Detección de cryptojacking a través del tráfico de red usando técnicas de *Machine learning*

Steven Bernal Tovar

Universidad Icesi
Facultad de Ingeniería
Ingeniería de Sistemas
Cali
2020

Detección de cryptojacking a través del tráfico de red usando técnicas de *Machine learning*

Steven Bernal Tovar

Proyecto de grado

Christian Urcuqui López, MsC
Andres Navarro Cadavid, PhD

Universidad Icesi
Facultad de Ingeniería
Ingeniería de Sistemas
Cali
2020

Agradecimiento

Dedico de manera especial esta tesis a mi madre, puesto su empeño y dedicación por sacarme adelante con principios y valores, me han permitido llegar hasta este punto como profesional y ser humano. A mis tutores por haberme tenido paciencia en este camino, haberme enseñado y corregido cuando era necesario. A mis maestros por instruirme en los conocimientos necesarios para afrontar la vida como profesional. Y a todos los que me apoyaron de forma directa o indirectamente.

Abstract

Cryptojacking is a threat that degrades the processor of a computer with the ability to connect to the internet, due to its intensive use in solving mathematical puzzles for the validation of cryptoactive transactions. causing increases in energy costs. Its ease of deployment regardless of the platform makes this threat very dangerous. During this project, certain techniques are explored to develop a machine learning model for the detection of miners through variables at the Network level, as well as an initial exploration of hardware variables against the activity of mining processes.

Resumen

El cryptojacking es una amenaza que degrada el procesador de un equipo con capacidad de conectarse a internet, por su uso intensivo en la resolución de acertijos matemáticos para la validación de transacciones de criptoactivos, provocando incrementos en los costos de energía. Su facilidad de despliegue sin importar la plataforma, hacen que esta ciberamenaza sea muy peligrosa. Durante este proyecto se exploran ciertas técnicas para desarrollar un modelo de aprendizaje automático para la detección de mineros a través de variables a nivel de red, así como una exploración inicial de variable de hardware frente actividad de procesos de minería.

Índice general

Lista de acrónimos

- IP: *Internet Protocol*
- DPI: *Deep Packet Inspection*
- DNS: *Domain Name System*
- P2P: *Peer to Peer*
- LAN: *local area network*
- MAN: *Metropolitan Area Network*
- WAN: *wide area network*
- OSI: *Open Systems Interconnection*
- TCP/IP: *Transmission Control Protocol/Internet Protocol*
- UDP: *User Datagram Protocol*
- TCP: *Transmission Control Protocol*
- JSON-RPC: *remote procedure call protocol encoded in JSON*
- EDA: *Exploratory Data Analysis*
- CDA: *Confirmatory Data Analysis*
- CRISP-DM: *Cross-Industry Estándar Process for Data Mining*
- USUM-DM: *Analytics Solutions Unified Method for Data Mining*
- SEMMA: *Sample, Explore, Modify, Model, and Assess*
- DT: *Decision Tree*
- SVM: *Support Vector Machines*
- KNN: *k-nearest neighbors*
- ROC: *Receiver Operating Characteristic*
- AUC: *Area Under Curve*

Índice de figuras

Índice de cuadros

Motivación y antecedentes

1.1. Contexto

En la actualidad la adopción de las criptomonedas ha venido siendo más significativa en el entorno financiero, puesto que representan una alternativa al dinero convencional y a los sistemas de pago tradicional, debido a que permiten realizar sin un intermediario financiero intercambios de bienes y servicios, transferencias de fondos o inversiones. Una gran cantidad de estas criptomonedas aplican algoritmos complejos de criptografía, para cifrar sus respectivas transacciones con la finalidad de reservar la privacidad de sus emisores y receptores, causando que sea complejo realizar una trazabilidad de sus movimientos, lo que ha incentivado su uso para actividades delictivas como evasión de impuestos, lavado de activos, financiación del terrorismo, entre otras actividades ilícitas (Arango, Ramírez, Ortiz, y Rego, 2018).

Existen dos formas de adquirir una criptomoneda. La primera es comprarla en un mercado de criptodivisas. La segunda es obtenerla por medio de la minería. El proceso de minería (**Figura 1.1**) consiste en validar una transacción de criptomonedas entre dos usuarios en una conexión *pear to pear*, por medio de la resolución de un problema matemático, por parte de otros usuarios conocidos como mineros, que utilizan sus dispositivos computacionales para tal fin. Una vez validada la transacción se crea un bloque con información de los movimientos relacionados con las monedas virtuales transferidas, que luego se verifican por el resto de usuarios, para que el bloque sea anexado al sistema de *Blockchain* (Nakamoto, 2008). El minero que resuelva el problema obtiene una nueva criptomoneda, y a su vez, el problema de validación incrementa su dificultad para la siguiente iteración (Yang, Chen, y Chen, 2019). Cabe resaltar que la recompensa, así como la dificultad pueden variar, dependiendo del tipo de criptodivisa.

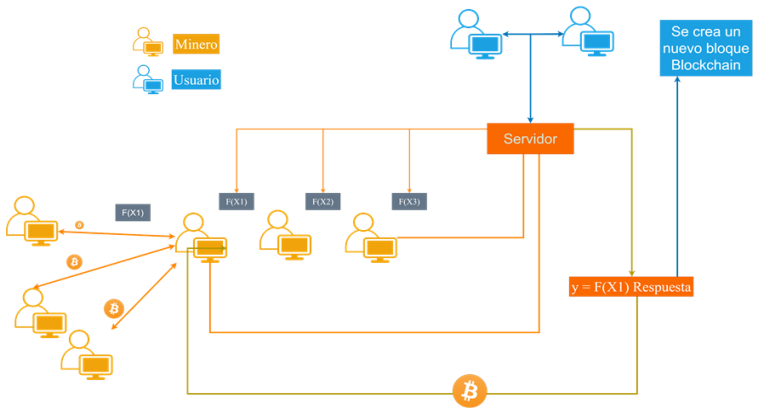

Figura 1.1: Proceso de minería

El proceso para extraer una moneda virtual requiere un uso intensivo de unidades de procesamiento, lo que causa un incremento en el consumo de electricidad, lo que hace inviable la minería para un solo dispositivo. Para el año 2018 extraer un Bitcoin por medio de una computadora sin modificaciones de herramientas mineras, podría tomar cerca de 425 años (Eskandari, Leoutsarakos, Mursch, y Clark, 2018). Por eso algunos mineros optaron por el trabajo colectivo por medio de piscinas mineras, donde la carga de trabajo se distribuye entre los participantes para minar la criptomoneda. Si un integrante obtiene la solución del problema, la recompensa se distribuye entre todos los mineros de la piscina con base en el aporte de recursos computacionales (Rosenfeld, 2011).

Mineros con dudosa moralidad extraen criptoactivos con recursos computacionales de otras personas sin autorización, esta actividad se denomina cryptojacking. Inicialmente los ataques de esta índole eran con ejecutables, después el enfoque cambio hacia la minería a través del navegador, donde los atacantes o criptojackers usan procesos benignos y autorizados del equipo afectado, con *scripts* de minería que se ejecutan en el navegador. Pero en los últimos años los criptojackers han venido atacando a la infraestructura de proveedores de servicios de computación en la nube (Jayasinghe y Poravi, 2020).

El cryptojacking se hizo popular durante el 2017, con el uso indebido del *script* de Coinhive. Coinhive fue una firma de minería de navegador, que se especializaba en extraer Monero una criptomoneda que utiliza el protocolo CryptoNote y aprovecha la firma de anillo enlazable (Ring CT), para proteger la identidad del emisor de las transacciones (Li y cols., 2019). Durante el funcionamiento de Coinhive se reportaban 10 millones de usuarios víctimas cada mes del *script* de esta compañía, antes de su cierre que ocurrió en marzo de 2019 (Varlioglu, Gonen, Ozer, y Bastug, 2020).

En la actualidad el cryptojacking afronta un futuro incierto, debido al cierre de Coinhive, el precio volátil de las criptomonedas y su complejidad para extraerlas. Pese a lo anterior, se han detectado versiones nuevas de los mineros conocidos, además de *scripts* únicos, que demostrarían que los cibercriminales están especializándose en la extracción de criptoactivos. Un estudio realizado por investigadores de las universidades de Cincinnati y Lakehead, demostró que el 99 % de los sitios web analizados por la herramienta de detección minera CMTracker durante el año 2018, ya no usan código minero. Solo el 1 % de los sitios web aún ejecutan código de minería, de este grupo se detectó 8 *scripts* únicos que permitieron rastrear 632 sitios web de cryptojacking únicos. Los investigadores concluyeron a partir de los resultados mencionados anteriormente, que el cryptojacking no terminó después del cierre de Coinhive (Varlioglu y cols., 2020).

Cabe resaltar que la razón de existir del cryptojacking es la popularidad de las criptomonedas, cuyo mercado está creciendo por diversas razones, especialmente por su aceptación por diferentes países, como China que está regulando una moneda virtual para competir con el dólar (Peters, Green, y Yang, 2020). También por la acogida de diferentes empresas como PayPal, Facebook, Microsoft, Shopify, JPMorgan y Tesla que hacen uso de estas criptomonedas o las han incluido en sus ecosistemas comerciales (Ali, 2020). Además la recuperación o incremento del precio de estos criptoactivos producto de la pandemia COVID-19.

1.2. Antecedentes del problema

El problema del cryptojacking persiste porque esta ciberamenaza es multiplataforma, en su gran mayoría no requiere usar ejecutables, basta con visitar un sitio corrupto sin descargar un archivo para comprometer el sistema (Carlin, O'Kane, Sezer, y Burgess, 2018). Los criptojackers también implementan ataques de *Living off the Land*, donde hacen uso de herramientas instaladas por los sistemas nativos, para ejecutar código minero y explotar vulnerabilidades, con directivas *bash* de Linux o *powershell* de Windows (Jayasinghe y

Poravi, 2020), sobre todo para atacar la infraestructura de servicios de computación en la nube.

Existen diversos métodos para detectar mineros en la red. El primer método consiste en detectar la dirección IP (IP, *Internet Protocol*) de mineros en el tráfico de red (Muñoz y Ros, 2019). Este método presenta algunos inconvenientes, ya que, los servidores de minería pueden cambiar la dirección IP del sitio donde están alojados. El segundo mecanismo es la inspección profunda de paquetes (DPI, *Deep Packet Inspection*) para detectar los protocolos utilizados por mineros. Este mecanismo representa mucha carga computacional, ya que todos los paquetes deben ser analizados para detectar posible actividad minera. El tercer método es la solicitud del sistema de nombres de dominio (DNS, *Domain Name System*), donde se busca detectar las direcciones que pueden estar relacionadas con minería. Este método no siempre da indicios correctos de que exista actividad minera en la red, ya que, algún usuario puede tan solo estar visitando sitios que ofrecen servicios de minería (i. Muñoz, Suárez-Varela, y Barlet-Ros, 2019), además las direcciones pueden ser modificadas por el atacante. Pese a que existan métodos para la detección del cryptojacking, el problema aún persiste. También se tiene que considerar el constante crecimiento de las redes, el cual se estima en 37 % para el año 2023, que hace cada vez más complejo analizar el tráfico de red (CISCO, 2020).

1.3. Justificación

Es importante tener una solución basada en firmas o perfiles que permita la detección de *malware* conocido, el problema con estas soluciones radica en su ineficiencia ante nuevas amenazas (O'Kane, Sezer, McLaughlin, y Im, 2013), sobre todo aquellas que no requieran de archivos para comprometer la máquina de una víctima, como el cryptojacking. De esta manera no se pueda capturar y asociar una firma a la actividad del *malware* minero, el cual se aprovecha de procesos benignos y autorizados por el sistema para llevar a cavo su cometido de extraer criptomonedas (Carlin y cols., 2018). Por lo anterior, es necesario realizar un análisis dinámico que permita determinar las trasmisiones de información en la red, entre el equipo que procesa los datos de los mineros (puede ser un servidor, como la computadora del cryptojacker) y la víctima. Una vez procesada la información se aplica técnicas de analítica de datos, para identificar los patrones mineros y evaluar un modelo que tenga capacidad de detectar amenazas mineras en tiempo real.

El reconocimiento de patrones y características en común de actividad minera de criptomonedas permitirá la identificación oportuna del cryptojacking, de esta manera un administrador de red podrá eliminar la actividad minera de los equipos afectados, previniendo que los dispositivos tecnológicos degraden su *hardware* y desperdicien energía eléctrica, evitando tanto costos por el remplazo de equipos deficientes, así como el incremento de la factura de energía causada por la actividad minera, además de reducir la huella ambiental por el consumo inútil de electricidad.

Descripción del problema

2.1. Identificación del problema

El cryptojacking es un ataque que normalmente se ejecuta en segundo plano, no necesariamente requiere de archivos binarios para infectar a un equipo, pero también puede usar *malware*, presenta variabilidad en los vectores de ataques, utiliza llamadas al sistema usando procesos autorizados y es multiplataforma. Esta amenaza suele evadir los mecanismos establecidos y genera un tráfico que puede pasar desapercibido en una red grande.

2.2. Formulación del problema

El cryptojacking parte del consumo de recursos de una víctima la cual debe recibir y enviar información a un tercero (cibercriminal u otro agente malicioso), toda transmisión se lleva a cabo a través de la red, es decir, que si ocurrió un acto malicioso los registros deben tener la historia del evento, independientemente del estatus del minero (ya identificado o no) y del tráfico que exista en una red.

Objetivos

3.1. Objetivo general

Desarrollar un modelo *Machine learning* que tenga la capacidad para detectar actividad de cryptojacking a través de registros de red.

3.2. Objetivos específicos

1 Crear un *dataset* de pcaps de tráfico de cryptojacking.

2 Crear un *dataset* de ventanas de tiempo de actividad de cryptojacking.

3 Encontrar un patrón en los registros de red, que permitan identificar indicios de minería.

4 Evaluar cuatro diferentes modelos de clasificación, que tengan la capacidad de predecir actividad minera.

5 Hacer un análisis exploratorio básico de variables de *hardware* con actividad minera y normal

Marco teórico

4.1. Redes de computadoras

Las redes permiten la interconexión entre sistemas de información independientemente del *hardware* y el sistema operativo, usando tecnologías de comunicación y un medio que puede ser físico o inalámbrico. El propósito de las conexiones es compartir información y recursos en cualquier parte, siempre que se tengan los permisos necesarios. Las redes pueden ser clasificadas dependiendo de su tamaño: P2P (*Peer-to-Peer*) red de pares, LAN (*Local Area Networks*) son redes que operan en espacios reducidos; MAN (*Metropolitan Area Network*) redes que cubren una zona metropolitana; WAN (*Wide Area Network*) redes que operan en extensas áreas (Tanenbaum y Wetherall, 2012).

4.1.1. Primitivas

Un servicio de conexión puede especificarse como un conjunto de primitivas que indican cuando puede desarrollar acciones o informar alguna eventualidad. Las primitivas son llamadas al sistema que causan un salto al modo kernel, al mismo tiempo que retorna el control de la máquina al sistema operativo para enviar los paquetes necesarios. Las primitivas pueden variar dependiendo del tipo de conexión, ya sea, un servicio orientado a conexión, así como un servicio no orientado a conexión.(Tanenbaum y Wetherall, 2012). En el **Cuadro 4.1** se observa algunas de las primitivas más importantes.

Cuadro 4.1: Primitivas
(Tanenbaum y Wetherall, 2012)

Primitivas	
LISTEN	Inicia el bloqueo de una conexión entrante, hasta que un proceso requiera conectarse
CONNECT	Intenta establecer alguna conexión entre nodos
ACCEPT	Acepta una conexión entrante
SEND	Envía información
RECEIVE	Bloquea en espera de información entrante
DISCONNECT	Finaliza la conexión

4.1.2. Modelos de referencia

Existen dos modelos de referencia que se usan para explicar cómo deberían de ser las conexiones entre los nodos de una red. El primero es el modelo de referencia OSI (*Open Systems Interconnection*) y el segundo es el modelo de referencia TCP/IP (*Transmission Control Protocol/Internet Protocol*). En la **Figura 4.1** se muestra la comparación de los dos modelos.

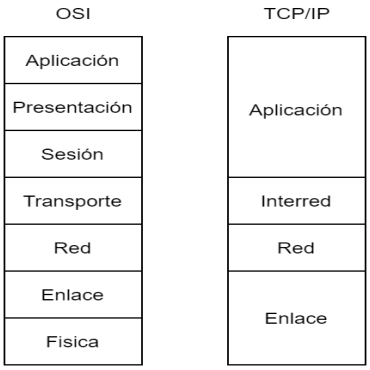

Figura 4.1: Comparación del modelo OSI y TCP/IP

El modelo OSI propone como se deben realizarse las comunicaciones en sistemas abiertos. Este modelo contiene siete capas: capa física, explica la transmisión de *bits* entre dispositivos; capa de enlace, se encarga de la trasmisión de información libre de errores entre la capa física y la capa de red; capa de red, establece el enrutamiento de los paquetes entre una o más redes; capa de transporte, permite la transacción de los datos del dispositivo origen al dispositivo de destino; capa de sesión, gestiona el enlace entre los sistemas de información que se estén comunicando; capa de presentación, tiene como objetivo representar la información de los datos, para que los dispositivos interconectados independientemente del intérprete sean capaces de entender la información; capa de aplicación, permite el consumo de los servicios de las capas inferiores a las aplicaciones que los requieran (ISO/IEC, 1994).

El modelo TCP/IP indica como deben realizarse las interconexiones de redes a través de diferentes protocolos y metodologías. Este modelo contiene cuatro capas: capa de enlace, indica como debe llevarse a cabo la transferencia de datos entre los dispositivos que se estén comunicando, esta capa es una interfaz entre hosts y enlaces de trasmisión; capa de interred, permite que los hosts inyecten paquetes en cualquier red y que viajen a sus respectivos destinos gracias al protocolo IP, el cual establece la ruta que debe seguir un paquete en la comunicación de host a host, estos paquetes están formados por datagramas que representan el mínimo bloque de información; capa de transporte, garantiza que los paquetes de datos lleguen en secuencia y sin errores, aplicando un protocolo orientado a la comunicación conocido como TCP (*Transmission Control Protocol*), el cual permite que los dispositivos se comuniquen enviando datos desde un punto de inicio que establece una comunicación y un punto de cierre que finaliza la conexión, también se utiliza un protocolo no orientado a la comunicación denominado UDP (*User Datagram Protocol*), el cual no válida las conexiones entre los hosts transmisores y receptores; capa de aplicación, establece los servicios que puede utilizar el usuario y las aplicaciones de red (Tanenbaum y Wetherall, 2012).

4.1.3. *Stratum*

Stratum es el protocolo que permite a los mineros recibir carga de trabajo de manera confiable y eficiente de servidores mineros. Este protocolo está construido sobre TCP/IP y trabaja con el formato JSON-RPC (*remote procedure call protocol encoded in JSON*), que permite hacer múltiples llamadas al servidor, sin esperar una respuesta sincrónica. (Recabarren y Carbunar, 01 Jul. 2017). Se utiliza en la comunicación entre dispositivos mineros, servidores y servicios de piscinas mineras. Este protocolo no produce demasiado tráfico de red y permite que los resultados de *hash* se trasmitan con mayor frecuencia, esto permite una medición *hash rate* más precisa (stratum protocolo, 2020).

4.1.4. *NetFlow*

Netflow es una función de enrutador CISCO con el propósito de analizar el tráfico de red, que agrupa los paquetes que tienen en común valores como origen y destino IP y puerto, protocolo y tipo de servicio, en un solo flujo. Posteriormente se cuentan los paquetes y *bytes*, con el fin de tomar registros que resuman la inmensa cantidad de información. Durante este proceso se pierde datos que no son relevante (D. I. CISCO, 2012). En la **Figura 4.2** se muestra la representación.

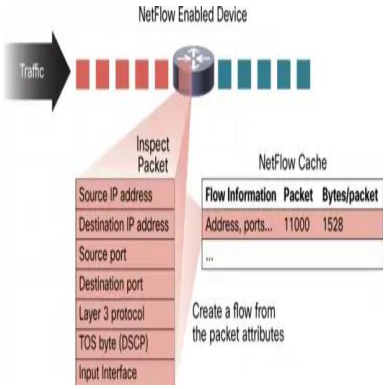

Figura 4.2: Netflow
(D. I. CISCO, 2012)

4.2. Ciencia de datos

La ciencia de datos busca identificar y comprender patrones en un conjunto de datos con el propósito de construir modelos que representen el contexto de la información. Esta ciencia tiene como propósito recopilar datos de diversas fuentes de información, por medio de técnicas y herramientas que incluyen métodos de estadística, *machine learning*, minería de datos y visualización.

4.2.1. Modelos

Un modelo define la relación entre atributos de datos y una función matemática o estadística. Existen dos modelos: descriptivos, cuyo objetivo es proveer más información sobre el contexto de los datos; y predictivos, encargados de estimar un objetivo a partir de una serie de valores(Urcuqui, Navarro, Osorio, y García, 2018).

4.2.2. Enfoques

En la ciencia de datos hay dos enfoques: el análisis exploratorio de los datos (EDA, *Exploratory Data Analysis*), que tiene como finalidad descubrir la relaciones o patrones que existen en los datos, la producción de indicadores y resúmenes basados en la aplicación de métodos estadísticos aplicable cuando no se tiene una hipótesis o un entendimiento de ellos; y análisis de datos confirmatorios (CDA, *Confirmatory Data Analysis*), que asume que el científico posee una hipótesis acerca de la información y tiene como objetivo probarla o descartarla a partir de los modelos creados (Urcuqui y cols., 2018).

4.2.3. Tipos de modelos

Los modelos en la ciencia de datos son de dos tipos: modelos basados en la estadística y modelos basados en algoritmos de *machine learning*. Los modelos basados en estadística,

son desarrollados con base en análisis de la distribución de los datos y de la probabilidad de predicción de posibles resultados, a partir de una ecuación matemática y de los parámetros que mejor estén relacionados con los datos analizados. Los modelos basados en algoritmos de *machine learning*, cuyo objetivo es encontrar los datos mejor relacionados (patrones y reglas) y evaluar los resultados de los algoritmos que mejor se ajusten a la solución del problema (Urcuqui y cols., 2018).

4.2.4. Metodologías

En la actualidad se pueden encontrar varias propuestas metodológicas para la aplicación de la ciencia de datos: CRISP-DM (*Cross-Industry Standard Process for Data Mining*) , ASUM-DM (*Analytics Solutions Unified Method for Data Mining*) y SEMMA (*Sample, Explore, Modify, Model, and Assess*)(Urcuqui y cols., 2018).

4.3. Ciberseguridad

La ciberseguridad es el área de las ciencias de la computación, cuyo objetivo es el desarrollo e implementación de métodos, que garanticen la protección de la información y la infraestructura tecnológica, asegurando la confidencialidad, integridad y disponibilidad de los datos. También se le da importancia al no repudio, que los actores no puedan negar haber realizado alguna acción, por ejemplo, enviar un correo electrónico (Christen, Gordijn, y Loi, 2019).

En el campo de la ciberseguridad se han realizado diferentes propuestas entre las cuales se encuentran las siguientes: marcos de trabajo para la evaluación de ciber amenazas; prácticas y herramientas para el desarrollo de *software* y *hardware* seguros, entre otros. Lo anterior ha permitido defender los sistemas contra cibercriminales y *software* malicioso; sin embargo, el desarrollo de las tecnologías de la información (TIC), la presencia de nuevas vulnerabilidades y los ataques del día cero han hecho que la ciberseguridad requiera de investigación constante para mitigar los riesgos (Urcuqui y cols., 2018). Actualmente hay diferentes herramientas y técnicas para el análisis de amenazas cibernéticas, algunas de ellas se pueden apreciar en la **Tabla 4.2**.

Cuadro 4.2: Métodos para el análisis de amenazas cibernéticas (Urcuqui y cols., 2018)

Métodos	Descripción
Análisis estático	Técnica que evalúa los comportamientos maliciosos en el código fuente, los datos o los archivos binarios, sin ejecutar directamente la aplicación. Su complejidad ha aumentado debido a la experiencia que han adquirido los cibercriminales en el desarrollo de aplicaciones. Se ha demostrado que es posible evadir este análisis a partir de técnicas de ofuscación.
Análisis dinámico	Métodos automatizados que estudian el comportamiento del *malware* en ejecución mediante un análisis de la interactividad del atacante y permiten evaluar características que solo pueden ser obtenidas mientras el *software* está en funcionamiento, como, por ejemplo: la inyección de código en ejecución, los procesos en ejecución, la interfaz de usuario, las conexiones de red y la apertura de sockets. Existen técnicas que permiten evadir este análisis donde el *malware* tiene la capacidad de detectar ambientes sandbox y detener su comportamiento malicioso
Análisis híbrido	Método que combina las ventajas de la aplicación de los análisis dinámico y estático.
Inteligencia artificial	Área que provee de una serie de técnicas para dar soluciones aproximadas a problemas complejos. Una de ellas, el *machine* learning tiene como propósito proveer a los sistemas de la capacidad de aprender cómo identificar a un *malware* sin ser programado de forma explícita. Gran cantidad de propuestas usan algoritmos de clasificación, tales como: Support Vector Machines (SVM), Neural Network NN), Decision Tree (DT) y Naïve Bayes (NB). Existen otras aplicaciones de la inteligencia artificial, como, por ejemplo, las técnicas de programación genética para la detección de anomalías en peticiones HTML.

Para la detección de ciberataque se han propuestos dos marcos de trabajo: el método basado en firmas, que tiene como finalidad la detección de amenazas a partir de una base de datos que contiene distintas características (firmas) de peticiones o archivos, maliciosos; y el método de detección por anomalías, que tiene dos actividades, una dedicada a la construcción de un perfil del sitio de análisis a partir de ciertas variables, y otra enfocada en el monitoreo y la detección de anomalías (cambios no registrados) en un perfil previamente creado.

4.3.1. Ataques ciberneticos "sin malware"

Existen ciberataques que no necesitan usar archivos en el disco de dispositivo, ya que se aprovechan de procesos, aplicaciones y protocolos autorizados para comprometer la seguridad de un sistema. Este tipo de ataques cibernéticos tienen la capacidad de obtener el control de computadoras o partes de ellas, como el procesador (Viscuso, 2017).

4.3.2. Mitigaciones existentes contra el *criptoyacking*

Ante la amenaza que representa el criptoyacking importantes entidades de seguridad como el Centro Nacional de Seguridad Cibernetica del Reino Unido, han emitido recomendaciones para desarrolladores y administradores sobre la Integridad de Sub-Recursos (SRI) y Política de Seguridad de Contenido (CSP). SRI es un protocolo que verifica scripts basados en hash que permite a un sitio validar la integridad del script al cual se emite peticiones. CSP es un servicio de lista blanca para descargar scripts de terceros (Carlin y cols., 2018).

La mayoría de los navegadores han implementado complementos de listas negras con la finalidad de bloquear servicios de minería reportados. Además algunos de los navegadores han bloqueado extensiones mineras.

Algunos ataques de criptoyacking basados en navegador crean ventanas emergentes translúcidas u ocultas para que se siga ejecutando el minero después de que la víctima haya cerrado el navegador (Carlin y cols., 2018).

A nivel de red existe la inspección de paquetes que se enfoca en identificar protocolos de minería como Stratum en cada paquete. También está la estrategia de identificar direcciones de IP relacionadas con servidores de minería. Además de la resolución de dominios que busca encontrar llamados a direcciones de sitios de minería. Lamentablemente no son siempre factibles en la detección de parásitos mineros.

4.4. Machine learning

Machine Learning es el área de la inteligencia artificial que busca que un sistema tenga la capacidad de aprender, sin que sea programado de forma explícita.

El aprendizaje supervisado junto al aprendizaje no supervisado y por refuerzo son tres técnicas para el entrenamiento de modelos. El aprendizaje supervisado se tiene conocimiento de los datos de entrada y la respuesta asociada, con la finalidad de que el modelo aprenda gradualmente para realizar predicciones a partir de nuevos datos de entrada. En el aprendizaje no supervisado el algoritmo aprende a partir de datos de entrada, pero no de respuesta asociada, por lo que su objetivo es encontrar la estructura y los patrones en la información; en el aprendizaje por refuerzo busca que el algoritmo tenga la capacidad de encontrar un conjunto de operaciones, para el cumplimiento de una tarea, por medio de aprendizaje de reglas y acciones.

Un algoritmo de *machine learning* puede ser usado dependiendo del contexto y del tipo de aprendizaje. Estos algoritmos presentan la siguiente clasificación: algoritmos de clasificación, que se utilizan para realizar la predicción de una categoría; algoritmos de regresión, que predicen un valor continuo; algoritmos de detección de excepciones, que son aplicados para la identificación de anomalías; clustering, que se usan para encontrar grupos de elementos similares; algoritmos de asociación, que se aplican para identificar reglas de concurrencia; algoritmos de secuencia, que se utilizan en requerimientos o problemas en los cuales se requieren identificar sucesiones de eventos; algoritmos de resumen, que se usan para simplificar la representación de una información; y algoritmos de visualización, que se aplican para el descubrimiento e interpretación (Urcuqui y cols., 2018).

Los problemas de clasificación se pueden evaluar con diferentes medidas de eficacia y desempeño (ver **Tabla 4.3**) en la aplicabilidad de los algoritmos. Entre estas medidas, se pueden identificar cuatro muy usuales que hacen parte de la tabla de confusión (**Figura: 4.3**).

Cuadro 4.3: Medidas de evaluación y de desempeño de la eficacia de los algoritmos de *machine learning*

(Urcuqui y cols., 2018)

Tipo	Medida	Descripción
Confusión	Verdaderos Positivos (VP)	Tasa de instancias identificadas correctamente y que hacen parte de su respectiva clase.
	Falsos Negativos (FN).	Tasa de instancias que se identificaron incorrectamente, pero no hacen parte de una clase especifica.
	Falsos Positivos (FP).	Tasa de datos que fueron identificadas incorrectamente y que pertenecen a una clase especifica.
	Verdaderos Negativos (VN).	Tasa de instancias que fueron identificadas correctamente, pero que no pertenecen a una clase especifica.
Desempeño	Desempeño	Probabilidad de obtener un verdadero positivo.
	Error	Tasa de instancias incorrectamente identificadas de todos los datos estudiados.
	Exactitud (*Accuracy*).	Proporción de datos que fueron correctamente identificados a través de todas las instancias utilizadas.
	Especificidad	Probabilidad de obtener un verdadero negativo.
	Recuperación (*recall*).	Proporción de datos correctamente clasificados contra el total de datos de su clase
	Precisión.	Tasa de datos identificados que son realmente relevantes.

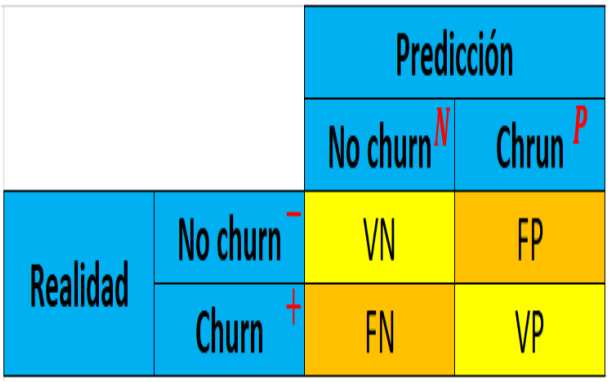

Figura 4.3: Medidas de confusión para problemas de dos clases

Estado del arte

Para el estado del arte de este proyecto, incluimos cinco trabajos que abordan de manera total o parcial una propuesta similar a la del presente proyecto.

5.1. *Detection of Bitcoin miners from network measurements*

Este es un proyecto (Muñoz y Ros, 2019) de investigación donde se presenta un método basado en el aprendizaje automático, usando el algoritmo J48 para la detección de mineros utilizando mediciones de *Netflow/IPFIX*. Con una precisión cercana a la inspección de paquetes y ahorrando recursos considerables, puesto que lo hace sin inspeccionar todo el contenido del paquete.

El proyecto se realizó en un ambiente controlado para simular actividad de cryptojacking, donde se utilizó Netflow v5, dos computadoras que representaban el servidor de minería y el equipo que estaba minando, además de utilizar los siguientes software: *Softflowd* como exportador, *nfcapd* como recopilador y *nfdump* para transformar los archivos *nfcapd* en datos legibles.

Durante su ejecución, se hicieron capturas de tráfico con la finalidad de identificar el protocolo minero *stratum* y cuáles son los métodos que utiliza para permitir la comunicación entre el servidor-cliente. El investigador pudo determinar que el servidor trasmite datos a un ritmo constantemente, ya que, sigue trasmitiendo los datos necesarios para extraer la criptomoneda, mientras que el equipo afectado trasmite poca información, normalmente durante el inicio de la conexión o cuando comparte recursos con otro malware minero. También se identificó variantes del protocolo *stratum*, los cuales usan variaciones de los métodos de comunicación. El uso de *Netflow* permitió extraer datos que podrían ser relevante para identificar *stratum* de tráfico, como el tiempo de inicio y finalización de la conexión, número de paquetes, número de *bytes*, protocolos y banderas.

Se determinó que el tráfico generado por mineros de las criptomonedas podría llegar a ser menor, al tráfico generado por un usuario que ingresa a sitios de juegos y películas. También se evidenció que no hay un tráfico homogéneo cuando se extrae diversas criptomonedas. Además, existe el riesgo de tener tráfico encriptado, que no se pueda identificar.

El investigador utilizó algoritmos de clasificación como DT *Decision tree learning*, SVM *Support Vector machine*, *Naïve Bayes* y C4.5. El autor probo los algoritmos en un entorno real, donde comprobó que *Support Vector machine* y *Naïve Bayes* tienen pésimo rendimiento, y que los algoritmos basados en árboles de decisión tienen un mejor rendimiento con un *Accuracy* del 99 % y una precisión del 98 %, como la implementación del algoritmo J48, dentro de un conjunto de 1795394 datos.

5.2. *How You Get Shot in the Back: A Systematical Study about Cryptojacking in the Real World*

Este proyecto (Hong y cols., 2018) de investigación se enfocó en la naturaleza del criptojacking y en componentes de la carga de trabajo de minería: computación regular, repetida y basada en *hash*. Para el desarrollo de este proyecto, los investigadores crearon CMTracker, un detector basado en el comportamiento con dos perfiladores de tiempo de ejecución, para rastrear de forma automática *scripts* de minería y sus dominios relacionados. El primer perfilador, basado en *hash*, aprovecha la naturaleza de un sistema de prueba de trabajo, para la detección de funciones de *hash* de bajo nivel. El segundo perfilador, basado en estructuras de pilas de llamadas de *scripts* de minería, registra pilas de tiempo de ejecución e identifica las páginas de minería buscando el punto de acceso de los contextos de llamadas.

El generador de perfiles basados en *hash* se centra en funciones de *hash* de bajo nivel. Usaron nueve interfaces de biblioteca de *hash* accesibles comunes, que se identifican mediante un conjunto de firmas fijas (por ejemplo, *"cryptonight_hash", "sha256", "crypto"*) de múltiples criptomonedas de código abierto o servicios de minería comercial. Después calcularon el tiempo acumulado de los sitios web que dedicaron al *hash* para identificar si una página web está minando. Como los sitios web normales generalmente dedican muy poco tiempo a procesar funciones de *hash*. Por otro lado, los *scripts* de minería de criptomonedas dedican la mayor parte del tiempo al *hash*.

El generador de perfiles basado en estructura de pila se usa para la detección de patrones repetidos, revelados por la pila de ejecución de sitios mineros, ya que rara vez un sitio web repite la misma pila de llamadas durante más del 5,60 % del tiempo de ejecución. Dado que la minería de criptomonedas es pesada, para evitar cualquier cosa notable por el usuario, la mayoría de las tareas de minería no se realizarán en el hilo principal al cargar la página web. En cambio, los mineros prefieren crear uno o más hilos dedicados.

Para la época en que se desarrolló esté proyecto, CMTracker identifico 868 dominios que contenían cryptojacking entre todos los principales sitios web de Alexa de 100K. También detectó 1902 dominios en links externos, el 53,9 % de estas muestras identificadas no se hubieran podido detectar con listas negras. A partir de los 2.770 sitios web de cryptojacking detectados por CMTracker, estimaron que afectan a 10 millones de usuarios web por mes.

Los investigadores recolectaron 853.936 muestras de páginas web como el conjunto de datos. Aprovecharon la interfaz para registrar y perfilar las páginas visitadas por muestreo de pila. En segundo lugar, detectaron páginas afectadas por la amenaza minera usando dos perfiladores basados en comportamiento. Después se identificó si el sitio correspondía a minería benigna. Por último, detectaron las páginas con presencia de actividad de cryptojacking.

Los investigadores determinaron que las cargas de trabajo de las muestras de criptojacking costarían un promedio de 278K kWh de energía eléctrica por día, equivalente al consumo de energía de una pequeña ciudad con 9,3K personas. Para realizar la estimación anterior se tuvo en cuenta la cantidad de visitantes de cada dominio identificado, la duración promedio que permanece un visitante, y la potencia de la CPU disponible para extraer en los navegadores.

$$Energy = \sum Visitors \times Duration \times PowerCPU$$

5.3. *The Browsers Strike Back: Countering Cryptojacking and Parasitic Miners on the Web*

Esta investigación (Tahir y cols., 2019) explora el cryptojacking del navegador donde los mineros se implementan en secreto dentro del código de navegador, sin el conocimiento del usuario.

Se analizaron 50.000 sitios web de la lista de Alexa, donde se encontró un porcentaje notable de sitios que practican cryptojacking a menudo usando código ofuscado. También se afirma que complementos como NoMiner no pueden mostrar instancias ocultas. Por lo tanto, se propuso una solución de aprendizaje automático basado en el perfil asistido por *hardware* de código del navegador en tiempo real. Además de proponer una microarquitectura de grano fino que permite calcular las aplicaciones de minería con 99 % de precisión e incluso conocer si el código de minería ha sido ofuscado o encriptado. Se creó una extensión de navegador que aplica todo el conocimiento anterior con una sobrecarga insignificante en la máquina del usuario.

Para desarrollar el producto los investigadores recopilaron un conjunto de datos de sitios web y se analizó cada uno para encontrar mineros ocultos. Identificando las principales categorías de sitios web (transmisión de video, sitios para adultos, torrents, etc.) que son predominantemente abusados por cryptojackers. Aprovechando los desarrollos de hardware de los últimos años, se propició el uso de contadores de rendimientos de *hardware* (HPC, *hardware performance counters*) para realizar actividades en el navegador y crear un clasificador de bosque aleatorio con una precisión del 99,35 %. El esquema de monitoreo se basa en el algoritmo de PoW de una moneda, por lo tanto, detecta la actividad minera independientemente de la ofuscación o las técnicas evasivas.

Los Contadores de Rendimiento de Hardware (HPC) son registros internos de un procesador que representan el estado del sistema en un momento dado. Los valores de estos registros resaltan las características de los programas que ejecuta el sistema. Estos contadores se pueden sondear con bastante rapidez y pueden proporcionarnos una idea de los comportamientos de los programas que se ejecutan (a nivel de microarquitectura).

El enfoque del proyecto fue ejecutar la muestra de sitios web y registrar los valores de los HPC, para luego clasificar el comportamiento normal del usuario y la actividad minera. Primero, el sitio web se ejecuta durante un cierto tiempo para cargar y representar el DOM (*Document Object Model*) HTML(*HyperText Markup Language*). En segundo lugar, los contadores de rendimiento de *hardware* de todo el sistema se sondean, registran y marcan para cada sitio web. Por último, los datos se pasan a un clasificador de aprendizaje automático para su predicción. Este algoritmo se denomina Random Forest, y funciona al construir un conjunto de árboles de decisión y decidir el resultado final basado en la votación del conjunto. Se dividió el conjunto de entrenamiento del 80 % y 20 % de prueba, para después hacer cross validation 10 veces.

Para la época en que se desarrolló este proyecto tenía una precisión del 99,35 % en un conjunto de prueba. El ROC demuestra que el rendimiento del clasificador pasa la tasa de falsos positivos casi al 0 % y verdaderos positivos 100 %.

Este documento presentó un enfoque de aprendizaje automático para marcar y mitigar actividades mineras secretas y ocultas. Mediante el uso de contadores de rendimiento de *hardware* (HPC), identificando con éxito aplicaciones de minería incluso cuando se utilizan técnicas de ofuscación. La precisión fue de 99 %.

5.4. *Detecting cryptocurrency miners with NetFlow/IPFIX network measurements*

En este artículo (i. Muñoz y cols., 2019), se presenta un método basado en aprendizaje automático capaz de detectar criptomonedas utilizando mediciones de red NetFlow / IPFIX. El método no requiere inspeccionar la carga útil de los paquetes, como resultado se logra una detección de mineros rentable y similar a las técnicas basadas en DPI.

Los investigadores identificaron que el protocolo *Stratum* es el responsable de permitir la comunicación entre el *software* y los servidores de minería. Este protocolo tiene un conjunto limitado de mensajes que se utilizan en la comunicación. Los investigadores determinaron que estos son los mensajes utilizados:

- Mining.subscribe: se usa al comienzo de la conexión para indicar al servidor que el cliente está listo para comenzar la minería.
- Mining.authorize: envía información de identificación al servidor.
- Mining.notify: indica al cliente los datos que se van a utilizar para minar.
- Mining.set_difficulty: mensaje corto utilizado por el servidor para indicar el nivel de dificultad del minero

La **Figura 5.1** muestra los datos de la capa TCP trasmitidos entre minero (rojo) y un servidor de grupo de minería (azul) durante una comunicación.

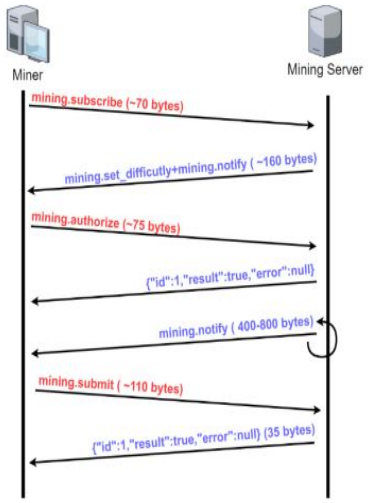

Figura 5.1: TCP connection between a miner (red) and a pool server (Blue) using the Stratum protocol

Los investigadores aclaran que, aunque *Stratum* es el protocolo más utilizado para minar, existen otros protocolos como getblocktemplate.

Los investigadores procesaron los datos generados por el tráfico de minería a través *Netflow*, donde seleccionaron los datos relacionados con el enrutamiento, ignorando datos relacionados con la administración de red. También tuvieron en cuenta datos relacionados con el número de paquetes y *bytes* relacionados con la duración total de los flujos del procesamiento de *Netflow*. Se determinó que las conexiones de *Stratum* son asimétricas, mientras el servidor transmite una gran cantidad de datos a los clientes, el cliente envía una

cantidad muy pequeña de datos al servidor. Decidieron combinar los registros de *Netflow* correspondiente a la salida y flujos entrantes para explotar la asimetría.

Lo primero que realizaron fue recoger el flujo de las mediciones del tráfico obtenido. Cada flujo tomó entre 5 y 30 minutos. Para recolectar suficientes datos para entrenar a un modelo de aprendizaje automático generando tráfico de minería por su propia cuenta, ya que no tenían acceso a una gran cantidad de capturas de tráfico. De esta manera, realizaron minería en diferentes servidores de criptomonedas y capturaron el tráfico. El conjunto de datos recolectados era de 1.795.408.

Los investigadores probaron diferentes modelos de *Machine Learning* en WEKA. Los modelos seleccionados y evaluados fueron:Máquinas de vectores de soporte (SVM), CART, árbol de decisión C4.5 y Naïve Bayes. Entre los modelos evaluados el que tuvo mejor resultado fue el árbol C4.5. La máxima profundidad del árbol es de 13 nodos. Además, en la mayoría de los casos emplea solo 5 operaciones divididas para clasificar correctamente el tráfico.

Los resultados de la evaluación muestran que el método basado en C4.5 fue capaz de detectar con precisión el tráfico generado por la minería. La principal ventaja del método es que solo necesita procesar una cantidad reducida de datos de *NetFlow* para clasificar el tráfico de minería.

5.5. *Dine and Dash: Static, Dynamic, and Economic Analysis of In-Browser Cryptojacking*

Este trabajo(Saad, Khormali, y Mohaisen, 2019) investigativo analiza los aspectos de los ataques del criptoyacking de manera estática, dinámica y económicamente. Para el análisis estático realizaron categorizaciones basadas en contenido, moneda y código de muestras de criptojacking para medir su distribución en sitios web, afinidad con la plataforma minera y la complejidad del código. Los investigadores aplicaron aprendizaje no supervisado para el análisis estático, con la finalidad de distinguir los *scripts* de *Criptojacking* de los benignos y otros ejemplos maliciosos de JavaScript, con una precisión del 96,4 %.

Para el análisis dinámico analizaron el efecto del criptojacking en los recursos críticos del sistema, como uso de CPU y batería. Además, realizaron huellas digitales del navegador web para analizar intercambios entre nodo víctima y el servidor de minería. También construyeron un modelo analítico para evaluar la viabilidad del cryptojacking como alternativa a la publicidad en línea.

5.6. Matriz del estado del arte

En la tabla **Tabla 5.1** están los criterios por los cuales fueron seleccionados los trabajos cientificos del estado del arte.
Criterios de selección:

- *Machine Learning*: este criterio indica si el trabajo utilizo técnicas de *Machine Learning*.
- *Behavior Hardware*: criterio que hace referencia si en la investigación abordó las variables de *hardware*.
- *Netflow*: criterio que demuestra el uso de este protocolo para resumir el trafico.
- *Network Features*: este criterio indica el uso de variables de red.

Cuadro 5.1: Criterios de comparación del estado del arte

	5.1	5.2	5.3	5.4	5.5
Machine Learning	SI	NO	SI	SI	SI
Behavior Hardware	NO	SI	SI	NO	SI
Netflow	SI	NO	NO	SI	NO
Network Features	SI	NO	SI	SI	SI

Metodología

6.1. CRISP-DM

Para este proyecto de investigación se aplicó la metodología CRISP-DM (*Cross-Industry Estándar Process for Data Mining*), puesto que su estructura beneficia la resolución de problemas relacionados con ciencia de datos, *Machine Learning* y minería de datos. Esta metodología consta de seis etapas que se pueden ver en la **Figura 6.1**.

Figura 6.1: Metodologia CRISP-DM
Pete Chapman et al., 2000

6.2. Entendimiento de negocio

Durante esta etapa se procedió a recolectar datos a nivel de red y hardware de actividad minera y normal. Después se procedió a realizar un análisis exploratorio univariado y bivariado para entender el comportamiento de los datos, como se relacionan entre ellos y detectar anomalías presentes. Cabe resaltar que solo se llegara a esta parte con las variables de *hardware*.

6.2.1. Nivel de red

Durante esta fase, el investigador Zayuelas Muñoz del proyecto (Muñoz y Ros, 2019) brindo acceso a una parte del *dataset* de capturas de tráfico que usaron en su respectiva investigación. Este conjunto se denominó minería pura, puesto que se conoce que criptomoneda le corresponde a cada registro. El conjunto de capturas es de cinco tipos de criptoactivos:

- Bitcash: 79 archivos pcap
- Bitcoin: 41 archivos pcap
- Ethereum: 106 archivos pcap
- Litecoin: 77 archivos pcap
- Monero: 55 archivos pcap

También se recopiló archivos pcaps de tráfico de red que están almacenados en sitios web como packetTotal, malware-traffic, contagiodump, entre otros, que tienen evidencia de actividad de cryptojacking en entornos no controlados. Pese a que son sitios con una popularidad respetable no son apoyados por una entidad académica o gubernamental, a pesar de esto la evidencia presentada esta lo suficientemente documentada para tenerla en cuenta en esta investigación y crear un conjunto de datos, al cual se le denominó minería no pura, ya que no se conoce que criptomoneda se obtuvo. Este conjunto consta de treinta y nueve archivos, cabe resaltar que algunos contienen también tráfico de otros *malware* o actividad maliciosa.

Para el tráfico normal se desarrolló un conjunto de herramientas para automatizar el proceso de captura, usando como fuente los sitios más visitados según Alexa. La primera herramienta comprueba el estatus de los sitios web, para saber si está activa o no. La segunda utiliza el API de VirusTotal, para determinar si la URL es benigna o no, en este proceso se toma como criterio las puntuaciones de todos los motores de antivirus de VirusTotal, si uno califica la URL con una puntuación negativa es descartada del proceso. Con la lista de sitios web benignos, la tercera herramienta ejecuta Selenium para abrir el navegador durante seis minutos redireccionando el sitio web, y a su vez ejecuta tshark para hacer la captura de tráfico. De este proceso se obtuvo un total de 1284 archivos pcap. Este proceso se realizó en un entorno semicontrolado donde se usó una computadora con un sistema operativo Linux Ming 19.3 Tricia, donde solo se mantuvieron los procesos vitales para su funcionamiento.

Se utilizó un conjunto de herramientas que hacen parte del proyecto nfdump (Haag y otros, 2015), para procesar los tres conjuntos de datos en archivos nfpcaps. Se procedió a utilizar una herramienta desarrollada en el proyecto de investigación (Gaviria, Ramirez, Urcuqui, y Navarro, 2020) para transformar los archivos nfpcacps en ventanas de tiempo, con la finalidad de resumir las comunicaciones de los flujos activos.

Del anterior proceso de transformación se obtuvo lo siguiente:

- El conjunto normal tiene 2501 registros de ventanas de tiempo.
- El conjunto de minería no pura tiene 37053 registros de ventanas de tiempo.
- El conjunto de minería pura tiene 2232 registros de ventanas de tiempo.
    - Bitcash con 485 registros.
    - Bitcoin con 247 registros.
    - Ethereum con 959 registros.
    - Litecoin con 349 registros.
    - Monero con 192 registros.

Los tres conjuntos de datos contienen las siguientes variables:

1. Name: Nombre de la ventana de tiempo.
2. Netflows: Cantidad Netflows en la ventana de tiempo.
3. First_Protocol: Top 1 de los protocolos usados en la ventana de tiempo.
4. Second_Protocol: Top 2 de los protocolos usados en la ventana de tiempo.
5. Third_Protocol: Top 3 de los protocolos usados en la ventana de tiempo.
6. P1_d: 25 % de los percentiles de todas las duraciones en la ventana de tiempo.
7. P2_d: 50 % de los percentiles de todas las duraciones en la ventana de tiempo.
8. P3_d: 75 % de los percentiles de todas las duraciones en la ventana de tiempo.
9. Duration: Duración total de la ventana de tiempo.
10. Max_d: Valor máximo de todas las duraciones en la ventana de tiempo.
11. Min_d: Valor mínimo de todas las duraciones en la ventana de tiempo.
12. Packets: Número total de paquetes en la ventana de tiempo.
13. Avg_bps: Promedio de *bits* por segundo en la ventana de tiempo.
14. Avg_pps: Promedio de paquetes por segundo en la ventana de tiempo.
15. Avg_bpp: Promedio de *bytes* por paquete en la ventana de tiempo.
16. Bytes: Número total de *bytes* en la ventana de tiempo.
17. Number_sp: Número total de puertos de origen usados en la ventana de tiempo.
18. Number_dp: Número total de puertos de destino usados en la ventana de tiempo.
19. First_sp: Top 1 de los puertos de origen en la ventana de tiempo.
20. Second_sp: Top 2 de los puertos de origen en la ventana de tiempo.
21. Third_sp: Top 3 de los puertos de origen en la ventana de tiempo.
22. First_dp: Top 1 de los puertos de destino en la ventana de tiempo
23. Second_dp: Top 2 de los puertos de destino en la ventana de tiempo
24. Third_dp: Top 3 de los puertos de destino en la ventana de tiempo
25. P1_ip: 25 % de los percentiles de todas las entradas de paquetes en la ventana de tiempo.

26. P2_ip: 50 % de los percentiles de todas las entradas de paquetes en la ventana de tiempo.
27. P3_ip: 75 % de los percentiles de todas las entradas de paquetes en la ventana de tiempo.
28. P1_ib: 25 % de los percentiles de todas las entradas de bytes en la ventana de tiempo.
29. P2_ib: 50 % de los percentiles de todas las entradas de bytes en la ventana de tiempo.
30. P3_ib: 75 % de los percentiles de todas las entradas de bytes en la ventana de tiempo.
31. Type: Tipo de ventana de tiempo (Minera/Normal)

6.2.2. Nivel de *hardware*

Se exploró la actividad del cryptojacking a nivel de *hardware* usando un conjunto de datos de una cuenta de *Kaggle*, la cual pertenece a la investigadora Keshani Jayasinghe, aunque el *dataset* no está asociado directamente a un proyecto de investigación, la autora es reconocida por investigar esta temática.

El conjunto de datos incluye dos archivos CSV, como se describe a continuación:

- Datos anormales con 14461 registros.
- Datos normales con 80851 registros.

Estos conjuntos de datos contienen las siguientes variables identificadas:

- cpu_idle: Tiempo en que un procesador está inactivo.
- iowait: Medida del tiempo que los procesos de la CPU pasan sin hacer nada.
- cpu_nice: Tiempo de CPU que está actualmente "en uso".
- cpu_softirq: Interrupciones.
- cpu_total: Uso total de la CPU.
- diskio_sda1_write_bytese: Escritura de *bytes*.
- diskio_sda1_read_bytes: Lectura de *bytes*.

6.3. Definición de experimentos

Durante este proyecto se planteó tres experimentos basados en la información recolectada, la formulación del problema y el entendimiento de los datos.

6.3.1. Primer experimento

Partiendo del hecho de que en la comunicación de dispositivos debe existir un intercambio de información en la red. Un servidor de minería debería de enviar una cantidad de datos al minero instalado en el dispositivo de la víctima, con información del problema matemático, de igual forma el nodo minero respondería con la resolución del problema o realizando peticiones. Un modelo de *machine learning* debería tener la capacidad de identificar un flujo minero de uno normal.

Se procederá a entrenar cuatro modelos de *machine learning* para escoger el mejor modelo que pueda explicar o refutar la hipótesis planteada. Para esta parte se concatenará el conjunto de minería pura y los registros normales. Una vez seleccionado el mejor modelo

se pondrá nuevamente a prueba para comprobar si conserva su rendimiento con la nueva partición, la cual consiste en combinar el 70 % de registros normales con el 100 % de mineros puros.

6.3.2. Segundo experimento

El mejor modelo del primer experimento será lo suficientemente bueno para detectar actividad minera recolectada en ambientes de incertidumbre.

En este experimento se pondrá a prueba el mejor modelo con datos de minería no pura, los cuales fueron recolectados en entornos reales. Este conjunto se le añadió el 30 % de los registros normales descartados en el primer experimento, haciéndolo un conjunto desbalanceado, lo cual no importa, ya que esta unión será usada para validar el mejor modelo, no para entrenar.

6.3.3. Tercer experimento

La combinación de características de los conjuntos de datos de minería pura y no pura, debería permitir a un modelo la identificación de registros mineros, independientemente del estatus o de que entorno fue recolectada la actividad de cryptojacking.

Para responder la hipótesis se procederá a reentrenar el mejor modelo con características de los dos conjuntos de datos de minería, pero como los mineros no puros tienen más cantidad de ejemplos se procedió a estructurar nuevamente los *dataset*, de la siguiente manera: se combinara el 70 % de los registros normales, el 100 % de los registros mineros puros y el 6 % de los registros mineros no puros (equivalente a la misma cantidad de los mineros puros) para entrenar el modelo reforzado. Esta última porción quedó excluida del grupo de minería no pura. Pese a esta combinación el conjunto estaba desbalanceado con respecto a los registros normales, por lo tanto se aplicó un sobremuestreo al 70 % de los datos normales ya seleccionado para tratar de balancear las clases.

6.4. Preparación de los datos

En esta fase se procedió a realizar estructuraciones en los conjuntos de datos para llevar a cabo los experimentos anteriormente mencionados. También se procedió a corregir anomalías y aspectos que afectaran la integridad de los datos, además de aplicar técnicas de ingeniería de características para seleccionar posibles predictores para el entrenamiento de los modelos. Además se estandarizó el conjunto de variables cuantitativas y se aplicó onehotencoder con las variables cualitativas.

Con respecto a la selección de posibles predictores se procedió a utilizar dos métodos: el Anova F-Value y ExtraTreesClassifer. El primero, porque durante la exploración de datos se evidenció que todas las posibles variables de entrada a excepción de una, son de naturaleza numérica. El segundo, porque aleatoriza subconjuntos de datos para reducir el sobreajuste. También se escogió un tercer grupo de características bajo criterio de un experto descrito en (Muñoz y Ros, 2019), en donde se argumenta que la información útil de la comunicación entre los mineros en un flujo se da por medio de los bytes y paquetes, el cual está alineado con el primer experimento, esto se realizó de esta forma, ya que es posible que los métodos usados incluyan variables que no aporten al modelo.

6.5. Modelado

Los modelos seleccionados para el presente proyecto son los siguientes:

- *KNN*
- *SVM*
- *Naive Bayes*
- *Decision Tree*

En esta fase también se procedió a entrenar los modelos con los tres conjuntos de predictores mencionados en la anterior etapa, con la finalidad de descubrir que conjunto de variables es el más óptimo, aunque se espera seleccionar el conjunto basado por el criterio de experto, ya que se alinea con la hipótesis del primer experimento. Todo esto se realizó de la siguiente forma, puesto que es posible que halla una variable que aporte información a los modelos, que no se considere en el criterio del experto.

Se procedió a entrenar cada modelo del primer experimento aplicando un hold-out con una partición del 70 % para entreno y un 30 % para validar. También se aplicó esta estrategia con el mejor modelo del primer experimento y el modelo reforzado del tercer experimento. En el tercer experimento se utilizó cross-validation con diez pliegues con la finalidad de instanciar diez modelos para seleccionar aquel que tuviera un mejor rendimiento, para ponerlo a prueba con los datos que no se usaron en ningún entrenamiento o validación de los modelos previamente.

6.6. Evaluación

En esta fase se utilizaron las siguientes métricas:

- *Baseline*
- *Recall*
- Matriz de confusión
- Especificidad
- *Accuracy*
- *F1-score*
- *Kappa*
- ROC
- *Precision*
- Area Under the Curve

Durante el entrenamiento de todos los modelos en el primer experimento, se le dio importancia al *Accuracy*, puesto que indica que fracción de las predicciones que realizó cada modelo fue correcta. Cabe resaltar que el conjunto que se usó para entrenamiento y prueba esta cerca de estar balanceado.

La matriz de confusión permitió utilizar estimadores de la realidad y estimadores de predicción, para clasificar cuando un registro es procedente de actividad minera o no. El resto de métricas se usarán durante el desarrollo del proceso de experimentación.

6.7. Despliegue

Durante esta fase, el mejor modelo de *Machine learning* se pone en producción en un ambiente real. Este trabajo no llegó hasta esta instancia por su carácter investigativo.

Experimentos y resultados

7.1. Tecnologías empleadas

Se usaron las siguientes tecnologías, descritas a continuación:

Cuadro 7.1: Tecnologías *Hardware*

Tecnología	Sistema operativo	versión	Propósito
Computadora XPS	Linux Mint	19.3 Tricia	Entorno de capturas y procesamiento de datos.
Computadora ASUS	Windows	10	Ejecución de modelos, programación y evaluación.

Cuadro 7.2: Tecnologías *Software*

Tecnología	Versión	Propósito
Python	3.6.9	Codificación
Tshark	2.6.10	Captura de trafico
Selenium	3.141.0	Ejecuta navegador
scikit-learn	0.24.0	algoritmos de machine learning

7.2. Análisis univariado

Este análisis se realizó solo con la primera estructuración de los datos del primer experimento para observar aspectos en la distribución de las variables. Donde se usó el coeficiente de asimetría y curtosis. Para las variables cualitativas se procedió hacer un conteo. Estos fueron los resultados más relevantes:

- Las columnas con datos faltantes son Second_Protocol, Third_Protocol, second_sp, third_sp, second_dp y third_dp. Con un porcentaje de datos perdidos del 52 %, 97 %, 7 %, 46 %, 7 % y 45 % respectivamente.
- El conjunto de datos está casi balanceado.
- El protocolo que más aparece es el TCP, el segundo es UDP y el tercero es ICMP6.
- Todas las características tienen una asimetría hacia la derecha, lo que significa que el promedio es mayor que la mediana en diferentes proporciones respectivamente.
- La única característica cercana a la simetría es Avg_bpp.
- Las características con forma leptocurtica (alta concentración de los valores cerca a su media) son las siguientes: Netflows, duration, packets, Avg_bps, Avg_pps, Bytes, sp, dp, p1_ip, p2_ip, p3_ip, p1_ib, p2_ib y p3_ib.

- Las características con forma Platicúrtica (baja concentración de los valores en torno a su media) son: p1_d, p2_d, p3_d, max_d, min_d, Avg_bpp, first_sp, second_sp, third_sp, first_dp, second_dp y third_dp.
- No hay ninguna variable con una distribución normal.

Este análisis no permite hacer una conclusión, ni tampoco identificar un patrón, ya que el problema que sé está abordando es de naturaleza dicótoma.

7.3. Análisis bivariado

Se procedió hacer un análisis sobre los tres conjuntos (mineria pura, mineria no pura, trafico normal) tomando como referencia las características usadas en el proyecto (Muñoz y Ros, 2019). Cabe resaltar que está investigación usa el concepto de ventanas de tiempo, por lo tanto las conclusiones se hacen a partir de las variables equivalentes a las proporcionadas por *Netflow*.

Características usadas en la investigación (Muñoz y Ros, 2019):

- *Packets/second Inbound*
- *Bits/second Inbound*
- *Bits/Packet Inbound*
- *Packets/second Outbound*
- *Bits/second Outbound*
- *Bits/Packet Outbound*
- *Packets Inbound/Packets Outbound*
- *Bits Inbound/Bits Outbound*

Variables que se analizaron:

- Promedio de bits por segundo (Avg bps)
- Promedio de paquetes por segundo (Avg pps)
- Promedio de bytes por paquete (Avg bpp)
- Percentil 75 % de todas las entradas de paquetes (p3 ip)
- Percentil 75 % de todas las entradas de bytes (p3 ib)

Aspectos relevantes encontrados:

El tráfico de los registros normales tiene una probabilidad más alta de aparecer que el de los mineros, además de que tienen un rango más amplio en los promedios de bits por segundo (**Figura 7.1**). Sin embargo, se puede observar que los registros de minería pura presentan unos picos en la probabilidad de presentarse en los primeros intervalos de tiempo.

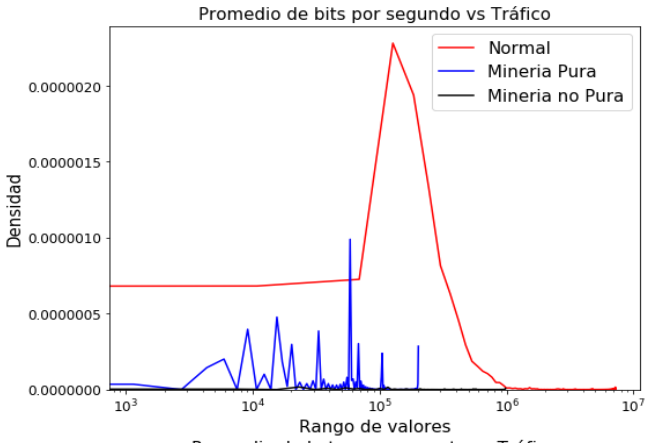

Figura 7.1: Promedio de bits por segundo vs Tipo tráfico

El tráfico de los registros mineros puros tiene una probabilidad mayor en los primeros promedios de paquetes, que el tráfico normal. La probabilidad de que aparezca minería no pura es casi nula, por lo menos en los flujos activos de las muestras (**Figura 7.2**).

Figura 7.2: Promedio de paquetes por segundo vs Tipo tráfico

En la **Figura 7.3** se aprecia que en la trasmisión de bytes en promedio fue mayor en los mineros no puros. Seguido de los mineros puros, lo cual indica que hubo una mayor trasmisión de bytes por paquetes en procesos de minería.

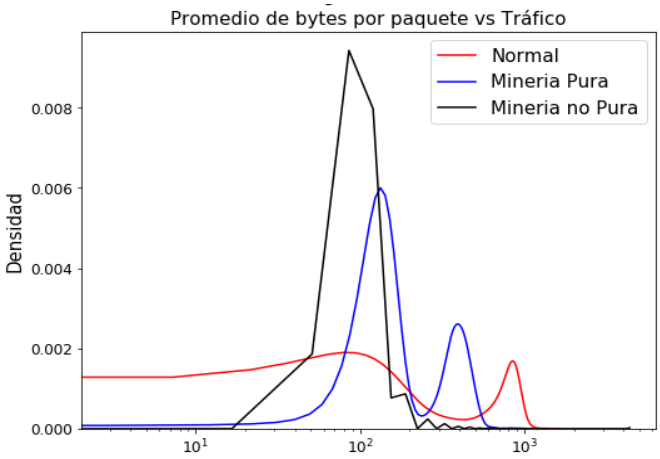

Figura 7.3: Promedio de bytes por paquete vs Tipo tráfico

En la figura **Figura 7.4** se muestra que la probabilidad más alta se presenta en los registros normales en las primeras entradas de paquetes. A pesar de tener poca probabilidad, los mineros puros pueden tomar una ocurrencia en un rango más amplio de valores en paquetes. En cambio la presencia de minería no pura es muy baja, al igual que su ocurrencia.

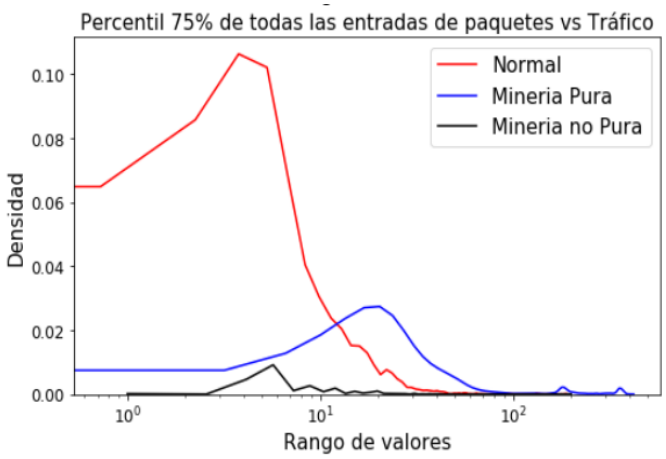

Figura 7.4: Percentil 75 % de todas las entradas de paquetes vs Tipo tráfico

El 75 % de los percentiles de todas las entradas de bytes de los registros normales es más alta en las primeras instancias de paquetes. Los mineros no puros incrementan su ocurrencia, cuando se reduce la de los registros normales, con respecto a los datos mineros puros se presenta una probabilidad menor, pero más prolongada en aparecer (**Figura 7.5**).

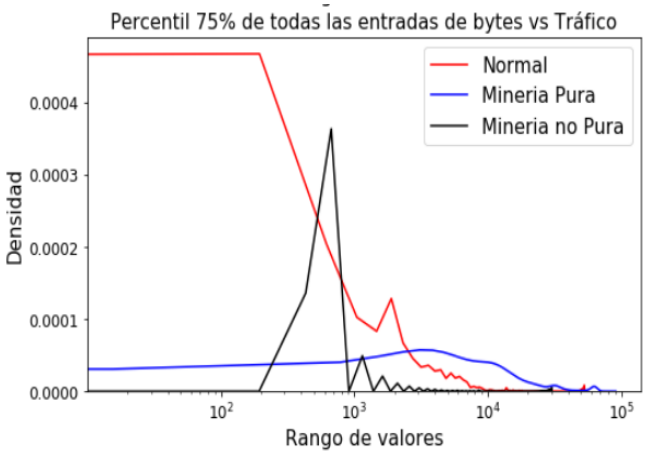

Figura 7.5: Percentil 75 % de todas las entradas de bytes vs Tipo tráfico

7.4. Preprocesamiento

Durante la fase de preprocesamiento se procedió a imputar los valores faltantes de las características second_sp, third_sp, second_dp y third_dp con el criterio de la mediana, por su robustez ante valores atípicos. En cambio, se tuvo que eliminar las características de Second_Protocol y Third_Protocol debido a que sus registros tenían un porcentaje de valores perdidos del 52 % y 97 %, ante lo cual no es factible eliminar sus registros, ni tampoco usar otras estrategias de imputación como el dato más frecuente, constantes o implementar modelos de aprendizaje automático, por su complejidad o incremento de sesgo.

Con respecto a los valores atípicos se decidió no eliminarlos (la mayoría de variables superan el 9 %) o modificarlos, ya que el proceso de inferencia podría verse afectado, puesto que se puede introducir sesgo, afectar la distribución de las varianzas o el tamaño muestral. Además se debería tener una autoridad en la materia para determinar hasta que punto se podría tolerar lo atípico con respecto a la temática abordada, lamentablemente no se encontró en la literatura un límite ante lo anterior. Se optó por dejar los datos atípicos para identificar posible anomalías en el tráfico.

Se identificó que la mayoría de las variables de este conjunto presentan una alta multicolinealidad por medio de la inflación de la varianza, ya que los resultados superan el cinco, valor usado como referencia para la multicolinealidad. La siguiente es la fórmula:

$$\frac{1}{1 - R^2}$$

Donde R^2 es el coeficiente de determinación.

7.5. *Feature selection*

En el conjunto de datos procesados, las variables de entrada son de naturaleza numérica y la variable objetivo es categórica. Para poder seleccionar los mejores predictores se escogió principalmente el método de ANOVA-F para calcular la relación entre los valores de la varianza (Brownlee, 2020). Donde se ajustó la función seleccionada con el parámetro K de la clase SelectKBest de la librería scikit-learn. Este método indicó como mejores predictores las siguientes variables: p1_d, p2_d, min_d, second_sp y second_dp.

El segundo método *ExtraTreesClassifier* selecciono las siguientes características: p1_d, p2_d, min_d, Avg_bpp, first_sp, second_sp, first_dp,second_dp, p2_ib y p3_ib.

Se puede apreciar que solo fue seleccionada una variable (Avgbpp), equivalente a las mencionadas en el trabajo de referencia (Muñoz y Ros, 2019) y otro proyecto del estado del arte (i. Muñoz y cols., 2019). Por tal razón se escogió otros cuatro predictores equivalentes a los mencionados en los trabajos anteriores para tenerlos en cuenta por criterio de experto, por lo tanto las variables son las siguientes: Avgbps, Avgbpp, p3_ip y p3_ib.

7.6. Primer experimento

Se procedió a entrenar los modelos previamente seleccionados en tres iteraciones usando los conjuntos de predictores. Al mismo tiempo se valido cada modelo usando el *Accuracy* para verificar el porcentaje de predicciones que realizaron correctamente.

Metrica	Anova F				ExtraTreesClassifier				Criterio de experto			
	KNN	NB	SVM	DT	KNN	NB	SVM	DT	KNN	NB	SVM	DT
Accuracy	0.99	0.92	0.94	0.99	0.99	0.97	0.99	1.0	0.99	0.78	0.97	0.99

Cuadro 7.3: Comparación de los modelos frente a diferentes predictores

Donde:

- KNN: *k-nearest neighbors*
- NB: *Naive bayes*
- SVM: *Support Vector Machines*
- DT *Decision tree*

Como se aprecia en la tabla (**Cuadro 7.3**) los modelos tuvieron un *Accuracy* muy alto, lo que significa que clasificaron de manera correcta casi todos los registros, sin importar el conjunto de predictores. Cabe resaltar que el modelo de Naive Bayes es el que tuvo el peor desempeño, sobre todo en el tercer grupo de predictores.

Pese a que la tabla sugiere que los dos mejores modelos son el KNN y árbol de decisión, se selecciona el segundo, puesto que es más eficiente, ya que no necesita procesar todos los datos, mientras que KNN en cada iteración tiene que trabajar con todos los datos siendo costoso computacionalmente.

El árbol de decisión fue seleccionado como el mejor modelo, pero como se mencionó en la definición de experimentos se tuvo que hacer estructuraciones a los conjuntos de datos, por tal razón se instancio nuevamente el mejor modelo para comprobar si seguía teniendo el buen rendimiento obtenido previamente.

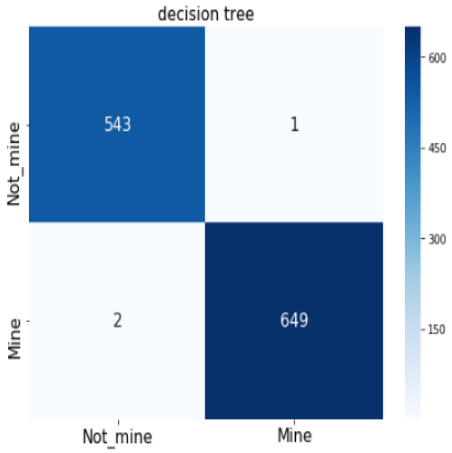

Figura 7.6: Matriz de confusión del mejor modelo

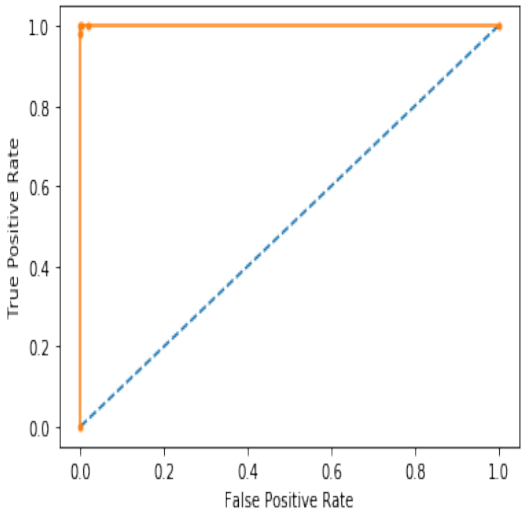

Figura 7.7: ROC mejor modelo

En la matriz de confusión (**Figura 7.6**) se observa que el modelo se equivocó solo tres veces y en sus métricas obtuvo 99 %. Además de que la curva ROC (**Figura 7.7**) indica que la prueba diagnostica tiene una probabilidad del 99 % de discernir cuando hay un registro minero y uno normal. Pese al desbalance que se pueda presentar debido a la estructuración mencionada en la definición de experimentos, el modelo tuvo una métrica excelente en el F1-score del 99 % (ver **Tabla 7.4**).

Este modelo apoya la hipótesis del experimento, ya que a través de los bytes, bits y paquetes en los archivos de red se puede identificar un evento de cryptojacking en un 99 % de las veces.

Cuadro 7.4: Métricas del mejor modelo

Algoritmo	Mineros				Normales				Accuracy	Kappa
	Precisión	Recall	Especificidad	F1-score	Precisión	Recall	Especificidad	F1-score		
Decision Tree	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99

7.7. Segundo experimento

El mejor modelo se puso a prueba con el conjunto de datos de minería no pura, el cual contiene el 30 % de los registros normales. Se espera que conserve su eficiencia para clasificar mineros, aun si es con datos nunca vistos.

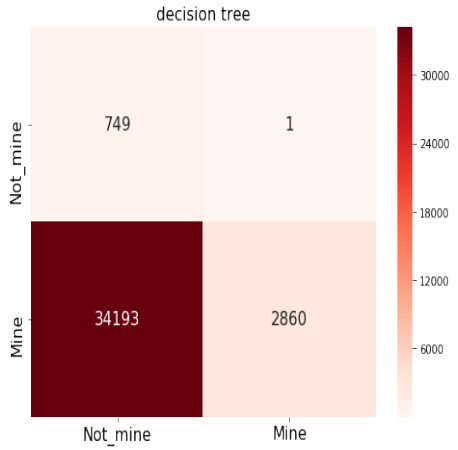

Figura 7.8: Matriz de confusión del mejor modelo vs datos minería no pura

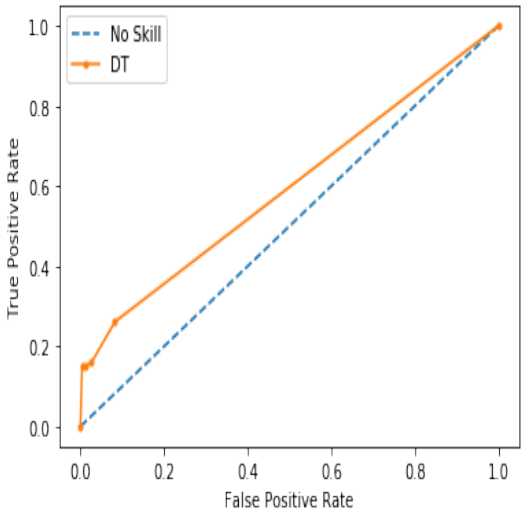

Figura 7.9: ROC Mejor modelo vs datos minería no pura

En la matriz de confusión (ver **Figura 7.8**) el modelo se equivocó 34194 veces a la hora de clasificar los registros mineros de los normales. También se puede observar en la curva ROC (ver **Figura 7.9**) que el AUC (el rendimiento fue de 59 %) no tiene mucha diferencia con respecto a la prueba sin habilidad de diagnóstico, lo que contradice la hipótesis inicial, indicando que el modelo es ineficiente e incapaz de clasificar mineros en ambientes de incertidumbre. Otro aspecto relevante es que el recall en la detección de actividad minera esta por debajo del 10 % (ver **Tabla 7.5**), el interés ante estas métricas es aumentar el recall de la detección de los mineros, ya que si no se detectan de manera oportuna pueden ser causantes de que el equipo afectado se deteriore y se incrementen costos por conceptos de energía.

Cuadro 7.5: Métricas del mejor modelo vs datos minería no pura

Algoritmo	Mineros				Normales				Accuracy	Kappa
	Precisión	Recall	Especificidad	F1-score	Precisión	Recall	Especificidad	F1-score		
Decision Tree	0.99	0.07	0.99	0.14	0.02	0.99	0.0.07	0.04	0.09	0.003

7.8. Tercer experimento

Se procedió a entrenar un nuevo modelo, con la hipótesis de que tendría mejores estadísticas a la hora de predecir, si se combinaba las características de los conjuntos de minería. Se aplicaron las mismas técnicas de ingeniería de características en el conjunto de minería no pura, aquellos predictores fueron: p3_d,min_d,first_sp,first_dp, second_sp y second_dp. Este conjunto se combinó con las variables del tercer conjunto anteriormente mencionado.

Las variables de tiempo fueron seleccionadas porque el modelo se verá beneficiado al tener en cuenta cuanto dura el flujo de trasmisión de bytes en la comunicación de los nodos de minería. Con respecto a al top 1 y 2 tanto de puertos de origen como destino, es por el hecho de que existen cryptojackers con bajas nociones de programación que no configuran los puertos de los mineros adecuadamente. Recordemos que en muchos casos basta con pegar un script en un sitio web para iniciar el proceso de minería.

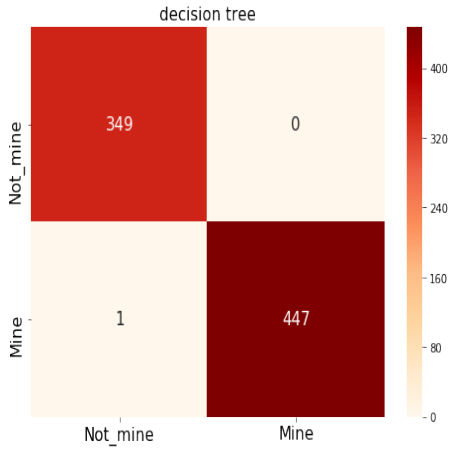

Figura 7.10: Matriz de confusión del modelo robusto

La matriz de confusión (ver **Figura 7.10**) revela que el modelo solo se equivocó una vez, lo que significa que el modelo reforzado es excelente a la hora de clasificar los mineros y no mineros. Pese a que los resultados son muy buenos, existe el riesgo de sobreajuste, por tal motivo se procede usar una validación cruzada para reducir el overfitting con un KFold de diez pliegues, donde se instancia diez modelos que se entrenaron por cada partición, donde se escogió el que tuviera mejor rendimiento. Cabe recordar que el entrenamiento de los modelos en este experimento se realizó con la tercera estructuración mencionada en la definición de experimentos.

El mejor modelo obtenido del *Cross Validation Kfold* se equivocó 122 veces con respecto a 36055 aciertos en la clasificación de mineros y registros normales no procesados en la etapa de entrenamiento (ver **Figura 7.11**). El área de bajo la curva ROC (**Figura 7.12**) indica que la prueba de diagnóstico para determinar si el registro es un minero o no es excelente con una probabilidad del 97 %. Se podría decir que el modelo es muy bueno para la detección de cryptojacking debido a que se ha realizado un proceso y se han obtenido buenas métricas (ver **Tabla 7.6**) como un recall y un accuracy del 99 %, además del F1-score del 99 %.

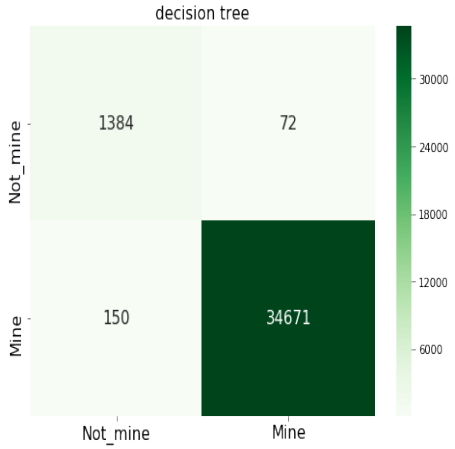

Figura 7.11: Matriz de confusión del modelo robusto vs datos de minería no pura

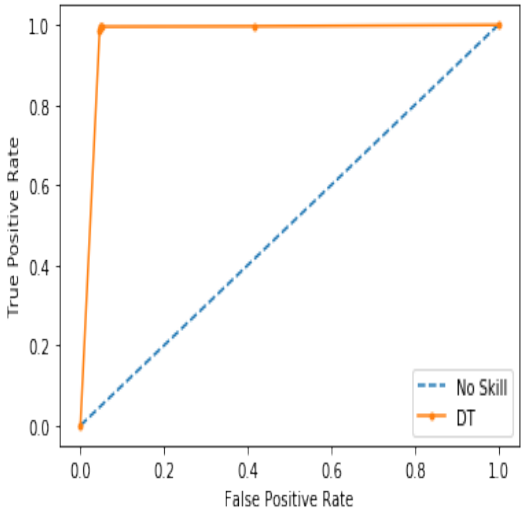

Figura 7.12: ROC modelo robusto vs datos de minería no pura

Sin embargo solo se podrá demostrar la efectividad de este estimador en modo producción. Lamentablemente este proyecto no tiene ese alcance por cuestiones de tiempo y recursos, además de que el propósito inicial fue de investigación.

Cuadro 7.6: Métricas de modelo robusto vs datos de minería no pura

Algoritmo	Mineros				Normales				Accuracy	Kappa
	Precisión	Recall	Especificidad	F1-score	Precisión	Recall	Especificidad	F1-score		
Decision Tree	0.99	0.99	0.95	0.99	0.90	0.95	0.99	0.92	0.99	0.92

7.9. Análisis de variables de hardware

Se analizará el comportamiento de mineros en algunas variables del conjunto de *hardware*:

De acuerdo a la gráfica (**Figura 7.13**) existe una probabilidad alta de que los proceso benignos cuando terminan sus tareas dejan de usar la CPU del equipo, mientras que la CPU afectada por los mineros no suelen estar inactivas durante tanto tiempo, ya que la probabilidad es mínima. Este hecho junto a que algunas CPU´S cuando están suspendidas reducen el consumo de voltaje en tiempo de inactividad, lo que alude al desperdicio de consumo energético por parte de un minero ilegal.

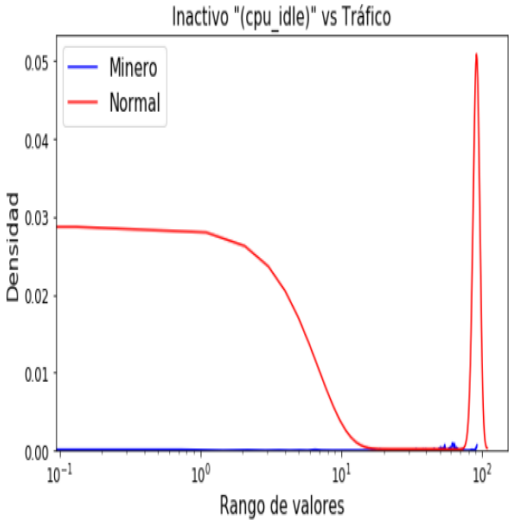

Figura 7.13: Inactivo (cpu_idle) VS Tráfico

De acuerdo a la medida de iowait (**Figura 7.14**), hay más probabilidad de encontrar proceso en espera a leer y escribir en el disco, mientras que los procesos de minería no suelen tardar tanto tiempo. Es probable que un minero cause un cuello de botella en el sistema.

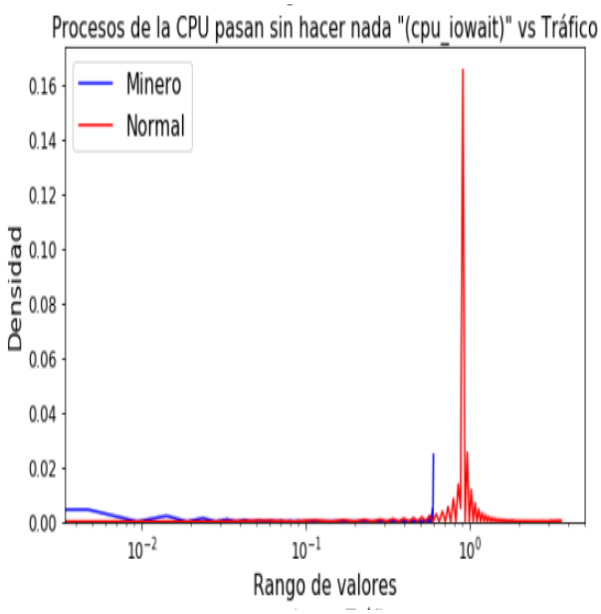

Figura 7.14: Procesos de la CPU inactivos(iowait) VS Tráfico

De acuerdo a este valor (**Figura 7.15**), hay una probabilidad mayor, pero poco prolongada de que un minero esté usando la CPU que un proceso benigno cuando se tomaron estos registros. Es posible que la investigadora no dejara ejecutar el código minero por tanto tiempo. Si la métrica llegara ser negativa el minero estaría consumiendo todo el tiempo en el uso de la CPU, lo cual podría estar relacionado con los cuellos de botella.

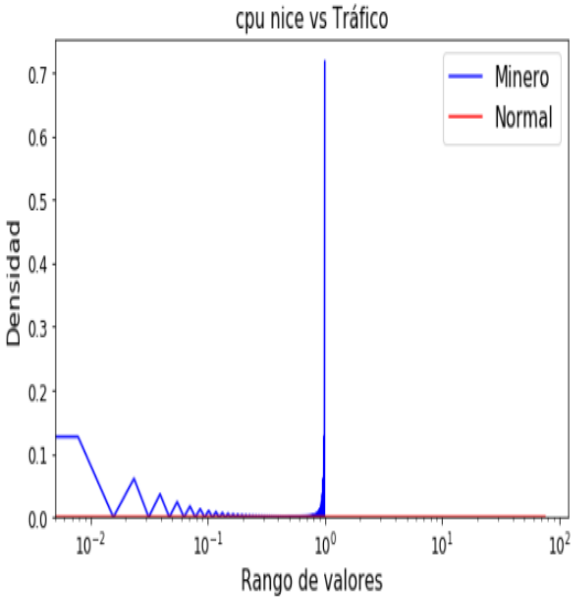

Figura 7.15: cpu nice vs Tráfico

Los mineros tienen picos de probabilidades altos en comparación de los registros normales en el uso total de la CPU, lo interesante es que la actividad minera se dispara en un determinado y corto tiempo **Figura 7.16**.

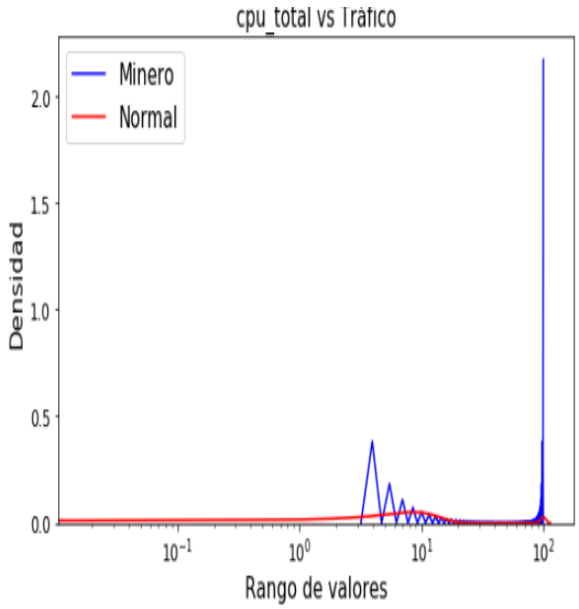

Figura 7.16: cpu total vs Tráfico

Los mineros tienen probabilidades pequeñas y prolongadas en el tiempo en la lectura de bytes. Esto es interesante, es probable que esto se relacione con el hecho que las comunicaciones son asincrónicas y la lectura sea la información que el servidor de minería envíe al minero (ver **Figura 7.17**).

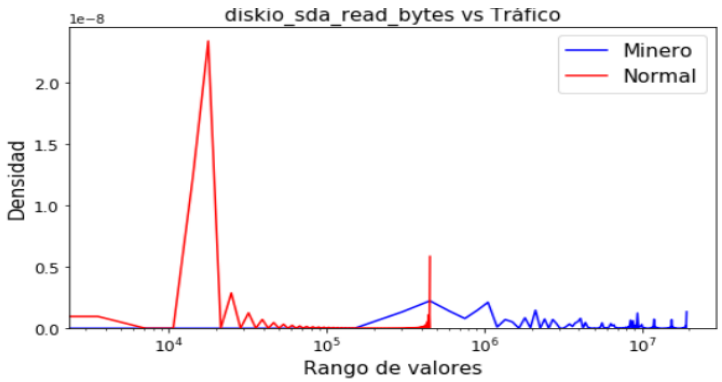

Figura 7.17: Lectura de bytes vs Tráfico

Contribuciones y entregables

El proyecto se desarrolló con la finalidad de investigar el cryptojacking a nivel de red usando ciencia de datos, pero sentando algunas bases para iniciar un análisis de variables de *hardware*.

8.1. Contribuciones

- Un modelo robusto que demostró tener una alta capacidad de clasificar de manera correcta registros mineros y normales, no solo de conjuntos de datos tomados en ambientes controlados, además con datos combinados con otros *malwares* en ambientes de incertidumbre (en entornos reales).
- Una exploración previa de variables de *hardware*, donde se puede evidenciar gran potencial para detectar la presencia de un minero.
- Conjunto de herramientas para gestionar capturas de tráfico de manera eficiente, descrito en la fase de entendimiento de los datos.

8.2. Entregables

- Los conjuntos de datos en formato pcap.
- Los conjuntos de datos de ventanas de tiempo.
- Los cuadernos (jupyter notebook) de código donde están implementados los modelos.
- El conjunto de herramientas desarrolladas para la captura de datos normales.

Conclusiones y trabajo futuro

9.1. Conclusiones

Las siguientes son concluciones que se han llegado con este proyecto:

- Un modelo de *machine learning* puede detectar con una alta exactitud de un 99 % (ver **Tabla 7.4**) actividad minera por medio de flujos de bytes entre el servidor y el nodo minero. Puesto que el servidor y el nodo intercambian información en los primeros instantes de la comunicación, debido a la transferencia de datos relacionada con el problema matemático, así como el umbral para hacer uso del procesador de la víctima.
- Los datos recolectados en ambientes controlados no necesariamente hacen a un modelo robusto ante los datos de entornos reales, esto lo observamos en el experimento dos, el cual demostró que el mejor modelo del experimento uno falla ante los datos de minería no pura, ya que tiene un recall del 7 % (ver **Tabla 7.5**) que indica que no predice a los mineros de manera adecuada.
- Combinar las características de los conjuntos de datos de minería, incrementa la probabilidad de que el árbol de decisión detecte un minero sin importar su estatus (identificado o desconocido), lo que da pie de que tener muestras con diversidad en los datos propicia modelos robustos, esto se aprecia cuando se pasó el modelo por la técnica de cross-validation y se probó con los datos que quedaron excluidos durante los entrenamientos y validaciones de todos los modelos usados en esta investigación, donde el modelo obtuvo un recall del 99 % (ver **Tabla 7.5**) y un rendimiento del 97 % en la curva ROC (ver **Figura 7.12**).
- Los mineros pueden hacer uso de la CPU en su totalidad. Esto se evidencia en la **Figura 7.16** y en el estado del arte.
- Los mineros presentan un patrón, el cual consiste en la transferencia de información en las primeras instancias de tiempo en el tráfico de red. Luego este flujo reduce su afluente de información. Posiblemente por la transferencia de información del Servidor al nodo de minería.

9.2. Trabajo futuro

Se propone como trabajo futuro lo siguiente

- Reforzar el modelo propuesto en este trabajo.
- Explorar con más detalle como se puede contribuir en la explicación del patrón identificado, para incrementar las posibilidades de los modelos de identificar minería.

- Entrenar un modelo de *machine learning* para la detección de actividad minera en *hardware*.
- Entrenar o hacer la propuesta de un modelo que tenga la capacidad de detección de cryptojacking a nivel de *hardware*, así como de red.
- Desarrollar un laboratorio donde se explique como es el proceso para crear una criptomoneda, minarlas, el estudio y capturas de tráfico con fines pedagógicos.

Referencias

Ali, I. Y. . S. (2020). Discovering interlinkages between major cryptocurrencies using high-frequency data: new evidence from covid-19 pandemic. Descargado de https://arxiv.org/abs/1112.4980 doi: 10.1186/s40854-020-00213-1

Arango, C. A. A., Ramírez, J. F. B., Ortiz, A. B., y Rego, M. M. B. (2018). Criptoactivos. *Documentos Técnicos o de Trabajo BANCO DE LA REPÚBLICA - COLOMBIA*.

Brownlee, J. (2020). *Machine learning mastery with python discover the fastest growing platform for professional machine learning with step-by-step tutorials and end-to-end projects*.

Carlin, D., O'Kane, P., Sezer, S., y Burgess, J. (2018). Detecting cryptomining using dynamic analysis. En *2018 16th annual conference on privacy, security and trust (pst)* (p. 1-6). doi: 10.1109/PST.2018.8514167

Christen, Gordijn, y Loi. (2019). *The ethics of cybersecurity*. Springer.

CISCO. (2020). Cisco annual internet report (2018–2023) white paper. Descargado de https://www.cisco.com/c/en/us/solutions/collateral/executive-perspectives/annual-internet-report/white-paper-c11-741490.html

CISCO, D. I. (2012). *Introduction to cisco ios netflow - a technical overview, document id:1518925696681207.* Descargado de https://www.cisco.com/c/en/us/products/collateral/ios-nx-os-software/ios-netflow/prod_white_paper0900aecd80406232.html

Eskandari, S., Leoutsarakos, A., Mursch, T., y Clark, J. (2018). A first look at browser-based cryptojacking. , 58-66. doi: 10.1109/EuroSPW.2018.00014

Gaviria, Ramirez, Urcuqui, y Navarro. (2020). Analysis of time windows to detect botnet's behaviour.

Haag, P., y otros. (2015). Descargado de https://github.com/phaag/nfdump

Hong, G., Yang, Z., Yang, S., Zhang, L., Nan, Y., Zhang, Z., ... Duan, H. (2018). How you get shot in the back: A systematical study about cryptojacking in the real world. En *Proceedings of the 2018 acm sigsac conference on computer and communications security* (p. 1701–1713). New York, NY, USA: Association for Computing Machinery. Descargado de https://doi.org/10.1145/3243734.3243840 doi: 10.1145/3243734.3243840

i. Muñoz, J. Z., Suárez-Varela, J., y Barlet-Ros, P. (2019). Detecting cryptocurrency miners with netflow/ipfix network measurements. En *2019 ieee international symposium on measurements networking (m n)* (p. 1-6). doi: 10.1109/IWMN.2019.8804995

ISO/IEC. (1994). *Iso/iec 7498-1:1994 information technology — open systems interconnection — basic reference model: The basic model.* Descargado de https://www.iso.org/standard/20269.html

Jayasinghe, K., y Poravi, G. (2020). A survey of attack instances of cryptojacking targeting cloud infrastructure. , 100–107. Descargado de https://doi.org/10.1145/3379310.3379323 doi: 10.1145/3379310.3379323

Li, Y., Yang, G., Susilo, W., Yu, Y., Au, M. H., y Liu, D. (2019). Traceable monero: Anonymous cryptocurrency with enhanced accountability. *IEEE Transactions on Dependable and Secure Computing*, 1-1. doi: 10.1109/TDSC.2019.2910058

Muñoz, J. Z. I., y Ros, P. B. (2019). Detection of bitcoin miners from network measurements.

Nakamoto, S. (2008). Bitcoin: A peer-to-peer electronic cash system. Descargado de https://bitcoin.org/

O'Kane, P., Sezer, S., McLaughlin, K., y Im, E. G. (2013). Svm training phase reduction using dataset feature filtering for malware detection. *IEEE Transactions on Information Forensics and Security*, *8*(3), 500-509. doi: 10.1109/TIFS.2013.2242890

Peters, M. A., Green, B., y Yang, H. M. (2020). Cryptocurrencies, china's sovereign digital currency (dcep) and the us dollar system. *Educational Philosophy and Theory*, *0*(0), 1-7. Descargado de https://doi.org/10.1080/00131857.2020.1801146 doi: 10.1080/00131857.2020.1801146

Recabarren, R., y Carbunar, B. (01 Jul. 2017). Hardening stratum, the bitcoin pool mining protocol. *Proceedings on Privacy Enhancing Technologies*, *2017*(3), 57 - 74. Descargado de https://content.sciendo.com/view/journals/popets/2017/3/article-p57.xml doi: https://doi.org/10.1515/popets-2017-0028

Rosenfeld, M. (2011). Analysis of bitcoin pooled mining reward systems. Descargado de https://arxiv.org/abs/1112.4980

Saad, M., Khormali, A., y Mohaisen, A. (2019). Dine and dash: Static, dynamic, and economic analysis of in-browser cryptojacking. En *2019 apwg symposium on electronic crime research (ecrime)* (p. 1-12). doi: 10.1109/eCrime47957.2019.9037576

stratum protocolo. (2020). Descargado de https://braiins.com/stratum-v2

Tahir, R., Durrani, S., Ahmed, F., Saeed, H., Zaffar, F., y Ilyas, S. (2019). The browsers strike back: Countering cryptojacking and parasitic miners on the web. En *Ieee infocom 2019 - ieee conference on computer communications* (p. 703-711). doi: 10.1109/INFOCOM.2019.8737360

Tanenbaum, y Wetherall. (2012). *Redes de computadoras* (Vol. Quinta Edición). PEARSON.

Urcuqui, Navarro, Osorio, y García. (2018). *Ciberseguridad: un enfoque desde la ciencia de datos - primera edición.* Universidad Icesi.

Varlioglu, S., Gonen, B., Ozer, M., y Bastug, M. (2020). Is cryptojacking dead after coinhive shutdown? En *2020 3rd international conference on information and computer technologies (icict)* (p. 385-389). doi: 10.1109/ICICT50521.2020.00068

Viscuso, M. (2017). *what is a non-malware (or fileless) attack?, carbon black.* Descargado de www.carbonblack.com/2017/02/10/non-malware-fileless-attack/

Yang, X., Chen, Y., y Chen, X. (2019). Effective scheme against 51 % attack on proof-of-work blockchain with history weighted information. , 261-265. doi: 10.1109/Blockchain.2019.00041