplemente cambiando el valor de una variable.

Otro *ransomware* que muestra diferencias entre su versión Windows y Linux es DarkSide [484]. Tal como se recoge en la Tabla 4.1, se observa cómo

	Variante de Windows	**Variante de Linux**
Mecanismo de cifrado	Salsa20 con RSA-1024	ChaCha20 con RSA-4096
Bloques de cifrado	Matriz Salsa20 personalizada y generada aleatoriamente con `RtlRAnsomExW`	Bloque inicial ChaCha20 estándar, y construido con la constante `expand 32-byte k`
Configuración	Cifrada	No cifrada
Finaliza la MV	No	Si
Archivos objetivo	Todos salvo los especificados en la configuración	Archivos relacionados con MV (servidores VMWare ESXi) con las extensiones dadas en la configuración
Extensiones	Generadas aplicando varias veces CRC32 sobre el identificador hardware de la máquina victima	Embebido en la configuración como .darkside o pasado como parámetro de ejecución
Nombre del archivo de la nota de rescate	README.ID.txt: nombre embebido en el código y un identificador generado para la víctima	Embebido en la configuración (darkside_readme.txt) o pasado como parámetro en la ejecución

Tabla 4.1: Comparación de características de DarkSide entre sus variantes para Windows y Linux.

utilizan un cifrado diferente, la versión de Linux sí esta preparada para finalizar máquina virtuales, los archivos objetivos incluyen rutas específicos de este tipo de máquinas, las extensiones de los archivos cifrados son diferentes y la nota de rescate se genera de forma diferente.

Ademas de sistemas Linux, otro sistema tipo Unix afectado por el *ransomware* es FreeBSD. Estos sistemas ha sido objetivo de la familia *ransomware* Hive, que en este caso finaliza todos los procesos que no son del administrador, escanea y cifra todos los archivos en el directorio raíz, o de los directorios que se le indiquen en la orden de invocación. Para evitar la restauración del sistema de archivos, puede rellenar el espacio de disco con datos aleatorios [485].

4.2. Defensas contra el *ransomware* en sistemas Linux

Como vimos en el capítulo anterior, la medidas preventivas por sí solas no son suficientes para detener este tipo de ataques. Por tanto, necesitamos

un segundo nivel de protección basado en la detección del comportamiento del *ransomware* en relación a su interacción con el sistema de archivos, las llamadas al sistema, el acceso al Registro, las comunicaciones con su C&C, o el procedimiento de cifrado, y a ser posible permitan que una detección temprana que evite el daño en el sistema. En este sentido, la mayoría de las propuestas de detección descritas con anterioridad se ha desarrollado para el sistema operativo Windows. En este apartado, veremos algunas soluciones específicas para el sistema Linux.

La propuesta blkCtrace [486] aborda un mecanismo de análisis del comportamiento del *ransomware* mediante el rastreo de las entradas/salidas realizadas sobre disco con el diseño de una capa ligera de E/S para dispositivos de bloques. A la misma vez trata de evitar los altos costes de memoria necesarios para realizar dicha labor; de hecho, su ejecución solo supone un 11,2 % de penalización sobre el sistema base con una carga de 1 GB de lecturas y escrituras (50:50) aleatorias en un disco SSD.

Otra forma de defensa contra el *ransomware* es corregir posibles vulnerabilidades del sistema con una gestión automatizada de las actualizaciones del mismo, como es el caso de la propuesta que se recoge en [487]. Este trabajo aborda la actualización simultánea de varios servidores utilizando herramientas de fuentes abiertas como son Puppet y Mcollective. Además, la herramienta es capaz de evaluar la seguridad del sistema que se está actualizando usando el escaneo de CVE.

Otro aspecto a considerar en los ataques de *ransomware* a sistemas Linux es cuando el sistema utiliza una capa de virtualización para ejecutar aplicaciones Windows, como puede ser Wine^{HQ}(`https://www.winehq.org/`) [488]. El trabajo [489] muestra que, efectivamente, el impacto del *ransomware* diseñado para sistemas Windows es elevado en diferentes componentes del sistema como el sistema de archivos, la red, los servicios, los procesos, etc. En concreto, sus resultados muestran que el 50 % de las muestras ejecutadas afectaban a todos las carpetas del sistema salvo las que requieren permisos de administrador (*/etc*). Por tanto, concluye la necesidad de mantener las medidas preventivas generales comentadas en el Capítulo 3 en sistemas Linux que utilizan software de virtualización de aplicaciones.

Pensado para la detección de *malware* en general, pero aplicable también a *ransomware*, los autores de [490] han creado un prototipo para detectar y recuperar programas ejecutables en formato ELF capaz de determinar con exactitud cuál es el ejecutable que tiene un comportamiento malicioso en base a su comportamiento estático y dinámico. Esta técnica es especialmente efectiva en el caso de *malware* sin archivos, es decir, en memoria. Las únicas ejecuciones excluidas del análisis son las hebras *kernel*. Para ello, la fuente de información procede del seudo-sistema de archivos */proc/<pid>/*, junto con la descripción a nivel *kernel* del espacio de memoria virtual del proceso, a través de la estructura de datos `mm_struct`. En una línea similar pero usando ML, encontramos el trabajo [491], donde se utilizan las llamadas el

sistema Linux, capturadas mediante un entorno de *sandbox*, para alimentar tres algoritmos (J48, Adaboost y *Random Forest*) para generar los modelos predictivos.

En el Apartado 3.3.1 se describían diferentes técnicas que utilizan archivos especiales como trampa o señuelos. Dichos enfoques pueden detectar la ejecución del *ransomware* cuando este ha iniciado su actividad de cifrado o no suelen ser autónomos para provocar su detención en el momento de ser detectado. En este contexto, el apartado siguiente describe un nuevo enfoque de detección y reacción frente al cripto-*ransomware* basado en el uso de *honeyfiles* con tres importantes beneficios: *i)* la ejecución del *ransomware* se bloquea completamente cuando este accede a un archivo trampa, *ii)* las contramedidas se activan inmediatamente de modo automático para solventar la infección, y *iii)* la complejidad y sobrecarga de la solución es muy baja. Este enfoque se ha implementado y evaluado en sistemas Linux mediante la herramienta bautizada como R-Locker.

4.3. Un nuevo enfoque basado en *honeyfiles* para frustrar las acciones del cripto-*ransomware*

En este apartado introduciremos primero la metodología funcional general cuyo objetivo es frustrar las acciones del cripto-*ransomware*. Esta debe ser ligera mantenimiento la exactitud y eficiencia para combatir la amenaza. Basándonos en estos principios, se describirá a continuación la herramienta R-Locker, una implementación específica de la metodología propuesta por los autores para plataformas Linux.

4.3.1. Enfoque basado en *honeyfiles* para solventar el problema del cripto-*ransomware*

La operación última del cripto-*ransomware* descansa en el escaneo del sistema de archivos de la máquina infectada, ya sea de forma aleatoria o selectiva como vimos en la Sección 2.4.8, en busca de archivos a los que acceder y cifrar su contenido. Basándonos en esta característica general, proponemos una solución anti-*ransomware* para crear un *honeyfile* (archivo 'atractivo' que atraerá al atacante) que sirva como trampa para capturar las muestras de este tipo de *malware*. Esta propuesta presenta dos características beneficiosas $\mathbf{F}=\{F1, F2\}$:

F1. La muestra de *ransomware* quedará bloqueada cuando acceda al *honeyfile*, de forma que el resto del sistema permanecerá sin daños.

F2. Además de bloquear la muestra, el evento maliciosos se notificará adecuadamente y/o se desplegará la contramedida automáticacamente para solventar la amenaza.

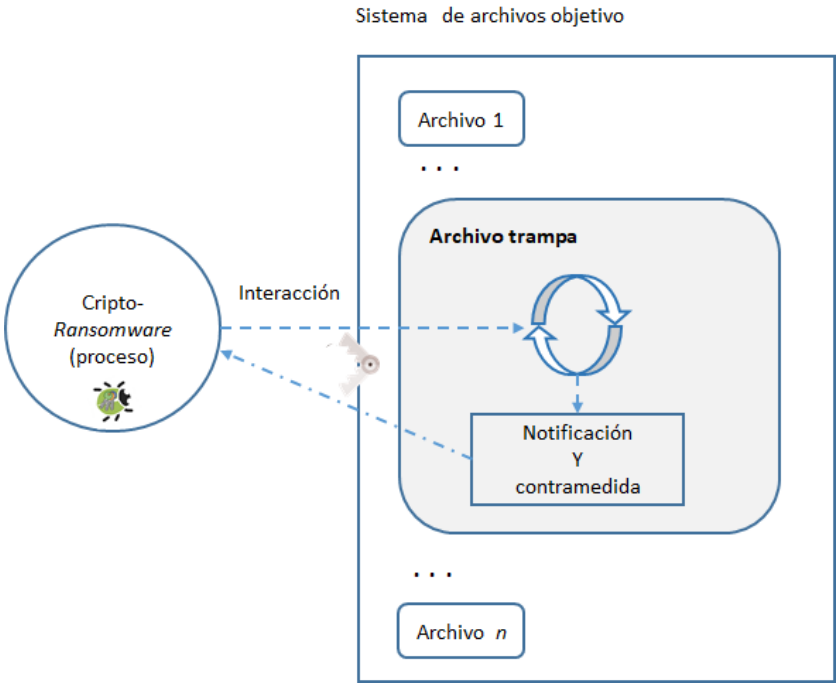


Figura 4.5: Metodología funcional de un *honeyfile*.

La metodología comentada se corresponde con la arquitectura funcional mostrada en la Figura 4.5. El procedimiento operacional es conceptual y, por tanto, independiente de la plataforma específica donde se implemente (Linux, Windows, Android, etc.). Además de las propiedades anti-*ransomware* antes mencionadas, se deben satisfacer otras características con el objeto de que sea escalable, utilizable y, por tanto, una solución válida para entornos reales. En esta línea, los principales requisitos son **R**={R1-R5}:

R1. *Efectividad*, de forma que las acciones dañinas del *ransomware* sobre el sistema sean nulas o se minimicen. De otra forma, no sería una solución válida.

R2. *Consumo bajo*, tanto desde el punto de vista computacional como de uso de memoria y almacenamiento. De otra forma, no sería escalable.

R3. *Claridad*, no requiera privilegios espaciales para su instalación o ejecución, ya que, si no, no sería utilizable por usuarios finales.

R4. *Transparencia*, desde la perspectiva del funcionamiento normal de entorno de ejecución; esto es, la solución no debe afectar al resto de aplicaciones y servicios. De otra forma, podrían producirse comportamientos inesperados o mal funcionamiento del sistema.

R5. *Simplicidad*, de forma que no se necesiten procedimientos complejos. Si bien R5 podría no ser obligatoria, parece una propiedad adicional atractiva para aplicarla en sistemas reales.

Desarrollar una solución anti-*ransomware* que satisfaga los beneficios básicos **F** y los requisitos **R** no es una tarea trivial. Si embargo, como prueba de concepto que pueda completarse y extenderse a otras plataformas, describiremos en la siguiente sección una implementación para sistemas Linux/Unix. Esta solución se sustenta en el uso del mecanismo de comunicación entre procesos FIFO o *cauce con nombre*, presente en los sistemas tipo Unix, y que pasamos a describir.

4.3.2. R-Locker: Implantación en plataformas Linux/Unix

Como hemos indicado, vamos a comentar la implementación de la metodología propuesta para las plataformas Unix, en particular para los sistemas Linux. Para ello, primero discutiremos la solución de detección en sí, y luego cómo esta puede desplegarse en sistemas Linux para protegerlos adecuadamente.

Al objeto de alcanzar la característica principal o beneficiarse de la solución de un *honeyfile*, F1 (es decir, bloquear al *ransomware* mientras que accede a él), mientras cumple con los requisitos fijados (en particular R3 y R4 relacionados con la ausencia de privilegios especiales y de no interacción con el resto del sistema), pensamos inicialmente utilizar una archivo 'infinito' para distraer al *malware* y frustrar sus acciones. Por ejemplo, se podría crear un archivo de tipo enlace al dispositivo cero (*/dev/zero*), que es una fuente infinita de ceros. Lamentablemente, aunque esta solución puede funcionar ya que mantendría al *ransomware* cifrando un archivo de tamaño infinito, no sería efectiva por una razón básica: el enorme espacio de disco necesario para almacenar la versión cifrada del archivo trampa dejaría al sistema operativo sin almacenamiento. Esto es, el requisito R2 no se podría alcanzar.

También podríamos simular un archivo infinito modificando funciones relacionadas con lecturas de archivos de las bibliotecas. Sin embargo, esto daría lugar a algunos problemas: (a) no conocemos cuáles de las bibliotecas (que son diversas) usará el *ransomware*, y (b) las técnicas para modificar las funciones son complejas (por ejemplo, instrumentación binaria [492]). Como consecuencia, R4 y R5 no se alcanzarían en este caso.

Una solución simple y elegante para alcanzar nuestros objetivos, tanto F1 y F2, mientras se satisfacen todos lo requerimientos **R** establecidos para la solución, es hacer uso del mecanismo FIFO. Un FIFO, aunque es un mecanismo de comunicación entre procesos (IPC - *Inter-Process Communication*), se crea como un objeto con nombre en el sistema de archivos y que tiene dos propiedades interesantes y útiles para nuestro propósito debido a

su naturaleza dual [493]:

- Como es un objeto en el sistema de archivos, un FIFO es (en cierto sentido) un archivo convencional accesible en la forma general por las aplicaciones y servicios. Es decir, sería visible por el *ransomware* cuando explora el sistema de archivos y resulta manipulable con operaciones convencionales de archivos (`open`, `close`, `read`, `write`, etc.).
- Como los cauces, un FIFO también implica un canal de comunicación de tamaño finito entre dos procesos: un lector y un escritor. En este punto, es importante resaltar que la sincronización entre ambos procesos, lector y escritor, es automáticamente gestionada por el *kernel* del sistema. De esta forma, si un proceso intenta leer un cauce vacío, o que contiene menos información de la que se desea leer, el *kernel* bloqueará al proceso lector hasta que el proceso escritor inserte los *bytes* necesarios en él. Por otro lado, un proceso que intente escribir en un cauce lleno se verá bloqueado por el sistema operativo hasta que se libere el contenido que almacena.

De lo comentado, nuestra propuesta anti-*ransomware* para sistemas Linux consta de una aplicación, denominada R-Locker, implementada en C y Java. Dados sus objetivos (a saber, bloquear el *ransomware*), opera como sigue (ver Figura 4.6 y Algoritmo 1):

1. Primero, el proceso despliega la solución creando un cauce con nombre utilizando la llamada al sistema `mkfifo()`. Este archivo será el *honeyfile* central o archivo trampa (línea 10).
2. En segundo lugar, el proceso abrirá el cauce en modo solo escritura (`O_WRONLY`) y escribe en él los *bytes* necesarios para llenarlo (líneas 14 y 16, respectivamente). Los *bytes* escritos deben ser diferentes de `EndOfFile` y su número mayor que el tamaño de los que admite el cauce, el cual puede depender del sistema específico, aunque un valor típico es 4 KB.
3. El fragmento *[...]* de la Figura 4.6 indica el estado 'durmiendo' en la terminología Unix (bloqueado, en la terminología general de sistemas operativos). Esto es, ya que inicialmente no hay lector accediendo al archivo trampa, el mencionado proceso escritor es bloqueado por el sistema. En este punto, la trampa está lista y esperando una presa (línea 16).
4. A partir de aquí, cuando un proceso externo (un supuesto *ransomware*) acceda a la trampa e inicie la lectura de la misma, el sistema operativo lo bloqueará cuando el número de *bytes* leídos supere el tamaño del cauce. Simultáneamente, el proceso escritor previamente detenido es

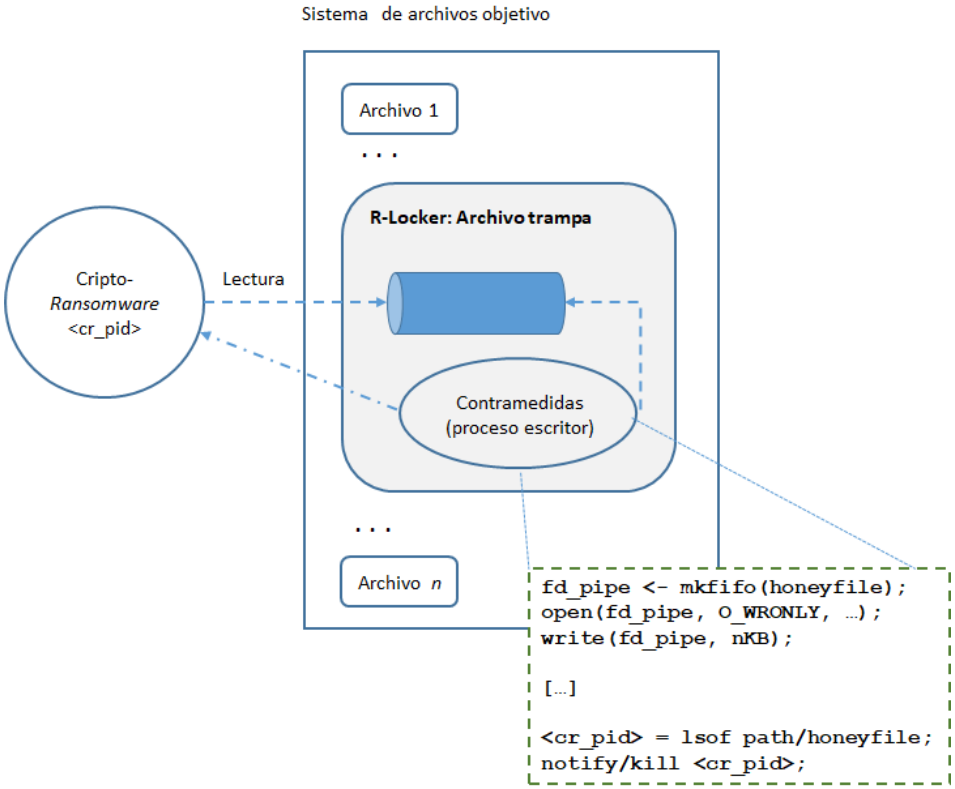


Figura 4.6: Anatomía de R-Locker.

automáticamente despertado (línea 17) por el *kernel* y sigue su flujo de ejecución podrá ejecutar las contramedidas como siguen:

- Primero, se determina el identificador (PID) de la aplicación que esta accediendo al *honeyfile*, bien a través de `/proc/<pid>/fd` o de la orden `lsof` (línea 17).
- Segundo, se notifica adecuadamente al usuario al objeto, si es necesario (línea 19), de eliminar al proceso correspondiente (<*cr_pid*> del *ransomware* en la Figura 4.6) e incluso desinstalarlo del sistema.

Resumiendo, podemos ver en la Figura 4.7 el flujo operacional general de R-Locker. Después de su despliege, la instalación del mismo es muy simple:

1. Después de ejecutar el binario asociado, aparecerá la pantalla principal que se muestra en la Figura 4.8, donde el botón 'Iniciar Servicio' inicializa la herramienta y el botón 'Detener Servicio', la finaliza.

Algoritmo 1: Seudocódigo de *R-Locker para Linux.*

```
   Entrada: nula
   Salida: nula
 1 ruta_trampa                    // Enlace simbólico al honeyfile
 2 lista_extensiones_trampa ← {.pdf .jpg .doc ...}
 3 Honeyfile                                  // Nombre del FIFO
 4 Función PoblarTrampas(ruta_trampa):
 5     lista_carpetas ← Generar la lista de carpetas
 6     para cada carpeta en lista_carpetas hacer
 7         si no existe ruta_trampa entonces
 8             symlink(Honeyfile, ruta_trampa.extension_trampa);
 9     Función InstanciarHoneyfile(rutaFIFO):
10         mkfifo(Honeyfile,...)                    // Crear el FIFO
11     Función Principal:
12         InstanciarHoneyfile(Honeyfile)
13         PoblarTrampas(Honeyfile)
14         dF = open("Honeyfile, O_WRONLY) // Abrimos el FIFO
               de escritura
15         para (;;) hacer
16             write(dF, 4 KB + 1) // El proceso queda bloqueado
                   hasta que llegue un lector
               // Cuando se desbloque el proceso...localizar su
                   ID a través de su inodo
17             cr_pid ← lsoftHoneyfile
               // Terminar proceso lector
18             kill(cr_pid, señal_finalizar)
19             ... notificar usuario
```

2. En el primer caso, se crea un único archivo trampa central como se ha descrito recientemente.

3. Con el objetivo de maximizar la protección del sistema de archivos mientras se minimizan los recursos consumidos, se despliegan por el sistema de archivos un conjunto de enlaces simbólicos que apuntan a la trampa central. Estos actuarán como una solución *honeyfile* distribuida por todo el sistema de archivos (Figura 4.9(a)).

4. Si en lugar de iniciar el servicio, pulsamos el botón 'Detener servicio' se eliminan tanto el *honeyfile* como los enlaces simbólicos, y el proceso/aplicación finaliza.

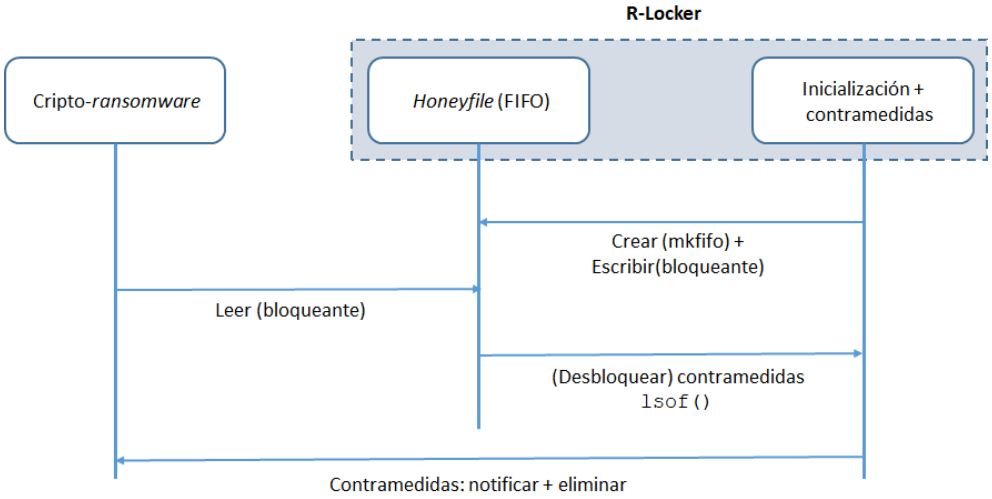


Figura 4.7: Flujo operacional de R-Locker.

Respecto a los anteriormente mencionados enlaces simbólicos que actúan como archivos trampa distribuidos, es importante remarcar que:

1. Con el propósito de realizar detección temprana, los enlaces se crean en la primera entrada del directorio correspondiente.

2. El uso de nombres y/o extensiones atractivas (por ejemplo, *videopersonal.mpg*) pueden ser interesantes para capturar, en este caso, la atención de un potencial *ransomware* selectivo, como vimos en la Tabla 2.8 de la Sección 2.8.

3. Los enlaces pueden distribuirse por todo el sistema de archivos, incluidos directorios relacionados con el *kernel* y sistema operativo, o solo cubriendo el directorio */home* del usuario (Figura 4.9(b)). En el primer caso, se necesitan privilegios especiales durante la instalación.

Un vez descrita la metodología global y cómo se ha particularizado para sistemas Linux, evaluaremos el enfoque en un escenario real en la siguiente sección.

4.4. Evaluación experimental

Después de describir la solución anti-*ransomware*, esta sección esta dedicada a evaluar el rendimiento de R-Locker con muestras reales de *ransomware*. Con este objetivo, se describe inicialmente el entorno experimental. Después de ello, se realizarán algunos experimentos para obtener algunas

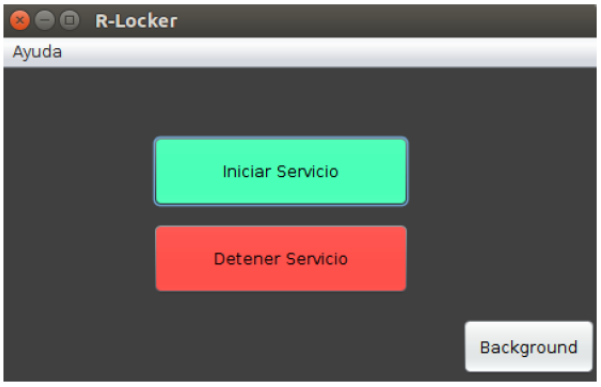


Figura 4.8: Pantalla principal de R-Locker.

métricas operacionales en términos de exactitud de la detección, consumo de recursos e impacto general en el entorno objetivo. Finalmente, abordaremos la discusión de las conclusiones principales y algunos aspectos relevantes de la propuesta.

4.4.1. Entorno experimental y muestras de cripto-*ransomware*

El entorno desplegado para realizar los experimentos de detección de *ransomware* es una única máquina corriendo Ubuntu 16.04 de 64 bits con acceso a Internet. Sin servicios especiales instalados salvo la excepción de un servidor Apache para HTTP con información personal del usuario.

Respecto a las muestras de cripo-*ransomware* con propósito de evaluación, hay tres casos que considerar:

- *C1*: Como primera aproximación al problema, hemos usado Linux GPG Suite (`https://gpgtools.org/`) para desarrollar una muestra de *ransomware* genérico ideado para cifrar el sistema de archivos de usuario de una manera sistemática.

- *C2*: Como prueba de concepto para plataformas Linux, la herramienta abierta Bash-Ransomware (`https://github.com/SubtleScope/bash-ransomware`) se basa en OpenSSL para cifrar archivos y su operación es similar a *Cryptowall*, el cripto-*ransomware* bien conocido y dañino que apareció a finales de 2013.

- *C3*: Más específico que los anteriores, y motivados por el impacto conocido en sistemas finales, hemos considerado *Linux.Encoder.1* (`https://vms.drweb.com/virus/?i=7704004&lng=en`) y *WannaCry* (`https://github.com/aguinet/wannakey`) como muestras de *ransomware* para la experimentación. Respecto a WannaCry, desarrollado es-

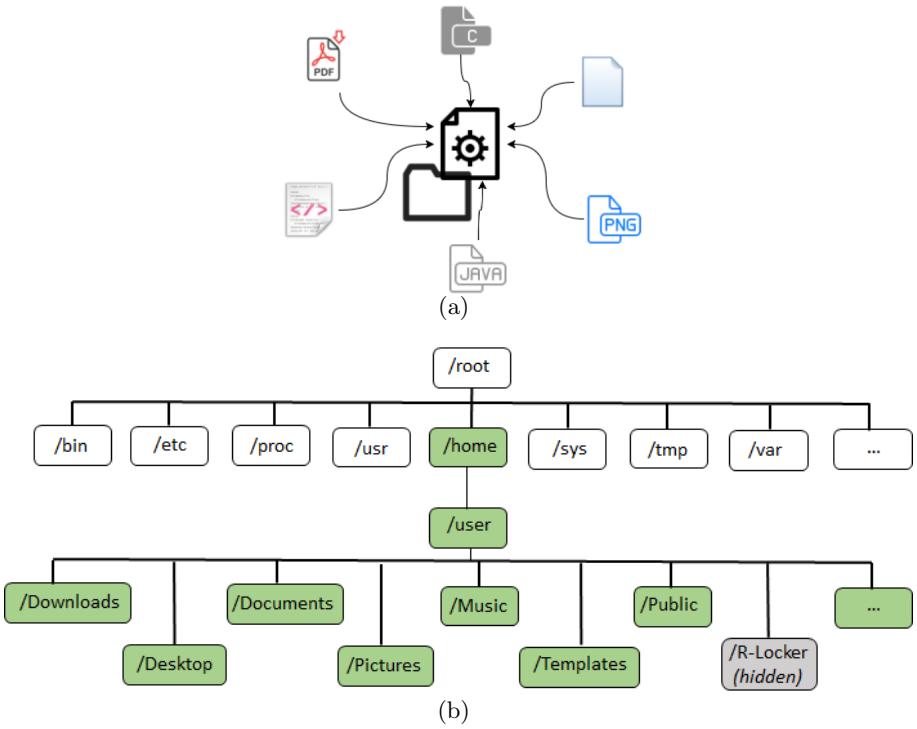


Figura 4.9: Despliegue de *honeyfiles* en el sistema de archivos como enlaces simbólicos a una trampa central única en R-Locker: enlace conceptuales (a), y directorio del usuario */home/user* en nuestro escenario concreto (b).

pecíficamente para sistemas Windows, nuestra experimentación lo ha ejecutado en la máquina Linux a través del software de virtualización de aplicaciones Wine.

4.4.2. Resultados de detección y rendimiento de R-Locker

Con el entorno operacional base funcionando y con R-Locker instalado y funcionamdo, hemos ejecutado cada una de las muestras de los *ransomware* considerados en los casos C1 a C3. En todos los casos, el comportamiento de R-Locker fue el esperado respecto a las características deseadas: las muestras de *ransomware* fueron inmediatamente bloqueadas (F1) y, por consiguiente, se notificó su eliminación (F2). Desde la perspectiva de los requisitos operacionales R1-R5, R-Locker cumple con:

R1. *Efectividad*: Debido principalmente a la distribución de los enlaces al *honeyfile* central repartidos por todo el sistema de archivos de usuario, junto con su inclusión en la primera entrada de cada directorio, las muestras de *ransomware* fueron detectadas/notificadas y bloqueadas inmediatamente (ver Figura 4.10 como un ejemplo para el caso de

WannaCry). Por tanto, podemos concluir una efectividad máxima de R-Locker de acuerdo con R1 en la Sección 4.3.1. Primero, la exactitud de la detección es del 100 % puesto que todas las muestras fueron detectadas. Segundo, el tiempo en frustrar sus efectos dañinos (y así el daño causado en el sistema) es mínimo ya que todos los directorios incluyen un enlace al *honeyfile*, que permite la frustración de la actividad del *ransomware* desde su inicio.

R2. *Bajo consumo*: Gracias al diseño de R-Locker basado en un FIFO como trampa central, solo consume unos 2 KB de almacenamiento en disco (los enlaces simbólicos no suelen necesitar almacenamiento extra salvo una entrada de directorio *entrada de i-nodo*), espacio que suele estar computado a la carpeta correspondiente. Es más, la detección y captura del *ransomware* tiene un coste computacional muy bajo (una vez que ha sido bloqueado gracias a la distribución de las trampas por el sistema de archivos). Como ejemplo, indicar que en nuestro caso son solo necesarios entre 500 y 800 ms para identificar (orden `lsof`) y notificar (interfaz gráfica) el proceso malicioso. También es importante resaltar, como indicábamos en la el punto 3 de la Sección 4.3.2, que R-Locker no consume recursos computacionales mientras espera la acción del *ransomware* (esto es, el proceso de la herramienta espera en el FIFO).

R3. *Sencillez*: R-Locker se puede instalar y ejecutar sin requerir privilegios o permisos especiales. A este respeto, la única, y obvia, limitación es que solo una parte de la totalidad del sistema de archivos será protegida (la que corresponde al usuario que instala la herramienta, ver Figura 4.9(b)). Esto sería compatible con el comportamiento de muchas muestras de *ransomware* tienden a no alterar archivos del sistema para mantener este operacional.

R4. *Transparencia*: Para comprobar comportamientos no deseados o malfuncionamiento, hemos probado el acceso de programas legítimos a los enlaces a la trampa: Adobe Acrobat para extensiones *pdf*, Microsoft Word para archivos *doc*, etc. Todas ellas abortan sin consecuencias, ya que el proceso correspondiente que lee la trampa produce un 'error de formato'. En otras palabras, R-Locker alcanza el objetivo deseado sin interferir un el funcionamiento normal del resto del sistema. Si embargo, algunas consideraciones sobre este aspecto se abordan en la Sección 4.4.3.

R5. *Simplicidad*: R-Locker ha demostrado ser simple y autónomo, sin necesidad de procesos adicionales que complementen su funcionalidad.

En comparación con otras herramientas probadas en nuestra experimentación, por ejemplo, Cyptostalker, R-Locker ha mostrado un mejor funcio-

Figura 4.10: Detección de la amenaza realizada por R-Locker.

namiento en varios aspectos. De esta forma, aunque se ha detectado el *ransomware* y notificado por herramientas alternativas, el efecto sufrido por el sistema es más severo que con R-Locker. De hecho, era necesario un proceso de monitorización activo (y por tanto consumiendo CPU) para determinar la ocurrencia de un ataque potencial de cripto-*ransomware*.

4.4.3. Consideraciones adicionales

Como consecuencia del análisis previo, debemos concluir las bondades de R-Locker desde varios aspectos. Además, debemos remarcar que el enfoque descansa en el hecho de que el *ransomware* debe acceder a los archivos para cifrarlos, y no en comportamientos complejos o conocimiento previo. Como consecuencia principal, nuestra solución es capaz de atrapar muestras de día cero. Por supuesto, los experimentos de detección realizados anteriormente corresponden a ataques desconocidos, día cero, en tanto en cuanto no se han observado previamente ni usados para entrenar la herramienta.

A pesar de lo dicho, se necesitan consideraciones adicionales para un despliegue exitoso del enfoque en entornos reales. Estas son como sigue.

Como hemos mencionado, la disposición y distribución de los enlaces simbólicos a la trampa central es primordial para proteger el sistema completo. De esta forma, esta tarea debe abordarse de forma cuidadosa para un sistema objetivo específico de cara a maximizar la oportunidad de frustrar el ataque. Para ello, entre otras cuestiones, deberíamos preguntarnos ¿dónde ubicar las trampas, solo en el directorio raíz o en todo el árbol de directorios?, ¿qué extensiones y nombres de archivos debemos usar?, y deben abordarse adecuadamente.

También relacionado con los enlaces, es conveniente monitorizarlos al objeto de determinar la ocurrencia potencial de cambios en su distribución. Si es así, este hecho debe corregirse. Como caso particular, debemos tener

espacial cuidado en la eliminación potencial de la trampa central. En nuestro caso, hemos intentado minimizar el riesgo al crear un archivo oculto (con prefijo '.' en Linux) en un directorio específico donde se instala la aplicación R-Locker. A pesar de ello, un proceso de monitorización sigue siendo necesario para detectar un posible borrado. En particular, usamos el mecanismo *i-notify* para ello, que permite al sistema operativo detectar eventos sobre objetos del sistema de archivos cuando se accede a ellos y notificarlos a la herramienta [493].

Somos conscientes que nuestra defensa puede sobrepasarse parcialmente accediendo de forma aleatoria a los archivos dentro de un directorio. En el peor caso, la muestra se bloqueará después de cifrar todos los archivos de la carpeta. Para evitar esto, se puede desplegar más de un enlace a la trampa por carpeta. Tal vez un procedimiento más crítico para eludir la defensa es estudiar los metadatos del archivo *honeyfile* para corroborar su correspondencia con la extensión asociada (pdf, jpg, ...), que no se respeta en nuestro caso. Sin embargo, los enlaces simbólicos ofuscan el tipo de archivo específico al que apuntan, cambiando la extensión del mismo, para que así sea más compleja su verificación por parte del *ransomware*; lo que favorece nuestra solución.

Otro aspecto a considerar relativo a los enlaces simbólicos en su implementación en Linux es que estos son archivos regulares cuyo contenido es la ruta del objeto del sistema de archivos al que apuntan. Esta característica hacen que su tamaño sea no nulo y se pueda aumentar si se almacena una ruta larga, lo que es una propiedad adecuada para la detección ya que, como vimos en la Sección 4.1, algunas muestras de *ransomware* permiten al operador del mismo establecer el tamaño mínimo de los archivos potencialmente seleccionables como archivos víctima.

Existen aún un aspecto adicional sobre R-Locker que necesita un breve comentario. En su estado actual, terminar y/o desinstalar el proceso sospechoso de ser *malware* descansa sobre el usuario final, que, si es necesario, requiere introducir el PID del procesos detectado y confirmar las acciones subsecuentes (como se puede observar en la Figura 4.10). Para mejorar la facilidad de uso de la propuesta, la intervención del usuario puede reemplazarse o complementarse mediante el uso (semi)automático de listas blancas/negras de aplicaciones conocidas tanto benignas como malignas. De esta forma, el funcionamiento de órdenes del sistema como `copy`, o las aplicaciones utilizadas para generar copias de seguridad, pueden acceder a los datos a través de su inclusión en una lista blanca. Por el contrario, aplicaciones como una muestra de *Wannacry* serán filtradas mediante la lista negra.

A pesar de que todos los aspectos previos son relevantes para una nueva versión de R-Locker, pensamos firmemente que ninguno de ellos devalúa el prometedor rendimiento alcanzado por este enfoque en su actual estado. En concreto, si tenemos en consideración que ninguna de las soluciones actuales son definitivas para resolver este acuciante problema social.

4.5. Conclusiones

Se ha introducido en este capítulo una metodología pensada para detectar de modo temprano y frustrar las acciones del cripto-*ransomware*. Está basada en el despliegue de una estructura de *honeyfiles* para bloquear *ransomware* cuando accede a un archivo trampa, de forma que preserve intactos el resto de archivos del sistema. Es más, mientras la muestra maligna es bloqueada, sería deseable que se lance de forma automática una contramedida destinada a erradicar el problema del entorno. Como prueba de concepto, hemos implementado la metodología para plataformas Linux haciendo uso de cauces con nombre o FIFO. La herramienta resultante se denomina R-Locker.

Hemos demostrado mediante la experimentación el comportamiento correcto del enfoque desde diferentes perspectivas, que incluyen la exactitud de la detección y la eficiencia del funcionamiento. De esta forma, R-Locker alcanza los objetivos de detección establecidos con un consumo bajo de recursos y sin interferir en el funcionamiento normal del entorno. También hemos discutido algunos aspectos prácticos principalmente relacionados con el despliegue y mantenimiento de la estructura de *honeyfiles* con el objetivo de maximizar el éxito del método en entornos reales.

Las mejoras propuestas se abordan en los próximos capítulos junto con la migración a otras plataformas, en particular para Windows (Capítulo 5) y Android (Capítulo 6) dada su aceptación general entre usuarios y compañías, además de su importante afectación por el *ransomware* en la actualidad. Si bien la solución basada en *honeyfiles* es aplicable en ambos casos (el mecanismo de FIFO existe en ambas plataformas), debemos abordar algunos aspectos específicos para construir una solución real. En particular, el espacio de nombres de los cauces con nombre en Windows es diferente del espacio de nombres de archivos, mientras que los enlaces simbólicos y cauces tiene serias limitaciones en Android desde el punto de vista de su uso.

Más allá de los desarrollos específicos, serían deseables acciones de I+D más genéricas. La práctica muestra que la tecnología de decepción no es una estrategia única, sino una capa adicional compatible con las defensas en profundidad existentes. De esta forma, será interesante desplegar y combinar mecanismos de rastreo asociados con los archivos trampa desplegados para fortalecer la seguridad general en varios aspectos. En particular, ser capaces de aprender sobre la actividad del atacante con respecto a los procedimientos y mecanismos de infección y penetración, como de las técnicas potenciales de ofuscación. Por eso, debemos desarrollar enfoques de defensa nuevos y más avanzados desde la comunidad investigadora.

Capítulo 5

Detección temprana de *ransomware* en Windows con R-Locker

En el capítulo anterior introducíamos la herramienta anti-*ransomware* R-Locker para sistemas Unix. La propuesta se sustenta en una enfoque basado en *honeyfiles*, donde archivos trampa 'infinitos' se diseminan por el sistema de archivos para detectar y bloquear de forma efectiva las acciones del *ransomware*. Esta herramienta ha sido extendida a sistemas Windows con tres nuevas contribuciones. Primero, R-Locker en su implementación en este sistema operativo debe conectar objetos que pertenecen a diferentes espacios de nombres para gestionar los FIFO y hacerlos accesibles desde el sistema de archivos. Segundo, se mejora la gestión general de los *honeyfiles* desplegados por el sistema de archivos para maximizar la protección. Finalmente, se bloquean procesos sospechosos de forma (semi)automática mediante el uso de listas blancas/negras de forma dinámica. Como en el trabajo anterior, la nueva versión de R-Locker para Windows se demuestra efectiva y eficiente para frustrar las acciones del *ransomware*.

Antes de entrar en los detalles de la herramienta desarrollada, revisaremos algunas soluciones específicas para la detección del *ransomware* en plataformas Windows que, como vimos en la Sección 2.2, es la plataforma más utilizada a nivel mundial y, por tanto, el objetivo más atacado.

5.1. Cripto-*ransomware* en Microsoft Windows

Un aspecto inicial importante previo a establecer mecanismos de detección es la caracterización del comportamiento del cripto-*ransomware* durante su ejecución sobre el sistema operativo, como pueden ser las llamadas a la

API usada con más frecuencia [494], los archivos creados o el proceso de cifrado [495], entre otras.

Como se indicaba en el Capítulo 3, gran parte de los mecanismos o herramientas de detección analizados se han desarrollado para el sistema operativo Windows, que es la plataforma más extendida a nivel mundial. En este apartado daremos un breve repaso de los mismos a modo de recordatorio pero vamos a dar un nuevo enfoque. En el capítulo dedicado a la detección abordábamos la revisión de los mecanismos desde el punto de vista de las fuentes de datos y del tipo de procesamiento realizado sobre los mismos. Ahora, realizamos una aproximación desde la etapa del ataque del cripto-*ransomware*, como se refleja en la Figura 5.1 [496].

Los vectores de infección son complejos de rastrear debido a que, en general, el análisis de *ransomware* se hace sobre muestras encontradas en un sistema y estudiadas en una *sandbox*. Los mecanismos de prevención/detección (etapa de entrega en la Figura 5.1) más importantes son la concienciación sobre la posibilidad de un ataque de este tipo, la realización de copias de seguridad (como por ejemplo [287]), establecer los controles de acceso adecuados, tener aseguradas copias de Volumen Compartido, o la detección basada en firmas (como se vio en [171, 149, 150, 354]).

En la fase de despliegue, que incluiría tanto la explotación de las vulnerabilidades del sistema como la instalación de los programas para la infección, tenemos la monitorización del sistema, bien de las llamadas a la API (ejemplos en [206, 382, 390, 417, 470]) o de eventos producidos en el mismo (como el caso de [207]).

En la fase de destrucción (C&C y acciones) podemos clasificar las herramientas de detección por: análisis de red ([235, 394, 344, 445, 469]), *honeypots* basados en red ([166]) o en sistemas ([337, 338, 340]), defensa de objetivo móvil ([276]), monitorización de archivos ([316, 362, 461, 466]), contadores hardware de alto rendimiento ([391]), múltiples indicadores de compromiso ([315, 330, 450, 453, 461]).

Dentro de las propuestas de detección basadas en *honeypots*, citadas en el párrafo anterior, se incluyen las basadas en *honeyfiles* donde se pretenden crean 'archivos' utilizados como trampas para atrapar al *ransomware*, como es el caso de la propuesta descrita en las secciones siguientes.

5.2. Herramienta anti-*ransomware* basada en *honeyfiles* para Windows

En [215] podemos encontrar que el ciclo de vida del *ransomware* para el marco *.NET* se puede estructurar en ocho pasos: *(1)* infiltración en el sistema a atacar, *(2)* obtener privilegios de ejecución, *(3)* crear unas claves criptográficas únicas, *(4)* enumerar los archivos a cifrar, *(5)* cifrar dichos archivos con las claves, *(6)* eliminar el acceso a los archivos originales, *(7)*

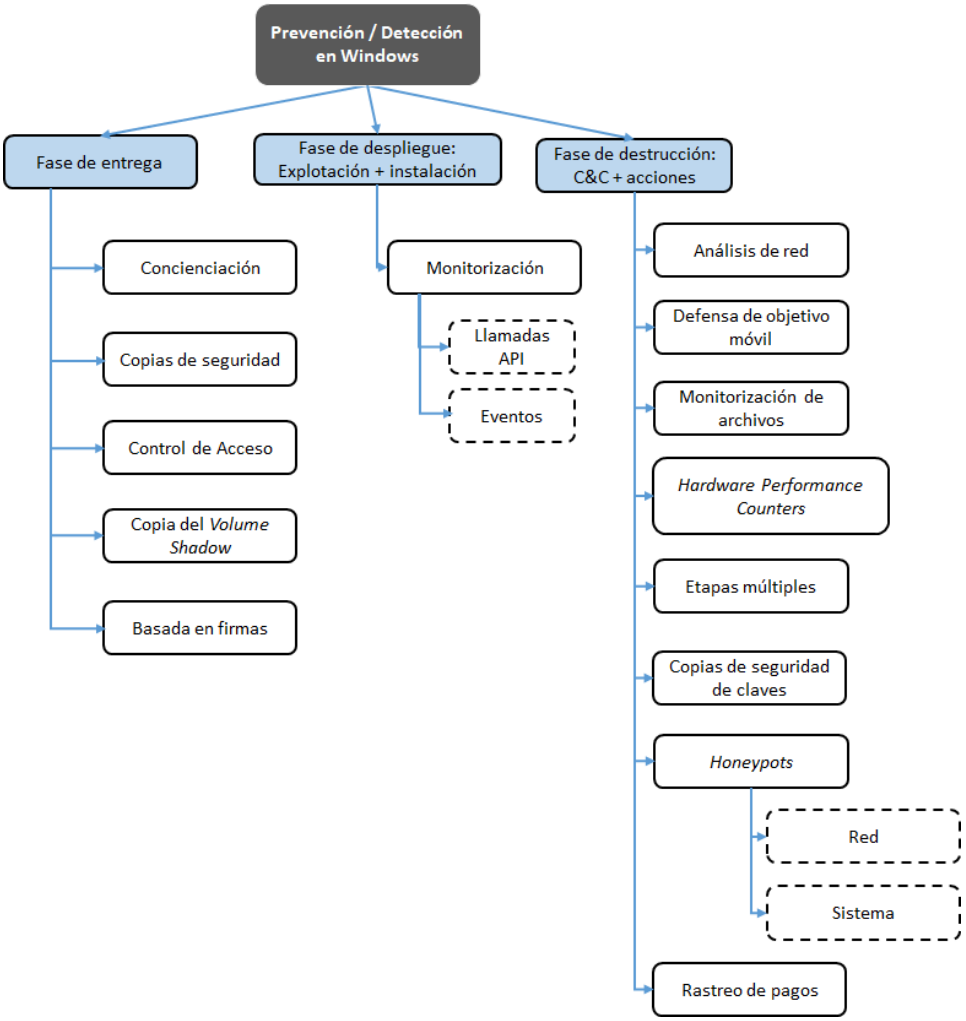


Figura 5.1: Mecanismos de detección de *ransomware* en Windows según las fases de un ataque.

proteger las claves hasta que se realice el pago, y *(8)* mantener un canal para la extorsión. En base a ellas, cualquier mecanismo destinado a prevenir el éxito del *ransomware* debería estar basado en detener su ejecución en, al menos, una de las etapas que acabamos de citar.

Como ya es conocido, nuestro enfoque está pensado para operar entre las etapas 4 y 5. Dado que para cifrar un archivo primero es necesario leer su contenido, nuestra propuesta descansa en el uso de archivos trampa, o *honeyfiles*, cuyo objetivo es mantener al proceso lector de forma indefinida en la fase de lectura previa al cifrado. Esto es, la llamada a *read()* nunca retornará el control al llamador (es decir, al *ransomware*), de forma que se frustrará el último objetivo del *ransomware*: cifrar el archivo.

Aceptado pues que las acciones del cripto-*ransomware* descansan en escanear el sistema de archivos de la máquina infectada y cifrar la información contenida en él, nuestro enfoque de detección tiene como objetivo satisfacer algunas propiedades funcionales y requisitos operacionales (denominados *F1* to *F2*, y *R1* a *R5*) en [471] con respecto a la efectividad, escalabilidad, transparencia, consumo de recursos, etc.

Para alcanzar los objetivos *Fx* y *Rx*, una solución sencilla y elegante es hacer uso del mecanismo de comunicación entre procesos denominado FIFO o *cauces con nombre* [471], como ya ha sido descrito en el capitulo previo.

Un cauce con nombre necesita un proceso escritor en el otro extremo. Este proceso puede controlar la interacción con el FIFO, así que es el lugar idóneo para ubicar el proceso de monitorización para la herramienta de detección que vamos a desplegar. De acuerdo a esto, en las Figuras 4.6 y 4.7 del Capítulo 4 se muestra la arquitectura y el flujo operación para el detector de *ransomware* propuesto. De estas figuras: *(i)* cuando se instala la herramienta, se crea un FIFO bloqueante y un archivo trampa se inserta en todas y cada una de las carpetas del sistema de archivos apuntando al FIFO central; *(ii)* una vez creados todos los archivos trampa y distribuidos por el sistema de archivos, el proceso monitor espera bloqueado en el cauce con nombre; *(iii)* cuando un proceso (por ejemplo, *ransomware*) accede a algún *honeyfile* para leerlo, es redirigido al FIFO y automáticamente es bloqueado por el sistema; *(iv)* en este momento, el sistema operativo desbloquea al proceso de monitorización que lanza un proceso destinado a adoptar las contramedidas correspondientes [497]. En base a esta descripción, el Algoritmo 2 describe la operativa general de R-Locker.

5.2.1. Implementación de R-Locker en Windows

En base a la operación general de R-Locker mencionada, en esta sección describiremos la implementación específica aquí desarrollada para la plataforma Windows, donde existen dos diferencias fundamentales en la gestión de los FIFO en comparación con los sistemas Unix:

Algoritmo 2: Algoritmo relativo a *R-Locker*.

Etrada: nula
Salida: nula

```
1  Archivo honey ← crear FIFO en modo bloqueo
2  Trampa ← lista de archivos con extensiones pdf, jpg, ...
3  inicio
       // Crea y despliega las trampas
4      para cada carpeta en el sistema de archivos hacer
5          Crear enlace simbólico: trampa → FIFO;
       // Las trampas son el extremos de lectura, accedido
          por el ransomware
6      para (;;) hacer
7          Crear instancia del hilo de detección;
8          Conectar el hilo instanciado con el extremo de escritura del
            FIFO;
9          Hilo espera a que otros lo desbloqueen al leer en FIFO;
           // Ahora, los hilos bloqueados están esperando que
              la muestra de ransomware caiga en la trampa
10         El hilo desbloqueado por el SO procede a la fase de
            detección;
```

1) Los archivos y los cauces tienen diferentes espacios de nombre por parte del sistema operativo.

2) Solo se permite que un proceso lea simultáneamente del FIFO.

Por tanto, el Algoritmo 3 muestra con detalle la implementación final en Windows, donde resaltamos que el desarrollo se ha realizado sobre la interfaz de programación de Windows de 32 bits (API Win32) por razones de eficiencia, por lo que el usuario solo tiene que ejecutar la aplicación sin que sea necesario modificar o configurar el *kernel* del sistema operativo.

De acuerdo con el algoritmo mencionado, la función `Principal()` (línea 27) de R-Locker se inicia llamando a la función `PoblarListaBlanca()` (línea 28), cuya implementación aparece en la línea 25, y que esta destinada a generar la lista de aplicaciones legales instaladas en el sistema. A continuación, el procedimiento principal llama a la función `PoblarTrampas()` (línea 4), que es responsable de obtener la estructura de carpetas del sistema de archivos (línea 5). Entonces, la herramienta crea un enlace simbólico al cauce con nombre en cada una de las carpetas (llamada al sistema `CreateSymbolicLink` - línea 8); así se despliega por el sistema de archivos un árbol de rutas directas a la herramienta de detección hacia el FIFO central.

Algoritmo 3: Seudocódigo de *R-Locker para Windows.*

```
   Entrada: nula
   Salida: nula
 1 ruta_trampa                    // Enlace simbólico al honeyfile
 2 lista_extensiones_trampa ← {.pdf .jpg .doc ...}
 3 honeyfile ← \\\\.\\pipe\\nombre_trampa    // Nombre del FIFO
 4 Función PoblarTrampas(ruta_trampa):
 5     lista_carpetas ← Generar la lista de carpetas
 6     para cada carpeta en lista_carpetas hacer
 7         si no existe ruta_trampa entonces
 8             CreateSymbolicLink(..\ruta_trampa.extension_trampa,
                honeyfile)
 9 Función InstanciarHilo(rutaFIFO):
10     hP=CreatePipe(honeyfile,.,PIPE_WAIT,...)       // Crear el
        FIFO
11     fConnec = ConnectNamedPipe(hP)  // Conectar con el FIFO
12     CreateThread(..., Deteccion,..)          // Hilo de detección
13 Procedimiento Detección(hPipe):
14     GetNamePipeClientProcessId(hP, ppid)
15     hProcess = OpenProcess(..., *ppid)
16     program_name ← QueryFullProcessImageName(hProcess, ..)
17     si (nombre_programa está en ListaBlanca) entonces
18         continuar ...
19     si (nombre_programa está en ListaNegra) entonces
20         Terminar el proceso (hProcess)
21     en otro caso
22         si (messageBox == Terminate) entonces
23             Añadir nombre_programa a ListaNegra
24             TerminateProcess(hProcess)
25 Función PoblarListaBlanca():
26     ListaBlanca ← Generar lista de aplicaciones instaladas
27 Función Principal:
28     PoblarListaBlanca()
29     PoblarTrampas(nombre_pipe)
30     num_hilo ←
        parámetro ∗ obtener número de procesadores del SO
31     para (;;) hacer
32         hilos_terminados = 0
33         para (i=0; i++; i < número_hilos) hacer
34             VectorHilos[i] = CreateThread(...,InstanciarHilo,Pipe,...)
35         mientras hilos_terminados < numero_hilos - 1 hacer
36             WaitMultipleObjects(...,VectorHilos,...)
37         si (periodo) entonces
38             CarpetaMonitor = CreateThread(...,PoblarTrampas,...)
```

Actualmente, muchas familias de *ransomware* (por ejemplo, Cerber) utilizan procesos multihebrados para optimizar el cifrado de los archivos. Para tratar con estas variantes de *malware*, R-Locker crea a continuación una bolsa de hilos (líneas 32-34). Para realizar operaciones multihebradas en Windows, es necesario crear múltiples instancias del mismo FIFO y desplegar un hilo monitor por cada instancia (línea 34). El número de hilos se determina de manera experimental, para lo cual el algoritmo declara `parámetro` para ajustarlo (línea 30). Una buena estimación de `parámetro` es 2, dado que el máximo rendimiento se obtiene con 2 hilos por núcleo (*core*) de procesamiento. De esta forma, desde el punto de vista del *ransomware*, mientras que un hilo está esperando la lectura de un archivo, el otro puede usar la CPU para cifrar datos. Considerar un número mayor de hilos en una muestra de *ransomware* no mejorará el rendimiento de este, ya que competirán unos hilos con otros por el uso de la CPU.

Cada uno de los hilos creados llaman a la función `InstanciarHilo()` (líneas 9-12) que está diseñada para desplegar las instancias respectivas del cauce con nombre. En este punto, hay que resaltar el hecho de que, como mencionamos previamente, los archivos del sistema y los cauces en Windows tienen espacios de nombres separados. Para solventar este inconveniente, necesitamos usar las reglas y convenciones de designación de Windows para nombrar dispositivos como archivos regulares o directorios (línea 3). La Figura 5.2 muestra el esquema general de R-Locker visualizando los espacios de nombres involucrados: *(i)* el espacio de nombres de procesos (recuadro amarillo más externo), *(ii)* espacio de nombres de archivos (recuadro azul intermedio); y *(iii)* el espacio de nombres de dispositivos (recuadro verde más interno).

Para nuestro propósito principal, otra propiedad interesante de los FIFO es que la sincronización de las comunicaciones es automáticamente gestionada por el sistema operativo. De esta forma, cuando un proceso lee un FIFO vacío (es decir, nadie ha escrito aún en él), el sistema operativo bloquea al proceso lector. Para ello, es necesario instruir al sistema operativo que la gestión del FIFO es síncrona (indicador `PIPE_WAIT` en la llamada `CreatePipe()` en línea 10). Después de esto, el hilo se conecta de forma síncrona con el cauce (línea 11), esperando que el *ransomware* caiga en la trampa. De esta forma, el proceso monitor de R-Locker actuará como escritor en el cauce, mientras que el *ransomware* actuará como lector sobre el canal de comunicación.

Cuando un proceso lector accede al *honeyfile*, se ejecutará la función `Detección()` (líneas 13-24) por el hilo correspondiente. Esta función es responsable de determinar la identidad del proceso que intenta leer la trampa y después poner en marcha la contramedida adecuada para dar respuesta al incidente de acuerdo a la naturaleza real del mismo, legítima o maliciosa.

La Figura 5.3 representa el diagrama de flujo específico de R-Locker para Windows (*caja gris*). Como discutíamos en los párrafos previos, está com-

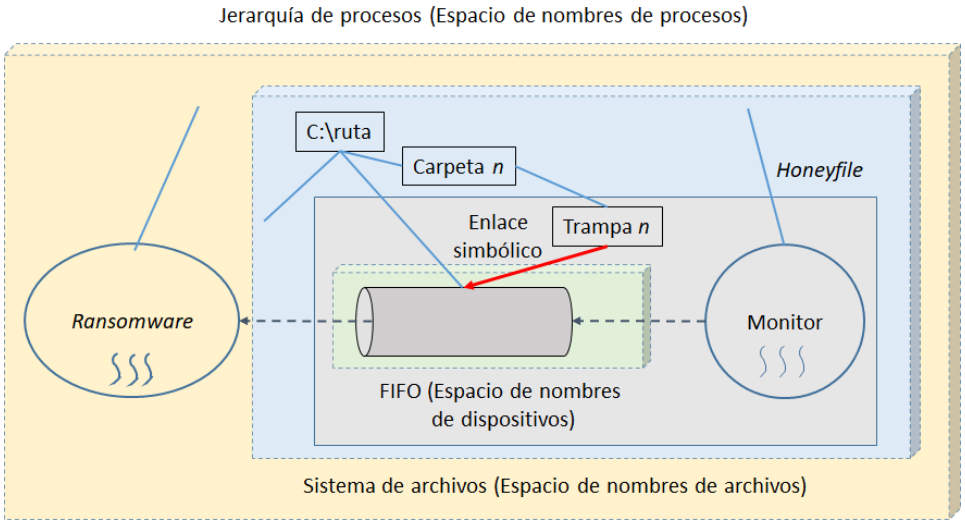


Figura 5.2: Espacios de nombres involucrados en la herramienta *R-Locker*.

puesto de un proceso de monitorización conectado al cauce con nombre. Este monitor consta de un proceso escritor en el cauce, cuya acción tendrá éxito cuando un proceso lector interactúe con el otro extremo del cauce. Dado que nuestro interés es proteger el sistema de archivos completo, replicaremos las trampas en múltiples directorios. Si bien es posible replicar el propio cauce, la gestión asociada del proceso maestro podría ser más compleja. Por ello, como se ha indicado, replicamos las trampas mediante el uso de enlaces simbólicos al mismo FIFO central. De esta forma, una muestra de *ransomware* que intenta acceder al enlace simbólico en cualquier parte del sistema de archivos se redirigirá al único FIFO. Por otra parte, como la acción de lectura sobre el FIFO es bloqueante en nuestra implementación sobre Windows, R-Locker creará varios hilos para gestionar *ransomware* multihebrado.

Comentar de nuevo el bajo consumo de recursos puestos en juego por R-Locker. Primero, en cuanto al espacio en disco, indicar que por una parte los enlaces simbólicos en Windows no consumen espacio de disco; de otro lado, los cauces con nombre solo requieren el espacio para almacenar los metadatos en el sistema de archivos, ya que no son realmente archivos sino dispositivos. Segundo, en cuanto al consumo de memoria principal, el sistema solo necesita espacio para asignar $2 * n$ hilos, donde n es igual al número de procesadores del sistema. Tercero, el uso de CPU es nulo cuando el monitor está esperando, dado que el sistema operativo lo pone en estado bloqueado y el trabajo real reside en la lectura futura de un proceso que lea el FIFO.

Embebido en el flujo operacional previo, en la siguiente sección presentaremos dos mejoras concretas con respecto a la herramienta original para Unix introducida en el Capítulo 4. Por una parte, la gestión dinámica de

la estructura de carpetas y el correspondiente despliegue de *honeyfiles*, que suministrará una protección mejor al sistema de archivos objetivo a lo largo del tiempo. Por otro lado, la adopción de las acciones correctivas consecuentes a la detección de un evento *ransomware*, donde se ha implementado un procedimiento (semi)automático que agiliza la respuesta del sistema.

5.2.2. Gestión dinámica de *honeyfiles*

Más allá del interesante comportamiento general de los *honeyfiles* desplegados por el sistema de archivos para su protección, debemos resaltar algunos aspectos prácticos específicos sobre los mismos. Uno hace referencia a la ubicación de estos para una protección efectiva, esto es, cómo seleccionar y manejar las carpetas en las que ubicar las trampas. En la Sección 2.4.8 se analizó la interacción del algunos *ransomware* con los archivos y el sistema de archivos, y se resaltaron dos cuestiones importantes. Primero, el orden seguido por las muestras de *ransomware* para la selección de las carpetas donde buscar archivos. Segundo, el orden de la selección de archivos dentro de una carpeta dependiendo del tipo de sistema de archivos [222].

En el primer caso, como comentamos en la sección indicada, nos encontramos con que no existe un orden único para la selección de carpetas: algunas muestras, como GandCrab o TeslaCrypt, hacen primero una selección en profundidad, luego alfabéticamente; otras, como Osiris, realizan una selección aleatoria. Estos comportamientos tan diferentes hacen concluir la conveniencia de desplegar las trampas por todas las carpetas de cara a alcanzar una protección completa. Por esto, necesitamos crear enlaces simbólicos al FIFO central en todas y cada una de las carpetas del sistema de archivos. Este es el propósito de la función `PoblarTrampas()` en el seudocódigo del Algoritmo 3, que crea tales enlaces en la fase de inicialización de la herramienta. Además, esta función explora dinámicamente el árbol de directorios del sistema objetivo para incluir enlaces simbólicos en las nuevas carpetas que vayan creándose conforme evoluciona el sistema.

Como el tamaño asociado a un enlace simbólico en Windows de 0 *bytes*, no se consume espacio de disco en la creación de todos los enlaces necesarios. Además, debido al hecho de que todos los enlaces simbólicos apuntan al FIFO central para el proceso lector, la única acción pro-activa que consume recursos adicionales del sistema es el hilo periódico que comprueba la existencia de nuevas carpetas en el sistema de archivos. El periodo de ejecución de cada hilo es actualmente ajustado para minimizar la exposición de nuevos directorios al *ransomware* a un coste computacional razonable (como veremos en breve en la Sección 5.3).

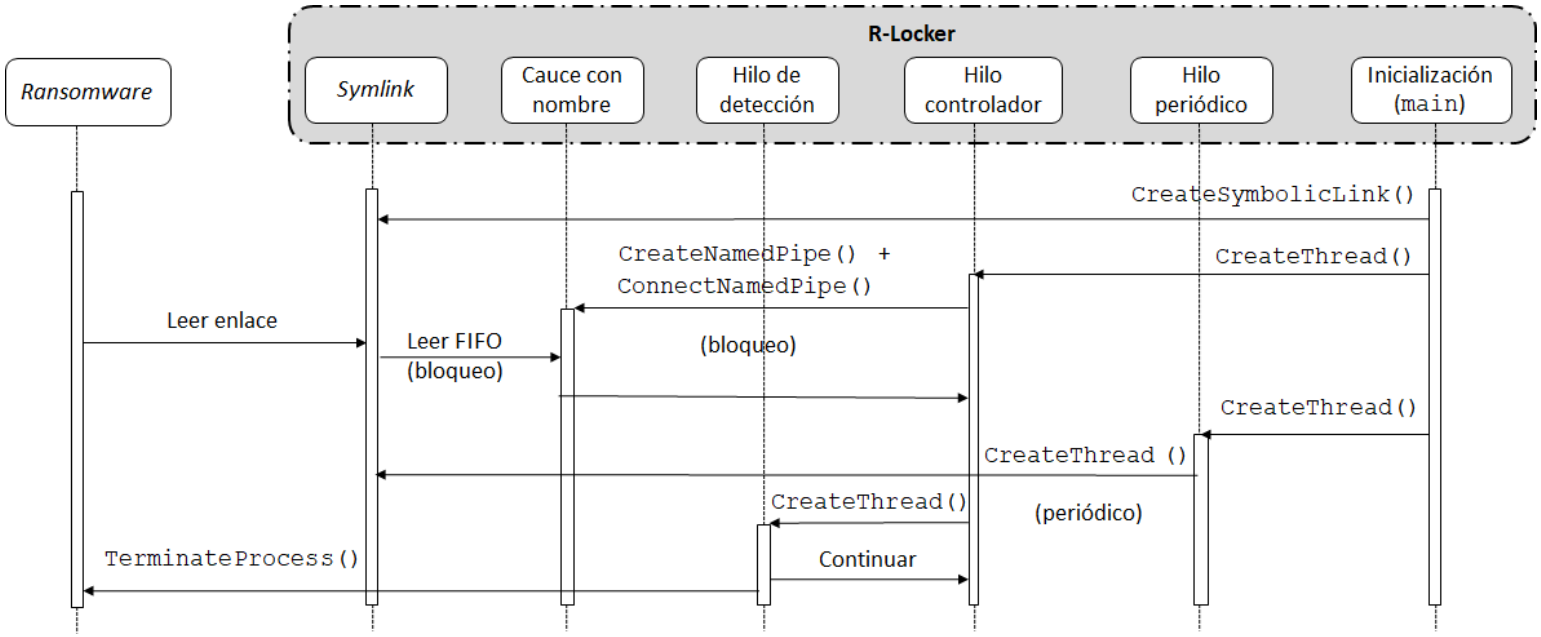


Figura 5.3: Diagrama de flujo de *R-Locker*.

Respecto al proceso de selección de archivos dentro de una carpeta, las muestras de *ransomware* seleccionan los archivos de acuerdo a diferentes criterios. En algunos casos, como GandCrab o TeslaCrypt, las funciones de la API Win32 comúnmente usadas para localizar archivos, como `FindFirstFile()` y `FindNextFile()` [494], devuelven un resultado que depende del tipo de sistema de archivos. Por ejemplo, en el caso de NTFS, el tipo de sistemas de archivos nativo de las versiones actuales de Windows, se devuelven las entradas de una carpeta ordenadas alfabéticamente. En este caso, para detener la acción del *ransomware* al inicio del proceso de lectura de archivos, los nombres de los archivos simbólicos creados se eligen para que sean el primer valor devuelto alfabéticamente (por ejemplo, "!..!"). En otros casos, por ejemplo Osiris, los archivos son primero priorizados por extensión y luego seleccionados alfabéticamente. Para abordar esta situación podemos crear múltiples enlaces con nombres "!..!" y extensiones como '.doc', '.pdf', '.jpg', etc. Para reforzar la transparencia desde la perspectiva del usuario, los enlaces simbólicos pueden marcarse con el atributo de ocultos de forma que sean invisibles a las operaciones normales del usuario.

5.2.3. Respuesta después de la detección

Cuando un proceso lector accede al *honeyfile*, R-Locker en su versión original (Capítulo 4) lanzaba una notificación para que el usuario adoptara manualmente las potenciales acciones correctivas subsecuentes. El medio más factible para implementar este proceso de notificación es hacer uso del procedimiento `NotifyUser()`. Esta función es responsable de determinar la identidad del proceso que está intentando leer de la trampa y, en consecuencia, notificar al usuario, quién decidirá la acción concreta o contramedida a tomar, si es necesario. El mecanismo de reacción usual adoptado hasta el momento es parar y/o finalizar al programa supuestamente malicioso.

Como mejora del procedimiento de reacción manual, la actual versión de R-Locker para Windows realiza un proceso semi-automático basado en la gestión dinámica de listas blancas y negras como sigue:

1. La herramienta crea en tiempo de instalación una lista blanca con los programas legítimos a los que les estará permitido acceder al sistema de archivos. Para esto, R-Locker explora las carpetas que contienen los programas del sistema, ubicadas en *C:\Archivos de programa* y *C:\Archivos de programa (x86)*, e incorpora los ejecutables correspondientes a la lista blanca.

2. En base a esto, cuando un programa sospechoso accede al *honeyfile* la herramienta primero comprueba la lista blanca: *(i)* si el proceso está en la lista, R-Locker no actúa de ninguna forma contra dicho proceso; *(ii)* en otro caso, se notificará al usuario para que decida cómo proceder.

3. Tras la notificación, si el usuario decide no actuar contra el incidente dado que considera que el programa es legítimo, el sistema incorporará automáticamente dicho programa a la lista blanca.

4. En caso contrario, si el usuario decide parar/terminar el proceso, R-Locker lo añadirá a la lista negra. De esta forma, la próxima vez que el mismo programa sea detectado como *ransomware*, será automáticamente terminado por la herramienta sin requerir la acción del usuario.

5.3. Evaluación experimental

Después de la descripción de R-Locker para Windows, esta sección está dedicada a la evaluación experimental de la herramienta confrontándola con muestras de *ransomware* real en una plataforma Windows.

5.3.1. Entorno experimental

Las pruebas con muestras de *ransomware* real requieren un entorno seguro y reusable. Seguro para evitar afectar realmente a los servicios y sistemas desplegados, y reusable para restaurar el sistema experimental a su estado original de una forma eficiente y dinámica en caso de que la prueba falle y el sistema se vea afectado.

Una forma usual de probar muestras de *malware* es ejecutarlas y analizarlas en una entorno virtualizado [498]. En este caso, debemos tener en cuenta algunas consideraciones:

- Primero, la máquina virtual (VM) debe aislarse del sistema anfitrión, mientras mantiene el acceso a Internet para permitir potenciales comunicaciones con un C&C.
- Segundo, la VM debe desplegarse en una red aislada para evitar que afecte a otros sistemas.
- Tercero, la VM debe suministrar un entorno lo más realista posible para evadir potenciales técnicas anti-*sandbox* empleadas por el *malware*.
- Finalmente, el entorno de pruebas es un sistema propietario que debe estar licenciado.

Para satisfacer los requisitos indicados anteriormente, el entorno experimental desplegado aquí para la detección de *ransomware* está basado en Malboxes [499], en conjunción con Vagrant [500]. Malboxes es un proyecto de fuentes abiertas destinado a construir máquinas virtuales de Windows para el análisis de *malware*. Su misión es crear una caja Vagrant mediante la descarga y creación de una VM basada en VirtualBox [501] para su provisión y configuración. La caja Vagrant es el estado inicial de las pruebas, y

permite la gestión de la máquina virtual en un flujo simple más fácil para la automatización del proceso. El uso conjunto de las dos herramientas mencionadas permite la creación, la destrucción o la parada de las VM mediante una orden sencilla.

El archivo de configuración de Vagrant ayuda a configurar los recursos prestados a la VM, la configuración de red, los archivos que deben copiarse desde el anfitrión a la VM y los *scripts* que se ejecutarán en el despliegue de la VM. Necesitamos todas estas características para automatizar el proceso general. En nuestro entorno experimental creamos una VM con 4 CPU (este número de núcleos permitirá potencialmente el paralelismo, no solo concurrencia, entre la ejecución de la muestra de *ransomware* y R-Locker) y 4 GB de RAM, para deshabilitar la carpeta compartida con Vagrant, para copiar el código de R-Locker en la VM, y para ejecutar un *script* de PowerShell para descargar las muestras de cripto-*ransomware* que se vayan a probar.

Debemos mencionar que no se ha automatizado completamente el despliegue del entorno experimentar debido a las razones que se indican a continuación. Primero, R-Locker debe ejecutarse manualmente dado que Vagrant no tiene una opción para hacerlo automáticamente. Segundo, en caso de que el programa analizado sea identificado como malicioso por el sistema, Windows Defender [502] debe deshabilitarse manualmente. Tercero, algunas muestras necesitan una configuración previa o resolución de dependencias para que puedan ejecutarse correctamente.

5.3.2. Muestras de evaluación y resultados

Una vez ajustado el entorno experimental, es momento de ejecutar las pruebas con muestras de cripto-*ransomware* reales. Para ello, hay disponibles algunos repositorios públicos de muestras de *malware*. Un ejemplo es el repositorio de [463], pero en este caso es solo de archivos *pcap* correspondientes al tráfico de red generado por *malware*, en lugar de los ejecutable en sí mismos. Por tanto, hemos obtenido las muestras experimentales del repositorio *TheZoo* [503]. La razón se debe a que sus muestras de *malware* están clasificadas por familias, lo cual es útil para el propósito de las pruebas y esto no ocurre con otros *dataset* bien conocidos como VirusShare [504] o VirusTotal [505].

La Tabla 5.1 recoge las muestras probadas en nuestra experimentación. Estas hacen referencia a conocidos casos de *ransomware* a nivel mundial como, por ejemplo, Cerber, Jigsaw y WannaCry. Además, como prueba de concepto hemos probado un *ransomware* genérico [506], cuya funcionalidad básica es similar al famoso CryptoLocker: cifra con una clave pública RSA-4096 (RSA-OAEP-4096 + SHA256) cualquier carga y sube los archivos a un servidor remoto. Si bien se han descargado otras muestras, estas no se ha ejecutado o no han funcionado adecuadamente en el entorno desplegado (aspecto que suele ser frecuente como indicábamos en la Sección 3.5.1).

Tabla 5.1: Muestras de cripto-*ransomware* probadas.

Nombre	Fuente	MD5	Detectado
Cerber	TheZoo	8b6bc16fd137c09a08b02bbe1bb7d670	No
Genérico	GitHub	Compilado *in situ*	Si
Jigsaw	TheZoo	3ad6374a3558149d09d74e6af72344e3	Si
WannaCry	TheZoo	84c82835a5d21bbcf75a61706d8ab549	Si

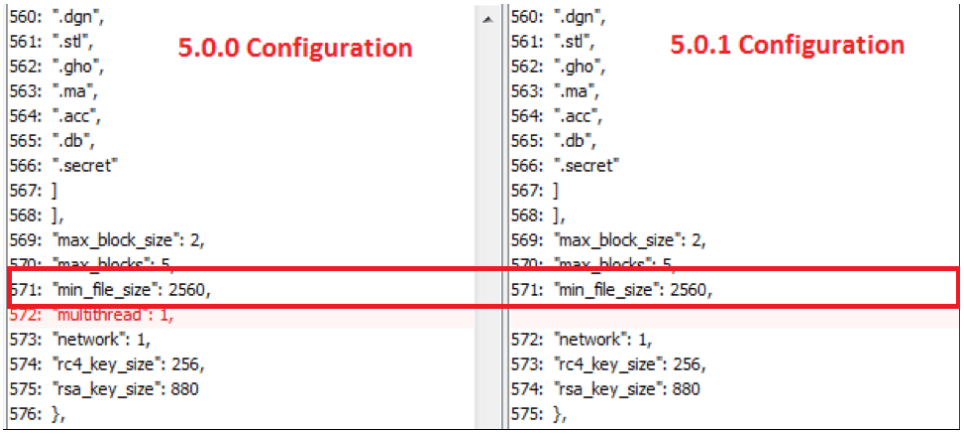


Figura 5.4: Parámetro de selección del tamaño mínimo de archivo víctima en la configuración de Cerber [507].

Como se indica en la Tabla 5.1, las muestras correspondientes a Wannacry, Jigsaw y Genérico han sido detectadas y bloqueadas por R-Locker, como se esperaba. Además, la afectación en todos los casos del sistema es nula, esto es, ningún archivo ha sido cifrado por las muestras gracias al despliegue de *honeyfiles* por todo el sistema de archivos. Por tanto, la herramienta de detección ha tenido éxito al derrotar al *ransomware*, lo que permite concluir que la solución FIFO es efectiva y las trampas desplegadas cubren correctamente las carpetas del sistema afectado por las muestras de *ransomware*.

Sin embargo, no todas las muestras de *ransomware* han sido detectadas por R-Locker. En particular, la muestra de Cerber no lo ha sido, hecho que se justifica como sigue. La ingeniería inversa de Cerber [507] muestra que los autores del *malware* han definido un parámetro de tamaño mínimo de archivo para que un archivo sea candidato para ser cifrado (Figura 5.4). Dado que los enlaces simbólicos en Windows tienen tamaño 0, los *honeyfiles* (enlaces simbólicos) desplegados por R-Locker no son seleccionados para cifrado por el *ransomware*, lo que impide la detección de su ejecución en el sistema.

```
[10:7:5] Log Initialized
[10:7:13] C:\Program Files (x86)\Microsoft Visual
Studio\2019\Community\Common7\IDE\CommonExtensions\Microsoft\TeamFoundation\ Team
Explorer\Git\mingw32\bin\git.exe was added to white list
[10:7:13] C:\Program Files (x86)\Microsoft Visual
Studio\2019\Community\Common7\IDE\CommonExtensions\Microsoft\TeamFoundation\ Team
Explorer\Git\mingw32\bin\git.exe was added to white list
[10:7:19] C:\Windows\explorer.exe was added to White list
. . .
[10:7:28] C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe was terminated and added to black list
[10:7:31] Program in the black list "C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe" tried to Access the
trap
[10:7:40] C:\Windows\SysWOW64\icacls.exe was terminated and added to black list
[10:7:44]
C:\Users\User\Downloads\Ransomware.Wannacry\ed01ebfbc9eb5bbea545af4d01bf5f1071661840480439c6e5babe8e08
0e41aa.exe was terminated and added to black list
```

Figura 5.5: Contenido parcial del archivo de registro generado por R-Locker.

Las principales operaciones de R-Locker en el tiempo se registran en un archivo de *log*. La Figura 5.5 muestra un fragmento de un ejemplo de este registro. En las primeras líneas, vemos cómo se construye la lista blanca a partir de los programas legítimos (por ejemplo, *explorer.exe*, *git.exe*). Cuando un programa es detectado como muestra sospechosa de ser *ransomware* y no está incluida en la lista blanca, R-Locker genera una ventana de notificación como la mostrada en la Figura 5.6. Esta notificación informa sobre el identificador de proceso (PID) del supuesto *ransomware* y la ruta desde donde se lanzó, preguntando al usuario sobre la posibilidad de parar el proceso potencialmente malicioso.

Desde la primera elipsis en el archivo de registro de la Figura 5.5, podemos ver cómo la gestiona la lista negra. Primero, la ejecución de PowerShell generará una notificación, de forma que el programa es etiquetado como malicioso por el usuario y, así, se incluirá automáticamente en la lista negra por R-Locker. Después de esto, cualquier nueva aparición de este programa será automáticamente bloqueada por R-Locker. De la misma manera, el archivo de registro también recoge la detección de una muestra de WannaCry así como una versión infectada de este, *iCacl.exe*.

En resumen, de todo lo anterior podemos concluir que el funcionamiento general de R-Locker es tan eficiente como esperábamos. El consumo de memoria RAM es pequeño dado que en Windows cada hilo puede consumir aproximadamente 1 MB [508], por lo que en el caso de una máquina con 8 núcleos el consumo sería de unos 8 MB. Una vez instalada la herramienta el consumo de tiempo de CPU es prácticamente 0, ya que la hebra de detección esta bloqueada, y solo se ejecutará puntualmente la hebra periódica para comprobar el estado de las trampas durante una fracción despreciable de tiempo.

Con el propósito de prueba y posteriores desarrollos, R-Locker para Windows esta disponible públicamente para la comunidad en GitHub [509].

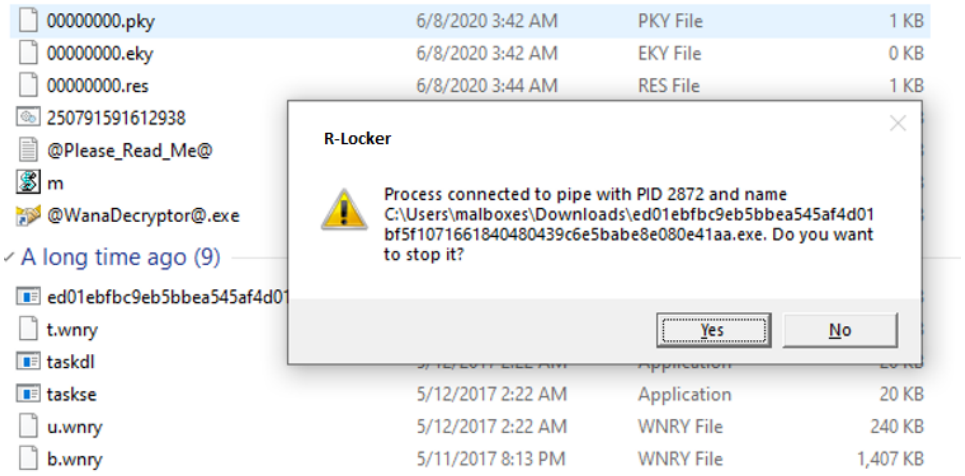


Figura 5.6: Notificación de *R-Locker* para el usuario de un incidente sospechoso de *ransomware*.

5.4. Conclusiones y trabajo futuro

Este capítulo ha descrito la nueva versión de R-Locker para plataformas Windows, una solución anti-*ransomware* propuesta por los autores y descrita en el capítulo anterior para Unix. Al igual que la versión original, está basada en el despliegue de *honeyfiles* por el sistema de archivos objetivo para atrapar procesos potencialmente maliciosos que acceden a la información del disco. Sin embargo, son tres los aspectos principales que mejoran la herramienta original. Primero, la implementación en Windows necesita solventar dos diferencias importantes en el manejo de los FIFO respecto a las plataformas Unix: *(i)* los espacios de nombres separados para gestionar los FIFO respecto de Unix, donde los diferentes objetos residen en el mismo espacio de nombres; y *(ii)* solo se permite a un único proceso acceder al FIFO de forma simultánea. Segundo, como apuntábamos en el capítulo anterior, se ha desarrollado un nuevo procedimiento para el despliegue y mantenimiento de los *honeyfiles* por el sistema de archivos de cara a una mejor protección a lo largo del tiempo. Tercero, se introduce el uso de listas blancas/negras para que de una forma (semi)automática se determine y actúe contra procesos legítimos frente a *ransomware* real que accede al sistema de archivos, reduciendo así la intervención del usuario final.

La publicación de la herramienta en el repositorio de GitHub [509], después de evaluarla experimentalmente tiene varios beneficios: *(i)* su funcionamiento es efectivo, *(ii)* la instalación no requiere privilegios, *(iii)* simple con bajo consumo de recursos, y *(iv)* no interfiere con el resto del sistema.

Como trabajo futuro, avanzaremos en dos direcciones principales. Por

una parte, derrotar al *ransomware* independientemente del tamaño de los *honeyfiles* desplegados (para atrapar a muestras como Cerber, Sección 5.3). Por otro lado, pensamos que es de interés general extender la propuesta a plataformas móviles, como veremos en el capítulo próximo, dado su amplio uso actualmente y, consiguientemente, el alto riesgo que supone esta amenaza para este tipo de dispositivos.

Capítulo 6

Detección de *ransomware* en Android

El uso de dispositivos móviles como *smartphones* o *tablets* está muy extendido entre los usuarios a nivel mundial. De acuerdo con GSMA [510], en 2021 el número de usuarios de móviles se encuentra alrededor de 5,3 mil millones, un 53 % de la población mundial, con la previsión de que crezca el 1,8 % para 2025.

Dado el enorme uso de dispositivos móviles, el desarrollo de *malware* asociado también experimenta un incremento notable. De acuerdo con [511], en 2021 se detectaron más de 2 millones de nuevas muestras de *malware*, afectando a más de 10 millones de dispositivos. Otros datos que muestran el interés de los cibercriminales por este tipo de dispositivos es el descubrimiento de un 30 % de vulnerabilidades de día cero específicas de móviles. Ha habido un aumento del 466 % en las vulnerabilidades de día cero explotadas utilizadas en ataques activos contra terminales móviles, y existe un 75 % de sitios de *phishing* especialmente dedicados a móviles. En lo que respecta al *ransomware*, el informe de Kaspersky [512] refleja un incremento para móviles del 0,5 % de ataques de dicho *malware* en el tercer trimestre respecto del segundo en 2021. Conforme aumenta la adopción de este tipo de dispositivos y servicios móviles, los incidentes de seguridad también se incrementan, no solo mediante diferentes tipos de *malware* (*adware*, troyanos, *ransomware*, *dropper*) sino con otro tipo de riesgos y ataques (exfiltración de datos, robo del terminal, *flashing* malicioso, etc.) [513].

Desde hace años, la plataforma móvil dominante es Android (69,74 %), seguida de iOS (25,49 %) y un porcentaje poco significativo de otras plataformas (0,77 % - Symbian, BlackBerry, Microsoft, etc.) [514].

Por este motivo, nos centraremos en este capítulo en los mecanismos de defensa frente al *ransomware* en el ecosistema Android. Como discutiremos

en breve, la solución que propusimos en los capítulos anteriores, para Unix y Windows, es de aplicación muy limitada en Android, por lo que en lo que resta nos centraremos en abordar la detección desde el punto de vista de la monitorización. Abordaremos dos enfoques, una monitorización activa y otra reactiva, y analizaremos los pros y contras de cada una de ellas.

6.1. *Ransomware* en Android

Las primeras familias de *ransomware* para Android aparecen en 2013 y pertenecen a la clase de *ransomware* de bloqueo. Estas aplicaciones maliciosas tomaban la forma de falsos antivirus, que notificaban al usuario de una infección para convencerlo de que pagase por limpiar el dispositivo, como es el caso de FakeDefender o FakeAV [515, 516].

En 2014 encontramos otras familias como: ScareOakage, que infecta a los usuarios haciéndose pasar por una app de Adobe Flash y una aplicación antivirus; Android.Lcoker.38.origin, que bloquea dos veces el dispositivo, se distribuye mediante ingeniería social, una vez instalada solicita permisos de administrativos y envía un mensaje a un servidor remoto; Worm.Koler, que se distribuye mediante un mensaje de texto, mantiene bloqueado el dispositivo hasta el pago del rescate y cuando infecta un dispositivo envía mensajes SMS a todos los contactos de la víctima con un texto indicando que se ha creado un perfil con su nombre y que se han subido sus fotos, seguido de un enlace acortado [517].

También en 2014 aparece la primera muestra de cripto-*ransomware* denominado SimpleLocker. Inicialmente distribuido en Rusia, mostraba la nota de extorsión y lanzaba un hilo de fondo para cifrar archivos con extensiones comunes (jpg, pdf, txt, etc.) [518]. Dicho *ransomware* utilizaba un servidor de C&C en la red TOR y, además, recolectaba información del dispositivo como el IMEI, el modelo de móvil y el fabricante. Este mismo año aparecen Locker, Svpeng, Small, Jisut [519], Pletor y LockDorid.

En 2015 tenemos nuevas familias como Fusob, LockScreen, Xbot y LockerPin. Xbot se distribuye como una falsa app bancaria y roba credenciales bancarias imitando el pago de Google Play, que envía a un servidor C&C. En 2017 tenemos Charger, WannaLocker y DoubleLocker. En el caso de WannaLocker, que se distribuye por un foro de juegos chino enmascarado como el juego *King of Glory*, cambia el fondo de pantalla y cifra los archivos almacenados en la memoria externa utilizando un algoritmo AES.

Más próximo en el tiempo, año 2020, encontramos ejemplos como CovidLocker, tanto en versión de bloqueo como de cifrado, y Dubbed Black Rose Lucy (de cifrado), que roba credenciales de cuentas de redes sociales y datos. Los vectores de infección del primero son los SMS o e-mail, y del segundo las redes sociales o la mensajería. En ambos casos se necesitan permisos del servicio de accesibilidad y acceso administrativo.

Un análisis comparativo de diferentes muestras de *ransomware* para Android muestra cómo no solo es un hecho su crecimiento, sino que los ciberdelincuentes están constantemente investigando nuevas formas de ataque para tener un rédito económico. En plataformas Android un medio común de infección es la visita a sitios web inseguros, normalmente al pinchar un enlace malicioso [520].

Antes de pasar a ver el comportamiento de *ransomware* en Android y describir soluciones para su detección, es necesario ver algunos aspectos del modelo de seguridad de Android.

6.1.1. Modelo de seguridad en Android

Esquemáticamente, Android tiene una arquitectura de cuatro capas, a saber [521]:

- Capa 1 - *Kernel* de Linux modificado para incluir algunos controladores hardware como cámaras, *bluetooth*, USB, etc. Desde el punto de vista de seguridad, el *kernel* complementa el control de acceso discrecional (DAC) de Linux con el control de acceso obligatorio (MAC) suministrado por SELinux [522].

- Capa 2 - Bibliotecas y *runtime* de Android, relativas a bibliotecas escritas en C/C++ que gestionan las ventanas y medios, la base de datos SQLite, etc. Esta capa contiene ART (Android *runtime*) para la máquina virtual Dalvik (DVM), que es responsable de convertir los *byte-code* generados por el compilador de Java en archivos ejecutables Dalvik (.dex). Inicialmente, cada aplicación de Android se ejecuta en su DVM separada, pero en versiones superiores a la 5.0 cada app es un proceso con su instancia de ART (al objeto de consumir menos memoria).

- Capa 3 - Marco de aplicaciones, que suministra servicios a las aplicaciones de Android desarrolladas para la capa 4. Entre estos servicios tenemos el gestor de actividades (gestiona el ciclo de vida de una actividad), el gestor de ventanas (gestiona la estructura y visibilidad de las ventanas), el suministrador de contenidos (suministra una interfaz para compartir datos entre aplicaciones), la vista del sistema (para construir la interfaz de usuario de las apps), el gestor de notificaciones (muestra las alertas), el gestor de recursos (acceso a gráficos, cadenas, estructura de archivos y ajustes de color), el gestor de telefonía (gestiona llamadas de voz), gestor de ubicación y el servicio de protocolo de presencia y mensajería extensible (para mensajería instantánea).

- Capa 4 - Aplicaciones, capa más externa que permite a los usuarios y desarrolladores instalar las aplicaciones e interaccionar con el dispo-

sitivo. Contiene tanto las aplicaciones nativas como las desarrolladas por terceros.

El nivel básico de seguridad del sistema operativo Android descansa en su *kernel* de Linux que suministra seguridad mediante el aislamiento de los recursos. El Linux de Android asigna a cada aplicación un UID (identificación de usuario) y utiliza una *sandbox* para aislar las aplicaciones. Podemos describir brevemente el modelo de seguridad como sigue:

- *Sandboxing* de aplicaciones - Se basa en aislar la ejecución de las aplicaciones unas de otras para evitar que una app acceda a los datos de otra sin autorización. Para alcanzar este objetivo se asigna un UID a una aplicación instalada para la ejecución en un proceso aislado donde cada aplicación tiene un directorio aislado con permisos de lectura y escritura. De esta forma, la aplicación está aislada tanto a nivel de proceso como de archivos, lo que previene la interacción entre aplicaciones benignas y malignas (ver Figura 6.1). La inclusión de SELinux en versiones superiores a la 5.0 asegura un control de acceso obligatorio entre el dispositivo y las apps.

- Permisos - El mecanismo de *sandboxing* restringe la compartición de recursos entre aplicaciones, lo que hace que uno de los aspectos importantes de seguridad sea precisamente la asignación de permisos a las aplicaciones ya que estos permiten la compartición de recursos. Los permisos se dividen en cuatro grupos: tres de ellos con un bajo nivel de riesgo, *Normales*, *Signature* y *SignatureOrSystem*; y uno con un nivel de riesgo alto (acceso a recursos sensibles), *Dangerous*. Los permisos suministran seguridad al usuario apoyando su privacidad después de informarle de los permisos otorgados a una aplicación.

- Cifrado - Android soporta el cifrado del sistema de archivos para asegurar la privacidad del usuario. A partir de la versión 5.0, se apoya el cifrado completo de la memoria con una clave única protegida por la clave del dispositivo. A partir de la vesión 7.0 se permite el bloqueo de archivos con diferentes claves que pueden desbloquearse independientemente.

- Play Store - El Mercado Android permite la descarga de aplicaciones que han pasado ciertos controles de seguridad. A partir de la versión 7.0 Google desarrolló Play Protect, que escanea constantemente el dispositivo para detectar y eliminar aplicaciones maliciosas.

El modelo de seguridad es robusto pero tiene algunos problemas que pueden ser aprovechados por los atacantes [523]: aplicaciones maliciosas que superan los controles de Mercado Google o que se instalan desde fuentes

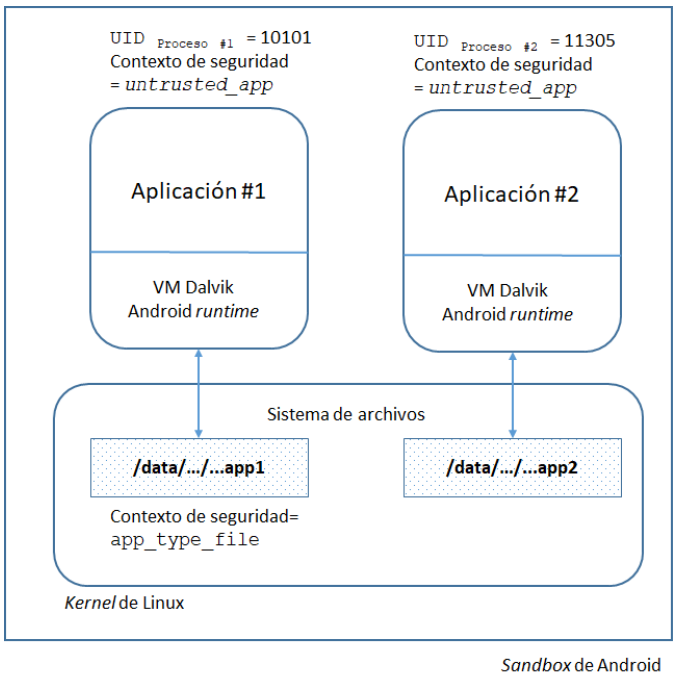


Figura 6.1: *Sandboxing* en Android.

desconocidas, la sobre-solicitud de permisos (vistos como vulnerabilidades) de muchas aplicaciones, que rompe el principio de menor privilegio y cuya aplicación no es bien entendida por los usuarios, que suelen conceder todos los permisos solicitados, hecho que puede ser aprovechado por los atacantes y, por último, el código nativo se puede ejecutar fuera de la máquina Dalvik, con lo que tiene una menor protección de la memoria.

6.1.2. Seguridad y almacenamiento de datos en Android

La memoria de un dispositivo Android está dividida en dos partes, como muestra la Figura 6.2, y que consta de:

- *Memoria interna* - Memoria donde se almacenan los datos privados, es decir, aquellos datos de aplicaciones (directorio */data/*) cuyo acceso se encuentra bajo el control del *kernel* y los datos del sistema (directorio */*). Esta es una parte de la memoria incluida en el dispositivo.
- *Memoria externa* - En esta memoria se guardan los datos públicos, es decir, que pueden ser compartidos por las aplicaciones, como datos personales, fotografías, documentos, etc. y cuyo acceso se encuentra controlado por los permisos concedidos a las aplicaciones. Esta memoria está compuesta por parte de la memoria incluida en el dispositivo y las tarjetas SD externas.

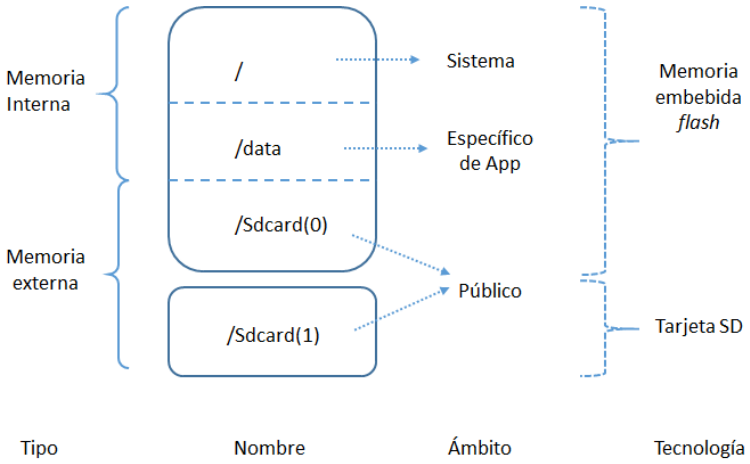


Figura 6.2: Estructura de la memoria en Android.

Respecto a los permisos que protegen el almacenamiento externo, indicar que estos afectan a todos los archivos que este contiene. Es decir, no es posible otorgar permisos a una aplicación para que escriba en un directorio concreto de este almacenamiento, sino que si se le otorga es para toda la memoria externa, tanto si utilizados el permiso de lectura (`READ_EXTERNAL_STORAGE`), el de escritura (`WRITE_EXTERNAL_STORAGE`), o el de gestión de dicho almacenamiento (`MANAGE_EXTERNAL_STORAGE`), que da acceso de lectura y escritura a la tabla *MediaStore.Files* que contiene un índice a todos los archivos del almacenamiento de medios, acceso al directorio raíz y a todos los directorios del almacenamiento interno salvo */Android/data* y *sdcardAndroid*.

6.1.3. Comportamiento del *ransomware* en Android y mecanismos de detección

Durante la instalación de una aplicación maliciosa en Android, uno de los primeros pasos de la misma es obtener los permisos necesarios para realizar las acciones dañinas. Numerosos trabajos se centran en el estudio de los permisos solicitados por las aplicaciones y se utiliza para diferenciar e identificar aplicaciones maliciosas de las que tienen un comportamiento adecuado. La Figura 6.3 [524] compara la frecuencia entre permisos utilizados por aplicaciones maliciosas y benignas. En ella se puede apreciar cómo el *ransomware* tiene preferencia por ciertos permisos frente a las aplicaciones benignas, como por ejemplo `RECIVE_BOOT_COMPLETED`, `WAKE_LOCK` o `GET_TASKS`.

La Tabla 6.1 muestra los diez permisos más usados por el *ransomware* junto a su frecuencia de uso (de una muestra de 2.050 apps maliciosas y otras tantas benignas, obtenidas de los *dataset* RansomProber [525] y AndroZoo [526]) y una breve descripción del propósito de los mismos.

Una vez que hemos dado una rápida visión de los mecanismos de segu-

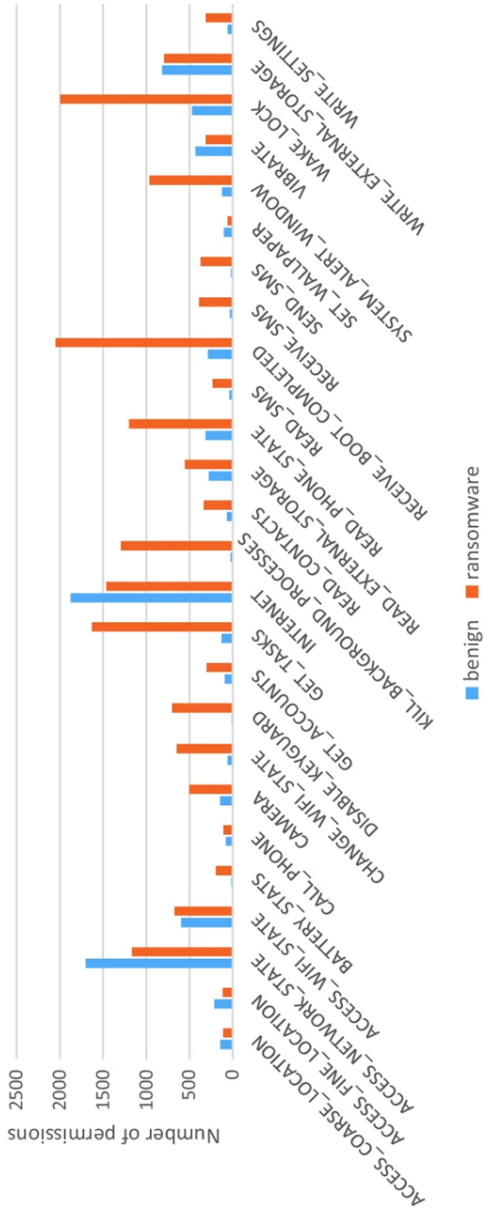


Figura 6.3: Comparativa de los permisos más usados por el *ransomware* en aplicaciones benignas y maliciosas [524].

Tabla 6.1: Los 10 permisos más usados por el *ransomware* en Android.

Permiso	Frecuencia	Propósito
RECEIVE_BOOT_COMPLETED	2049	Comprueba cuando se inicia el dispositivo
WAKE_LOCK	1998	Mantiene encendida la pantalla
GETS_TASKS	1631	Obtiene información de las tareas ejecutándose
INTERNET	1465	Abre *sockets* de red
KILL_BACKGROUND_PROCESSES	1295	Mata a procesos ejecutándose de fondo
READ_PHONE_STATE	1200	Obtiene acceso al estado del teléfono
ACCESS_NETWORK_STATE	1165	Accede a la información sobre la red
SYSTEM_ALERT_WINDOW	965	Crea ventana encima de las existentes
WRITE_EXTERNAL_STORAGE	796	Escribe en memoria externa
DISABLE_KEY_GUARD	701	Deshabilita la protección del teclado

ridad de Android y analizado el comportamiento del *ransomware* en dicho sistema, seguidamente vamos a indicar los problemas encontrados en los intentos de re-implementar R-Locker en Android, que hacen inviable el trasvase de dicho enfoque a pesar de que la capa 1 de Android es un *kernel* de Linux. Tras ello, veremos otras posibilidades de detección basadas en monitorización, en especial un tipo de monitorización reactiva de los eventos del sistema de archivos para muestras que tienen permisos sobre el almacenamiento externo.

6.2. Problemas de la migración de R-Locker a Android

Como hemos visto en los capítulos anteriores, la implementación de R-Locker en un sistema requiere que el sistema operativo soporte la creación de un FIFO y de algún objeto tipo enlace, ya sea simbólico o duro, en el sistema de archivos. El primer requisito es factible pero el segundo no en sistemas Android.

Android permitirá la creación de un objeto tipo FIFO en la carpeta */data/user/0/...* siempre y cuando el almacenamiento interno está formateado con el tipo *ext4*. El tipo de sistema de archivos utilizado podemos verlo

simplemente ejecutando la orden `mount | grep /data"` en un terminal. Lamentablemente no todos los dispositivos tienen formateada la memoria en un sistema de archivos que permita la creación del objeto FIFO, independientemente de que estos se intenten crear mediante interfaz de Java o mediante una aplicación nativa utilizando JNI [527].

Si bien esto sería bastante restrictivo debido al *sandboxing* y al móvil concreto que vayamos a utilizar, podría ser suficiente para extender la solución R-Locker a algunos modelos siempre que fuésemos capaces de crear enlaces simbólicos o duros hacia el FIFO desde el sistema de archivos a monitorizar.

Desgraciadamente, el requisito de creación de enlaces está en una situación aún peor que los FIFO. Hay que indicar que el sistema de archivos con en el que suele formatearse la memoria externa es una variante del sistema FAT, conocida como exFAT (*Extensible File Allocation Table*) [528], que no soporta enlaces simbólicos, ni la creación de objetos FIFO. Por tanto, en este tipo de memoria es inviable construir una solución basada en la propuesta de R-Locker. Si bien sería potencialmente posible formatear la memoria externa con un tipo de sistema de archivo que permita dichos mecanismos y que esté soportado en Android, por ejemplo Ext4, es cierto que no todos los sistemas Android lo permiten. Además, esto rompería la transparencia y sencillez requeridas a R-Locker a la hora de instalar dicha solución, ya que el usuario debería hacer copia de seguridad de su memoria externa para poder formatearla al nuevo tipo.

En este punto hemos de buscar una alternativa a R-Locker. Para ello, podemos adoptar dos enfoques: una monitorización dinámica continua en la que se recolecte información sobre el funcionamiento del sistema y las aplicaciones para un análisis posterior, que denominaremos 'activa'; y otro enfoque en el que la monitorización funciona de forma 'reactiva', es decir, se indica al sistema operativo qué se desea monitorizar y la aplicación esperará, no de forma activa, a que ocurran los eventos monitorizados para gestionarlos. En las secciones siguientes vamos a analizar cada uno de estos enfoques comenzando por la monitorización activa, viendo los problemas que plantea, para pasar posteriormente a la descripción e implementación de una monitorización reactiva.

6.3. Monitorización 'activa' del sistema

Al revisar los trabajos de detección de *ransomware* en Android [521, 529, 530], como vimos en el Capítulo 3 para otras plataformas, el proceso de detección en general pasa por realizar una recolección de parámetros del dispositivo que luego son procesados para determinar si su comportamiento puede considerarse benigno o malicioso.

En esta línea se podría utilizar la aplicación AMon (*Android Monitoring*)

[310, 531] desarrollada por el grupo de investigación NESG, para el proyecto MDSM (*Mobile device Dynamic Security Management*) y disponible en `https://nesg.ugr.es/mdsm/`.

AMon tiene como objetivo recolectar información dinámica sobre numerosos aspectos del funcionamiento del sistema móviles tales como comunicaciones, aplicaciones, estado de seguridad y estado de sus interfaces. Debemos resaltar que AMon no requiere privilegios especiales o acceso de *root* para operar y que recoge un número más amplio de parámetros que los que solemos encontrar en otras herramientas de la literatura. Además, permite que se le añadan nuevos parámetros de monitorización de manera sencilla.

Esta herramienta está escrita en Java y se orienta a la recogida de datos de múltiples fuentes para plataformas Android. Por ello, la mayoría de la funcionalidad esta implementada utilizando la API para desarrolladores [532]. Sin embargo, Android es un sistema operativo que está en continua evolución, implementando nueva funcionalidad y nuevos cambios de una versión a la siguiente. Además, la operativa de AMon descansa en la accesibilidad a la información del sistema, cuya obtención es difícil y diferente dependiendo de la versión de Android, especialmente si consideramos que es una aplicación que no requiere privilegios especiales (*root*). Por tanto, la selección de la versión del SDK (*Software Development Kit*) mínimo no es una cuestión trivial.

En Android Oreo el acceso a los datos de red a través del seudo-sistema de archivos */proc/net* está deshabilitado y el acceso a */proc* es muy limitado. Esto, por un lado, impide el acceso a las estadísticas de CPU en futuras versiones y, por otro, obliga a acceder a los datos de red a través del uso de una VPN (*Virtual Private Network*) local.

AMon se ha desarrollado para la versión Oreo como objetivo, que es compatible con la versión Pie. La versión mínima soportada es Android Marshmallow, pero podría reducirse a costa de perder algo de funcionalidad como de obtención de las direcciones IP de los dispositivos.

Dado el amplio alcance de AMon en cuanto a la recogida de datos, su funcionalidad está distribuida en cuatro módulos que agrupan funcionalidades similares:

- *Módulo de comunicaciones* - Responsable de capturar el tráfico de red y algunas estadísticas asociadas al mismo.
- *Módulo de aplicaciones* - Destinado a obtener las aplicaciones instaladas en el dispositivo y los permisos declarados por las mismas.
- *Recursos hardware* - Permite acceder al estado de los recursos hardware del dispositivo.
- *Mecanismos de protección* - Posibilita determinar los mecanismos de protección activos en el sistema.

A modo de resumen, en la Tabla 6.2 [533] se recogen los parámetros que recolecta la aplicación hasta el momento, agrupados por categorías.

Esta herramienta permite múltiples aplicaciones relacionadas con la monitorización de dispositivos Android, como son: detección de *malware* mediante el análisis de los permisos declarados de las aplicaciones instaladas, detección de *ransomware* a partir del consumo de recursos como el uso de CPU [534], etc. También es posible su utilización para determinar el nivel de seguridad de un dispositivo al objeto de realizar un control de acceso basado en permisos; por ejemplo, sistema ARANAC [535].

Dicho todo lo anterior, esta solución de monitorización activa resulta bastante flexible pero presenta algunos problemas como son:

- Recursos - Si bien el uso de recursos no es muy elevado dada la optimización que se hace de los mismos, debemos indicar que no es muy elevado en términos absolutos pero sí en términos relativos. En el caso de AMon, el consumo de batería en un régimen de uso normal es del 0,4 %, que no es muy elevado por sí solo pero es el doble comparado con el consumo de una aplicación antivirus (medidos ambos consumos en un teléfono Samsung S9 con una batería de 3.000 mAh).

- Procesamiento *offline* de atributos - Dependiendo del método de detección utilizado podremos hacer un procesamiento de los datos recogidos dentro o fuera del teléfono. En el caso mayoritario, que utiliza técnicas basadas en ML, el procesamiento debe hacerse fuera del teléfono para no agotar los recursos del dispositivo.

- Consumo de datos - Si necesitamos realizar el procesamiento fuera del dispositivo, tenemos que contar con un incremento del tráfico de datos y un mayor uso del servicio de comunicación, lo que supone mayor consumo de batería.

Ante esta situación, nuestra propuesta se orienta hacia una monitorización reactiva que permita a la aplicación de detección consumir recursos sólo cuando se producen aquellos eventos de interés para la misma. El siguiente apartado justifica en mayor profundidad esta propuesta, para luego abordar la implementación concreta propuesta.

6.4. Monitorización 'reactiva' del sistema

Como hemos señalado, la monitorización 'activa' tiene algunos inconvenientes. El principal de ellos radica en ser un proceso que consume recursos del dispositivo como CPU y batería. Otro aspecto a tener en cuenta es que el uso de apps como AMon generan gran cantidad de datos que repercuten en el aumento del tráfico de datos del dispositivos y el correspondiente coste monetario.

Grupo	Subgrupo	Atributo	Descripción
Configuración (21)	Aplicación (5)	*appIs*	Identificador aplicación
		macId	Dirección MAC
		name/version	Nombre del paquete/versión
		permissions	Lista permisos del paquete
	Estado (3)	*ramUsage*	Uso de RAM
		batteryLevel	Nivel de batería
		cpuUsage	% CPU en uso
	Seguridad (4)	*unknownSources*	Software fuente desconocida?
		developerOptions	Opción de desarrollo activa?
		secure	Mecanismo de bloqueo?
		rooted	Dispositivo *rooteado*?
	Dispositivo (9)	*mac*	Dirección MAC
		device (4)	Descripción (modelo, etc.)
		sdk	SDK usado
		cpuCores	# de núcleos
		ramTotal	Total de RAM instalada
		batteryTotal	Total de capacidad batería
Comunicaciones (25)	Tráfico (14)	*macID*	Identificación de la MAC
		packageName	Paquete responsable flujo
		time	Tiempo de la comunicación
		duration	Duración del flujo
		protocol	Protocolo de comunicación
		saddr / daddr	Dirección fuente / destino
		sport / dport	Puerto fuente / destino
		sentBytes	# de *bytes* enviados
		receiveBystes	# de *bytes* recibidos
		sentPackets	# de paquete enviados
		receivePackets	# de paquetes recibidos
		tcpFlags	Estado flujo: nuevo, activo, cerrado, ...
	Bluetooth (2)	*macID*	ID de la dirección MAC
		bluetoothDevice	Descripción del *bluetooth*
	WiFi (3)	*macID*	ID dirección MAC
		SSID	SSID de la WiFI conectada
		security	Mecanismo seguridad WiFi
	Conectividad (6)	*macId*	Dirección MAC
		data	Datos de red activos?
		wifi	WiFi activa?
		bluetooth	*Bluetooth* activo?
		gps	GPS activo?
		airplane	Modo avión activo?

Tabla 6.2: Atributos recolectados por AMon.

Estos inconvenientes nos hace plantearnos un mecanismo de detección 'reactiva' de *ransomware*, es decir, monitorización que se realiza solo cuando hay un evento que requiere atención y, por tanto, este mecanismo requeriría unos recursos mínimos mientras ese evento no ocurra.

6.4.1. Monitorización reactiva del sistema de archivos

Para el desarrollo de un mecanismo de detección de *ransomware* en Android mantenemos uno de los principios establecidos por R-Locker en las plataformas Linux y Windows, a saber: detectar los programas con comportamiento malicioso cuando interaccionan con los objetos del sistema de archivos, concretamente cuando se procede a cifrar el contenido de los mismos. Para ello, manteniendo la premisa de desplegar una aplicación sin privilegios de administrador, analizamos los mecanismos que el sistema operativo suministra al usuario mediante su API para evaluar otras alternativas. Esto nos lleva a determinar que el mecanismo de monitorización de archivos suministrado a través de la interfaz dada por `FileObserver()` parece ser adecuado para tal fin, como muestran los trabajos que utilizan señuelos en Android: RW-Guard [328], donde son necesarios privilegios de administrador; KR-Protector [536], que tienen una complejidad `O(n)` con `n` igual al número de carpetas en la memoria. Seguidamente vamos a describir el mecanismo de monitorización de archivos soportado por Android y vamos a implementar una aplicación que lo utilice en la detección de *ransomware* aproximando su forma de trabajo a la ya establecida por R-Locker, para concluir analizando sus pros y contras.

6.4.2. La clase `FileObserver()`

En Android, la clase `FileObserver()` [537] encapsula el mecanismo `inotify()` de Linux [493]. Este mecanismo notifica al *kernel* cuáles son los archivos/directorios objetivos de la monitorización mediante la llamada `inotify_add_watch()`, que añade una entrada a la lista de observación para cada instancia de *inotify*. La documentación indica que la función no registra cuál es el proceso o hilo que accede a la ruta monitorizada. La función `inotify_add_watch()` no es recursiva, hecho que se traslada a `FileObserver()` de Android, aunque la documentación indica lo contrario (ver la nota para desarrolladores [538]). Cada entrada de la lista de observación contiene la ruta del archivo a monitorizar y una máscara de *bits* que representa los eventos a observar. La función devuelve un descriptor de observación que será el que se utilice en operaciones posteriores como `read()`.

La ventaja de este mecanismo frente a otros tipos de monitorización es que los descriptores de observación pueden leerse mediante algún mecanismo de E/S asíncronas, o controladas por señales, como `select()`, `poll()` o

Tabla 6.3: Eventos monitorizables con `FileObserver()`.

Evento	Descripción
`ACCESS`	Se han leído datos del archivo
`ALL_EVENTS`	Todos los tipos válidos de eventos combinados
`ATTRIB`	Han cambiado los metadatos de forma explícita
`CLOSE_NOWRITE`	Alguien tiene archivo/directorio abierto solo-lectura y lo cerró
`CLOSE_WRITE`	Alguien tiene archivo/directorio abierto para escritura y lo cerró
`CREATE`	Se ha creado un archivo/directorio en la carpeta monitorizada
`DELETE`	Un archivo se ha borrado del directorio monitorizado
`DELETE_SELF`	Se ha borrado el archivo/directorio en el directorio monitorizado; detiene la monitorización
`MODIFY`	Los datos del archivo se han escrito
`MOVED_FROM`	Se movió un archivo o directorio desde la carpeta monitorizada
`MOVED_TO`	Se movió un archivo o directorio a la carpeta monitorizada
`MOVED_SELF`	Se ha movido el archivo o directorio observado; continúa la monitorización
`OPEN`	Se ha abierto un archivo o directorio

`epoll()`, donde el *kernel* indica al proceso de observación cuándo un evento está preparado para lectura, liberando a dicho proceso de estar constantemente sondeando al sistema para ver si se ha producido el evento.

Volviendo a Android, `FileObserver()` es una clase abstracta; se debe establecer un gestor del evento con `onEvent()`. Cada instancia de la clase puede monitorizar una ruta y usa una máscara de eventos para especificar sobre qué cambios o acciones informar. La lista de eventos observables en Android se pueden ver en la Tabla 6.3, junto a una descripción de su significado.

La tabla anterior recoge los eventos monitorizados y, como puede observarse, ni se monitoriza el contenido de los archivos ni los procesos o aplicaciones que interaccionan con ellos.

Como indicábamos anteriormente, *FileOberserver* monitoriza archivos de forma individual, por lo que si queremos monitorizar la memoria externa completa deberemos recorrer recursivamente el árbol de directorios de la misma y establecer instancias para cada uno. El uso de la función tal como establece la documentación es bastante simple, como muestra el fragmento de código del Listado 6.1. En este código se muestra cómo, tras establecer la función para gestionar los eventos sobre los archivos a monitorizar, el

Listado 6.1: Uso de `FileObserver` en Android.

```
1        new FileObserver ("ruta a directorio") {
2                @Override
3                public void onEvent(int evento, String archivo){
4                        ... Lógica de gestión del evento
5                }
6        };
7        observer.startWatching{}; // Inicio de la monitorización
```

proceso invocador queda a la espera de que dichos eventos tengan lugar.

Con el objetivo de probar la validez del uso de `FileObserver` como mecanismo reactivo para la detección de *ransomware*, se ha desarrollado una aplicación, denominada *FileMonitor* [539]. Dicha aplicación tiene una interfaz de usuario sencilla tal como puede verse en la Figura 6.4. En ella encontramos, en la parte inferior, botones para iniciar/detener la monitorización, para comprobar su funcionamiento y para limpiar la lista de monitorización. En la parte superior podemos ver qué aplicaciones tenemos instaladas con permisos peligrosos y, por último, permite filtrar los eventos a monitorizar (ver Tabla 6.3).

Nuestro objetivo es, primero, validar el uso de esta clase para la detección de actividades maliciosas; luego, explorar las posibilidades para la detección de *ransomware* y poder activar las contramedidas para detenerlo.

6.4.3. Detección de actividades maliciosas con `FileObserver`

Vamos a describir la aplicación desarrollada para ver las posibilidades de dicha funcionalidad para sustituir a R-Locker, utilizando la función *FileObserver*. Esta aplicación, FileMonitor, a la vista de lo anterior, va a monitorizar la memoria externa para detectar actividades sospechosas.

Para el desarrollo de dicha aplicación establecemos los siguientes requisitos funcionales y no funcionales:

RF1 - El usuario puede arrancar/detener la monitorización de archivos.

RF2 - El usuario puede limpiar la lista de eventos de monitorización.

RF3 - La aplicación mostrará el tipo de evento registrado y la ruta afectada, pudiéndose filtrar por tipo de evento.

RF4 - La aplicación mostrará la lista de aplicaciones peligrosas con los permisos correspondientes.

NF1 - Tanto la interfaz como el uso de la aplicación deben ser simples y fáciles de comprender.

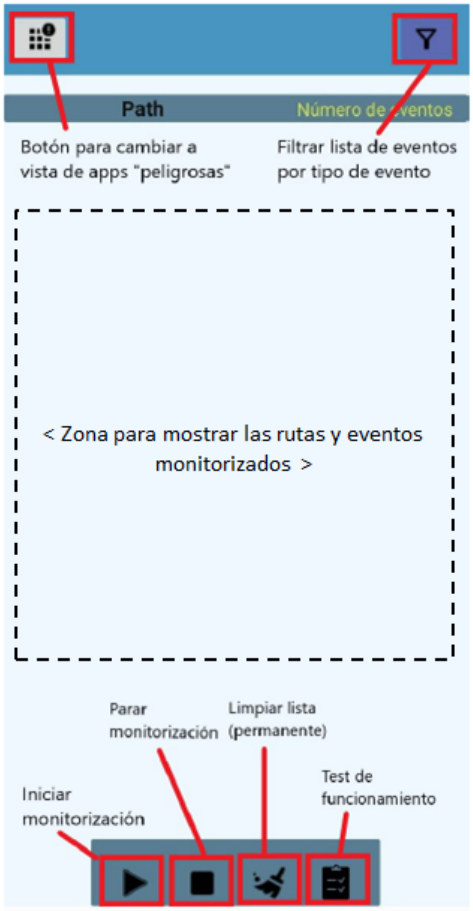


Figura 6.4: Interfaz gráfica de la aplicación *FileMonitor*.

NF2 - La lista de eventos se debe mostrar de forma organizada. Un primer nivel mostrará un resumen de los eventos sobre un archivo; un segundo nivel, la lista detallada de los mismos.

NF3 - La lista de aplicaciones peligrosas también tiene dos niveles: primero, se muestra de forma resumida; segundo, pulsando sobre ella se desplegarán los detalles.

NF4 - La monitorización de archivos continuará aunque la aplicación se ponga en segundo plano.

El funcionamiento de la aplicación es sencillo una vez que hemos comentado cómo funciona la clase `FileObserver()`, si bien es necesario implementarla como una actividad y un servicio (para cumplir RNF4, debe funcionar en segundo plano). El servicio realiza la monitorización en sí, la actividad se encarga de recibir los datos del servicio y mostrarlos al usuario.

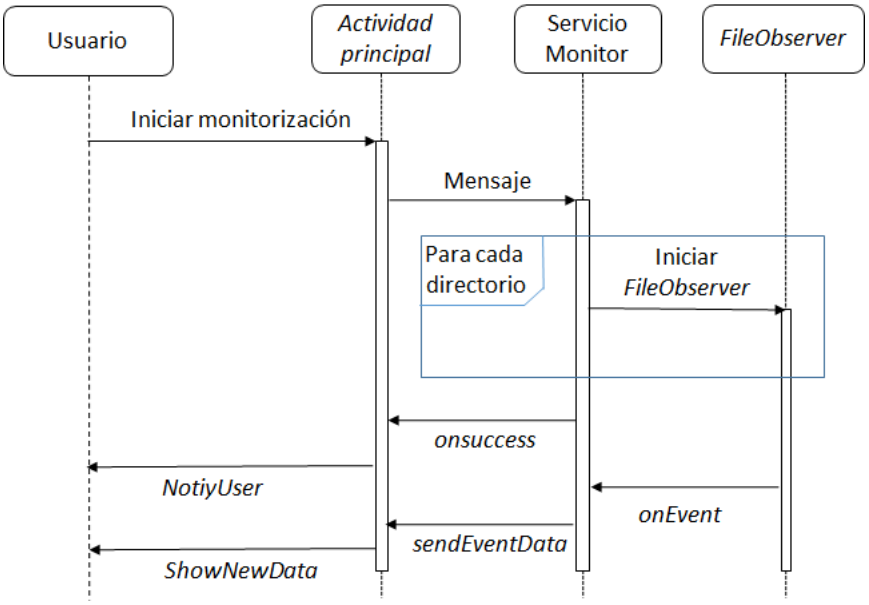


Figura 6.5: Diagrama de secuencia de la inicialización de la monitorización y la notificación de un evento.

La Figura 6.5 representa el diagrama de secuencias de las acciones tanto de inicialización de la monitorización como de notificación de un evento.

Para el desarrollo del sistema se ha utilizado Java con Android Studio por ser el IDE oficial que ofrece estabilidad, permite desarrollar la interfaz gráfica y la emulación en diferentes dispositivos. La implementación del servicio es convencional como muestra el Listado 6.2, donde hay que resaltar el uso del indicador `START_STICKY` para que dicho servicio no se detenga al estar en segundo plano.

El servicio debe declararse en el manifiesto con el permiso correspondiente (`android.permission.FOREGROUND_SERVICE`), como aparece en el Listado 6.3. También se puede observar cómo se solicitan por parte de la app los permisos para leer/escribir en el almacenamiento externo, consultar sobre los paquetes existentes en la plataforma y activar la pantalla del dispositivo.

Como indicábamos anteriormente, la función `FileObserver()` no es recursiva, por lo que debemos modificar el Listado 6.1, que implementa una monitorización sencilla, para se realice en todo el sistema de archivos: Listado 6.4.

La comunicación entre el servicio y la actividad se realiza mediante un `LocalBroadManager` suficiente para comunicar componentes de la misma aplicación.

La API de Android suministra funcionalidad para recabar información de las aplicaciones que se están ejecutando a través de las funciones de `PackageManager`: `getPackageManager()` y `getInstalledPackages()` . Es-

Listado 6.2: Creación del servicio utilizado por *FileMonitor*.

```
1          @Override
2         public int onStartCommand(Intent intent, int flags, int
              startId) {
3
4                 if (Build.VERSION.SDK\_INT >= Build.VERSION\_CODES.O)
                      {
5                         createNotificationChannel();
6                 } else {
7                         CHANNEL\_ID= "";
8                 }
9
10                Notification notification = new NotificationCompat.
                      Builder(this, CHANNEL\_ID).build();
11                startForeground(1, notification);
12
13                for (File f : observed\_files){
14                        Log.i("Service:", f.getAbsolutePath());
15                        observers.add(new singleFileObserver(f));
16                }
17                Toast.makeText(this,"Iniciando Fileobserver en el
                      directorio" + root.getAbsolutePath() + "/",Toast.
                      LENGTH\_SHORT).show();
18                return Service.START\_STICKY;                //
                      Servicio tipo "sticky" para que no sea parado por
                      el sistema antes de tiempo
19        }
```

Listado 6.3: Declaración del servicio levantado por *FileMonitor* en segundo plano

```
1 <\\application>
2         . . .
3         <service android:name="com.filemonitor.prueba.fileobserver.
              FileObserverService"
4         android:enabled="true"></service>
5
6         </aplication>
7
8         <uses-permission android:name="android.permission.WRITE\
              _EXTERNAL\_STORAGE" />
9         <uses-permission android:name="android.permission.WAKE\_LOCK"
              />
10        <uses-permission android:name="android.permission.READ\
              _EXTERNAL\_STORAGE" />
11        <uses-permission android:name="android.permission.FOREGROUND\
              _SERVICE"/>
12        <uses-permission android:name="android.permission.QUERY\_ALL\
              _PACKAGES" />
```

Listado 6.4: Implementación de *FileObserver* para monitorizar el árbol de directorios

```
public void recursiveFileObserver(File root, List<File> files) {
  File[] = list = root.listFiles();

  if (list != null) {
        for (File f : list) {
                if (f.isDirectory()) {
                        FileObserver = new FileObserver(f.
                            getAbsolutePath()) {
                                @Override
                                public void onEvent)int event,
                                    @Nullable String path) {
                                        // Operaciones para gestionar
                                            el evento sobre esa ruta
                                }
                                FileObserver.startWatching();

                                files.add(f);
                                recursiveFilesList(f, files);
                        }
                }
        }
  }
```

te paquete devuelve las apps instaladas y los permisos de las mismas, tanto los declarados en el manifiesto como los concedidos. Esto sirve para seleccionar aquellas apps que tienen permisos peligrosos. El código de dicha aplicación puede encontrarse en `https://github.com/JA-Gomez-Hernandez/FileMonitor`.

Como ejemplo de monitorización, podemos ver en la Figura 6.6 cómo se registran los eventos tal como se establecen en los requisitos RF3 y RNF2.

Experimentación

Para realizar las pruebas con *malware* se ha desplegado el siguiente entorno de ejecución:

- Una máquina virtual con VirtualBox para aislar el entorno de ejecución del de trabajo.
- Una máquina virtual Android dentro de la MV anterior. Esta virtualización se realiza con GenyMotion [540].

Para realizar las pruebas se han utilizado muestras de *malware* obtenidas del repositorio AndroidMalware [541] que sabemos que interaccionan con la memoria externa y que se recogen en la Tabla 6.4.

En concreto, si nos fijamos en la prueba realizada con el *ransomware* Cyberpunk (una app que se hace pasar por un juego de moda del mismo

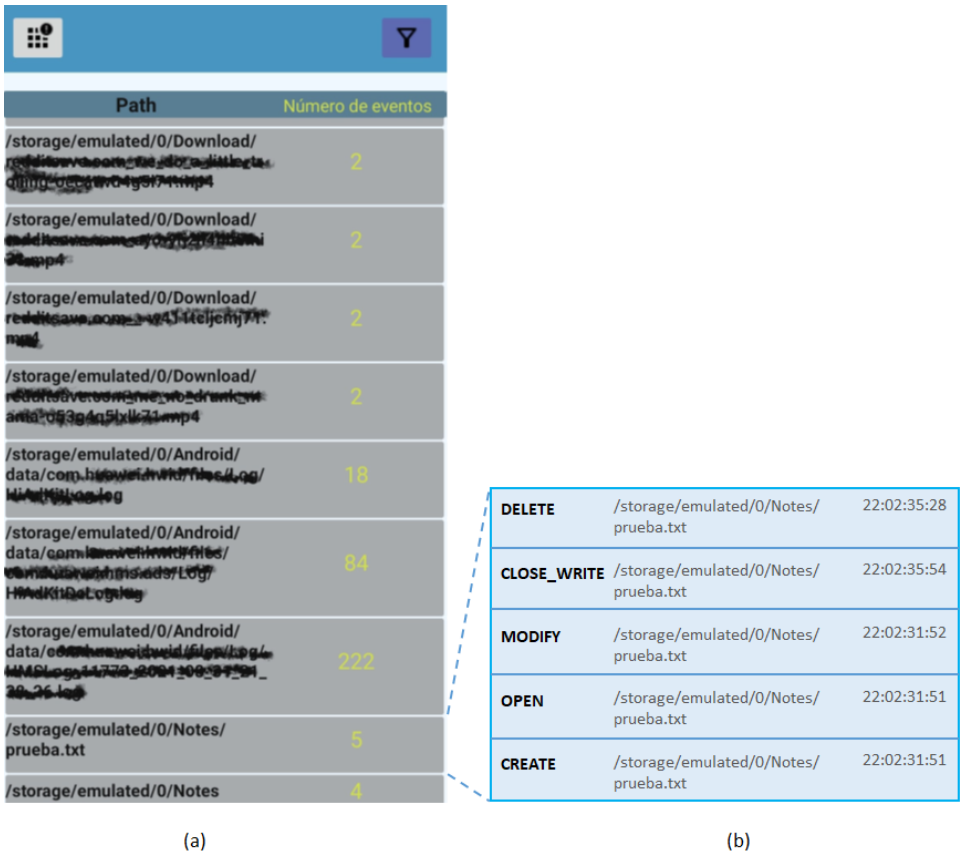


Figura 6.6: Vista resumen de eventos monitorizados (a) y vista detallada de los eventos sobre el archivo *prueba.txt* (b).

Tabla 6.4: Muestras del *malware* utilizadas en las pruebas de *FileMonitor*.

Nombre	***Hash MD5***	**Tipo**
CookierStealer	65a92baefd41eb8c1a9df6c266992730	*Spyware*
Covid_SpyPhone	3288a6cb81bc3e928e438fa280fec847	*Riskware*
Covid_Cerberus	66c4513025128719dda018820cc0987e	*Spy/Dropper*
Crydroid	381134ea0f0be535b9d2ce8a94093576	*Ransomware*
Cyberpunk	cbd92757051490316de527a02ac17947	*Ransomware*
Joker	44faa3de0f17491557a3a775c88e7e33	*Spy/Dropper*
Shopaholic	0a421b0857cfe4d0066246cb87d8768c	*Dropper*
ThiefBot	e88867956017bbe5b633811885c87018	*Spyware*
Trickbot	05c0c1bb23cc06474c3fd3ba51e4e4c6	*Spy/Dropper*

nombre, lo que permite que el usuario le otorgue permisos de escritura), la aplicación desarrollada permite caracterizar su comportamiento, tal como se ve en la Figura 6.7. En ella se aprecia que los archivos monitorizados se ven sujetos a tres operaciones:

1. El evento `DELETE` indica una operación de borrado del archivo (el borrado del archivo -`unlink()`- indica que se borra la información en disco, sus datos permanecen intactos mientras haya una referencia activa a él en el *kernel*, es decir, mientras que el archivo esté abierto).

2. El evento `MODIFY` refleja que se modifica el contenido del archivo, incluido un cambio en la extensión.

3. Se crea (evento `CREATE`) la versión cifrada del archivo, que tiene el mismo nombre y la extensión *.coderCrypt*.

Este esquema de comportamiento encaja en el tipo visto en la Figura 2.20(b) del Capítulo 2, que se denominaba 'lectura-cifrado-borrado'.

Por tanto, la aplicación *FileMonitor* permite detectar muestras de *ransomware* ejecutándose en un dispositivos mediante la caracterización del comportamiento de dicho procesos durante el proceso de cifrado en base a los patrones vistos en la Sección 2.4.8. Además, también podemos determinar el tipo de *ransomware* en función de la extensión en el nombre de los archivos que va cifrando.

Las medidas realizadas para determinar el tiempo de respuesta del mecanismo de notificación indican que la monitorización de 200 eventos sobre un archivo monitorizado requiere un tiempo de notificación de: entre 1 y 200 ms, si solo se ejecuta una aplicación; 1-450 ms, si se ejecutan cuatro; y, 1-700, cuando se ejecutan nueve. Como se puede observar, a mayor número de eventos de monitorización más tarda el sistema-aplicación en responder a ellos. Esto era de esperar dado que, como comentábamos al describir *inotify*, el mecanismo dispone de una cola para guardar qué eventos debe atender, por lo que a mayor número de eventos mayor tiempo de respuesta. Además, *FileObserver* descansa en un mecanismo de *callback* que notificará los eventos con un retardo mayor en función de la carga del sistema. En el dispositivo de prueba se ha medido el coste de cifrar un archivo: 10 ms para un archivo de 8KB; 70 ms, para un archivo de 60 KB. Por tanto, una muestra de *ransomware* podría cifrar unos diez archivos si distribuimos señuelos de 60KB, por lo que podemos considerar que se comporta como un mecanismo de detección temprana.

Una vez mostrada la capacidad de detección de *ransomware* en Android analizando la interacción del mismo con el sistema de archivos, podemos proceder como en el caso de R-Locker para los sistemas operativos Linux y Windows, que analizamos en los Capítulos 4 y 5, respectivamente:

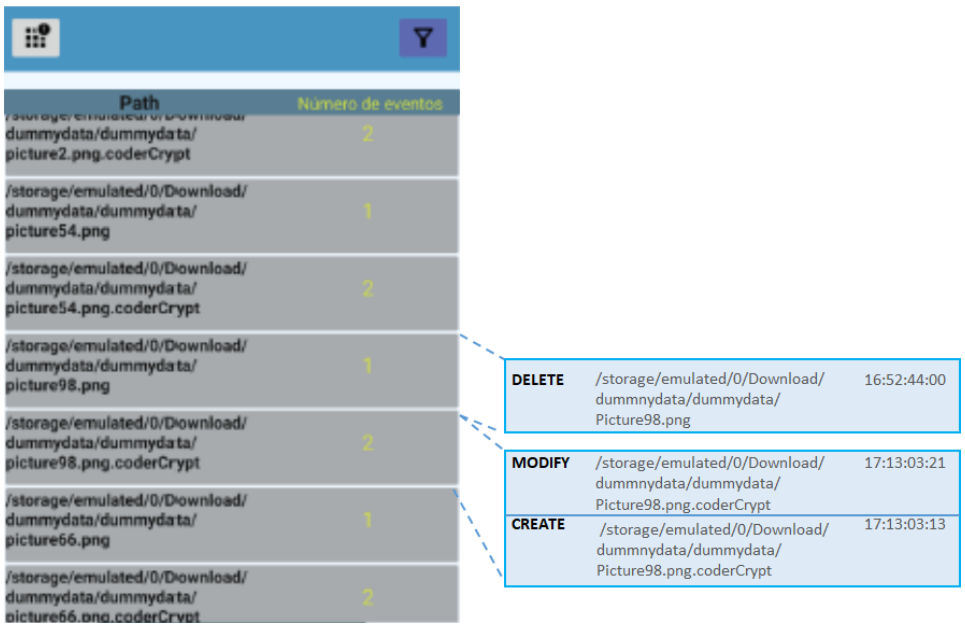


Figura 6.7: Monitorización del proceso de cifrado de archivos por CyberPunk.

- En primer lugar, necesitamos desplegar archivos trampa/señuelo por todas las carpeta del árbol de directorios de la memoria externa, debido a que no todas las familias de *ransomware* siguen el mismo patrón para seleccionar archivos víctima como mostrábamos en la Tabla 2.7 de la Sección 2.4.8. Estos archivos señuelo son los objetivos de la monitorización. Para hacerlos transparentes al usuario tendrán un nombre con la forma *'.<nombre.extensión>'*. El '.' es la forma convencional de ocultar los archivos a ciertas órdenes en Linux para que el usuario no los visualice, es decir, queden ocultos al usuario. Si bien el *nombre* del archivo puede ser cualquiera, las extensiones deberán ser atractivas para el *ransomware*, como indicábamos en la Tabla 2.8 de la sección recientemente comentada. Una vez desplegados dichos archivos por todo el sistema de archivos solo queda esperar a la interacción de programas potencialmente dañinos con dichos archivos.

- Cuando se detecta el acceso a algunos de los archivos monitorizados, debemos desplegar las contramedidas necesarias que permitan, por una parte, determinar qué aplicación tiene este comportamiento maliciosos y, por otra, finalizar dicho proceso notificando al usuario como ya hicimos en R-Locker.

  Para determinar el proceso malicioso y, siguiendo la propuesta de R-

Locker, utilizamos una lista blanca de aplicaciones benignas. Como se describió en el Capítulo 5, nuestra propuesta lo que hace es determinar qué aplicaciones están instaladas en el momento que instalamos la aplicación *FileMonitor* y se almacenan en una lista blanca, donde suponemos que, dado que el dispositivo funciona correctamente, son aplicaciones benignas.

En el caso de Android utilizamos el permiso `QUERY_ALL_PACKAGES`, que se ha debido conceder a la aplicación, para recabar la lista de aplicaciones instaladas en el dispositivo antes de comenzar la monitorización. Cuando se produce la detección volvemos a recoger la dicha información y, por tanto, localizar la app (o apps) que se ha instalado con posterioridad a la de monitorización y que es la candidata a tener el comportamiento malicioso. Siguiendo en esquema de R-Locker, podemos matar dicho proceso dándole a nuestra aplicación el permiso `KILL_BACKGROUND_PROCESSES`.

A diferencia de la propuesta realizada para Linux y Windows, la propuesta para Android es capaz de construir una lista blanca más selectiva, ya que podemos seleccionar solo aplicaciones que tienen los permisos necesarios para causar daño en la memoria externa y que denominamos aplicaciones peligrosas. La Figura 6.8 muestra una lista blanca con aplicaciones potencialmente 'peligrosas' (a) y el detalle de los permisos de una de estas aplicaciones (b).

- Por otro lado, con el objetivo de detectar y detener diferentes hilos de una muestra de *ransomware* multihebrada, podemos usar, como ya hicimos, la solución de la lista negra, donde se anotan aquellas aplicaciones detectadas como maliciosos de cara a detener nuevas instancias de la misma haciendo transparente dicho proceso al usuario.

Una ventaja de *FileMonitor* es que puede detectar otros comportamientos maliciosos, no solo *ransomware*. Para mostrarlo vamos a utilizar una aplicación, denominada ThiefBot, que es un *spyware*. Dicha aplicación apareció en septiembre de 2020 y su objetivo es obtener credenciales bancarias, si bien recopilaba todo tipo de credenciales e información personal. Esta app se hace pasar por la aplicación Google Play y solicita permisos para acceder al almacenamiento, a los SMS, al teléfono, a los contactos y a la cámara. Al instalar dicha app, esta muestra un mensaje de error, en el que se indica que no está realizada para funcionar en versiones más antiguas de Android, que no funciona correctamente y que se deben localizar actualizaciones de la misma. Tras ello parece que se cierra, cuando en realidad pasa a trabajar en segundo plano.

Si utilizamos *FileMonitor* tras la instalación de este *spyware*, obtenemos los resultados que muestra la Figura 6.9, donde podemos ver un número

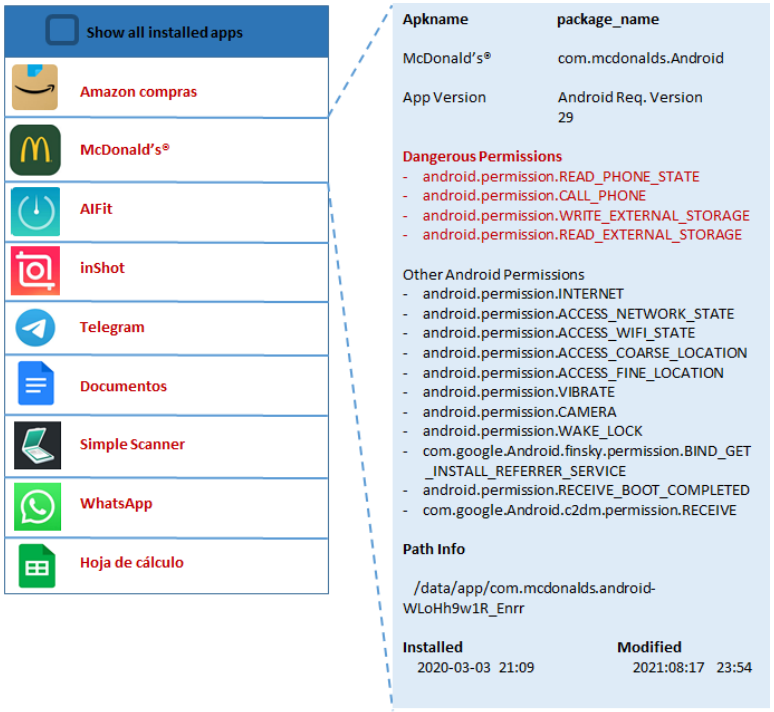


Figura 6.8: Lista blanca de aplicaciones potencialmente peligrosas (a) y detalle de los permisos de una de ellas (b).

muy elevado de accesos de tipo `ACCESS` y `OPEN` a una carpeta creada por la app maliciosa, denominada *downloads*.

Como se puede observar, en un periodo pequeño de tiempo se accede a la carpeta con un número muy elevado de eventos (del orden de 5.000 eventos por minuto). Ello nos hace sospechar del comportamiento malicioso de la misma.

6.5. Conclusiones

En este capítulo hemos analizado el problema que supone el *ransomware* en sistemas móviles, especialmente en la plataforma Android. Tras ver el modelo de seguridad de Android, que es bastante robusto, hemos esbozado algunos de los huecos de seguridad que plantea, abordando especialmente el problema de sobre-otorgamiento de permisos que ofrece una mayor superficie de ataque de los dispositivos.

Además, se han comentado los problemas encontrados al tratar de implementar el enfoque R-Locker debido a la política de seguridad de Android para la creación de FIFO, así como a las limitaciones del sistema de archivos

Figura 6.9: Resultado de la monitorización del *spyware* ThiefBot.

de la memoria externa para la creación de enlaces. Dado que estos problemas son insalvables salvo que se *rootee* el dispositivo, hemos optado por recurrir a una detección basada en la monitorización del mismo.

La primera aproximación de monitorización abordada ha sido utilizar una app de monitorización activa, en nuestro caso AMon, que recoja de forma dinámica información del dispositivo para caracterizar el comportamiento del mismo desde el punto de vista de seguridad. El principal problema de este enfoque es que, si bien la app desarrollada usa los mínimos recursos posibles, el consumo de los mismos es considerable comparado con otras aplicaciones del sistema. Por otra parte, tiene problemas adicionales el costo de transmisión de datos y la necesidad de utilizar recursos fuera del dispositivo para el análisis del comportamiento del móvil.

En base a los problemas del primer enfoque, hemos analizado también una monitorización reactiva, donde indicamos al sistema operativo qué eventos del sistema de archivos queremos gestionar y quedamos a la espera de que este los notifique para gestionarlos. Para ello se ha desarrollado la aplicación *FileMonitor* basado en la función `FileObserver()`, que permite un mecanismo de detección de comportamientos maliciosos como *ransomware* desde sus primeras acciones sobre el sistema de archivos. Este mecanismo se completa con el de listas blancas, para localizar la aplicación maliciosa y poder detenerla, y con el de listas negras, para registrar la aplicación detectada y poder detener múltiples hilos de la misma.

El problema de dicha aplicación es que el mecanismo subyacente, *inotify*, no detiene al proceso que interacciona con el archivo monitorizado, por lo

que el tiempo de respuesta no está acotado (no como el R-Locker, donde al tener el proceso bloqueado no encontramos limitaciones en el tiempo de reacción). En todo caso, un despliegue y configuración adecuados de File-Monitor permitirá tiempos de respuesta en la detección breves, de forma que la afectación del sistema de archivos sea reducida.

Capítulo 7

Conclusiones y trabajo futuro

En este capítulo final se resumen las principales conclusiones extraídas tras la realización del trabajo de Tesis aquí expuesto y que se han ido detallando en los capítulos anteriores, presentándose de forma unificada y abreviada. Además, describe brevemente las líneas de trabajo futuro al objeto de continuar con la investigación aquí iniciada.

7.1. Conclusiones

El cripto-*ransomware* se ha convertido en una de las forma prevalentes de ciberamenaza, especialmente para las organizaciones, que resultan un objetivo prioritario dada su mayor propensión al pago del rescate debido al coste e impacto negativo que les supone no hacerlo. Los atacantes están capitalizando los avances tecnológicos, como la adopción de la nube y los entornos de trabajo híbridos, mediante campañas de *ransomware* dirigidas y persistentes operadas por humanos. Por ello somos conscientes de que se debe de seguir mejorando los mecanismos de defensa. En este sentido presentamos:

- Un estudio de la evolución de esta amenaza que ayude a realizar una proyección a corto y medio plazo de del tipo de extorsión y forma de actuación antes esta amenaza.
- Una caracterización de los elementos constitutivos del *ransomware* basada en el modelo de ataque *Cyber Kill Chain*, para comprender mejor todas las posibles técnicas usadas por los programas maliciosos y que permita comprender dónde podemos establecer las defensas y de qué tipo.

- Una revisión de las propuestas actuales para la lucha contra el cripto-*ransomware*, especialmente aquellas que permitan desarrollar una herramienta anti-*ransomware* que sea simple, eficiente y que no requiera privilegios especiales para su instalación y operación. Además, dicha herramienta debería poder desplegarse en un amplio número de plataformas para alcanzar los fines previstos.

- Una revisión de las propuestas realizadas hasta la fecha en relación a la prevención, detección y respuesta a esta amenaza, lo que nos ha permitido determinar que nuestra herramienta debe estar más enfocada a la detección y respuesta, dado que la prevención aborda problemas más generales de ciberseguridad.

- Una propuesta de mejora de las taxonomías existentes que permiten comprender mejor las múltiples soluciones existentes en base a qué datos y procesamiento utilizan para realizar la detección.

Realizada una revisión de la literatura especializada acerca de la caracterización y detección del *ransomware*, las principales contribuciones de este trabajo están centradas en la detección de cripto-*ransomware* basadas en la utilización de archivos trampa, y son:

- Desarrollamos un método que permite la detección de la ejecución de una muestra de cripto-*ransomware* utilizando únicamente los servicios ofrecidos por el sistema operativo ofertados por su API. Esto permite que la herramienta no necesite privilegios especiales y que su instalación no requiere una operativa compleja, por lo que el usuario del sistema puede instalarla y operarla de forma sencilla.

- Utilizando solo dos mecanismos, FIFO y enlaces, ofrecidos por los sistemas operativos Windows y Linux, hemos desarrollado una trampa, que conceptualmente se comporta con un archivo infinito. De esta forma, el sistema bloquea las acciones del *ransomware* y permite accionar las contramedidas por parte del proceso monitor.

- En Windows se ha podido resolver el problema de la separación entre los espacios de nombres de archivos y de dispositivos mediante el uso de enlaces simbólicos, lo que ha permitido dar la misma estructura a la solución original de Linux.

- Se ha definido cómo ocultar las trampas al usuario, de forma que no interfieran sus actividades y, de esta forma, sea un mecanismo transparente.

- Al objeto de proteger todo el sistema operativo, se establece el método para desplegar las trampas por todo el sistema de archivos. Además,

este despliegue se supervisa periódicamente para detectar que no se ha borrado ninguna de ellas, de forma accidental o intencionada. Si esto ha ocurrido, se restaura de nuevo la trampa.

- Dada la naturaleza más restrictiva de la configuración de Android desde en punto de vista de la seguridad y de las características de los dispositivos usados, se ha tenido que migrar hacia un mecanismo de monitorización reactiva basado en la funcionalidad ofertada por *FileMonitor*.
- Hemos diseñado e implementado la herramienta semi-automática para las plataformas más comunes del mercado: Android, Windows y Linux. Se ha mantenido el método general de detección pero han sido necesarias algunas modificaciones en la implementación para adaptarla a las peculiaridades de cada sistema operativo. El método de respuesta sigue en todos los casos el mismo esquema.
- Hemos probado su eficiencia en la detección, su sencillez en el uso y el escaso consumo de recursos que utiliza, lo que la hace escalable para cualquier tamaño del sistema de archivos a proteger y a la limitación de recursos que tenga la plataforma objetivo.
- La herramienta está disponible públicamente para aquellos investigadores que deseen profundizar el este tema y para cualquiera que desee ejecutarla y, por qué no, mejorarla.

7.2. Líneas de trabajo futuro

La línea de investigación iniciada durante el trabajo de Tesis deja abiertas varias ideas de trabajo futuro. Entre otras posibles, algunas de las principales son:

- El método desarrollado es demasiado conservador por lo que puede dar algunos falsos positivos en tanto en cuanto cualquier proceso que acceda a la trampa sería detectado pese a que esta no es directamente visible. Eliminar estos casos produciría una herramienta automática, que no requeriría la intervención del usuario más allá de ser notificado.
- Dadas las similitudes entre los mecanismos utilizados en todos los casos, así como la forma de manipularlos, sería posible abordar el desarrollo de una única herramienta para la diferentes plataformas, que se distribuiría como ejecutable en Windows y Linux, y como apk en Android.
- En Android, y dada la naturaleza reactiva de la monitorización, habría que idear un mecanismo para minimizar el impacto del *ransomware* debido a los retardos producidos en la notificación de eventos.

- Actualmente, la herramienta está escritas en el lenguaje nativo del sistema operativo para los que se ha desarrollado. Re-implementar la herramienta en un lenguajes portable, como Java, simplificaría su uso y facilitaría su distribución.
- Sería beneficioso seguir extendiendo la herramienta a otras plataformas de uso frecuente, por ejemplo iOS.
- Dado el alto número de familias y muestras que se crean constantemente, sería necesario evaluar la herramienta frente estas nuevas amenazas.
- En relación al punto anterior, sería una gran ayuda para los investigadores disponer de una base de datos específica de muestras de *ransomware* para la experimentación.
- En el caso de Android, dadas sus restricciones de configuración, sería posible abordar qué adaptaciones se podrían hacer en el sistema para que permitir el bloqueo de las muestras de *ransomware*.
- En Windows también se podría abordar el uso de DLL *kernel* para solventar el problema del tamaño de los enlaces simbólicos.

Bibliografía

[1] Embroker, Top 10 Cybersecurity Threats in 2022. Disponible en https://www.embroker.com/blog/top-10-cybersecurity-threats-2022/.

[2] A. Zaharia, 300+ Terrifying Cybercrime and Cybersecurity Statistics (2022 EDITION), Comparitech, Feb. 2022. Disponible en https://www.comparitech.com/vpn/cybersecurity-cyber-crime-statistics-facts-trends/.

[3] J. Pracht, Five Threats That Will Drive the 2022 Cybercrime Economy, Mainstream Technologies, Jan. 2022. Disponibles en https://www.mainstream-tech.com/2022-cybercrime-economy/.

[4] Imperva, 2021 Cyberthreat Defense Report, Cyberedge Group, 2021. Disponible en https://www.imperva.com/resources/resource-library/reports/2021-cyberthreat-defense-report/.

[5] A. Morrison, Cyber Security Landscape 2022, Deloitte, Feb. 2022.

[6] Statcounter, Operating System Market Share World Wide, May 2022. Disponible en https://gs.statcounter.com/os-market-share.

[7] A. Liska, Ransomware: Understand, Prevent, Recover, ActualTech Media, 2021.

[8] S. Schaibly, Files for Ransom, Network World, Septiembre 2005. Disponible en https://www.networkworld.com/article/2314306/files-for-ransom.html.

[9] J. Canavan, The Evolution of Malicious IRC Bots, Symantec White Paper, 2005. Disponible en https://www.yumpu.com/en/document/view/13999738/the-evolution-of-malicious-irc-bots-symantec.

[10] S. Corbet, J.W. Goodell, The reputational contagion effects of ransomware attacks, Finance Research Letters, 102715, 2022. DOI: 10.1016/j.frl.2022.102715.

[11] B. A. S. Al-rimy, M. A. Maarof, S. Z. M. Shaid, Ransomware threat success factors, taxonomy, and countermeasures: A survey and research directions, Computers & Security, Volume 74: 144-166, 2018, DOI:10.1016/j.cose.2018.01.001.

[12] P. R. Kumar, H.R.E.B.H. Ramlie (2021) Anatomy of Ransomware: Attack Stages, Patterns and Handling Techniques. En: Suhaili W.S.H., Siau N.Z., Omar S., Phon-Amuaisuk S. (eds) Computational Intelligence in Information Systems. CIIS 2021. Advances in Intelligent Systems and Computing, vol 1321. Springer, Cham. DOI:10.1007/978-3-030-68133-3_20.

[13] C. Beaman, A. Barkworth, T. D. Akande, S. Hakak, M. K. Khan, Ransomware: Recent advances, analysis, challenges and future research directions, Computers & Security, Volume 111, 2021, 102490. DOI:10.1016/j.cose.2021.102490.

[14] Cyber Security, What Is Scareware? Examples & What to Know, Sectigo Store, Feb. 2020. Disponible en `https://sectigostore.com/blog/what-is-scareware-and-scareware-examples/`.

[15] R. Moussaileb, R.E. Navas, N. Cuppens, Watch out! Doxware on the way. . . , Journal of Information Security and Applications, Volume 55, 2020, 102668, DOI: 10.1016/j.jisa.2020.102668.

[16] A. Young, M. Yung, Cryptovirology: extortion-based security threats and countermeasures, Proceedings 1996 IEEE Symposium on Security and Privacy, 1996, pp. 129-140. DOI: 10.1109/SECPRI.1996.502676.

[17] S. Morgan, Global Ransomware Damage Costs Predicted To Reach $20 Billion (USD) By 2021, Cybercrime Magazin, 2019. Disponible en `https://cybersecurityventures.com/global-ransomware-damage-costs-predicted-to-reach-20-billion-usd-by-2021/`.

[18] M. Ryan, Ransomware Revolution: The Rise of a Prodigious Cyber Threat, Advances in Information Security Series (ADIS), vol. 85, Springer, 2021. DOI: 10.1007/978-3-030-66583-8.

[19] D. Formby, S.S. Durbha, R.A. Beyah, Out of Control : Ransomware for Industrial Control, Systems. RSA Conference, San Fancisco, 13-17 Feb. 2017.

[20] Dragos, EKANS Ransomware and ICS Operations, Dragos, Mar. 2020, Disponible en `https://www.dragos.com/blog/industry-news/ekans-ransomware-and-ics-operations/`.

[21] U.J. Butt, M. Abbod, A. Lors, H. Jahankhani, A. Jamal, A. Kumar, Ransomware Threat and its Impact on SCADA, 2019 IEEE 12th International Conference on Global Security, Safety and Sustainability (ICGS3), 2019, pp. 205-212. DOI: 10.1109/ICGS3.2019.8688327.

[22] N. Weiss, M. Schrötter, R. Hackenberg, On Threat Analysis and Risk Estimation of Automotive Ransomware, ACM Computer Science in Cars Symposium (CSCS '19). Association for Computing Machinery, New York, NY, USA, Article 6, 1–9, 2019. DOI: 10.1145/3359999.3360492.

[23] P. Bajpai, R. Enbody, B.H.C. Cheng, Ransomware Targeting Automobiles, Proceedings of the Second ACM Workshop on Automotive and Aerial Vehicle Security (AutoSec '20). Association for Computing Machinery, New York, NY, USA, 23–29, 2020. DOI: 10.1145/3375706.3380558.

[24] D. M. Nicol, The Ransomware Threat to Energy-Delivery Systems, IEEE Security & Privacy, vol. 19, no. 3, pp. 24-32, May-June 2021. DOI: 10.1109/MSEC.2021.3063678.

[25] Y. Su, B. Ahn, S. R. B. Alvee, T. Kim, J. Choi, S. C. Smith, Ransomware Security Threat Modeling for Photovoltaic Systems, 2021 6th IEEE Workshop on the Electronic Grid (eGRID), 2021, pp. 01-05. DOI: 10.1109/eGRID52793.2021.9662163.

[26] R. Richardson, M. North, Ransomware: Evolution, Mitigation and Prevention, Kennesaw State University, 4276, 2017. Disponible en `https://digitalcommons.kennesaw.edu/facpubs/4276/`.

[27] I.A. Chesti, M. Humayun, N.U. Sama, N. Zaman, Evolution, Mitigation, and Prevention of Ransomware. 2020 2nd International Conference on Computer and Information Sciences (ICCIS), 1-6, 2020. DOI: 10.1109/ICCIS49240.2020.9257708.

[28] A. Zimba, M. Chishimba, Understanding the Evolution of Ransomware: Paradigm Shifts in Attack Structures, International Journal of Computer Network and Information Security(IJCNIS), Vol.11, No.1, pp.26-39, 2019.DOI: 10.5815/ijcnis.2019.01.03.

[29] K.A Wagner Ramsdell, K.E. Esbeck, Evolution of Ransomware, The Mitre Corporation, Jul. 2021. Disponible en `https://healthcyber.mitre.org/wp-content/uploads/2021/08/Ransomware-Paper-V2.pdf`.

[30] A.K. Maurya, N. Kumar, A. Agrawal, R. A. Khan, Ransomware: Evolution, Target and Safety Measures, International Journal of Computer Sciences and Engineering, Vol.6, Issue.1, pp.80-85, 2018. DOI: 10.26438/ijcse/v6i1.8085.

[31] Zscaler, CovidLock: Android Ransomware Walkthrough and Unlocking Routine, Marh 2020. Disponible en https://www.zscaler.com/blogs/security-research/covidlock-android-ransomware-walkthrough-and-unlocking-routine.

[32] Goliate, Hidden-tear. Disponible en https://github.com/goliate/hidden-tear.

[33] S. Ryu, Anatomy of Chaos Ransomware builder and its origin (feat. Open-source Hidden Tear ransomware), S2W Blog, Aug 2021. Disponible en https://medium.com/s2wblog/anatomy-of-chaos-ransomware-builder-and-its-origin-feat-open-source-hidden-tear-ransomware-ffd5937d005f.

[34] X. Zhang, X. Xiao, Thoughts on Vulnerability Security by Ransomware Virus, International Journal of Social Science and Education Research Volume 5 Issue 1, 2022. DOI: 0.6918/IJOSSER.202201_5(1).0019.

[35] TrendMicro, The State of Ransomware 2020's - Catch-22, TrenMicro, Feb. 2021. Disponible en https://www.trendmicro.com/vinfo/us/security/news/cybercrime-and-digital-threats/the-state-of-ransomware-2020-s-catch-22.

[36] Security, COVID-19 pandemic sparks 72 % ransonmware frowth, mobile vulnerabilities grow 50 %, Security Magazine, July 2020. Disponible en https://www.securitymagazine.com/articles/92886-covid-19-pandemic-sparks-72-ransomware-growth-mobile-vulnerabilities-grow-50.

[37] Europol, COVID-19: Ransomware, Dic. 2021. Disponible en https://www.europol.europa.eu/covid-19/covid-19-ransomware.

[38] Coveware, Ransomware Attack Vectors Shift as New Software Vulnerability Exploits Abound, April, 2021. Disponible en https://www.coveware.com/blog/ransomware-attack-vectors-shift-as-new-software-vulnerability-exploits-abound.

[39] McAfee, Informe de McAfee Labs sobre amenazas - COVID-19, Jul. 2020. Disponible en https://www.mcafee.com/enterprise/es-es/assets/reports/rp-quarterly-threats-july-2020.pdf.

[40] H.S. Lallie, L.A. Shepherd, J.R.C. Nurse, A. Erola, G. Epiphaniou, C. Maple, X. Bellekens, Cyber security in the age of COVID-19: A timeline and analysis of cyber-crime and cyber-attacks during the pandemic, Computers & Security, Volume 105, 102248, 2021. DOI: 10.1016/j.cose.2021.102248.

[41] S. Gatlan, Qlocker ransomware returns to target QNAP NAS devices worldwide, BleepingComputer, Jan. 2022. Disponible en `https://www.bleepingcomputer.com/news/security/qlocker-ransomware-returns-to-target-qnap-nas-devices-worldwide/`.

[42] M.J. Schwartz, Cybercrime Moves: Conti Ransomware Absorbs TrickBot Malware, Baank Info Security, Feb. 2022. Disponible en `https://www.bankinfosecurity.com/cybercrime-moves-conti-ransomware-absorbs-trickbot-malware-a-18573`.

[43] Sachiel, Analysis of "Heaven's Gate" part 1, Jan.2021. Disponible en `https://sachiel-archangel.medium.com/analysis-of-heavens-gate-part-1-62cca0ace6f0`.

[44] Lifars, A Deep Dive into The Grief Ransomware's Capabilities, Lifars, 2021. Disponible en `https://lifars.com/wp-content/uploads/2022/01/GriefRansomware_Whitepaper-2.pdf`.

[45] G. Varma, R. Chauhan, Cybercriminals Strike Where It Hurts Most: SARS-Cov-2 Pandemic and its Influence on Critical Infrastructure Ransomware Attacks, 2022 16th International Conference on Ubiquitous Information Management and Communication (IMCOM), 2022, pp. 1-7. DOI: 10.1109/IMCOM53663.2022.9721721.

[46] A. Din, The Full Source Code for the Babuk Ransomware Published on a Russian Hacker Forum, Heimdal Security, Sep. 2021. Disponible en `https://heimdalsecurity.com/blog/the-full-source-code-for-the-babuk-ransomware-published-on-a-russian-hacker-forum/`.

[47] D. Tudor, Babuk Focuses On Data-Theft Extortion, Heimdal Security, May 2020. Disponible en `https://heimdalsecurity.com/blog/babuk-focuses-on-data-theft-extortion/`.

[48] A. Raheem, R. Raheem, T.M. Chen, A. Alkhayyat, Estimation of Ransomware Payments in Bitcoin Ecosystem, 2021 IEEE Intl Conf on Parallel & Distributed Processing with Applications, Big Data & Cloud Computing, Sustainable Computing & Communications, Social Computing & Networking (ISPA/BDCloud/SocialCom/SustainCom), 2021, pp. 1667-1674. DOI: 10.1109/ISPA-BDCloud-SocialCom-SustainCom52081.2021.00224.

[49] Y.K. Bin Mohamed Yunus, S. Bin Ngah, Ransomware: stages, detection and evasion, 2021 International Conference on Software Engineering & Computer Systems and 4th International Conference on Computational Science and Information Management (ICSECS-ICOCSIM), 2021, pp. 227-231. DOI: 10.1109/ICSECS52883.2021.00048.

[50] P. H. Meland, Y. F. F. Bayoumy, G. Sindre, The Ransomware-as-a-Service economy within the darknet, Computers & Security, Volume 92, 2020, 101762. DOI: 10.1016/j.cose.2020.101762.

[51] C. Karapapas, I. Pittaras, N. Fotiou, G. C. Polyzos, Ransomware as a Service using Smart Contracts and IPFS, 2020 IEEE International Conference on Blockchain and Cryptocurrency (ICBC), 2020, pp. 1-5. DOI: 10.1109/ICBC48266.2020.9169451.

[52] A. Kapoor, A. Gupta, R. Gupta, S. Tanwar, G. Sharma, I.E. Davidson, Ransomware Detection, Avoidance, and Mitigation Scheme: A Review and Future Directions. Sustainability 2022, 14, 8. DOI: 10.3390/su14010008.

[53] VirusTotal, Ransomware in a Global Context, Oct. 2021. Disponible en `https://www.virustotal.com/go/ransomware-in-a-global-context-2021`.

[54] D. Farhat, M. S. Awan, A Brief Survey on Ransomware with the Perspective of Internet Security Threat Reports, 2021 9th International Symposium on Digital Forensics and Security (ISDFS), 2021, pp. 1-6. DOI: 10.1109/ISDFS52919.2021.9486348.

[55] C. Mehra, A.K. Sharma, A. Sharma, Elucidating Ransomware Attacks In Cyber-Security, International Journal of Innovative Technology and Exploring Engineering, Vol. 9(1), 2019. DOI: 10.35940/ijitee.A8106.119119.

[56] M.J. Haber, Privileged Attack Vectors, Apress, Berkeley, CA, 2020. DOI: 10.1007/978-1-4842-5914-6.

[57] M. Kerner, Ransomware trends, statistics and facts in 2021, TechTarger, Nov. 2021. On line `https://www.techtarget.com/searchsecurity/feature/Ransomware-trends-statistics-and-facts`.

[58] D. Blessman, Protecting Your Software Supply Chain, Risk Management, vol 66(1): 10-11, New York, Jan-Feb, 2019.

[59] NIST, CISA, Defending Against Software Supply Chain Attacks, Cybersecurity and Infrastructure Security Agency, TLP:WHITE, Apr. 2021. Disponible en `https://www.cisa.gov/sites/default/files/p`

ublications/defending_against_software_supply_chain_attacks_508.pdf.

[60] K. Panetta, The Top 8 Cybersecurity Predictions for 2021-2022, Garner, Oct. 2021. Disponible en https://www.gartner.com/en/articles/the-top-8-cybersecurity-predictions-for-2021-2022.

[61] M.J. Haber, C. Hills, B. Chappell, J. Maude, BeyondTrust Cybersecurity Trend Predictions for 2022 & Beyond, BeyondTrust, Oct. 2021. Disponible en BeyondTrustCybersecurityTrendPredictionsfor2022&Beyond.

[62] L. Vaas, Ransomware Payments Explode Amid 'Quadruple Extortion', ThreatPost, 12 agosto 2021. Disponible en https://threatpost.com/ransomware-payments-quadruple-extortion/168622/.

[63] Radware, 2021–2022 Global Threat Analysis Report, Radware Ltd., 2022. Disponible en https://www.radware.com/2021-2022-global-threat-analysis-report/.

[64] D. Goodin, Cybercriminals who breached Nvidia issue one of the most unusual demands ever, Ars Technica, Apr. 2020. Disponible en https://arstechnica.com/information-technology/2022/03/cybercriminals-who-breached-nvidia-issue-one-of-the-most-unusual-demands-ever/.

[65] K. Collier, Ransomware hackers' new tactic: Calling you directly, NBC News, Jan. 2022. Disponible en https://www.nbcnews.com/tech/security/ransomware-hackers-new-tactic-calling-directly-rcna6466.

[66] M. Loman, LockFile ransomware's box of tricks: intermittent encryption and evasion, Sophos News, Aug. 2021. Disponible en https://news.sophos.com/en-us/2021/08/27/lockfile-ransomwares-box-of-tricks-intermittent-encryption-and-evasion/.

[67] S. Dalal, Z. Wang, S. Sabharwal, Identifying Ransomware Actors in the Bitcoin Network, 2021, eprint 2108.13807, arXiv. Disponible en https://arxiv.org/abs/2108.13807.

[68] S21sec, Threat Landscape Report, Segundo semestre de 2021. Disponible en https://www.s21sec.com/es/threat-landscape-report-es/.

[69] CrownStrike, 2022 Global Threat Report, 2022. Disponible en https://go.crowdstrike.com/rs/281-OBQ-266/images/Report2022GTR.pdf.

[70] Ransomwhere. Disponible en https://ransomwhe.re/.

[71] Purplesec, 2021 Ransomware Statistics, Data & Trends, 2022. Disponible en https://purplesec.us/resources/cyber-security-statistics/ransomware/.

[72] M. Midler, Ransomware as a Service (RaaS) Threats. Software Engineering Institute, Carnegie Mellon Institute, Oct. 2020. Disponible en https://insights.sei.cmu.edu/blog/ransomware-as-a-service-raas-threats/.

[73] K. Buker, Ransomware as a Service (RaaS) Explained, Crowstrike, Feb. 2022. Disponible en https://www.crowdstrike.com/cybersecurity-101/ransomware/ransomware-as-a-service-raas/.

[74] F. Barr-Smith, X. Ugarte-Pedrero, M. Graziano, R. Spolaor, I. Martinovic, Survivalism: Systematic Analysis of Windows Malware Living-Off-The-Land, 2021 IEEE Symposium on Security and Privacy (SP), 2021, pp. 1557-1574. DOI: 10.1109/SP40001.2021.00047.

[75] I. Vakilinia, M.M. Khalili, M. Li, A Mechanism Design Approach to Solve Ransomware Dilemmas. En: Bošanský B., Gonzalez C., Rass S., Sinha A. (eds) Decision and Game Theory for Security. GameSec 2021. Lecture Notes in Computer Science, vol 13061. Springer, Cham., 2021. DOI: 10.1007/978-3-030-90370-1_10.

[76] Team Cymru, Research Group, Analyzing ransomware negotiations with CONTI: An in-depth analysis, Research Papers and Analysis, s.a., Disponible en https://team-cymru.com/wp-content/uploads/2021/10/Conti_Paper_1.pdf.

[77] R. Lakshmanan, Dridex Malware Deploying Entropy Ransomware on Hacked Computers, The Hacker News, Feb. 2022. Disponible en https://thehackernews.com/2022/02/dridex-malware-deploying-entropy.html.

[78] SOCRadar, What Is Ransomware-as-a-Service (RaaS)?, Nov. 2021. Disponible en https://socradar.io/what-is-ransomware-as-a-service-raas/.

[79] Insikt Group, New Ransomware-as-a-Service Tool 'Thanos' Shows Connections to 'Hakbit', Recorded Future, Jun. 2020. Disponible en https://www.recordedfuture.com/thanos-ransomware-builder/.

[80] M. de Jesús, D.O. Ladores, Chaos Ransomware: A Proof of Concept With Potentially Dangerous Applications, Trend Micro, Aug. 2021. Disponible en https://www.trendmicro.com/en_us/research/21/h/chaos-ransomware-a-dangerous-proof-of-concept.html.

[81] Foro Vx-Underground. Chaos Ransomware Builder. Disponible en https://www.vx-underground.org/archive.html#builders.

[82] CSW, Ransomware - Through the Lens of Threat and Vulnerability Management, CSW SecurityWorks, 2022 Spotlight Report, 2022. Disponble en https://cybersecurityworks.com/ransomware/.

[83] S. Halder, Ransomware as a Service (RaaS) & Its Implications in 2021. AppKonx, Sept. 2021. Disponible en https://www.appknox.com/blog/ransomware-as-a-service.

[84] Chainalysis, As Ransomware Payments Continue to Grow, So Too Does Ransomware's Role in Geopolitical Conflict, Feb. 2022. Disponible en https://blog.chainalysis.com/reports/2022-crypto-crime-report-preview-ransomware/.

[85] K. Wang, J. Pang, D. Chen, Y. Zhao, D. Huang, C. Chen, W. Han, A Large-scale Empirical Analysis of Ransomware Activities in Bitcoin. ACM Trans. Web 16, 2, Article 7, May 2022, 29 pages. DOI: 10.1145/3494557.

[86] Microsoft, Destructive malware targeting Ukranian organizations, Jan. 2022. Disponible en https://www.microsoft.com/security/blog/2022/01/15/destructive-malware-targeting-ukrainian-organizations/.

[87] Symantec, Ukraine: Disk-wiping Attacks Precede Russian Invasion, Symantec Threat Hunter Team, Feb. 2022. Disponible en https://symantec-enterprise-blogs.security.com/blogs/threat-intelligence/ukraine-wiper-malware-russia.

[88] L. Abrams, New data-wiping malware used in destructive attacks on Ukranie, BleeppingComputer, Dec. 2022. Disponible en https://www.bleepingcomputer.com/news/security/new-data-wiping-malware-used-in-destructive-attacks-on-ukraine/.

[89] C. del Fierro, J. Dwyer, IBM Security X-Force Research Advisory: New Destructive Malware Used In Cyber Attacks on Ukraine, Feb. 2022, Security Intellignece. Disponible en https://securityintelligence.com/posts/new-destructive-malware-cyber-attacks-ukraine/.

[90] L.Abrams, Conti Ransomware source code leaked by Ukrainian researcher, BleepingComputer, Mar. 2022. Disponible en https://www.bleepingcomputer.com/news/security/conti-ransomware-source-code-leaked-by-ukrainian-researcher/.

[91] Vx-Underground, Conti. Disponible en https://share.vx-underground.org/Conti/.

[92] T. Dargahi, A. Dehghantanha, P.N. Bahrami, M. Conti, G. Bianchi, A Cyber-Kill-Chain based taxonomy of crypto-ransomware features. J. Comput. Virol. Hack Tech 15, 277–305, 2019. DOI: 10.1007/s11416-019-00338-7.

[93] Q. K. A. Mirza, M. Brown, O. Halling, L. Shand, A. Alam, Ransomware Analysis using Cyber Kill Chain, 2021 8th International Conference on Future Internet of Things and Cloud (FiCloud), 2021, pp. 58-65, DOI: 10.1109/FiCloud49777.2021.00016.

[94] A. Tandon, A. Nayyar, A Comprehensive Survey on Ransomware Attack: A Growing Havoc Cyberthreat, Proceedings of ICDMAI 2018, Volume 2, 2019. DOI: 10.1007/978-981-13-1274-8_31.

[95] Lokheed Martin, The cyber kill chain. Disponible en `http://www.lockheedmartin.com/us/what-we-do/aerospace-defense/cyber/cyber-kill-chain.html`.

[96] S. Barnum, Standardizing cyber threat intelligence information with the structured threat information expression (stixTM). MITRE Corp. 11, 1–22 2012. Disponible en `https://www.mitre.org/publications/technical-papers/standardizing-cyber-threat-intelligence-information-with-the`.

[97] E.M. Hutchins, M.J. Cloppert, R.M. Amin, Intelligence-driven computer network defense informed by analysis of adversary campaigns and intrusion kill chains. Bethesda, MD: Lockheed Martin Corporation, 2010.

[98] D. González, T. Hayajneh, Detection and prevention of crypto-ransomware, 2017 IEEE 8th Annual Ubiquitous Computing, Electronics and Mobile Communication Conference (UEMCON), 2017, pp. 472-478. DOI: 10.1109/UEMCON.2017.8249052.

[99] D. Caivano, G. Canfora, A. Cocomazzi, A. Pirozzi, C. A. Visaggio, Ransomware at X-Rays, 2017 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData), 2017, pp. 348-353. DOI: 10.1109/iThings-GreenCom-CPSCom-SmartData.2017.58.

[100] A. Adamov, A. Carlsson, The state of ransomware. Trends and mitigation techniques, 2017 IEEE East-West Design & Test Symposium (EWDTS), 2017, pp. 1-8. DOI: 10.1109/EWDTS.2017.8110056.

[101] M. Anghel, A. Racautanu, A note on different types of ransomware attacks, Cryptology ePrint Archive, Report 2019/605, 2019, `https://ia.cr/2019/605`.

[102] M. Damshenas, A. Dehghantanha, R. Mahmoud, A survey on malware propagation, analysis, and detection. Int. J. Cyber-Secur. Digit. Forensics (IJCSDF) 2(4), 10–29, 2013.

[103] A.K. Sood, R.J. Enbody, Crimeware-as-a-service — A survey of commoditized crimeware in the underground market, International Journal of Critical Infrastructure Protection, Volume 6, Issue 1, 2013, Pages 28-38. DOI: 10.1016/j.ijcip.2013.01.002.

[104] A. Zimba, Z. Wang, Malware-Free Intrusions: Exploitation of Built-in Pre-Authentication Services for APT Attack Vectors, International Journal of Computer Network and Information Security (IJCNIS), Vol.9, No.7, pp.1-10, 2017. DOI: 10.5815/ijcnis.2017.07.01.

[105] D. Kiwia, A. Dehghantanha, K. Choo, J. Slaughter, A Cyber Kill Chain Based Taxonomy of Banking Trojans for Evolutionary Computational Intelligence. J. Comput. Sci., 27, 394-409, 2018. DOI: 10.1016/j.jocs.2017.10.020.

[106] Crowstrike, How Ransomare Uses PowerShell, infografía, 2017. Disponible en `https://www.crowdstrike.com/wp-content/uploads/2017/10/crowdstrike-ransomware-infographic.pdf`.

[107] Z.A. Genç, G. Lenzini, P.Y.A. Ryan, Next Generation Cryptographic Ransomware. En: Gruschka N. (eds) Secure IT Systems. NordSec 2018. Lecture Notes in Computer Science, vol 11252. Springer, Cham., 2018. DOI: 10.1007/978-3-030-03638-6_24.

[108] M. N. Olaimat, M. Aizaini Maarof, B. A. S. Al-rimy, Ransomware Anti-Analysis and Evasion Techniques: A Survey and Research Directions, 2021 3rd International Cyber Resilience Conference (CRC), 2021, pp. 1-6. DOI: 10.1109/CRC50527.2021.9392529.

[109] A. Afianian, S. Niksefat, B. Sadeghiyan, D. Baptiste, Malware Dynamic Analysis Evasion Techniques: A Survey, ACM Comput. Surv. 52, 6, Article 126 , November 2020, 28 pages. DOI: 10.1145/3365001.

[110] C.S. Veerappan, P.L.K. Keong, Z. Tang, F. Tan, Taxonomy on malware evasion countermeasures techniques, 2018 IEEE 4th World Forum on Internet of Things (WF-IoT), 2018, pp. 558-563. DOI: 10.1109/WF-IoT.2018.8355202.

[111] J. Boutsikas, M.E. Eren, C. Varga, E. Raff, C. Matuszek, C. Nicholas, Evading malware classifiers via monte carlo mutant feature discovery, arXiv preprint arXiv:2106.07860, 2021.

[112] C. Wilson, Forensic Analysis of the *Zone.Identifier* Stream, Digital Forensic. Blog, Oct. 2021. Disponible en `https://www.digital-detective.net/forensic-analysis-of-zone-identifier-stream/`.

[113] A. Alsabeh, H. Safa, E. Bou-Harb, J. Crichigno, Exploiting Ransomware Paranoia For Execution Prevention, ICC 2020 - 2020 IEEE International Conference on Communications (ICC), 2020, pp. 1-6. DOI: 10.1109/ICC40277.2020.9149005.

[114] T. Apostolopoulos, V. Katos, K.R. Choo, C. Patsakis, Resurrecting anti-virtualization and anti-debugging: Unhooking your hooks, Future Generation Computer Systems, Volume 116, Pages 393-405, 2021. DOI: 10.1016/j.future.2020.11.004.

[115] J. Chatsomsanga, C. Benjangkaprasert, Malware Developing Guide: Encryption and Decryption, 2022 24th International Conference on Advanced Communication Technology (ICACT), 2022, pp. 275-278. DOI: 10.23919/ICACT53585.2022.9728944.

[116] A. Ghafarian, D. Keskin, G. Helton, An Assessment of Obfuscated Ransomware Detection and Prevention Methods. En: K. Arai (eds) Advances in Information and Communication. FICC 2021. Advances in Intelligent Systems and Computing, vol 1363. Springer, Cham, 2021. DOI: 10.1007/978-3-030-73100-7_56.

[117] A. Khraisat, I. Gondal, P. Vamplew, J. Kamruzzaman, Survey of intrusion detection systems: techniques, datasets and challenges. Cybersecur 2, 20, 2019. DOI: 10.1186/s42400-019-0038-7.

[118] H. Kılıç, N. S. Katal, A. A. Selçuk, Evasion Techniques Efficiency Over The IPS/IDS Technology, 2019 4th International Conference on Computer Science and Engineering (UBMK), pp. 542-547, 2019. DOI: 10.1109/UBMK.2019.8907177.

[119] N.A. Hassan, Ransomware Distribution Methods, Ransomware Revealed, Apress, Berkeley, CA., 2019. DOI: 10.1007/978-1-4842-4255-1_2

[120] K. Gangwar, S. Mohanty, A.K. Mohapatra, Analysis and Detection of Ransomware Through Its Delivery Methods. En: B. Panda, S. Sharma, N. Roy (eds) Data Science and Analytics, REDSET 2017, Communications in Computer and Information Science, vol 799. Springer, Singapore, 2018. DOI: 10.1007/978-981-10-8527-7_29.

[121] F. Salahdine, N. Kaabouch, Social Engineering Attacks: A Survey, Future Internet 2019, 11, 89. DOI: 10.3390/fi11040089.

[122] Z. Alkhalil, C. Hewage, L. Nawaf, I. Khan, Phishing Attacks: A Recent Comprehensive Study and a New Anatomy, Front. Comput. Sci., 09 March, 2021. DOI: /10.3389/fcomp.2021.563060.

[123] Veeam, 2021 Ransomware Retrospective, Veeam Software, 2021. Disponible en https://www.veeam.com/2021-ransomware-retrospective.html.

[124] G.Q. He, C. Liu, A. Huang, R. Lu, Ransomware Families: 2021 Data to Supplement the Unit42 Ransomware Threat Report, Unit42, Jul. 2021. Disponible en https://unit42.paloaltonetworks.com/ransomware-families/.

[125] Trend Micro, Exploit kit. Disponible en https://www.trendmicro.com/vinfo/us/security/definition/exploit-kit.

[126] C.P.R. Team: Inside nuclear's core: analyzing the nuclear exploit kit infrastructure — part I, Check Point, 2016). Disponible en https://blog.checkpoint.com/wp-content/uploads/2016/04/Inside-Nuclear-1-2.pdf.

[127] E. Suren, P. Angin, Know Your EK: A Content and Workflow Analysis Approach for Exploit Kits, Journal of Internet Services and Information Security (JISIS), volume: 9, number: 1, pp. 24-47, February, 2019. Disponible en http://isyou.info/jisis/vol9/no1/jisis-2019-vol9-no1-02.pdf.

[128] Microsoft, Exploits and Exploit Kits, Microsoft Documentation, Oct. 2021. Disponible en https://docs.microsoft.com/da-dk/windows/security/threat-protection/intelligence/exploits-malware.

[129] A. Liska, T. Gallo, Threat Intelligence and ransomware, O'Reilly, Dec. 2016. Disponible en https://www.oreilly.com/content/threat-intelligence-and-ransomware/.

[130] Trend Micro, New Exploit Kit Fallout Delivering Gandcrab Ransomware, Sept. 2018. Disponible en https://www.trendmicro.com/vinfo/es/security/news/cybercrime-and-digital-threats/new-exploit-kit-fallout-delivering-gandcrab-ransomware.

[131] Microsfot, Critical Security Update for Microsoft Windows SMB Server (4013389), Microsoft Security Bulletin MS17-010. Disponible en https://docs.microsoft.com/en-us/security-updates/securitybulletins/2017/ms17-010.

[132] J. Segura, Spelevo exploit kit debuts new social engineering trick, Dec. 2019. Disponible en https://blog.malwarebytes.com/threat-analy

sis/2019/12/spelevo-exploit-kit-debuts-new-social-engineering-trick/.

[133] G. Lionel, PetitPotam PoC. Disponible en https://github.com/topotam/PetitPotam.

[134] Kaspersky, The era of targeted ransomware: attacks on high-profile victims grows nearly eightfold from 2019 to 2020, April, 2021. Disponible en https://www.kaspersky.com/about/press-releases/2021_the-era-of-targeted-ransomware-attacks-on-high-profile-victims-grows-nearly-eightfold-from-2019-to-2020.

[135] K. Van Impe, How Attackers Exploit the Remote Desktop Protocol, Security Intelligence, Nov. 2021. Disponible en https://securityintelligence.com/articles/exploiting-remote-desktop-protocol/.

[136] V. Stocchetti (Ed.), Exploited Protocols: Server Message Block (SMB), Center for Internet Security (CSI), 2021. Disponible en https://learn.cisecurity.org/CIS_Controls_v8_Exploited_Protocols_Server_Message_Block_SMB.

[137] C. Cimpanu, Top exploits used by ransomware gangs are VPN bugs, but RDP still reigns supreme, ZDNet, Agu. 2020. Disponible en https://www.zdnet.com/article/top-exploits-used-by-ransomware-gangs-are-vpn-bugs-but-rdp-still-reigns-supreme/.

[138] E.C. Ogu, O.A. Ojesanmi, O. Awodele, S. Kuyoro, A Botnets Circumspection: The Current Threat Landscape, and What We Know So Far, Information, 10(11):337, 2019. DOI: 10.3390/info10110337.

[139] A. Zimba, M. Chishimba, S. Chihana, A Ransomware Classification Framework Based on File-Deletion and File-Encryption Attack Structures, arXiv, 2021. Disponible en https://arxiv.org/abs/2102.10632.

[140] DFIR, Exchange Exploit Leads to Domain Wide Ransomware, The DFIR Report, Nov. 2021. Disponible en https://thedfirreport.com/2021/11/15/exchange-exploit-leads-to-domain-wide-ransomware/.

[141] M. Logan, E. Mendoza, R.Maglaque, N. Tamaña, The State of Ransomware 2020's Catch-22, Trend Micro, Feb. 2021. Disponible en https://www.trendmicro.com/vinfo/us/security/news/cybercrime-and-digital-threats/the-state-of-ransomware-2020-s-catch-22.

[142] M. Conti, T. Dargahi, A. Dehghantanha, Cyber Threat Intelligence: Challenges and Opportunities. En: A. Dehghantanha, M. Conti, T. Dargahi (eds), Cyber Threat Intelligence, Advances in Information Security, vol 70. Springer, Cham., 2018, DOI: 10.1007/978-3-319-73951-9_1.

[143] C. Bansal, P. Deligiannis, C. Maddila, N. Rao, Studying Ransomware Attacks Using Web Search Logs, Proceedings of the 43rd International ACM SIGIR Conference on Research and Development. En Information Retrieval. Association for Computing Machinery, New York, NY, USA, 1517–1520, 2020. DOI: 10.1145/3397271.3401189.

[144] N. Ariffin, A. Zainal, M. A. Maarof, M. Nizam Kassim, A Conceptual Scheme for Ransomware Background Knowledge Construction, 2018 Cyber Resilience Conference (CRC), 2018, pp. 1-4. DOI: 10.1109/CR.2018.8626868.

[145] A. Kamal, M. Derbali, S. Jan, J.I. Bangash, F.Q. Khan, H. Jerbi, R. Abbassi, G. Ahmad, A User-friendly Model for Ransomware Analysis Using Sandboxing, Computers, Materials & Continua, 67(3). DOI: 10.32604/cmc.2021.015941.

[146] J. Haile, S. Havens, Identifying Ubiquitious Third-Party Libraries in Compiled Executables Using Annotated and Translated Disassembled Code with Supervised Machine Learning, 2020 IEEE Security and Privacy Workshops (SPW), 2020, pp. 157-162. DOI: 10.1109/SPW50608.2020.00042.

[147] R. Cooley, M. Cutshaw, S. Wolf, R. Foster, J. Haile and M. Borowczak, Comparing Ransomware using TLSH and @DisCo Analysis Frameworks, 2021 IEEE International Conference on Big Data (Big Data), 2021, pp. 2084-2091. DOI: 10.1109/BigData52589.2021.9671573.

[148] A. Cuzzocrea, F. Mercaldo, F. Martinelli, A Framework for Supporting Ransomware Detection and Prevention Based on Hybrid Analysis. En: O. Gervasi *et al.* (eds), Computational Science and Its Applications – ICCSA 2021. Lecture Notes in Computer Science, vol 12951. Springer, Cham., 2021. DOI: 10.1007/978-3-030-86970-0_2.

[149] K. P. Subedi, D.R. Budhathoki, D. Dasgupta, Forensic Analysis of Ransomware Families Using Static and Dynamic Analysis, 2018 IEEE Security and Privacy Workshops (SPW), 2018, pp. 180-185. DOI: 10.1109/SPW.2018.00033.

[150] N. Naik, P. Jenkins, N. Savage, L. Yang, Cyberthreat Hunting - Part 1: Triaging Ransomware using Fuzzy Hashing, Import Hashing and YARA Rules, 2019 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE), 2019, pp. 1-6. DOI: 10.1109/FUZZ-IEEE.2019.8858803.

[151] N. Naik, P. Jenkins, N. Savage, L. Yang, Cyberthreat Hunting - Part 2: Tracking Ransomware Threat Actors using Fuzzy Hashing and Fuzzy C-Means Clustering, 2019 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE), 2019, pp. 1-6. DOI: 10.1109/FUZZ-IEEE.2019.8858825.

[152] M. A. Aboud, K. Mariyappn, Investigation of Modern Ransomware Key Generation Methods: A Review, 2021 International Conference on Computer Communication and Informatics (ICCCI), 2021, pp. 1-5. DOI: 10.1109/ICCCI50826.2021.9402680.

[153] F. Cicala, E. Bertino, Analysis of Encryption Key Generation in Modern Crypto Ransomware, en IEEE Transactions on Dependable and Secure Computing. DOI: 10.1109/TDSC.2020.3005976.

[154] P. Bajpai, R. Enbody, An Empirical Study of Key Generation in Cryptographic Ransomware, International Conference on Cyber Security and Protection of Digital Services (Cyber Security), pp. 1-8, 2020. DOI 10.1109/CyberSecurity49315.2020.9138878.

[155] P. Bajpai, A. K. Sood, R. Enbody, A key-management-based taxonomy for ransomware, 2018 APWG Symposium on Electronic Crime Research (eCrime), 2018, pp. 1-12, DOI: 10.1109/ECRIME.2018.8376213.

[156] Fortinet, The Ins and Outs of the Ransomware: How to Mitigate Email-based Attacks, Fortinet White Paper, 2019.

[157] J. Yuste, S. Pastrana, Avaddon ransomware: An in-depth analysis and decryption of infected systems, Computers & Security, Volume 109, 102388,2021. DOI: 10.1016/j.cose.2021.102388.

[158] H.C. Yüceel, TTPs used by BlackByte Ransomware Targeting Critical Infrastructure, Pycus Security, Feb. 2022. Disponible en `https://www.picussecurity.com/resource/ttps-used-by-blackbyte-ransomware-targeting-critical-infrastructure`.

[159] R. Mendrez, BlackByte Ransomware – Pt. 1 In-depth Analysis, Trustwave, Oct. 2021. Disponible en `https://www.trustwave.com/en-us/resources/blogs/spiderlabs-blog/blackbyte-ransomware-pt-1-in-depth-analysis/`.

[160] J. Hill, ALPHV (BlackCat) Ransomware, Inside Out Security, Jan. 2022. Disponible en `https://www.varonis.com/blog/alphv-blackcat-ransomware`.

[161] A. Tanner, A. Hinchliffe, Threat Assessment: BlackCat Ransomware, Palo Alto Network, Jan. 2022. Disponible en `https://unit42.paloaltonetworks.com/blackcat-ransomware/`.

[162] I. Kara, M. Aydos, Static and Dynamic Analysis of Third Generation Cerber Ransomware, 2018 International Congress on Big Data, Deep Learning and Fighting Cyber Terrorism (IBIGDELFT), 2018, pp. 12-17. DOI: 10.1109/IBIGDELFT.2018.8625353.

[163] S. Pletinckx, C. Trap, C. Doerr, Malware Coordination using the Blockchain: An Analysis of the Cerber Ransomware, 2018 IEEE Conference on Communications and Network Security (CNS), 2018, pp. 1-9. DOI: 10.1109/CNS.2018.8433199.

[164] A. Kurniawan, I. Riadi, Detection and Analysis Cerber Ransomware Based on Network Forensics Behavior. International Journal of Network Security. 20, 2018. DOI: 10.6633/IJNS.201809.

[165] Ace, The Ins and Outs Of CryptoLocker Malware, Ace Cloud Hosting Editor, Apr. 2017. Disponible en `https://www.acecloudhosting.com/blog/ins-and-outs-of-cryptolocker-malware/`.

[166] K. Cabaj, P. Gawkowski, K. Grochowski, D. Osojca, Network activity analysis of CryptoWall ransomware. Przeglad Elektrotechniczny, 91(11), 201-204, 2015.

[167] K. Cabaj, W. Mazurczyk, Using software-defined networking for ransomware mitigation: the case of cryptowall, IEEE Netw, 30(6):14–20, 2016. DOI: 10.1109/MNET.2016.1600110NM.

[168] A. Pillai, R. Kadikar, M. S. Vasanthi, B. Amutha, Analysis of AES-CBC Encryption for Interpreting Crypto-Wall Ransomware, 2018 International Conference on Communication and Signal Processing (ICCSP), 2018, pp. 0599-0604. DOI: 10.1109/ICCSP.2018.8524494.

[169] R. Upadhyaya, A. Jain, Cyber ethics and cyber crime: A deep dwelved study into legality, ransomware, underground web and bitcoin wallet, 2016 International Conference on Computing, Communication and Automation (ICCCA), 2016, pp. 143-148. DOI: 10.1109/CCAA.2016.7813706.

[170] J. Wyke, A. Ajjan, The Current State of Ransomware, Sophos, 2015. Disponible en `https://www.sophos.com/en-us/medialibrary/PDFs/technical%20papers/sophos-current-state-of-ransomware.pdf`.

[171] M. Weckstén, J. Frick, A. Sjöström, E. Järpe, A novel method for recovery from Crypto Ransomware infections, 2016 2nd IEEE International Conference on Computer and Communications (ICCC), 2016, pp. 1354-1358, doi: 10.1109/CompComm.2016.7924925.

[172] DFIR, Diavol Ransomware, The DFIR Report, Dec. 2021. Disponible en https://thedfirreport.com/2021/12/13/diavol-ransomware/.

[173] D. Neemani, A. Rubinfeld, Diavol - A New Ransomware Used By Wizard Spider?, Forninet, Jul. 2021. Disponible en https://www.fortinet.com/blog/threat-research/diavol-new-ransomware-used-by-wizard-spider.

[174] L. Abrams, DMA Locker Ransomware targets Unmapped Network Shares, Bleepingcomputer, Feb. 2016. Disponible en https://www.bleepingcomputer.com/news/security/dma-locker-ransomware-targets-unmapped-network-shares/.

[175] Hasherezade, DMA Locker 4.0: Known ransomware preparing for a massive distribution, Malwarebytes Lab, May 2016. Disponible en https://blog.malwarebytes.com/threat-analysis/2016/05/dma-locker-4-0-known-ransomware-preparing-for-a-massive-distribution/.

[176] V.R. Paşca, E. Simion, Challenges in Cyber Security: Ransomware Phenomenon. En: Koç Ç.K. (eds) Cyber-Physical Systems Security, Springer, Cham, 2018. DOI:10.1007/978-3-319-98935-8_15.

[177] D. Masson, What the EKANS ransomware attack reveals about the future of OT cyber-attacks, Darktrace Blog, Jun. 2020. Disponible en https://www.darktrace.com/en/blog/what-the-ekans-ransomware-attack-reveals-about-the-future-of-ot-cyber-attacks/.

[178] A. Bradt, Dridex bots deliver Entropy ransomware in recent attacks, Sophos News, Feb. 2022. Disponible en https://news.sophos.com/en-us/2022/02/23/dridex-bots-deliver-entropy-ransomware-in-recent-attacks/.

[179] G. Palazolo, F. Duarte, Reverse Engineering Dridex and Automating IOC Extraction, Appgate, Sep. 2020. Disponible en https://www.appgate.com/blog/reverse-engineering-dridex-and-automating-ioc-extraction.

[180] R. de la Paz, FAKBEN Team Ransomware Uses Open Source "Hidden Tear" Code, Fortinet, Nov. 2015. https://www.fortinet.com/blog/threat-research/fakben-team-ransomware-uses-open-source-hidden-tear-code.

[181] VinRansomware, Fakben ransomware, VinRansomware. Disponible en http://www.vinransomware.com/fakben-ransomware.

[182] V.C. Craciun, A. Mogage, E. Simion, Trends in Design of Ransomware Viruses. En: J.L. Lanet, C. Toma (eds) Innovative Security Solutions for Information Technology and Communications. SECITC 2018. Lecture Notes in Computer Science, vol 11359, 2019. Springer, Cham. DOI:10.1007/978-3-030-12942-2_20.

[183] Huawei, Analysis of CASE A Latest Variant of the GlobeImposter Family, Hwawei, Feb. 2019. Disponible en https://isecurity.huawei.com/sec/web/viewBlog.do?id=1980.

[184] X. Zhang, Analysis of New GlobeImposter Ransomware Variant, Fortinet, Aug. 2017. Disponible en https://www.fortinet.com/blog/threat-research/analysis-of-new-globeimposter-ransomware-variant.

[185] CCN-CERT, Hive ransomware, CCN-CERT ID-15/21, Dic. 2021. Disponible en https://www.ccn-cert.cni.es/informes/informes-ccn-cert-publicos/6326-ccn-cert-id-15-21-hive-ransomware-1/file.html.

[186] M. Muir, Analysis of Novel Khonsari Ransomware Deployed by the Log4Shell Vulnerability, Cado Security, Dec. 2021. Disponible en https://www.cadosecurity.com/analysis-of-novel-khonsari-ransomware-deployed-by-the-log4shell-vulnerability/.

[187] A. Unnikrishnan, Technical Analysis of Khonsari Ransomware Campaign Exploiting the Log4Sehll Vulnerability, s.a. Disponible en https://cloudsek.com/technical-analysis-of-khonsari-ransomware-campaign-exploiting-the-log4shell-vulnerability/.

[188] J. MacRae, V.N.L. Franqueira, On Locky Ransomware, Al Capone and Brexit. En: P. Matoušek, M. Schmiedecker (eds) Digital Forensics and Cyber Crime. ICDF2C 2017. Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, vol 216, 2018. Springer, Cham. DOI: 10.1007/978-3-319-73697-6_3.

[189] R. Broadhurst, H. Trivedi, Malware in spam email: Risks and trends in the Australian spam intelligence database, Trends and Issues in Crime and Criminal Justice, (603), 1–18, 2020. DOI: 10.3316/agispt.20201215041188.

[190] Avast, A closer look at the locky ransomware, Avast, Mar. 2016. Disponible en https://blog.avast.com/a-closer-look-at-the-locky-ransomware.

[191] Acenture, Technical Analysis of Megacortex Vesion 2 Ransomware, Acenture Security Cyber Advisory, 2019. Disponible en https://www.accenture.com/_acnmedia/pdf-106/accenture-technical-analysis-megacortex.pdf.

[192] L. Abrams, Padcrypt: The first ransomware with live support chat and an uninstaller, Bleepingcomputer, Feb. 2016. Disponible en https://www.bleepingcomputer.com/news/security/padcrypt-the-first-ransomware-with-live-support-chat-and-an-uninstaller/

[193] GoldSparrow, Paycrypt ransomware description. EnigmaSoft, 2016. Disponible en https://www.enigmasoftware.com/paycryptransomware-removal/.

[194] J.S. Aidan, H.K. Verma, L.K. Awasthi, Comprehensive Survey on Petya Ransomware Attack, 2017 International Conference on Next Generation Computing and Information Systems (ICNGCIS), 2017, pp. 122-125. DOI: 10.1109/ICNGCIS.2017.30.

[195] A. Bezverky, Petya.A / NotPetya is an AI-powered cyber weapon, TTPs lead to Sandworm APT group, SOCPrime, Jul. 2017. Disponible en https://socprime.com/blog/petya-a-notpetya-is-an-ai-powered-cyber-weapon-ttps-lead-to-sandworm-apt-group/.

[196] R. Alvarez, Key Differences Between Petya and NotPetya, Fortinet, Jul. 2017. Disponible en https://www.fortinet.com/blog/threat-research/key-differences-between-petya-and-notpetya.

[197] S. Gupta, Kaseya VSA Downed by REvil in Monumental Supply-Chain Attack, CSOnline, Jul. 2021. Disponible en https://www.csoonline.com/article/3236183/what-is-ransomware-how-it-works-and-how-to-remove-it.html.

[198] A. Elshinbary, Deep Analysis of Ryuk Ransomware, GitHub, May 2020. Disponible en https://n1ght-w0lf.github.io/malware%20analysis/ryuk-ransomware/.

[199] B. Mason, Ryuk Malware – Analysis and Reverse Engineering, Ben's ideas and projects Blog, April 2020. Disponible en https://ben.the-collective.net/2020/04/08/ryuk-malware-analysis-and-reverse-engineering/#Initial_discovery.

[200] Avertium, An In-Depth Look at Ransomware Gang, Sabbath, Jan. 2002. Disponible en https://www.avertium.com/resources/threat-reports/in-depth-look-at-sabbath-ransomware-gang.

[201] Malwarebytes, Explained: Sage ransomware, Malwarebytes Laba, Mar. 2019. Disponible enhttps://blog.malwarebytes.com/threat-analysis/2017/03/explained-sage-ransomware/

[202] F. Bacurio, W. Low, J. de Manuel, Evasive Sage 2.2 Ransomware Variant Targets More Countries, Fortinet, Oct. 2017. Disponible en https://www.fortinet.com/blog/threat-research/evasive-sage-2-2-ransomware-variant-targets-more-countries.

[203] J. Jedynak, Sage 2.0 analysis, Cert.pl, Feb. 2017. Disponible en https://cert.pl/en/posts/2017/02/sage-2-0-analysis/.

[204] I. Arghire, Sage Ransomware Gets Anti-Analysis Capabilities, SecurityWeek, Oct. 2017. Disponible en https://www.securityweek.com/sage-ransomware-gets-anti-analysis-capabilities

[205] McAfee, McAfee ATR Analyzes Sodinokibi aka REvil Ransomware-as-a-Service – What The Code Tells Us, McAfee, Oct. 2019. Disponible en https://www.mcafee.com/blogs/other-blogs/mcafee-labs/mcafee-atr-analyzes-sodinokibi-aka-revil-ransomware-as-a-service-what-the-code-tells-us/.

[206] A. Palisse, H. Le Bouder, J.L. Lanet, C. Le Guernic, A. Legay, Ransomware and the Legacy Crypto API. En: F. Cuppens, N. Cuppens, J.L. Lanet, A. Legay (eds) Risks and Security of Internet and Systems, CRiSIS 2016, Lecture Notes in Computer Science, vol 10158. Springer, Cham, 2017. DOI: 10.1007/978-3-319-54876-0_2.

[207] S. Homayoun, A. Dehghantanha, M. Ahmadzadeh, S. Hashemi, R. Khayami, Know Abnormal, Find Evil: Frequent Pattern Mining for Ransomware Threat Hunting and Intelligence, IEEE Transactions on Emerging Topics in Computing, vol. 8, no. 2, pp. 341-351, 1 April-June 2020. DOI: 10.1109/TETC.2017.2756908.

[208] Y. Lemmou, E. M. Souidi, Infection, Self-reproduction and Overinfection in Ransomware: The Case of TeslaCrypt, 2018 International Conference on Cyber Security and Protection of Digital Services (Cyber Security), 2018, pp. 1-8. DOI: 10.1109/CyberSecPODS.2018.8560670

[209] A.B. Shushan, N. Lifshitz, A. Kushnir, M. Korman, B. Wasserman, Lazarus Group's Mata Framework Leveraged To Deploy TFlower Ransomware, Sygnia, Aug. 2021. Disponible en https://blog.sygnia.co/lazarus-groups-mata-framework-leveraged-to-deploy-tflower-ransomware.

[210] Hybrid Analysis, tflower.exe, Oct. 2019. Disponible en https://hybrid-analysis.com/sample/7ca3494c165647424222f80b8b61a9fb80ff695c2be77a9fb6a0a352f5df3140?environmentId=120.

[211] M.E.M. Léveillé, TorrentLocker: Ransomware en un país cercano, ESET, 2014. Disponible en `https://www.welivesecurity.com/wp-content/uploads/2015/01/TorrentLocker.pdf`.

[212] D. Kao, S. Hsiao, The dynamic analysis of WannaCry ransomware, 2018 20th International Conference on Advanced Communication Technology (ICACT), 2018, pp. 159-166. DOI: 10.23919/ICACT.2018.8323682.

[213] S. Hsiao, D. Kao, The static analysis of WannaCry ransomware, 2018 20th International Conference on Advanced Communication Technology (ICACT), 2018, pp. 153-158. DOI: 10.23919/ICACT.2018.8323680.

[214] R. Ploszeik, P. Svec, P. Debnár, Analysis of Ecryption Schemes in Modern Ransomware, Rad HAZU, Matematičke znanosti, Vol. 25, 1-13, 2021. DOI: 10.21857/mnlqgc58gy.

[215] P. Bajpai, R. Enbody, Dissecting .net ransomware: key generation, encryption and operation, Network Security, 2020(2):8–14, 2020. DOI: 10.1016/S1353-4858(20)30020-9.

[216] B. Filiz, B. Arief, O. Cetin, J. Hernández-Castro, On the Effectiveness of Ransomware Decryption Tools, Computers & Security, Volume 111, 102469, 2021. DOI: 10.1016/j.cose.2021.102469.

[217] StopRansomWare. Disponible en `https://www.cisa.gov/stopransomware`.

[218] No More Ransomware. Disponible en `https://www.nomoreransom.org/`.

[219] ID Ransomware. Disponible en `https://id-ransomware.malwarehunterteam.com`.

[220] VirusTotal. Disponible en `https://www.virustotal.com/gui/home/upload`.

[221] N.A. Hassan, Capter 8: Ransomware Decryption Tools. En N.A. Hassan, Ransomware Revealed: A Beginner's Guide to Protecting and Recovering from Ransomware Attacks, APress, 2019. DOI: 10.1007/978-1-4842-4255-1_8.

[222] L. Grant, S. Parkinson, Identifying file interaction patterns in ransomware behaviour. En: S. Parkinson, A. Crampton, R. Hill (eds.), Guide to Vulnerability Analysis for Computer Networks and System. An Artificial Intelligent Approach, Computer Communication and Network Series, pp. 317–335. Springer, 2018.

[223] Gobal Moderator, Which file types do common ransomwares target?, Bleeping Computer Blog, Mar. 2016. Disponible en https://www.bleepingcomputer.com/forums/t/607369/which-file-types-do-common-ransomwares-target/.

[224] The White House, Joint Statement of the Ministers and Representatives from the Counter Ransomware Initiative Meeting, Oct. 2021. Disponible en https://www.whitehouse.gov/briefing-room/statements-releases/2021/10/14/joint-statement-of-the-ministers-and-representatives-from-the-counter-ransomware-initiative-meeting-october-2021/.

[225] CRI, Ransomware and Federal Law: Cybercrime and Cybersecurity, Congressional Research Service, Oct. 2021. Disponible en https://crsreports.congress.gov/product/pdf/R/R46932.

[226] P. Paganini, The Dutch government will not tolerate ransomware attacks that could threaten national security, it will use intelligence or military services to curb them, Security Affairs, Oct. 2021. Disponible en https://securityaffairs.co/wordpress/123113/security/the-netherlands-war-ransomware-operations.html.

[227] Jdsupra, Ransomware Legislation: A Window Into Tomorrow's Compliance Obligations, Feb, 2022. Disponible en https://www.jdsupra.com/legalnews/ransomware-legislation-a-window-into-9217421/.

[228] R. Iyengar, Why it's so difficult to bring ransomware attackers to justice, CNN Business, Jul. 2021. Disponible en https://edition.cnn.com/2021/07/08/tech/ransomware-attacks-prosecution-extradition/index.html.

[229] S.M. Kerner, Ransomware trends, statics and facts in 2020, TechTarget, Feb. 2022. Disponible en https://www.techtarget.com/searchsecurity/feature/Ransomware-trends-statistics-and-facts.

[230] Panda, 73 Ransomware Statistics Vital for Security in 2022, Mar. 2022. Disponible en https://www.pandasecurity.com/en/mediacenter/security/ransomware-statistics/.

[231] J. D. Klusnick, Understanding Ransomware Trajectory to Create an Informed Prediction, University Honors Theses. Paper 1111, 2021. University Honors Theses. Paper 1111. DOI: 10.15760/honors.1138.

[232] M. Laitine, S. Armstrong-Smith, Tacklin cybercrime and ransomware head-on: Disrupting criminla networks and protecting organization, Cyber Security, Vol. 5(3), pgs. 190-205, 2022.

[233] J. Pont, O. Abu Oun, C. Brierley B. Arief, J. Hernández-Castro, A Roadmap for Improving the Impact of Anti-ransomware Research. En: A. Askarov, R. Hansen, W. Rafnsson (eds), Secure IT Systems, NordSec 2019, Lecture Notes in Computer Science, vol 11875. Springer, Cham. DOI: 10.1007/978-3-030-35055-0_9.

[234] E. Berrueta, D. Morato, E. Magaña, M. Izal, A Survey on Detection Techniques for Cryptographic Ransomware, IEEE Access, vol. 7, pp. 144925-144944, 2019. DOI: 10.1109/ACCESS.2019.2945839.

[235] M. M. Ahmadian, H. R. Shahriari, S. M. Ghaffarian, Connection-monitor & connection-breaker: A novel approach for prevention and detection of high survivable ransomwares, 2015 12th International Iranian Society of Cryptology Conference on Information Security and Cryptology (ISCISC), 2015, pp. 79-84. DOI: 10.1109/ISCISC.2015.7387902.

[236] A. Alqahtani, F.T. Sheldon, A Survey of Crypto Ransomware Attack Detection Methodologies: An Evolving Outlock, Sensor: 22, 1837, 2022. DOI: 10.3390.s22051837.

[237] H. Oz, A. Aris, A. Levi, A.S. Uluagac, A Survey on Ransomware: Evolution, Taxonomy, and Defense Solutions, ACM Comput. Surv., Aceptado en Jan. 2022. DOI: 10.1145/3514229.

[238] T. McIntosh, A.S.M. Kayes, Y.P. Chen, A. Ng, P. Watters, Ransomware Mitigation in the Modern Era: A Comprehensive Review, Research Challenges, and Future Directions, ACM Comput. Surv. 54, 9, Article 197, December, 2022, 36 pages. DOI: 10.1145/3479393.

[239] A. Liska, T. Gallo, Ransomware: Defending Against Digital Extortion, O'Reilly, 2017.

[240] W.C. Barker, W. Fisher, K. Scarfone, M. Souppaya, NIST 8374 - Ransomware Risk Management: A Cybersecurity Framework Profile, NIST, Feb. 2022. DOI: 10.6028/NIST.IR.8374.

[241] CCN-CERT, Medidas de seguridad contra ransomware: CCN-CERT IA-11/18, Centro Criptológico Nacional, 2018. Disponible en `https://www.ccn-cert.cni.es/informes/informes-ccn-cert-publicos/2877-ccn-cert-ia-11-18-medidas-de-seguridad-contra-ransomware.html`.

[242] W. C. Barker, K. Scarfone, W. Fisher, M. Souppaya, Draft NISTIR 8374 - Cybersecurity Framework Profile for Ransomware Risk Management, NIST 2021. DOI: 10.6028/NIST.IR.8374-draft.

[243] CCN-CERT, BP-04 - Ransomware: Informe de buenas prácticas, Centro Criptológico Nacional, 2021. Disponible en https://www.ccn-cert.cni.es/informes/informes-ccn-cert-buenas-practicas-bp/2088-ccn-cert-bp-04-ransomware.html.

[244] S. Bradley, Ransomware, SANS Whitepapers, 2021. Disponible en https://www.sans.org/white-papers/37317/.

[245] Ekta, U. Bansal, A Review on Ransomware Attack, 2021 2nd International Conference on Secure Cyber Computing and Communications (ICSCCC), 2021, pp. 221-226. DOI: 10.1109/ICSCCC51823.2021.9478148.

[246] CIS, Ransomware Impacts and Defense Controls. Disponible en https://www.cisecurity.org/insights/blog/ransomware-impacts-and-defense-controls.

[247] CIS, Security Primer, Disponible en https://www.cisecurity.org/insights/white-papers/security-primer-ransomware.

[248] D. Barroso, Fighting Ransomware with Active Defense, Counter Craft. Disponible en https://www.countercraftsec.com/blog/post/fighting-ransomware-with-active-defense/.

[249] CCCS, Ransomware Playbook, Canadian Centre for Cyber Security, 2021. Disponible en https://cyber.gc.ca/sites/default/files/2021-12/itsm00099-ransomware-playbook-2021-final3-en.pdf.

[250] N. Sharma, R. Shanker, Analysis of Ransomware Attack and Their Countermeasures: A Review, 2022 International Conference on Electronics and Renewable Systems (ICEARS), 2022, pp. 1877-1883. DOI: 10.1109/ICEARS53579.2022.9751949.

[251] S. Midtrapanon, G. Wills, Linux patch management: With security assessment features, 4th International Conference on Internet of Things, Big Data and Security, IoTBDS 2019, Heraklion, Crete, Greece, pp. 270-277, 01-03 May 2019.

[252] W. Liu, Modeling Ransomware Spreading by a Dynamic Node-Level Method, IEEE Access, vol. 7, pp. 142224-142232, 2019. DOI: 10.1109/ACCESS.2019.2941021.

[253] A. Nair, The Why and How of adopting Zero Trust Model in Organizations, TechRxiv, 2021, Preprint. DOI: 10.36227/techrxiv.14184671.v1. Disponible en https://www.techrxiv.org/articles/preprint/The_Why_and_How_of_adopting_Zero_Trust_Model_in_Organizations/14184671/1.

[254] N. Atanassov, M.M. Chowdhury, Mobile Device Threat: Malware, 2021 IEEE International Conference on Electro Information Technology (EIT), pp. 007-013 2021. DOI: 10.1109/EIT51626.2021.9491845.

[255] E. Galinkin, Winning the Ransomware Lottery. En: B. Bošanský, C. González, S. Rass, A. Sinha (eds), Decision and Game Theory for Security. GameSec 2021. Lecture Notes in Computer Science, vol 13061. Springer, Cham. DOI: 10.1007/978-3-030-90370-1_11.

[256] CIS, Ransomware: The Data Exfiltración and Double Extortion Trends, Center for Internet Security . Disponible en `https://www.cisecurity.org/insights/blog/ransomware-the-data-exfiltration-and-double-extortion-trends`.

[257] L.Y. Connolly, D. Wall, The rise of crypto-ransomware in a changing cybercrime landscape: Taxonomising countermeasures, Computers & Security, Volume 87, 101568, 2019. DOI: 10.1016/j.cose.2019.101568.

[258] H. Torres-Calderón, M. Velásquez, D. Mauricio, Method for Designing Countermeasures for Crypto-Ransomware Based on the NIST CSF. En: M. Ben Ahmed, H.N.L. Teodorescu, T. Mazri, P. Subashini, A.A. Boudhir (eds), Networking, Intelligent Systems and Security, Smart Innovation, Systems and Technologies, vol 237. Springer, Singapore, 2022. DOI: 10.1007/978-981-16-3637-0_26.

[259] M. Suppaya, K. Scarfne, Guide to Malware Incident Prevention and Handling for Desktop and Laptops, NIST Special Publication 800-83, Rev, 1, 2013. Disponible en `https://nvlpubs.nist.gov/nistpubs/specialpublications/nist.sp.800-83r1.pdf`. DOI: 10.6028/NIST.SP.800-83r1.

[260] A. Cartwright, E. Cartwright, L. Xue, J. Hernandez-Castro, An investigation of individual willingness to pay ransomware, Journal of Financial Crime, Vol. ahead-of-print No. ahead-of-print., 2022. DOI: 10.1108/JFC-02-2022-0055.

[261] Department of the Treasury, Advisory on Potential Snaction Risk for Facilitating Ransomware Payments, Oct. 2020. Disponible en `https://home.treasury.gov/system/files/126/ofac_ransomware_advisory_10012020_1.pdf`.

[262] T. Ahnert, M. Brolley, D.A. Cimon, R. Riordan, Do You Know Where Your Data Sleeps at Night? Cyber Security and Ransomware in Financial Markets, SSRN, March 14, 2022. DOI: 10.2139/ssrn.4057505.

[263] M. Wade, Digital hostages: Leveraging ransomware attacks in cyberspace, Business Horizons, Volume 64, Issue 6, Pages 787-797, 2021. DOI: 10.1016/j.bushor.2021.07.014.

[264] E. Cartwright, J. Hernandez Castro, A. Cartwright, To pay or not: Game theoretic models of ransomware. Journal of Cybersecurity, 5(1), 2021. DOI: 10.1093/cybsec/tyz009.

[265] R. Fang, M. Xu, P. Zhao, Determination of Ransomware Payment based on Bayesian Game Models, Computers & Security, 102685, 2022. DOI: 10.1016/j.cose.2022.102685.

[266] T. Hofmann, How organisations can ethically negotiate ransomware payments, Network Security, Volume 2020, Issue 10, Pages 13-17, 2020. DOI: /10.1016/S1353-4858(20)30118-5.

[267] G. Morgan, Ethical Issues in Cybersecurity: Employing red teams, responding to ransomware attacks and attempting botnet takedowns. PhD thesis, Dublin City University, 2021. Disponible en `https://doras.dcu.ie/26226/`.

[268] CISOMAG, Paying Ransom Doubles the Cost of Ransomware Attack: Research, Ciso Mag, May, 2020. Disponible en `https://cisomag.eccouncil.org/paying-ransom-doubles-the-cost-of-ransomware-attack-research/`.

[269] P.H. Chen, R. Bodak, N.S. Gandhi, Ransomware Recovery and Imaging Operations: Lessons Learned and Planning Considerations. J Digit Imaging 34, 731–740, 2021. DOI:10.1007/s10278-021-00466-x.

[270] S.J. Mierzwa, J.J. Drylie, C. Ho, D. Bogdan, K. Watson, Ransomware Incident Preparations With Ethical Considerations and Command System Framework Proposal. Journal of Leadership, Accountability and Ethics, 19(2), 2022. DOI: 10.33423/jlae.v19i2.5112.

[271] S. Parkinson, Use of access control to minimise ransomware impact, Network Security, 2017(7):5–8, 2017. DOI: 10.1016/S1353-4858(17)30069-7.

[272] O. Ami, Y. Elovici, D. Hendler, Ransomware prevention using application authentication-based file access control. En: Proceedings of the 33rd Annual ACM Symposium on Applied Computing, p. 1610–19, 2018. DOI: 10.1145/3167132.3167304.

[273] H. Turaev, P. Zavarsky, B. Swar, Prevention of Ransomware Execution in Enterprise Environment on Windows OS: Assessment of Application Whitelisting Solutions, 2018 1st International Conference on Data Intelligence and Security (ICDIS), pp. 110-118, 2018. DOI: 10.1109/ICDIS.2018.00024.

[274] D. Kim, J. Lee, Blacklist vs. Whitelist-Based Ransomware Solutions, IEEE Consumer Electronics Magazine, vol. 9, no. 3, pp. 22-28, 1 May 2020. DOI: 10.1109/MCE.2019.2956192.

[275] T. McIntosh, P. Watters, A. Kayes, A. Ng, Y. Chen, Enforcing situation-aware access control to build malware-resilient file systems. Future Generation Computer Systems, 115:568–82, 2021. DOI:10.1016/j.future.2020.09.035.

[276] S. Lee, H.K. Kim, K. Kim, Ransomware protection using the moving target defense perspective, Computers & Electrical Engineering, Volume 78, Pages 288-299, 2019. DOI: 10.1016/j.compeleceng.2019.07.014.

[277] Sophos, La mejor protección para endpoints del mundo: Malware, Ransomware, Exploits, Virus, Feb. 2022. Disponible en `https://www.sophos.com/es-es/products/endpoint-antivirus`.

[278] Microsoft, Proteger carpetas importantes con acceso controlado a carpetas, Feb. 2022. Disponible en `https://docs.microsoft.com/es-es/microsoft-365/security/defender-endpoint/controlled-folders?view=o365-worldwide`.

[279] Microsoft, Novedades de Microsoft Defender for Identity, Feb. 2022. Disponible en `https://docs.microsoft.com/es-es/defender-for-identity/whats-new`.

[280] Z. A. Genç, G. Lenzini, P. Ryan, No random, no ransom: a key to stop cryptographic ransomware, International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment. Springer, p. 234–55, 2018. DOI: 10.1007/978-3-319-93411-2_11.

[281] J. Ahn, D. Park, C. Lee, D. Min, J. Lee, S. Park, Q. Chen, Y. Kim, KEY-SSD: Access-Control Drive to Protect Files from Ransomware Attacks, ArXiv, 2019. DOI: 10.48550/arXiv.1904.05012.

[282] A. S. Siddiqui, C. Lee, F. Saqib, Hardware based protection against malwares by PUF based access control mechanism, 2017 IEEE 60th International Midwest Symposium on Circuits and Systems (MWSCAS), pp. 1312-1315, 2017. DOI: 10.1109/MWSCAS.2017.8053172.

[283] M. Akbanov, V.G. Vassilakis, M.D. Logothetis, Ransomware detection and mitigation using software-defined networking: The case of WannaCry, Computers & Electrical Engineering, Volume 76, Pages 111-121, 2019. DOI: 10.1016/j.compeleceng.2019.03.012.

[284] J. Huang, J. Xu, X. Xing, P. Liu, M.K. Qureshi, Flashguard: Leveraging intrinsic flash properties to defend against encryption

ransomware, Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security; 2017. p. 2231–44. DOI:10.1145/3133956.3134035.

[285] J. Thomas, G. Galligher, Improving backup system evaluations in information security risk assessments to combat ransomware. Computer and Information Science, 11(1), 2018. Disponible en `https://ssrn.com/abstract=3095629`.

[286] D. Min, D. Park, J. Ahn, R. Walker, J. Lee, S. Park, Y. Kim, Amoeba: an autonomous backup and recovery ssd for ransomware attack defense, IEEE Comput. Archit. Lett., 17(2):245–8, 2018. DOI: 10.1109/LCA.2018.2883431.

[287] M. Baykara, B. Sekin, A novel approach to ransomware: Designing a safe zone system, 2018 6th International Symposium on Digital Forensic and Security (ISDFS), 2018, pp. 1-5. DOI: 10.1109/ISDFS.2018.8355317.

[288] A. Kharraz, E. Kirda, Redemption: Real-time protection against ransomware at end-hosts, International Symposium on Research in Attacks, Intrusions, and Defenses. Springer, pp. 98–119, 2017. DOI: 10.1007/978-3-319-66332-6_5.

[289] W. Lao, Z. Chen, B. Gao, J. Wang, Y. Tao, R. Zhang, RAP: RAnsomware Protection Scheme Based on Blockchain, 2022 2nd International Conference on Consumer Electronics and Computer Engineering (ICCECE), pp. 13-20, 2022. DOI: 10.1109/ICCECE54139.2022.9712682.

[290] S.J. Sánchez, Líneas de defensa ante ataques tipo «ransomware», Dell, Mar. 2021. Disponible en `https://www.dell.com/es-es/blog/lineas-de-defensa-ante-ataques-tipo-ransomware/`.

[291] S. Tafkov, Z. Minchev, Decentralized File Storage and Ransomware Protection, The 12th International Conference on Business Information Security (BISEC-2021), Belgrade, Serbia, 3rd December 2021. DOI: 10.13140/RG.2.2.36728.78087.

[292] DFIR, Stolen Images Campaign Ends in Conti Ransomware, The DFIR Report, Apr. 2022. Disponible en `https://thedfirreport.com/2022/04/04/stolen-images-campaign-ends-in-conti-ransomware/?s=09`.

[293] A. Bello, A. Maurushat, Technical and Behavioural Training and Awareness Solutions for Mitigating Ransomware Attacks. En: R. Silhavy (eds), Applied Informatics and Cybernetics in Intelligent Systems, CSOC 2020, Advances in Intelligent Systems and Computing, vol 1226. Springer, Cham., 2020. DOI:10.1007/978-3-030-51974-2_14.

[294] J. Thomas, Individual Cyber Security: Empowering Employees to Resist Spear Phishing to Prevent Identity Theft and Ransomware Attacks, International Journal of Business and Management, Vol. 13, 20218. DOI: 10.5539/ijbm.v13n6p1.

[295] J. Ophoff, M. Lakay, Mitigating the Ransomware Threat: A Protection Motivation Theory Approach. En: H. Venter, M. Loock, M. Coetzee, M. Eloff, J. Eloff (eds), Information Security, ISSA 2018, Communications in Computer and Information Science, vol 973. Springer, Cham., 2019. DOI: 10.1007/978-3-030-11407-7_12.

[296] M. Chung, Why employees matter in the fight against ransomware. Computer Fraud & Security 2019;2019(8):8–11. DOI: 10.1016/S1361-3723(19)30084-3.

[297] G. Hull, H. John, B. Arief, Ransomware deployment methods and analysis: views from a predictive model and human responses. Crime Sci 8, 2, 2019. DOI: 10.1186/s40163-019-0097-9

[298] G.N. Angafor, I. Yevseyeva, Y. He, Bridging the Cyber Security Skills Gap: Using Tabletop Exercises to Solve the CSSG Crisis. En: M. Ma, B. Fletcher, S. Göbel, J. Baalsrud Hauge, T. Marsh (eds), Serious Games, JCSG 2020. Lecture Notes in Computer Science, vol 12434. Springer, Cham, 2020. DOI: 10.1007/978-3-030-61814-8_10.

[299] M. D. Salunke, P.B. Kumbharkar, K. Pramod, A Proposed Methodology to Mitigate the Ransomware Attack, en Recent Trends in Intensive Computing M. Rajesh et al. (Eds.), vol 39, pgs. 16-21, IOS Press, 2021. DOI: 10.3233/APC210173.

[300] K. Sokolov, Ransomware Activity and Blockchain Congestion, Capital Markets: Market Efficiency eJournal, 2018, DOI: 10.2139/ssrn.3175986.

[301] A. Balachandar, A. Alsowdh, K. Arumugam, Design and Development of Future Estimate in Confronting Ransomware, Journal of Physics Conference Series, 2021. DOI: 10.1088/1742-6596/1717/1/012063.

[302] B.N. Chaithanya, S.H. Brahmananda, Detecting Ransomware Attacks Distribution Through Phishing URLs Using Machine Learning. En: S. Smys, R. Bestak, R. Palanisamy, I. Kotuliak (eds), Computer Networks and Inventive Communication Technologies, Lecture Notes on Data Engineering and Communications Technologies, vol 75. Springer, Singapore, 2022. DOI: 10.1007/978-981-16-3728-5_61.

[303] A. Pagán, K. Elleithy, A Multi-Layered Defense Approach to Safeguard Against Ransomware, 2021 IEEE 11th Annual Computing and Communication Workshop and Conference (CCWC), pp. 0942-0947, 2021. DOI: 10.1109/CCWC51732.2021.9375988.

[304] G. Margarov, E. Mitrofanova, Management of Ransomware Detection and Prevention in Multilevel Environmental Monitoring Information System. En: A. Sidorenko, H. Hahn (eds), Functional Nanostructures and Sensors for CBRN Defence and Environmental Safety and Security; NATO Science for Peace and Security Series C: Environmental Security. Springer, Dordrecht, 2020. DOI: 10.1007/978-94-024-1909-2_10.

[305] Y.L. Tiu, M.F. Zolkipli, Study on Prevention and Solution of Ransomware Attack. Journal of IT in Asia, 9(1), 133-139, 2021. DOI:10.33736/jita.3402.2021.

[306] Z. Manjezi, R.A. Botha, Preventing and Mitigating Ransomware. En: H. Venter, M. Loock M., M. Coetzee, M. Eloff, J. Eloff (eds), Information Security, ISSA 2018, Communications in Computer and Information Science, vol 973. Springer, Cham., 2019. DOI: 10.1007/978-3-030-11407-7_11.

[307] A. Singh, A.R. Ikuesan, H.S. Venter, Digital Forensic Readiness Framework for Ransomware Investigation. En: F. Breitinger,I. Baggili (eds), Digital Forensics and Cyber Crime, ICDF2C 2018, Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, vol 259. Springer, Cham., 2019. DOI:10.1007/978-3-030-05487-8_5.

[308] I. Nadir, T. Bakhshi, Contemporary cybercrime: A taxonomy of ransomware threats & mitigation techniques, 2018 International Conference on Computing, Mathematics and Engineering Technologies (iCoMET), pp. 1-7, 2018. DOI: 10.1109/ICOMET.2018.8346329.

[309] J.A. Herrera Silva, L.I. Barona López, A.L. Valdivieso Caraguay, M.A. Hernández-Álvarez, A Survey on Situational Awareness of Ransomware Attacks—Detection and Prevention Parameters. Remote Sens.,11, 1168, 2019. DOI: 10.3390/rs11101168.

[310] J. A. Gómez-Hernández, P. García-Teodoro, J. A. Holgado-Terriza, G. Maciá-Fernández, J. Camacho-Páez and M. Robles-Carrillo, AMon: A Monitoring Multidimensional Feature Application to Secure Android Environments, 2021 IEEE Security and Privacy Workshops (SPW), 2021, pp. 31-36, doi: 10.1109/SPW53761.2021.00013.

[311] Monika, P. Zavarsky, D. Lindskog, Experimental analysis of ransomware on windows and android platforms: evolution and characterization. Procedia Comput Sci, 94:465–72, 2016. DOI: 10.1016/j.procs.2016.08.072.

[312] Q. Chen, R. A. Bridges, Automated Behavioral Analysis of Malware: A Case Study of WannaCry Ransomware, 2017 16th IEEE International

Conference on Machine Learning and Applications (ICMLA), pp. 454-460, 2017. DOI 10.1109/ICMLA.2017.0-119.

[313] Microsoft, File system minifilter drivers, Microsoft docs, 2017. Disponible en `https://docs.microsoft.com/en-us/windows-hardware/drivers/ifs/filter-manager-concepts`.

[314] A. Kharraz *et al.*, UNVEIL: A large-scale, automated approach to detecting ransomware, 25th USENIX Security Symposium (USENIX Security 16), pp. 757-772, 2016. Disponible en `https://www.usenix.org/system/files/conference/usenixsecurity16/sec16_paper_kharraz.pdf`.

[315] A. Continella *et al.*, Shieldfs: A self-healing, ransomware-aware filesystem, Proceedings of the 32nd Annual Conference on Computer Security Applications (ACSAC '16), pp. 336-347, New York, NY, USA: ACM, 2016.

[316] A. Palisse *et al.*, Data aware defense (DaD): towards a generic and practical ransomware countermeasure, Nordic Conference on Secure IT Systems, pp. 192-208, 2017.

[317] G. Bottazzi, G. Italiano, D. Spera, Preventing Ransomware Attacks Through File System Filter Drivers, Second Italian Conference on Cyber Security, HAl-01925958, Milan, Italy, 2018.

[318] N. Sabic. Fibratus. 2017. Disponible en `https://github.com/rabbitstack`.

[319] M. M. Ahmadian, H.R. Shahriari, 2entFOX: A framework for high survivable ransomwares detection, 2016 13th International Iranian Society of Cryptology Conference on Information Security and Cryptology (ISCISC), pp. 79-84, 2016. DOI: 10.1109/ISCISC.2016.7736455.

[320] B. Reidys, P. Liu, J. Huang, RSSD: defend against ransomware with hardware-isolated network-storage codesign and post-attack analysis. En Proceedings of the 27th ACM International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS 2022), Association for Computing Machinery, New York, NY, USA, 726–739, 2022. DOI: 10.1145/3503222.3507773.

[321] A. Groenewegen, M. Alqabandi, M. Elamin, P. Paardekooper. A behavioral analysis of the ransomware strain nefilim, 2020. DOI:10.13140/RG.2.2.18301.59360.

[322] A. Alzahrani *et al.*, RanDroid: Structural Similarity Approach for Detecting Ransomware Applications in Android Platform, 2018 IEEE International Conference on Electro/Information Technology (EIT), pp. 0892-0897, 2018. DOI: 10.1109/EIT.2018.8500161.

[323] L. J. García Villalba, A. L. Sandoval Orozco, A. López Vivar, E. A. Armas Vega and T.H. Kim, Ransomware Automatic Data Acquisition Tool, IEEE Access, vol. 6, pp. 55043-55052, 2018. DOI: 10.1109/ACCESS.2018.2868885.

[324] Y. Lemmou, J.L. Lanet, E.M. Souidi, In-Depth Analysis of Ransom Note Files, Computers, 10(11):145, 2021. DOI: 10.3390/computers10110145.

[325] C. Constantinescu, S. Seshadri, Sentinel: ransomware detection in file storage, Proceedings of the 14th ACM International Conference on Systems and Storage (SYSTOR '21). Association for Computing Machinery, New York, NY, USA, Article 28, 1, 2021. DOI: 10.1145/3456727.3463834.

[326] M.E. Ahmed, H. Kim, S. Camtepe, S. Nepal, Peeler: Profiling Kernel-Level Events to Detect Ransomware, En: E. Bertino, H. Shulman, M. Waidner(eds), Computer Security, ESORICS 2021, Lecture Notes in Computer Science, vol 12972. Springer, Cham, 2021. DOI: 10.1007/978-3-030-88418-5_12.

[327] M. J. May, E. Laron, Combating Ransomware using Content Analysis and Complex File Events, 2019 10th IFIP International Conference on New Technologies, Mobility and Security (NTMS), pp. 1-5, 2019. DOI: 10.1109/NTMS.2019.8763851.

[328] S. Mehnaz, A. Mudgerikar, E. Bertino, RWGuard: A Real-Time Detection System Against Cryptographic Ransomware. En: M. Bailey, T. Holz, M. Stamatogiannakis, S. Ioannidis (eds), Research in Attacks, Intrusions, and Defenses, RAID 2018, Lecture Notes in Computer Science, vol 11050. Springer, Cham, 2018. DOI10.1007/978-3-030-00470-5_6.

[329] G. Ramesh, A. Menen, Automated dynamic approach for detecting ransomware using finite-state machine. Decis Support Syst, 138:113400, 2020. DOI: 10.1016/j.dss.2020.113400.

[330] N. Scaife, H. Carter, P. Traynor, K. R. B. Butler, CryptoLock (and Drop It): Stopping Ransomware Attacks on User Data, 2016 IEEE 36th International Conference on Distributed Computing Systems (ICDCS), pp. 303-312, 2016. DOI: 10.1109/ICDCS.2016.46.

[331] S. Baek, Y. Jung, A. Mohaisen, S. Lee, D. Nyang, Ssd-insider: Internal defense of solid-state drive against ransomware with perfect data recovery, 2018 IEEE 38th International Conference on Distributed Computing Systems (ICDCS) p. 875–84, 2018. DOI: 10.1109/ICDCS.2018.00089.

[332] A. Natanzon, P. Derbeko, U. Stern, Bakshi, M., Manusov, Y., Ransomware detection using i/o patterns. US Patent 10,078,459, 2018.

[333] L. Iffländer, A. Dmitrienko, C. Hagen, M. Jobst, S. Kounev, Hands Off my Database: Ransomware Detection in Databases through Dynamic Analysis of Query Sequences, arXiv, 2019. Vol: abs/1907.06775.

[334] X. Han, N. Kheir, D. Balzarotti, Deception Techniques in Computer Security: A Research Perspective. ACM Comput. Surv. 51, 4, Article 80 (July 2019), 36 pages, 2018. DOI: 10.1145/3214305.

[335] Z. Wang, X. Wu, C. Liu, Q. Liu, J. Zhang, RansomTracer: Exploiting Cyber Deception for Ransomware Tracing, 2018 IEEE Third International Conference on Data Science in Cyberspace (DSC), pp. 227-234, 2018. DOI: 10.1109/DSC.2018.00040.

[336] A. Patel, J.P. Tailor, A malicious activity monitoring mechanism to detect and prevent ransomware. Computer Fraud & Security, 14-19, 2020. DOI: 10.1016/s1361-3723(20)30009-9.

[337] R. Moussaileb, B. Bouget, A. Palisse, H. Le Bouder, N. Cuppens-Boulahia, J.L. Lanet, Ransomware's Early Mitigation Mechanisms. ARES 2018 - 13th International Conference on Availability, Reliability and Security, Aug 2018, Hambourg, Germany. DOI: 10.1145/3230833.3234691.

[338] J. Lee, J. Lee, J. Hong, How to Make Efficient Decoy Files for Ransomware Detection? In Proceedings of the International Conference on Research in Adaptive and Convergent Systems (RACS '17). Association for Computing Machinery, New York, NY, USA, 208–212, 2017. DOI: 10.1145/3129676.3129713.

[339] Y. Feng, C. Liu, B. Liu, Poster: A new approach to detecting ransomware with deception. Proceedings of the 38th IEEE Symposium on Security and Privacy Workshops, San Jose, CA, USA, 22–24 May 2017.

[340] C. Moore, Detecting Ransomware with Honeypot Techniques, 2016 Cybersecurity and Cyberforensics Conference (CCC), pp. 77-81, 2016. DOI: 10.1109/CCC.2016.14.

[341] S. K. Shaukat, V.J. Ribeiro, RansomWall: A layered defense system against cryptographic ransomware attacks using machine learning, 10th International Conference on Communication Systems & Networks (COMSNETS), pp. 356-363, 2018. DOI: 10.1109/COMSNETS.2018.8328219.

[342] G. Al-Nemera, S. Al-Otaibi, R. Tahir, M. Alkhatib, Making Honey Files Sweeter: SentryFS - A Service-Oriented Smart Ransomware Solution, arXiv, 2021, 2108.12792.

[343] D. Bekerman, B. Shapira, L. Rokach, A. Bar, Unknown malware detection using network traffic classification, 2015 IEEE Conference on Communications and Network Security (CNS), pp. 134-142, 2015. DOI: 10.1109/CNS.2015.7346821.

[344] K. Cabaj, M. Gregorczyk, W. Mazurczyk, Software-defined networking-based crypto ransomware detection using http traffic characteristics. Computers & Electrical Engineering, 66:353–68, 2018. DOI: 10.1016/j.compeleceng.2017.10.012.

[345] A. Alzahrani *et al.*, RanDroid: Structural Similarity Approach for Detecting Ransomware Applications in Android Platform, 2018 IEEE International Conference on Electro/Information Technology (EIT), 2018, pp. 0892-0897, doi: 10.1109/EIT.2018.8500161.

[346] M.A. Sotelo Monge, J. Maestre Vidal, L.J. García Villalba, A novel Self-Organizing Network solution towards Crypto-ransomware Mitigation. En Proceedings of the 13th International Conference on Availability, Reliability and Security (ARES 2018), Association for Computing Machinery, New York, NY, USA, Article 48, 1–10, 2018. DOI:https://doi.org/10.1145/3230833.3233249.

[347] S. Chadha, U. Kumar, Ransomware: Let's fight back!, 2017 International Conference on Computing, Communication and Automation (ICCCA), pp. 925-930, 2017. DOI: 10.1109/CCAA.2017.8229926.

[348] S. Salehi, H. Shahriari, M. M. Ahmadian, L. Tazik, A Novel Approach for Detecting DGA-based Ransomwares, 2018 15th International ISC (Iranian Society of Cryptology) Conference on Information Security and Cryptology (ISCISC), pp. 1-7, 2018. DOI: 10.1109/ISCISC.2018.8546941.

[349] D. Morato, E. Berrueta, E. Magaña, M. Izal, Ransomware early detection by the analysis of file sharing traffic, Journal of Network and Computer Applications, Volume 124, pags. 14-32, 2018. DOI: 10.1016/j.jnca.2018.09.013.

[350] J. Lee, K. Lee, A Method for Neutralizing Entropy Measurement-Based Ransomware Detection Technologies Using Encoding Algorithms. Entropy 2022, 24, 239. DOI: 10.3390/e24020239.

[351] T. Xia, Y. Sun, S. Zhu, Z. Rasheed, K. Hassan-Shafique, A Network-Assisted Approach for Ransomware Detection, arXiv, 2020. Disponible en `https://arxiv.org/abs/2008.12428`.

[352] Virus Total. Disponible en `https://www.virustotal.com/gui/home/search`.

[353] Y.S. Joshi, H. Mahajan, S.N. Joshi *et al.*, Signature-less ransomware detection and mitigation. J Comput Virol Hack Tech 17, 299–306 (2021). https://doi.org/10.1007/s11416-021-00384-0.

[354] M. Medhat, S. Gaber, N. Abdelbaki, A New Static-Based Framework for Ransomware Detection, 2018 IEEE 16th Intl Conf on Dependable, Autonomic and Secure Computing, 16th Intl Conf on Pervasive Intelligence and Computing, 4th Intl Conf on Big Data Intelligence and Computing and Cyber Science and Technology Congress(DASC/PiCom/DataCom/CyberSciTech), 2018, pp. 710-715. DOI: 10.1109/DASC/PiCom/DataCom/CyberSciTec.2018.00124.

[355] Yara, Yara's Documentation, s.a. Disponible en `https://yara.readthedocs.io/en/stable/`.

[356] D. S. Keyes, B. Li, G. Kaur, A. H. Lashkari, F. Gagnon, F. Massicotte, EntropLyzer: Android Malware Classification and Characterization Using Entropy Analysis of Dynamic Characteristics, 2021 Reconciling Data Analytics, Automation, Privacy, and Security: A Big Data Challenge (RDAAPS), 2021, pp. 1-12, DOI: 10.1109/RDAAPS48126.2021.9452002.

[357] J. Zhu, J. Jang-Jaccard, A. Singh, I. Welch, H. AI-Sahaf, S. Camtepe, A Few-Shot Meta-Learning based Siamese Neural Network using Entropy Features for Ransomware Classification, arXiv preprint, arXiv:2112.00668, 2021.

[358] T. McIntosh, J. Jang-Jaccard, P. Watters, T. Susnjak. The Inadequacy of Entropy-Based Ransomware Detection. En: T. Gedeon, K. Wong, M. Lee (eds), Neural Information Processing. ICONIP 2019. Communications in Computer and Information Science, vol 1143. Springer, Cham, 2019. DOI: 10.1007/978-3-030-36802-9_20.

[359] R. D. Simon, R. Macfarlane, W.J. Buchanan, Differential area analysis for ransomware attack detection within mixed file datasets, Computers & Security, Volume 108, 2021. DOI: 10.1016/j.cose.2021.102377.

[360] C. M. Hsu, C. C. Yang, H. H. Cheng, P. E. Setiasabda, J.S. Leu, Enhancing File Entropy Analysis to Improve Machine Learning Detection Rate of Ransomware, IEEE Access, vol. 9, pp. 138345-138351, 2021, DOI: 10.1109/ACCESS.2021.3114148.

[361] J. Jiao, H. Zhao, Y. Liu, Analysis and Detection of Android Ransomware for Custom Encryption, 2021 IEEE 4th International Conference on Computer and Communication Engineering Technology (CCET), 2021, pp. 220-225, DOI: 10.1109/CCET52649.2021.9544366.

[362] K. Lee, S.Y. Lee, K. Yim, Machine Learning Based File Entropy Analysis for Ransomware Detection in Backup Systems, IEEE Access, vol. 7, pp. 110205-110215, 2019. DOI: 10.1109/ACCESS.2019.2931136.

[363] G.Y. Kim, J.Y. Paik, Y. Kim, E.S. Cho, Byte Frequency Based Indicators for Crypto-ransomware Detection from Empirical Analysis. Journal of Computer Science and Technology, 37(2):423-442, Mar. 2022. DOI: 10.1007/s11390-021-0263-x.

[364] M. Kakavand, L. Arulsamy, A. Mustapha, M. Dabbagh, A Novel Crypto-Ransomware Family Classification Based on Horizontal Feature Simplification. En: S.K. Bhatia, S. Tiwari, S. Ruidan, M.C. Trivedi, K.K. Mishra (eds), Advances in Computer, Communication and Computational Sciences. Advances in Intelligent Systems and Computing, vol 1158, Springer, Singapore, 2021. DOI: 10.1007/978-981-15-4409-5_1.

[365] J. Han, Z. Lin, D.E. Porter, On the Effectiveness of Behavior-Based Ransomware Detection. En: N. Park, K. Sun, S. Foresti, K. Butler, N. Saxena (eds), Security and Privacy in Communication Networks, SecureComm 2020, Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, vol 336. Springer, Cham., 2020. DOI: 10.1007/978-3-030-63095-9_7.

[366] S.H. Kok, A. Abdullah, N.Z. Jhanjhi, M. Supramaniam, Ransomware, Threat and Detection Techniques : A Review, International Journal of Computer Science and Network Security (IJCSNS), 136, Vol.19 No.2, Feb. 2019.

[367] F. De Gaspari, D. Hitaj, G. Pagnotta, L. De Carli, L.V. Mancini, Evading behavioral classifiers: a comprehensive analysis on evading ransomware detection techniques. Neural Comput & Applic, 2022. DOI: 10.1007/s00521-022-07096-6.

[368] N. Rani, S.V. Dhavale, A. Singh, A. Mehra, A Survey on Machine Learning-Based Ransomware Detection. En: D. Giri, K.K. Raymond Choo, S. Ponnusamy, W. Meng, S. Akleylek, S. Prasad Maity (eds),

Proceedings of the Seventh International Conference on Mathematics and Computing, Advances in Intelligent Systems and Computing, vol 1412. Springer, Singapore, 2022. DOI: 10.1007/978-981-16-6890-6_13.

[369] L. Chen, C.Y. Yang, A. Paul, R. Sahita, Towards resilient machine learning for ransomware detection, arXiv, 2019. DOI: 10.48550/arXiv.1812.09400.

[370] S. Malik, B. Shanmugam, K. Kannorpatti, S. Azam, Critical Feature Selection for Machine Learning Approaches to Detect Ransomware, International Journal of Computing and Digital Systems, vol. 11(1), Mar. 2022. DOI: 10.12785/ijcds/110195.

[371] U. Urooj, B.A.S. Al-rimy, A. Zainal, F.A. Ghaleb, M.A. Rassam, Ransomware Detection Using the Dynamic Analysis and Machine Learning: A Survey and Research Directions, Applied Sciences, 12(1):172, 2022. DOI: /10.3390/app12010172.

[372] I. Bello, H. Chiroma, U.A. Abdullahi, A.Y. Gital, F. Jauro, A. Khan, J.O. Okesola, S.M. Abdulhamid, Detecting ransomware attacks using intelligent algorithms: recent development and next direction from deep learning and big data perspectives. J Ambient Intell Human Comput 12, 8699–8717, 2021. DOI: 10.1007/s12652-020-02630-7.

[373] H. Zuhair, A. Selamat, O. Krejcar, A Multi-Tier Streaming Analytics Model of 0-Day Ransomware Detection Using Machine Learning. Applied Sciences, 10(9):3210, 2020. DOI: 10.3390/app10093210.

[374] D.W. Fernando, N. Komninos, T. Chen, A Study on the Evolution of Ransomware Detection Using Machine Learning and Deep Learning Techniques. IoT, 1(2):551-604, 2020. DOI:10.3390/iot1020030.

[375] A.M. Maigida, S.M. Abdulhamid, M. Olalere, J.K. Alhassan1, H. ChiromaE.G. Dada, Systematic literature review and metadata analysis of ransomware attacks and detection mechanisms, J. Reliable Intell. Environ. 5, 67–89, 2019. DOI:10.1007/s40860-019-00080-3.

[376] B. Khammas, Ransomware detection using random forest technique. ICT Express, 6(4):325–31, 2020. DOI: 10.1016/j.icte.2020.11.001.

[377] P. Shijo, A. Salim, Integrated Static and Dynamic Analysis for Malware Detection. Procedia Comput. Sci., 46:804–11, 2015. DOI: 10.1016/j.procs.2015.02.149.

[378] F. Khan, C. Ncube, L.K. Ramasamy, S. Kadry, Y. Nam, A digital dna sequencing engine for ransomware detection using machine learning, IEEE Access, 8:119710–19, 2020. DOI:10.1109/ACCESS.2020.3003785.

[379] S. Kok, A. Abdullah, N. Jhanjhi, Early detection of crypto-ransomware using pre-encryption detection algorithm. Journal of King Saud University-Computer and Information Sciences, 2020. DOI: 10.1016/j.jksuci.2020.06.012.

[380] S.H. Kok, A. Abdullah, N. Jhanjhi, M. Supramaniam, Prevention of Crypto-Ransomware Using a Pre-Encryption Detection Algorithm. Computers, 8(4):79, 2019. DOI: 10.3390/computers8040079.

[381] D. Sgandurra et al., Automated dynamic analysis of ransomware: Benefits, limitations and use for detection, arXiv preprint arXiv:1609.03020, 2016.

[382] Y. Takeuchi, K. Sakai, S. Fukumoto, Detecting ransomware using support vector machines. In: Proceedings of the 47th International Conference on Parallel Processing Companion, p. 1–6, 2018. DOI: 10.1145/3229710.3229726.

[383] A. Walker, S. Sengupta, Insights into malware detection via behavioral frequency analysis using machine learning. En: MILCOM 2019-2019 IEEE Military Communications Conference (MILCOM), IEEE, p. 1–62019.. DOI:10.1109/MILCOM47813.2019.9021034.

[384] B.A.S. Al-Rimy *et al.*, A Pseudo Feedback-Based Annotated TF-IDF Technique for Dynamic Crypto-Ransomware Pre-Encryption Boundary Delineation and Features Extraction, IEEE Access, vol. 8, pp. 140586-140598, 2020. DOI: 10.1109/ACCESS.2020.3012674.

[385] B. Qin, Y. Wang, C. Ma, API Call Based Ransomware Dynamic Detection Approach Using TextCNN, 2020 International Conference on Big Data, Artificial Intelligence and Internet of Things Engineering (ICBAIE), pp. 162-166, 2020. DOI: 10.1109/ICBAIE49996.2020.00041.

[386] M.A. Ayub, A. Continella, A. Siraj, An I/O Request Packet (IRP) Driven Effective Ransomware Detection Scheme Using Artificial Neural Network, Proceedings of 2020 IEEE 21st International Conference on Information Reuse and Integration for Data Science (IRI 2020), p. 319–24, Piscataway, NJ, 2020. DOI: 10.1109/IRI49571.2020.00053.

[387] S. Bae, G. Lee, E. Im, Ransomware detection using machine learning algorithms, Concurrency and Computation: Practice and Experience, 32(18):e5422, 2020. DOI: 10.1002/cpe.5422.

[388] D. Javaheri, M. Hosseinzadeh, A.M. Rahmani, Detection and Elimination of Spyware and Ransomware by Intercepting Kernel-Level System Routines, IEEE Access, vol. 6, pp. 78321-78332, 2018. DOI: 10.1109/ACCESS.2018.2884964.

[389] A. Cohen, N. Nissim, Trusted detection of ransomware in a private cloud using machine learning methods leveraging meta-features from volatile memory. Expert Syst. Appl., 102:158–78, 2018. DOI: 10.1016/j.eswa.2018.02.039.

[390] B. Al-rimy, M. Maarof, S. Shaid, Crypto-ransomware early detection model using novel incremental bagging with enhanced semi-random subspace selection, Future Generation Computer Systems, 101:476–91, 2019. DOI: 10.1016/j.future.2019.06.005.

[391] M. Alam, S. Sinha, S. Bhattacharya, S. Dutta, D. Mukhopadhyay, A. Chattopadhyay, RAPPER: ransomware prevention via performance counters. arXiv preprint arXiv:2004.01712, 2020. Disponible en `https://arxiv.org/abs/2004.01712`.

[392] J. A. Herrera Silva, M. Hernández-Alvarez, Large scale ransomware detection by cognitive security, 2017 IEEE Second Ecuador Technical Chapters Meeting (ETCM), 2017, pp. 1-4. DOI: 10.1109/ETCM.2017.8247484.

[393] A. Azmoodeh, A. Dehghantanha, M. Conti, K.K.R. Choo, Detecting crypto-ransomware in IoT networks based on energy consumption footprint, J. Ambient Intell. Human Comput. 9, 1141–1152, 2018. DOI: 10.1007/s12652-017-0558-5.

[394] G. Cusack, O. Michel, E. Keller, Machine Learning-Based Detection of Ransomware Using SDN, Proceedings of the 2018 ACM International Workshop on Security in Software Defined Networks & Network Function Virtualization (SDN-NFV Sec'18), Association for Computing Machinery, New York, NY, USA, 1–6, 2018. DOI:10.1145/3180465.3180467.

[395] B. Zhang, W. Xiao, X. Xiao, A.K. Sangaiah, W. Zhang, J. Zhang, Ransomware classification using patch-based CNN and self-attention network on embedded N-grams of opcodes, Future Generation Computer Systems, Volume 110, pgs. 708-720, 2020. DOI: 10.1016/j.future.2019.09.025.

[396] J. Baldwin, A. Dehghantanha, Leveraging Support Vector Machine for Opcode Density Based Detection of Crypto-Ransomware. En: A. Dehghantanha, M. Conti, T. Dargahi (eds), Cyber Threat Intelligence, Advances in Information Security, vol 70. Springer, Cham., 2018. DOI: 10.1007/978-3-319-73951-9_6.

[397] F. Manavi, A. Hamzeh, A New Method for Ransomware Detection Based on PE Header Using Convolutional Neural Networks, 2020 17th

International ISC Conference on Information Security and Cryptology (ISCISC), pp. 82-87, 2020. DOI: 10.1109/ISCISC51277.2020.9261903.

[398] S. Poudyal, K. P. Subedi, D. Dasgupta, A Framework for Analyzing Ransomware using Machine Learning, 2018 IEEE Symposium Series on Computational Intelligence (SSCI), pp. 1692-1699,2018. DOI: 10.1109/SSCI.2018.8628743.

[399] S. Poudyal, D. Dasgupta, Z. Akhtar, K. Gupta, A multi-level ransomware detection framework using natural language processing and machine learning, 14th International Conference on Malicious and Unwanted Software (MALCON), No. October 2019. DOI: 10.1109/SSCI.2018.8628743.

[400] S. Homayoun, A. Dehghantanha, M. Ahmadzadeh, S. Hashemi, R. Khayami, K.K.R. Choo, D.E. Newton, DRTHIS: Deep ransomware threat hunting and intelligence system at the fog layer, Future Generation Computer Systems, 90, 94-104, 2019. DOI: 10.1016/j.future.2018.07.045.

[401] L. Fernández Maimó, A. Huertas Celdrán, A.L. Perales Gómez, F.J. García Clemente, J. Weimer, I. Lee, Intelligent and Dynamic Ransomware Spread Detection and Mitigation in Integrated Clinical Environments. Sensors, 19, 1114, 2019. DOI: 10.3390/s19051114.

[402] G. Kathareios, A. Anghel, A. Mate, R. Clauberg, M. Gusat, Catch It If You Can: Real-Time Network Anomaly Detection with Low False Alarm Rates, 2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA), 2017, pp. 924-929. DOI: 10.1109/ICMLA.2017.00-36.

[403] A. Arabo, R. Dijoux, T. Poulain, G. Chevalier, Detecting Ransomware Using Process Behavior Analysis, Procedia Computer Science, Volume 168, Pages 289-296, 2020. DOI: 10.1016/j.procs.2020.02.249.

[404] R.M.A. Molina, S. Torabi, K. Sarieddine, E. Bou-Harb, N. Bouguila, C. Assi, On Ransomware Family Attribution Using Pre-Attack Paranoia Activities, IEEE Transactions on Network and Service Management, 2021. DOI: 10.1109/TNSM.2021.3112056.

[405] C. Moreira, C. Sales, D. Moreira, Understanding Ransomware Actions Through Behavioral Feature Analysis, JCIS, vol. 37, no. 1, pp. 61-76, Mar. 2022. DOI: 10.14209/jcis.2022.7.

[406] S. Poudyal, D. Dasgupta, Analysis of Crypto-Ransomware Using ML-Based Multi-Level Profiling, IEEE Access, vol. 9, pp. 122532-122547, 2021. DOI: 10.1109/ACCESS.2021.3109260.

[407] K.C. Roy, Q. Chen, DeepRan: Attention-based BiLSTM and CRF for Ransomware Early Detection and Classification. Inf. Syst. Front. 23, 299–315, 2021. DOI: 10.1007/s10796-020-10017-4.

[408] N. Pundir, M. Tehranipoor, F. Fahim, RanStop: A Hardware-assisted Runtime Crypto-Ransomware Detection Technique, arXiv, 2020, DOI: 10.48550/ARXIV.2011.12248.

[409] S. Maniath, A. Ashok, P. Poornachandran, V. G. Sujadevi, P. Sankar A.U., S. Jan, Deep learning LSTM based ransomware detection, 2017 Recent Developments in Control, Automation & Power Engineering (RDCAPE), pp. 442-446, 2017. DOI: 10.1109/RDCAPE.2017.8358312.

[410] R. Agrawal, J. W. Stokes, K. Selvaraj, M. Marinescu, Attention in Recurrent Neural Networks for Ransomware Detection, ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2019, pp. 3222-3226. DOI: 10.1109/ICASSP.2019.8682899.

[411] S. Sharmeen, Y. A. Ahmed, S. Huda, B. Ş. Koçer, M. M. Hassan, Avoiding Future Digital Extortion Through Robust Protection Against Ransomware Threats Using Deep Learning Based Adaptive Approaches, IEEE Access, vol. 8, pp. 24522-24534, 2020. DOI: 10.1109/ACCESS.2020.2970466.

[412] S.I. Bae, G.B. Lee, E.G. Im, Ransomware detection using machine learning algorithms, Concurrency and Computation: Practice and Experience, 32:18, 2020. DOI: 10.1002/cpe.5422.

[413] J. Zhou, M. Hirose, Y. Kakizaki, A. Inomata, Evaluation to Classify Ransomware Variants based on Correlations between APIs, roceedings of the 6th International Conference on Information Systems Security and Privacy (ICISSP 2020), pages 465-472, 2020. DOI: 10.5220/0008959904650472.

[414] Y.A. Ahmed, B. Koçer, B.A.S. Al-rimy, Automated Analysis Approach for the Detection of High Survivable Ransomware, KSII Transactions on Internet and Information Systems, vol. 14, no. 5, pp. 2236-2257, 2020. DOI: 10.3837/tiis.2020.05.021.

[415] Y.A. Ahmed, B. Koçer, S. Huda, B.A.S. Al-rimy, M.M. Hassan, System call refinement-based enhanced Minimum Redundancy Maximum Relevance method for ransomware early detection. J. Netw. Comput. Appl., 167, 102753, 2020. DOI: 10.1016/j.jnca.2020.102753

[416] U. Urooj, M.A.B. Maarof, B.A.S. Al-rimy, A proposed Adaptive Pre-Encryption Crypto-Ransomware Early Detection Model, 2021 3rd International Cyber Resilience Conference (CRC), pp. 1-6, 2021. DOI: 10.1109/CRC50527.2021.9392548.

[417] B.A.S. Al-rimy, M.A. Maarof, Y.A. Prasetyo, S.Z., Mohd Shaid, A.F.M. Ariffin, Zero-Day Aware Decision Fusion-Based Model for Crypto-Ransomware Early Detection, International Journal of Integrated Engineering, 2018: 10(6).

[418] N. Tasnim, I.H. Sarker, Ransomware Family Classification With Ensemble Model Based On Behavior Analysis. Preprints 2022, 2022010454. DOI: 10.20944/preprints202201.0454.v1.

[419] A. Bahrani, A.J. Bidgly, Ransomware detection using process mining and classification algorithms, 2019 16th International ISC (Iranian Society of Cryptology) Conference on Information Security and Cryptology (ISCISC), 2019, pp. 73-77. DOI: 10.1109/ISCISC48546.2019.8985149.

[420] S. Sharma, R. Kumar, C. Rama Krishna, A survey on analysis and detection of Android ransomware. Concurrency Computat Pract Exper. 2021; 33:e6272. DOI: doi.org/10.1002/cpe.6272.

[421] M.A. Saleh, A Proactive Approach for Detecting Ransomware based on Hidden Markov Model (HMM), International Journal of Intelligent Computing Research (IJICR), Volume 10, Issue 3, September 2019. DOI: 10.20533/ijicr.2042.4655.2019.0122.

[422] H. Zhang, X. Xiao, F. Mercaldo, S. Ni, F. Martinelli, A.K. Sangaiah, Classification of ransomware families with machine learning based on N-gram of opcodes, Future Generation Computer Systems, Volume 90, Pgs. 211-221, 2019. DOI: 10.1016/j.future.2018.07.052.

[423] S. Sharma, S. Singh, Texture-Based Automated Classification of Ransomware. J. Inst. Eng. India Ser. B 102, 131–142, 2021. DOI: 10.1007/s40031-020-00499-w.

[424] F. Quinkert, T. Holz, K. Hossain, E. Ferrara, K. Lerman, Raptor: ransomware attack predictor. arXiv preprint arXiv:1803.01598, 2018.

[425] A. Melaragno, W. Casey, Detecting Ransomware Execution in a Timely Manner, arXiv, 2022. Disponible en `https://arxiv.org/abs/2201.04424`.

[426] A. Arfeen, M. Asim Khan, O. Zafar, U. Ahsan, Process based volatile memory forensics for ransomware detection. Concurrency Computat Pract Exper., 34(4):e6672, 2022. DOI: 10.1002/cpe.6672.

[427] Arfeen A, Khan MA, Zafar O, Ahsan U. Virtual machine (VM) memory dumps for ransomware forensics. Harvard Dataverse, 2020. DOI: 10.7910/DVN/YVL3CW.

[428] A. Singh, R.A. Ikuesan, H. Venter, Ransomware Detection using Process Memory, International Conference on Cyber Warfare and Security, Volume 17(1): 413-422, 2022. Disponible en `https://arxiv.org/abs/2203.16871`.

[429] M. A. Ayub, A. Sirai, Similarity Analysis of Ransomware based on Portable Executable (PE) File Metadata, 2021 IEEE Symposium Series on Computational Intelligence (SSCI), 2021, pp. 1-6. DOI: 10.1109/SSCI50451.2021.9660019.

[430] V.G. Ganta, G. Harish, V. Kumar, G.R. Rao, Ransomware Detection in Executable Files Using Machine Learning. 2020 International Conference on Recent Trends on Electronics, Information, Communication & Technology (RTEICT), 282-286, 2020.

[431] U. Zahoora, M. Rajarajan, Z. Pan, A. Khan Zero-day Ransomware Attack Detection using Deep Contractive Autoencoder and Voting based Ensemble Classifier. Appl. Intell., 2022. DOI: 10.1007/s10489-022-03244-6.

[432] Q. Chen, S.R. Islam, H. Haswell, R.A. Bridges, Automated Ransomware Behavior Analysis: Pattern Extraction and Early Detection. En: F. Liu, J. Xu, S. Xu, M. Yung (eds), Science of Cyber Security, SciSec 2019, Lecture Notes in Computer Science, vol 11933, Springer, Cham, 2019. DOI: 10.1007/978-3-030-34637-9_15.

[433] P.S. Goyal, A. Kakkar, G. Vinod, G. Joseph, Crypto-Ransomware Detection Using Behavioural Analysis. En: P. Varde, R. Prakash, G. Vinod (eds), Reliability, Safety and Hazard Assessment for Risk-Based Technologies, Lecture Notes in Mechanical Engineering, Springer, Singapore, 2020. DOI: 10.1007/978-981-13-9008-1_20.

[434] S. Homayoun, A. Dehghantanha, M. Ahmadzadeh, S. Hashemi, R. Khayami, Know Abnormal, Find Evil: Frequent Pattern Mining for Ransomware Threat Hunting and Intelligence, arXiv, 2018. Disponible en `http://arxiv.org/abs/1808.01957`.

[435] Y.A. Ahmed, B. Koçer, B.A.S. Al-rimy, Automated Analysis Approach for the Detection of High Survivable Ransomware, KSII Transactions on Internet and Information Systems, vol. 14, no. 5, pp. 2236-2257, 2020. DOI: 10.3837/tiis.2020.05.021.

[436] J. Hwang, J. Kim, S. Lee, K. Kim, Two-Stage Ransomware Detection Using Dynamic Analysis and Machine Learning Techniques. Wireless Pers Commun 112, 2597–2609, 2020. DOI:10.1007/s11277-020-07166-9.

[437] A.M. Maigida, S.M. Abdulhamid, M., Olalere, I. Ismaila, An Intelligent Crypto-Locker Ransomware Detection Technique using Support Vector Machine Classification and Grey Wolf Optimization Algorithms, i-manager's Journal on Software Engineering, 13(3), 15-23, 2019. DOI:10.26634/jse.13.3.15685.

[438] M.M. Hasan, M.M. Rahman, RansHunt: A support vector machines based ransomware analysis framework with integrated feature set, 2017 20th International Conference of Computer and Information Technology (ICCIT), pp. 1-7, 2017. DOI: 10.1109/ICCITECHN.2017.8281835.

[439] S. Egunjobi, S. Parkinson, A. Crampton, Classifying Ransomware Using Machine Learning Algorithms. En: H. Yin, D. Camacho, P. Tino, A. Tallón-Ballesteros, R. Menezes, R. Allmendinger (eds), Intelligent Data Engineering and Automated Learning – IDEAL 2019, IDEAL 2019, Lecture Notes in Computer Science, vol 11872. Springer, Cham., 2019. DOI: 10.1007/978-3-030-33617-2_5

[440] M.S. Abbasi, H. Al-Sahaf, I. Welch, Particle Swarm Optimization: A Wrapper-Based Feature Selection Method for Ransomware Detection and Classification. En: P.A. Castillo, J.L. Jiménez Laredo, F. Fernández de Vega (eds), Applications of Evolutionary Computation, EvoApplications 2020, Lecture Notes in Computer Science, vol 12104. Springer, Cham., 2020. DOI: 10.1007/978-3-030-43722-0_12.

[441] A. Ashraf, A. Aziz, U. Zahoora, A. Khan, Ransomware Analysis using Feature Engineering and Deep Neural Networks, arXiv, 2019. Disponible en `http://arxiv.org/abs/1910.00286`.

[442] S. Aurangzeb, H. Anwar, M.A. Naeem, M. Aleem, BigRC-EML: big-data based ransomware classification using ensemble machine learning. Cluster Comput., 2022. DOI: 10.1007/s10586-022-03569-4.

[443] B. Jethva, I. Traoré, A. Ghaleb, K. Ganame, S. Ahmed: Multilayer ransomware detection using grouped registry key operations, file entropy and file signature monitoring, Journal of Computer Security, vol. 28, n. 3, pp. 337-373, 2020. DOI: 10.3233/JCS-191346.

[444] M. Almousa, J. Osawere, M. Anwar, Identification of Ransomware families by Analyzing Network Traffic Using Machine Learning Techniques, 2021 Third International Conference on Transdisciplinary AI (TransAI), 2021, pp. 19-24. DOI: 10.1109/TransAI51903.2021.00012.

[445] O.M.K. Alhawi, J. Baldwin, A. Dehghantanha, Leveraging Machine Learning Techniques for Windows Ransomware Network Traffic Detection. En: A. Dehghantanha, M. Conti, T. Dargahi (eds), Cyber Threat Intelligence, Advances in Information Security, vol 70. Springer, Cham. 2018. DOI: 10.1007/978-3-319-73951-9_5.

[446] J. Modi, I. Traore, A. Ghaleb, K. Ganame, S. Ahmed, Detecting Ransomware in Encrypted Web Traffic, Foundations and Practice of Security, Springer International Publishing, 2020.

[447] D.W. Fernando, N. Komninos, FeSA: Feature selection architecture for ransomware detection under concept drift, Computers & Security, Volume 116, 102659, 2022. DOI: 10.1016/j.cose.2022.102659.

[448] B. Purnama, D. Stiawan, D. Hanapi, M.Y. Idris, N. Afifah, S. Sharipuddin, R. Budiarto, Time Efficiency on Computational Performance of PCA, FA and TSVD on Ransomware Detection, Indonesian Journal of Electrical Engineering and Informatics (IJEEI), Vol 10, No 1, 2022. DOI: 10.52549/ijeei.v10i1.3481.

[449] M. Masum, M. J. Hossain Faruk, H. Shahriar, K. Qian, D. Lo, M. I. Adnan, Ransomware Classification and Detection With Machine Learning Algorithms, 2022 IEEE 12th Annual Computing and Communication Workshop and Conference (CCWC), pp. 0316-0322, 2022. DOI: 10.1109/CCWC54503.2022.9720869.

[450] K. Lee, K. Yim, J. Seo, Ransomware prevention technique using key backup. Concurrency and Computation: Practice and Experience, 30(3):e4337, 2018. DOI: 10.1002/cpe.4337.

[451] Microsoft, Cryptography API: Next Generation, 2021. Microsoft Documentation. Disponible en `https://docs.microsoft.com/en-us/windows/win32/seccng/cng-portal`.

[452] S. Z. Mohd Shaid, M. A. Maarof, In memory detection of Windows API call hooking technique, 2015 International Conference on Computer, Communications, and Control Technology (I4CT), 2015, pp. 294-298. DOI: 10.1109/I4CT.2015.7219584.

[453] E. Kolodenker, W. Koch, G. Stringhini, M. Egele, Paybreak: Defense against cryptographic ransomware, Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security; 2017. p. 599–611. DOI: 10.1145/3052973.3053035.

[454] P. Bajpai, R. Enbody, An empirical study of api calls in ransomware. 2020 IEEE International Conference on Electro Information Technology (EIT), p. 443–8, 2020, DOI:10.1109/EIT48999.2020.9208284.

[455] P. Bajpai and R. Enbody, Attacking Key Management in Ransomware, IT Professional, vol. 22, no. 2, pp. 21-27, 1 March-April 2020, DOI: 10.1109/MITP.2020.2977285.

[456] M. Al-Dwairi, A.S. Shatnawi, O. Al-Khaleel, B. Al-Duwairi, Ransomware-Resilient Self-Healing XML Documents, Future Internet, 2022, 14(4):115. DOI: 10.3390/fi14040115.

[457] A. Gupta, A. Prakash and N. Scaife, Prognosis Negative: Evaluating Real-Time Behavioral Ransomware Detectors, 2021 IEEE European Symposium on Security and Privacy (EuroS&P), 2021, pp. 353-368, DOI: 10.1109/EuroSP51992.2021.00032.

[458] Crystaldiskmark. Disponible en `https://crystalmark.info/en/software/crystaldiskmark/`

[459] Geekbench. Disponible en `https://www.geekbench.com/`

[460] PCMark10. Disponible en `https://benchmarks.ul.com/pcmark10`

[461] A. Kharraz, W. Robertson, D. Balzarotti, L. Bilge, E. Kirda: Cutting the Gordian Knot: A Look Under the Hood of Ransomware Attacks, 12th Conference on Detection of Intrusions and Malware & Vulnerability Assessment (DIMVA), pp. 1-20, 2015.

[462] K. Rieck, T. Holz, C. Willems, P. Düssel, P. Laskov, Learning and Classification of Malware Behavior. In Proceedings of the 5th international conference on Detection of Intrusions and Malware, and Vulnerability Assessment (DIMVA '08). Springer-Verlag, Berlin, Heidelberg, 108–125, 2008. DOI: 10.1007/978-3-540-70542-0_6.

[463] E. Berrueta, D. Morato, E. Magaña, M. Izal, Open Repository for the Evaluation of Ransomware Detection Tools, in IEEE Access, vol. 8, pp. 65658-65669, 2020. DOI: 10.1109/ACCESS.2020.2984187.

[464] ISOT Research Lab, Ransomware Dataset, 2020. Disponible en `https://www.uvic.ca/ecs/ece/isot/datasets/botnet-ransomware/index.php`.

[465] U. Bayer, C. Kruegel, E. Kirda, TTAnalyze: a tool for analyzing malware. En: Proceedings of the European Institute for Computer Antivirus Research Annual Conference, April 2006.

[466] F. Mbol, J.M. Robert, A. Sadighian, An Efficient Approach to Detect TorrentLocker Ransomware in Computer Systems. In: Foresti S., Persiano G. (eds) Cryptology and Network Security. CANS 2016. Lecture Notes in Computer Science, vol 10052. Springer, Cham., 2016. DOI: 10.1007/978-3-319-48965-0_32.

[467] Resilient Information Systems Security, Ransomware Dataset. Disponible en https://rissgroup.org/category/contributions/.

[468] A. Rege, Critical Infrastructure Ransomware Incident Dataset. Version 11.8. Temple University, 2022. Disponible en https://sites.temple.edu/care/ci-rw-attacks/. Funded by National Science Foundation CAREER Award #1453040. ORCID: 0000-0002-6396-1066.

[469] A. Tseng, Y. Chen, Y. Kao, T. Lin. Deep learning for ransomware detection. IEICE Tech. Rep., 116(282):87–92, 2016.

[470] R. Vinayakumar, K.P. Soman, K.K. Senthil Velan, S. Ganorkar, Evaluating shallow and deep networks for ransomware detection and classification. In International Conference on Advances in Computing, Communications and Informatics (ICACCI), pages 259–265. IEEE, 2017.

[471] J.A. Gómez-Hernández, L. Álvarez-González, P. García-Teodoro, R-Locker, Thwarting ransomware action through a honeyfile-based approach, Computers & Security, Volume 73, Pages 389-398, 2018. DOI: 10.1016/j.cose.2017.11.019.

[472] B. Botezatu, Third Iteration of Linux Ransomware Still not Ready for Prime-Time, Ditdefender, Jan. 2016. Disponible en https://www.bitdefender.com/blog/labs/third-iteration-of-linux-ransomware-still-not-ready-for-prime-time/.

[473] C. Cimpanu, KillDisk Ransomware Now Targets Linux, Prevents Boots-Up, Has Faulty Encryption, Beepingcomputer, Jan. 2017. Disponible en https://www.bleepingcomputer.com/news/security/killdisk-ransomware-now-targets-linux-prevents-boot-up-has-faulty-encryption/.

[474] Z. Chang, G. Sison, J. Jocson, Erebus Resurfaces as Linux Ransomware, TrendMicro, Jun. 2017. Disponible en https://www.trendmicro.com/en_se/research/17/f/erebus-resurfaces-as-linux-ransomware.html.

[475] Y. Wen, J. Wang, Analysis and Remodeling of the DirtyCOW Vulnerability by Debugging and Abstraction. En: H. Miao, C. Tian, S. Liu, Z. Duan (eds), Structured Object-Oriented Formal Language and Method, SOFL+MSVL 2019, Lecture Notes in Computer Science, vol 12028. Springer, Cham, 2020. DOI: 10.1007/978-3-030-41418-4_1.

[476] L. Abrams, Linux version of LockBit ransomware targets VMware ESXi servers, BeepingComputer, Jan. 2022. Disponible en https://www.bleepingcomputer.com/news/security/linux-version-of-lockbit-ransomware-targets-vmware-esxi-servers/.

[477] J. de la Cruz, Analysis and Impact of LockBit Ransomware's First Linux and VMware ESXi Variant, TrendMicro, Jan. 2022. Disponible en https://www.trendmicro.com/en_us/research/22/a/analysis-and-Impact-of-lockbit-ransomwares-first-linux-and-vmware-esxi-variant.html?utm_source=trendmicroresearch&utm_medium=smk&utm_campaign=0122_ImpactofLockbit.

[478] L. Abrams, Linux version of HelloKitty ransomware targets VMware ESXi servers, Bleepingcomputer, Jul. 2021. Disponible en https://www.bleepingcomputer.com/news/security/linux-version-of-hellokitty-ransomware-targets-vmware-esxi-servers/.

[479] Cyble, A Deep-dive Analysis of LOCKBIT 2.0, Aug. 2021. Disponible en https://blog.cyble.com/2021/08/16/a-deep-dive-analysis-of-lockbit-2-0/.

[480] Kaspersky, RansomEXX Trojan attacks Linux systems, SecureList, Nov. 2020. Disponible en https://securelist.com/ransomexx-trojan-attacks-linux-systems/99279/.

[481] C. Cimpanu, SFile (Escal) ransomware ported for Linux attacks, The Record, Jan. 2022. Disponible en https://therecord.media/sfile-escal-ransomware-ported-for-linux-attacks/.

[482] A. Maurya, TellYouThePass Ransomware Analysis Reveals a Modern Reinterpretation Using Golang, CrowdStrike, Jan. 2022 Disponible en https://www.crowdstrike.com/blog/tellyouthepass-ransomware-analysis-reveals-modern-reinterpretation-using-golang/.

[483] S. Hilt, F. Mercês, Backing Your Backup Defending NAS Devices Against Evolving Threats, TrendMicro Research, 2022. Disponible en https://www.trendmicro.com/en_us/research/22/a/defending-users-NAS-devices-from-evolving-threats.html.

[484] N. Naiim, DarkSide on Linux: Virtual Machines Targeted, TrendMicro, May. 2021. Disponible en https://www.trendmicro.com/en_us/research/21/e/darkside-linux-vms-targeted.html.

[485] D. Shestakov, A. Zhdanov, Inside the Hive: Deep dive into Hive RaaS, analysis of latest samples. Group-IB, Dec. 2021. Disponible en https://blog.group-ib.com/hive.

[486] D. Park, Y. Kim, S. Park, J. Lee, blkCtrace: Lightweight Block I/O Layer Content Tracing on Linux, Work-in-Progress Reports (WiPs) en 16th USENIX Conference on File and Storage Technologies, 12-15 Feb. Oakland, CA, USA, 2018.

[487] S. Midtrapanon, G. Wills, Linux patch management: With security assessment features. 4th International Conference on Internet of Things, Big Data and Security, IoTBDS 2019, pp. 270-277, Heraklion, Crete, Greece. 01-03 May 2019.

[488] M.R. Yaswinski, M.M. Chowdhury, M. Jochen, Linux Security: A Survey, 2019 IEEE International Conference on Electro Information Technology (EIT), pp. 357-362, 2019. DOI: 10.1109/EIT.2019.8834112.

[489] R. Septiasari, Y. Restu Pramadi, A study on windows-based ransomware implications on linux operating system using compatibility layer wine based on dynamic analysis, IOP Conference Series: Materials Science and Engineering, Vol. 1007, 012120, 2020. DOI: 10.1088/1757-899X/1007/1/012120.

[490] Z. Wang, B. Cui, Y. Zhang, An ELF Recovery Method for Linux Malicious Process Detection. En: L. Barolli, K. Yim, H.C. Chen (eds), Innovative Mobile and Internet Services in Ubiquitous Computing - Proceedings of the 15th International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS-2021), Asan, Korea, 1-3 July, 2021. Volume 279 of Lecture Notes in Networks and Systems, pages 285-294, Springer, 2021. DOI: 10.1007/978-3-030-79728-7_28.

[491] K.A. Asmitha, P. Vinod, Linux Malware Detection Using eXtended–Symmetric Uncertainty. En: R.S. Chakraborty, V. Matyas, P. Schaumont (eds), Security, Privacy, and Applied Cryptography Engineering, SPACE 2014, Lecture Notes in Computer Science, vol 8804. Springer, Cham, 2014. DOI: 10.1007/978-3-319-12060-7_21.

[492] M.A. Laurenzano, M.M. Tikir, L. Carrington, A. Snavely: PEBIL: Efficient Static Binary Instrumentation for Linux. *IEEE International Symposium on Performance Analysis of Systems & Software (ISPASS)*, pp. 175-183, 2010.

[493] M. Kerrisk, *The Linux Programming Interface*, No Starch Press, 2010.

[494] N. Hampton, Z. Baig, S. Zeadally, Ransomware behavioural analysis on windows platforms, Journal of Information Security and Applications, Volume 40, pp. 44-51, 2018. DOI: 10.1016/j.jisa.2018.02.008.

[495] A.H. Mohammad, Analysis of Ransomware on Windows platform, IJCSNS International Journal of Computer Science and Network Security, VOL.20 No.6, July 2020. DOI: 10.13140/RG.2.2.11150.59202.

[496] R. Moussaileb, N. Cuppens, J.L. Lanet, H. Le Bouder, A Survey on Windows-based Ransomware Taxonomy and Detection Mechanism: Case Closed?, ACM Computing Surveys, Vol. 54(6): 117, Jul. 2021. DOI:10.1145/3453153.

[497] Microsoft, Programming reference for the Win32 API, Microsoft Documentation. Disponible en https://docs.microsoft.com/en-us/windows/win32/api/.

[498] I. Vurdelja, I. Blažić, D. Bojić, D. Drašković, A framework for automated dynamic malware analysis for Linux, 2020 28th Telecommunications Forum (TELFOR), pp. 1-4, 2020. DOI: 10.1109/TELFOR51502.2020.9306520.

[499] Malboxes, Malware analysis Windows virtual machines. Disponible en https://github.com/GoSecure/malboxes.

[500] Vagrant, Development Environments Made Easy. Disponible en https://www.vagrantup.com.

[501] VirtualBox. Disponible en https://www.virtualbox.org/.

[502] Microsoft, Microsoft Defender Antivirus in Windows, Microsoft Documentations, Apr. 2022. Disponible en https://docs.microsoft.com/en-us/microsoft-365/security/defender-endpoint/microsoft-defender-antivirus-windows?view=o365-worldwide.

[503] TheZoo, A Live Malware Repository. Disponible en https://github.com/ytisf/theZoo.

[504] VirusShare, Repositorio de muestras de *malware*. Disponible en https://virusshare.com/.

[505] VirusTotal, Repositorio de muestras de *malware*. Disponible en https://www.virustotal.com.

[506] S. Mauri, A POC Windows crypto-ransomware. Disponible en https://github.com/mauri870/ransomware.

[507] S. Wu, J. Leong, Cerber 5.0.1 Arrives with New Multithreading Method, Fortinet, Dic. 2016. Disponible en https://www.fortinet.com/blog/threat-research/cerber-5-0-1-arrives-with-new-multithreading-method.

[508] Microsoft, Thread Stack Size, Windows App Development, Jul. 2021. Disponible en https://docs.microsoft.com/en-us/windows/win32/procthread/thread-stack-size.

[509] J.A. Gómez-Hernández, R-Locker: Anti-ransomware tool for Windows platforms. Recurso disponible en https://github.com/JA-Gomez-Hernandez/R-Locker.

[510] GSMA, The Mobile Economy 2022, GSMA Association, 2022. Disponible en https://www.gsma.com/mobileeconomy/.

[511] R. Relick, 2022 Global Mobile Threat Report: Key Insights on the State of Mobile Security, Zimperium, Mar. 2022. Disponible en `https://blog.zimperium.com/global-mobile-threat-report-key-insights/`.

[512] Kaspersky, IT threat evolution in Q3 2021. Mobile statistics, Nov. 2021. Disponible en `https://securelist.com/it-threat-evolution-in-q3-2021-mobile-statistics/105020/`.

[513] X. Wang, Security Threats and Protection Based on Android Platform. En: M. Atiquzzaman, N. Yen, Z. Xu, 2021 International Conference on Big Data Analytics for Cyber-Physical System in Smart City. Lecture Notes on Data Engineering and Communications Technologies, vol 103. Springer, 2022, Singapore. DOI: 10.1007/978-981-16-7469-3_19.

[514] F. Laricchir, Mobile operating systems' market share worldwide from January 2012 to January 2022, Statista, Feb 7, 2022. Disponible en `https://www.statista.com/statistics/272698/global-market-share-held-by-mobile-operating-systems-since-2009/`.

[515] C. Xu, Android ransomware trends and case studies: A reverse engineering approach, Thesis of Master of Science, Iowa State University, 2019.

[516] A. Tandon, A. Nayyar, A Comprehensive Survey on Ransomware Attack: A Growing Havoc Cyberthreat. En: V. Balas, N. Sharma, A. Chakrabarti (eds), Data Management, Analytics and Innovation. Advances in Intelligent Systems and Computing, vol 839. Springer, Singapore, 2019. DOI: 10.1007/978-981-13-1274-8_31.

[517] S. Sjouwerman, The Evolution of Mobile Ransomware, KnowBe4. Disponible en `https://blog.knowbe4.com/evolution-of-mobile-ransomware`.

[518] Secure Honey, How to Dissect Android SimpleLocker Ransomware, SecureHoney, Jun. 2014. Disponible en `https://securehoney.net/blog/how-to-dissect-android-simplelocker-ransomware.html#.Wi2nvraZMnU`.

[519] A. Martín, J. Hernandez-Castro, D. Camacho, An in-Depth Study of the Jisut Family of Android Ransomware, IEEE Access, vol. 6, pp. 57205-57218, 2018. DOI: 10.1109/ACCESS.2018.2873583.

[520] M. Ameer, S. Murtaza, M. Aleem, A Study of Android-based Ransomware: Discovery, Methods, and Impacts, Journal of Information Assurance & Security, 13(3), pp. 109-117, 2018.

[521] S. Sharma, R. Kumar, C. Rama Krishna, A survey on analysis and detection of Android ransowmare, Concurrency and Computarion: Practice and Experience, Wisley, Vol. 23(16), 2021. DOI: 10.1002/cpe.6272.

[522] Android, Linux con seguridad mejorada en Android. Disponible en `https://source.android.com/security/selinux`.

[523] G. Renjith, S. Aji, Unveiling the Security Vulnerabilities in Android Operating System. En: S. Shakya, K.L. Du, W. Haoxiang (eds), Proceedings of Second International Conference on Sustainable Expert Systems, Lecture Notes in Networks and Systems, vol 351. Springer, Singapore, 2022. DOI: 10.1007/978-981-16-7657-4_9.

[524] S. Sharma, R. Kumar, C.B. Krishna, RansomAnalysis: The Evolution and Investigation of Android Ransomware. En: M. Dutta, C. Krishna, R. Kumar, M. Kalra (eds), Proceedings of International Conference on IoT Inclusive Life (ICIIL 2019), NITTTR Chandigarh, India, Lecture Notes in Networks and Systems, vol 116. Springer, Singapore, 2020. DOI: 10.1007/978-981-15-3020-3_4.

[525] J. Chen, C. Wang, Z. Zhao, K. Chen, R. Du, G.J. Ahn, Uncovering the face of Android ransomware: characterization and real-time detection. IEEE Trans. Inf. Forensics Secur., 13(5), 1286–1300, 2018. DOI: 10.1109/TIFS.2017.2787905.

[526] K. Allix, T.F. Bissyandé, J. Klein, Y. Le Traon, Androzoo: collecting millions of Android apps for the research community, en Proceedings of the 13th International Conference on Mining Software Repositories, ACM, pp. 468–471, 2016.

[527] SO Documentation, Android Java Native Interface (JNI). Disponible en `https://sodocumentation.net/android/topic/8674/android-java-native-interface--jni-`.

[528] Microsoft, Especificación del sistema de archivos exFAT, Microsoft Docs. Disponible en `https://docs.microsoft.com/es-es/windows/win32/fileio/exfat-specification`.

[529] S. Alsoghyer, I. Almomani, Ransomware Detection System for Android Applications, Electronics, Vol. 8(8): 868, 2019. DOI: 10.3390/electronics8080868.

[530] N. Alzahrani, D. Alghazzawi, A Review on Android Ransomware Detection Using Deep Learning Techniques. Proceedings of the 11th International Conference on Management of Digital EcoSystems (MEDES '19), Association for Computing Machinery, New York, NY, USA, 330–335, 2019. DOI: 10.1145/3297662.3365785.

[531] J.A. Gómez-Hernández, P. García-Teodoro, J.A. Holgado-Terriza, G. Maciá-Fernández, J. Camacho-Páez, J.M. Noguera-Comino, Monitoring Android Communications for Security, IEEE INFOCOM 2021 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), pp. 1-2, 2021. DOI: 10.1109/INFOCOMWKSHPS51825.2021.9484574.

[532] Android, API reference. Disponible en `https://developer.android.com/reference`.

[533] P. García-Teodoro, J. Camacho, G. Maciá-Fernández, J.A. Gómez-Hernández, V.J. López-Marín, A Novel Zero-Trust Network Access Control Scheme based on the Security Profile of Devices and Users, Computer Networks, Vol. 212, 109068, 2022. DOI: 10.1016/j.comnet.2022.109068.

[534] R. Cabral, J. T. McDonald, L. M. Hively, R. G. Benton, Profiling CPU Behavior for Detection of Android Ransomware, SoutheastCon 2022, 2022, pp. 690-697. DOI: 10.1109/SoutheastCon48659.2022.9764053.

[535] J. A. Gómez-Hernández, J. Camacho, J. A. Holgado-Terriza, P. García-Teodoro, G. Maciá-Fernández, ARANAC: A Bring-Your-Own-Permissions Network Access Control Methodology for Android Devices, IEEE Access, vol. 9, pp. 101321-101334, 2021. DOI: 10.1109/ACCESS.2021.3097152.

[536] S. Wang *et al.*, KRProtector: Detection and Files Protection for IoT devices on Android without ROOT against Ransomware Based on Decoys, IEEE Internet of Things Journal. DOI: 10.1109/JIOT.2022.3156571.

[537] Android, FileObserver abstrac class, Developers manual. Disponible en `https://developer.android.com/reference/android/os/FileObserver`.

[538] IssueTracker, FileObserver says its recursive but it is not, Android Issue Trackers. Disponible en `https://issuetracker.google.com/issues/36949869`.

[539] J. Rodríguez-Pérez, Detección de *malware* en Android, TFM de Máster Propio en Ciberseguridad de la Universidad de Granada. Tutor: J.A. Gómez-Hernández, 2021.

[540] Genymotion, Android as a Service. Disponible en `https://www.genymotion.com/`.

[541] Sk3ptre, Repositorio de *malware*. Disponible en `https://github.com/sk3ptre/AndroidMalware_2020`.