Analysis of Time Windows to detect Botnet's Behaviour

Julio Cesar Gaviria Jaramillo – Anderson Ramírez

Dirigido por:

Christian Camilo Urcuqui López, MSc.

Andrés Navarro, PhD.

Facultad de Ingeniería

Programa de Ingeniería de Sistemas

Departamento TICS

Universidad Icesi

Proyecto de grado

Santiago de Cali, junio 2020

Dedicación

En dedicación a nuestros padres y abuelos, los cuales siempre creyeron en nosotros….

A nuestros tutores , profesores y colegas que hicieron parte de este proceso de aprendizaje….

A Dios por la vida y por la oportunidad de permitirnos evidenciar los frutos de nuestro aprendizaje….

Abstract

Decades ago, botnets have been a weapon for the rise of cybercrime. These mechanisms allow cybercriminals vulnerate and use without authorization, personal or business equipments to commit cybercrime, including DDoS, Spam, Cryptocurrency mining, information theft, among others. Today a black market exist, which has been essential to keep these mechanisms active through the sale of exploits that can be integrated into these networks, allowing cybercriminals adapt them dynamically as technology advances.

The purpose of this research was train and evaluate a set of Machine Learning models, in order to detect behavioral patterns in the infection and C&C phases of these mechanisms. We reach detection rates between 99,85% and 99,71% in the testing data.

Finally, a web application was developed, which allows people scan their network, allowing them evidenciate if there are abnormal behavior patterns, where in turn, if is positive, the application will notify them by email.

Resumen

Desde hace algunas décadas las botnets han sido un arma fundamental para el auge de la ciberdelincuencia. Estás han permitido a los cibercriminales vulnerar y utilizar sin ningún consentimiento equipos personales o empresariales para cometer ciberdelitos, entre los cuales se destacan DDoS, Spam, minado de Criptomonedas, robo de información, entre otras. Hoy en día se conoce de la existencia de un mercado negro, el cual ha sido fundamental para mantener estos mecanismos activos a través de la venta de exploits que se pueden integrar a estas redes, permitiendo a los ciberdelincuentes adaptarlas dinámicamente a medida que la tecnología avanza.

El propósito de esta investigación fue entrenar y evaluar un conjunto de modelos de Machine Learning, con el fin de detectar patrones de comportamiento en las fases de Infección y C&C de estos mecanismos. Alcanzamos unas tasas de detección que oscilan entre el 99.85% y 99.71% en los datos de testeo.

Finalmente se desarrolló una aplicación web que permite a una persona escanear su red, permitiéndole evidenciar si existen patrones de comportamiento anómalos, donde a su vez en caso de ser positivo le notificara por medio de un correo electrónico

Índice General

Lista de acrónimos

- **IRC** Internet Relay Chat
- **HTTP** Hypertext Transfer Protocol
- **TCP** Transmission Control Protocol
- **UDP** User Datagram Protocol
- **ICMP** Internet Control Message Protocol
- **ICMP6** ICMP para IPv6
- **IGMP** Internet Group Management Protocol
- **P2P** Topología de red descentralizada (Peer To Peer)
- **DoS** Ataques de denegación de servicio
- **DDoS** Ataques de denegación de servicio distribuido
- **ML** Machine Learning
- **IoT** Internet de las cosas
- **V.T** Ventana de Tiempo
- **VP** Verdaderos Positivos
- **VF** Verdaderos Negativos
- **FP** Falsos Positivos
- **FN** Falsos Negativos

Glosario de Términos

- **IoT:** Termino sugerido a dispositivos como sensores, electrodomésticos, etc, que tienen la capacidad de acceder a internet.

- **Unidad computacional**: Un dispositivo de cualquier tipo (laptop, de mesa, IoT, móvil)

- **Botmaster**: Persona encargada de manejar la botnet.

- **Botnet**: Red de equipos cuyo objetivo es realizar ataques de denegación de servicio, robo de información, envió de spam y en algunos casos, usados para consumir recursos ajenos con el fin de incrementar el minado de criptomonedas

- **Denegación de servicio distribuido**: Ataque realizado por un número indeterminado de unidades computacionales, con el fin de dejar un objetivo sin la capacidad de respuesta, provocando su caída momentáneamente

- **Spam**: Envió de información no deseada generalmente al correo. Puede contener archivos maliciosos

- **Explotación de vulnerabilidades**: o Inbound Scan, cuyo objetivo es buscar vulnerabilidades en el software para filtrar archivos binarios maliciosos o probar la seguridad de un sitio

- **Ciclo de vida**: Aparición, desarrollo y terminación de una determinada funcionalidad

- **Formación**: El primer estado del ciclo de vida de la botnet, donde se emplea el uso de ingeniería social o explotación de vulnerabilidades con el fin de comprometer nuevos equipos y agregarlos a la red de bots.

- **Comando y Control (C&C)**: Servidores usados con el fin de enviar comandos que permiten dar manejo a la botnet remotamente.

- **Topología**: Mapa físico o lógico que define la forma en la que se encuentran conectados los dispositivos en una red

- **Topología Centralizada**: Topología que generalmente tiene muchos equipos y una unidad computacional denominada concentrador, el cual reenvía la información al resto y administra los recursos de la red. Ejemplo: topología en estrella

- **Topología Descentralizada**: Topología que generalmente tiene desacoplados los equipos, no hace uso de un concentrador y cada dispositivo es capaz de administrar sus recursos

- **Mecanismos de mitigación**: Mecanismos de seguridad diseñados para mitigar ataques producidos por cibercriminales, ejemplo: Honeynets, Honeypots.

- **Modelo de Machine Learning**: Modelos que se entrenan con el fin de analizar, encontrar patrones y dar resultados sobre un conjunto de datos

- **Dataset**: Conjunto de datos generados a partir de algoritmos realizados en diferentes lenguajes, ejemplo Python, java, etc.

- **Script**: Modulo de programación con un objetivo en específico.

- **Pcaps**: Conjunto de paquetes encapsulados en un solo tipo de archivo. Generalmente tienen una duración predeterminada.

- **Paquete**: Datos obtenidos de la red a través de herramientas de sniffing

- **Puerto:** Son las interfaces que permiten la comunicación entre nodos computacionales. Existen 65535 puertos, donde los primeros 1024 son denominados bien conocidos, debido a que son usados exclusivamente por el sistema operativo y los demás pueden ser utilizados por cualquier tipo de aplicación.

- **Netflow:** Es un resumen de una comunicación P2P (punto a punto), el cual refleja varias características de la conexión como IPs, puertos de origen y destino, numero de paquetes intercambiados, numero de bytes enviados, entre otros.

- **Protocolo:** Es un sistema de normas que regula la comunicación entre 2 o más sistemas que transmiten información por diversos medios físicos

Índice de Figuras

Índice de Tablas

Página

Motivación y Antecedentes

1.1 Contexto

El conocimiento ha permitido a la humanidad crear nuevas tecnologías que le han ayudado a progresar en la historia. Uno de los problemas que ha sufrido es la capacidad para tergiversar el uso de ciertas herramientas tecnológicas. Un ejemplo de ello son las botnets, que surgieron como una nueva forma de comunicación hasta llegar al punto de que algunas personas malintencionadas usaran sus ventajas para cometer ciberdelitos. (Garcia, 2014)

El termino botnet surgió con el nacimiento de EggDrop, la primera utilizada con fines benignos (Garcia, 2014). Está administraba canales de chat a través del protocolo IRC (Internet Relay Chat). Con el paso del tiempo, algunos curiosos empezaron a utilizar las herramientas que proveía el protocolo IRC con el fin de crear botnets malignas; estas tenían la capacidad de expandirse por su cuenta a través de vulnerabilidades en el software, comprometer información valiosa o incluso realizar ataques de denegación de servicio distribuido (DDoS)

Hoy en día siguen existiendo familias de botnets malignas. Algunas de ellas ya han sido deshabilitadas gracias a la colaboración de investigadores en el área de seguridad informática y algunos gobiernos, como en el caso de la botnet GRUM (Mushtaq, 2012). Los Cibercriminales que aun controlan botnets activas, han desarrollado la capacidad de implementar nuevas técnicas con el fin de volverlas dinámicas en el tiempo. Lo anterior se debe a la inteligencia de los ciberdelincuentes que están a cargo de estos mecanismos, que ha permitido generar nuevos algoritmos que le facilitan a la botnet migrar sus servidores de Comando y Control hacia otros sitios de manera autónoma. Además, los ciberdelincuentes han logrado cambiar las topologías de sus redes, haciendo aún más difícil la identificación de todo el sistema.

En efecto, los cibercriminales que controlan las botnet que aún se encuentran activas siguen cometiendo ciberdelitos tales como ataques de denegación de servicio, robo de información, incremento de Spam. Al mismo tiempo, algunas de ellas han sido

liberadas con el fin de que personas curiosas puedan pagar por su uso (Garcia, 2014). Es pertinente saber que puede ser difícil eliminarlas por completo, ya que día tras día se siguen incorporando nuevos exploits (módulos de programación), con el fin de hacerlas aún más completas, como en el caso de la botnet Echobot (Jiménez, 2019). Pero no solo eso, según García, existen algunos gobiernos u organizaciones cibercriminales que financian esta clase de mecanismos (2014) . Por lo anterior, muchas de ellas aún siguen expandiéndose a través del uso de Spam o ingeniería social, incrementando el número de delitos informáticos exponencialmente.

Actualmente existen muchos sectores que emplean infraestructuras tecnológicas como parte de su organización, tales como el gobierno y empresas. Muchas de estas pueden estar en peligro de ser comprometidas por un mecanismo de este tipo, ya que podría usar sus equipos para llevar a cabo los objetivos del ciberdelincuente.

1.2 Antecedentes del problema

El problema de la detección de ciberataques de tipo botnet se presenta por diversos motivos, donde el más importante es el dinamismo constante que revela el mecanismo. Este comportamiento es provocado debido a que día tras día se identifican nuevas vulnerabilidades en el software, que permiten al ciberdelincuente abrir caminos hacia su objetivo. Malas técnicas de programación, baches de seguridad en las plataformas y mal diseño en la arquitectura del software, son algunas de las causas que incrementan la posibilidad de crear nuevos caminos hacia la ciberdelincuencia.

Un reporte realizado por spamhaus.org, muestra el incremento exponencial de estos mecanismos:

“Last year, compared to 2017, we saw a 100% increase in the number of the domain names registered and set up by cybercriminals for the sole purpose of hosting a Botnet C&C:

- 2017: 50,000 domains
- 2018: 103,503 domains”

Por otra parte, NSFOCUS Inc. una red global y líder en ciberseguridad, que protege empresas de ciberataques avanzados, menciona que en el 2018 se detectaron 111,472 mil instrucciones de ataque de diferentes familias de botnets, que fueron recibidas por un total de 451,187 objetivos de ataque, esto incrementó el porcentaje en un 66,4% respecto al año 2017 (Jain, 2019)

También es pertinente tener en cuenta que el financiamiento que reciben los grupos o los creadores de estos mecanismos provoca que cada vez se mejoren e incrementen las funcionalidades de la red, haciéndola aún más compleja de detectar.

Todo lo anterior provoca el incremento de actividades maliciosas tales como spam, DDoS, robo de información, e inclusive, puede dar paso a la creación de nuevas botnets, como ha sucedido con anterioridad.

1.3 Justificación

Actualmente existen muchas investigaciones que han abordado el análisis y detección de botnets desde distintas perspectivas. Algunas de ellas han permitido a los investigadores trabajar junto a entidades gubernamentales con el fin de desactivar algunos de estos mecanismos. El problema surge cuando la botnet se vuelve dinámica y muchas de estas investigaciones pasan al olvido debido a que se vuelven obsoletas.

Debido a esto y a la agilidad de los cibercriminales, los DDoS aún se siguen presentando en la actualidad, tal como lo refleja Kaspersky lab en uno de sus reportes. Este menciona que "El número de ataques dirigidos contra una región en particular, sigue siendo liderado por China (63,8%), seguida de Estados Unidos (17,57%) y Hong Kong (4,61%) en el tercer lugar" (Kupreev, Badovskaya, & Gutnikov, 2019)

Además, este reporte también menciona que "en la estadística de servidores de administración de botnets, el primer lugar lo ocupa los Estados Unidos (44,14%), los países bajos en segundo lugar (12,16%) y Reino Unido, el tercero (9,46%)" (Kupreev, Badovskaya, & Gutnikov, 2019)

Todo lo anterior da a evidenciar que aún siguen existiendo estructuras de este tipo, y por la alta tasa de servidores de administración de botnets, puede dar a entender que es difícil tener una prevención adecuada ante un posible ataque de este tipo. Por tal motivo, es necesario implementar un mecanismo que permita notificar a un conjunto de usuarios si presentan patrones de comportamiento malignos, propios de la fase de infección o C&C del ciclo de vida de la botnet, con el fin de reducir el índice de ciberdelincuencia a un nivel más adecuado.

Descripción del Problema

2.1 Formulación del problema

Las botnet son un mecanismo que ha tomado auge en los últimos años. Según García en sus inicios fueron utilizados como parte de múltiples proyectos que involucraban el manejo de canales de chats a través del protocolo IRC (2014), hasta que algunos "curiosos" se empezaron a dar cuenta de los beneficios que podría traer consigo manipular una red tan enorme de dispositivos.

Una vez trasladados al lado contrario de la seguridad informática, ha sido difícil detectar y predecir un ciberataque de tipo botnet. Esto es causado por la dificultad de mitigar a los cibercriminales denominados Botmaster, debido a su inteligencia y astucia; también por el dinamismo de la estructura de la botnet, ya que estas pueden aprovechar nuevas vulnerabilidades presentes en el software debido al error humano.

También, puede ser complejo llevar investigaciones de este tipo, debido al descontento por parte de agentes externos y la complejidad de los datos para su investigación. Por consiguiente, existe una clara ausencia de mecanismos que permitan predecir y prevenir sobre si un conjunto de usuarios hace parte de la estructura de la botnet (zombies) a través del análisis de tráfico de red. Todo lo anterior ha ayudado a incrementar la ciberdelincuencia en estos últimos años, incrementando los ataques DDoS, robo de información y el uso de spam con el fin de expandir aún más la red.

Objetivos

3.1 Objetivo General

Desarrollar una herramienta que permita detectar patrones de comportamiento malignos, propios de la fase del ciclo de vida de C&C de las botnets, haciendo uso de ventanas de tiempo.

3.2 Objetivos Específicos

- Obtener una muestra de pcaps a través del análisis de trafico de red

- Generar un Dataset con variables de trafico de red, clasificado en registros benignos y malignos.

- Evaluar un conjunto de modelos de Machine Learning que permitan predecir si un tráfico de es benigno o presenta evidencias de comportamientos malignos.

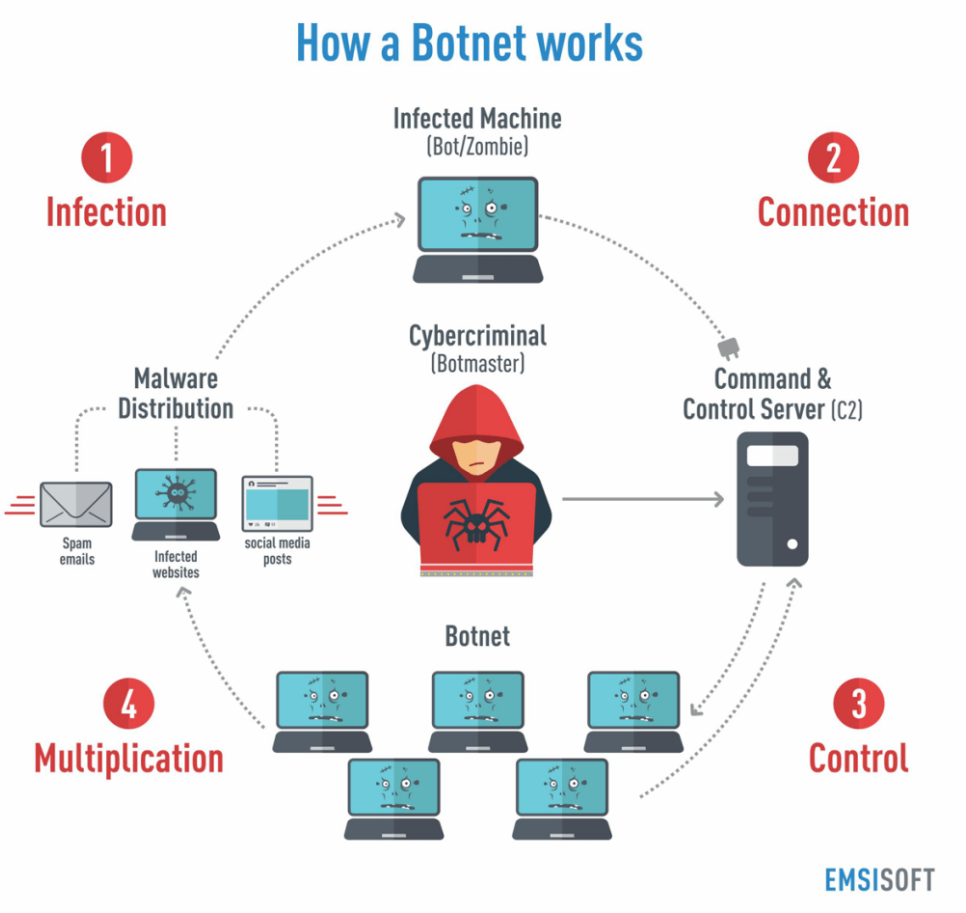
- Desarrollar una herramienta de software que permita notificar a un conjunto de usuarios si son parte de la estructura de la botnet (zombies).

Marco Teórico

4.1 Botnets

Las botnets son mecanismos compuestos por miles de equipos comprometidos ilegalmente denominados “zombies” o “bots”. Estas estructuras han sido utilizadas para múltiples fines, incluyendo DDoS, Spam, robo de información y hoy en día, para minado de criptomonedas ilegalmente. Son difíciles de detectar, ya que día tras día los cibercriminales las hacen mucho más potentes mediante módulos de programación, denominados Exploits, que son introducidos e integrados a sus sistemas. Muchos de estos módulos son generalmente producidos y vendidos en el mercado negro del malware, ya que así permite desacoplar las partes del mecanismo en funciones importantes.

Día tras día mediante técnicas sumamente creativas, los cibercriminales tienden a expandir aún más la red, haciéndola aún más fuerte, más madura y más difícil de erradicar. Al principio, era común ver mecanismos de esta índole utilizando topologías centralizadas. Hoy en día los cibercriminales se dieron cuenta que ese tipo de topología no era el adecuado, ya que si se detectaba el servidor de C&C (nodo central) podrían perder el control total de la botnet, por lo que empezaron a utilizar topologías basadas en P2P, lo cual permite que cada nodo administre sus propios recursos de manera autónoma, haciendo difícil la destrucción del mecanismo.



Grafica 1: Estructura de una botnet Maligna

Las botnet presentan un ciclo de vida particular, similar al de la *Grafica 1*, el cual consta de las siguientes etapas:

1. **Formation / Infection:** Fase donde la botnet incrementa su tamaño mediante ingeniería social o realizando "Inboud scan" directamente a las redes comprometidas.

2. **Command and Control (C&C):** Fase donde el cibercriminal sincroniza su estructura y los bots esperan órdenes para saber cuál será su siguiente objetivo

3. **Attack:** Fase donde los bots generan suficiente poder computacional para derribar por medio de DDoS a un objetivo en particular.

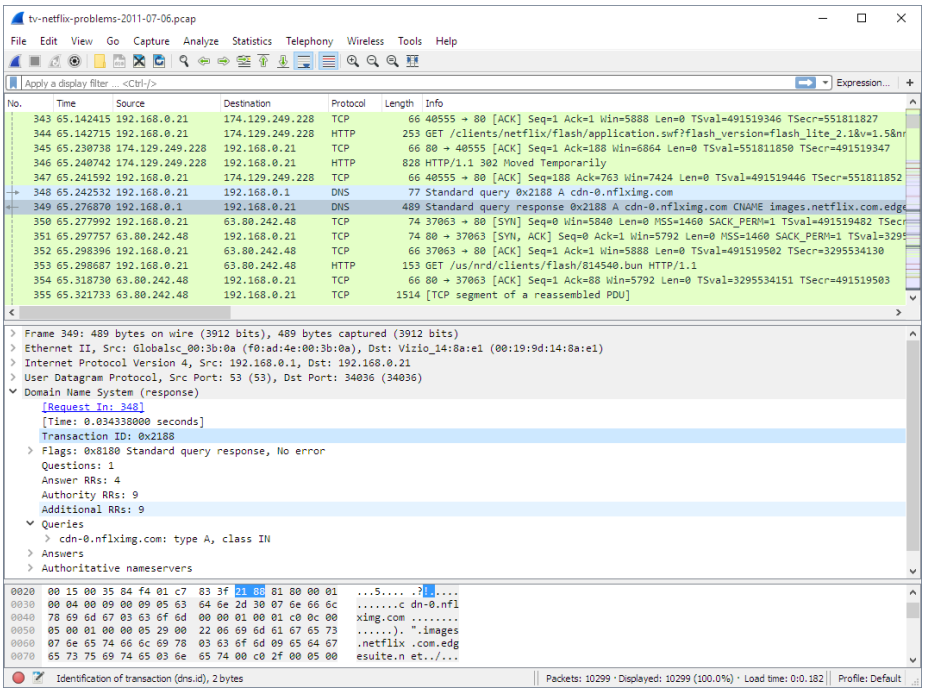
4. **Post-Attack:** El cibercriminal hace un barrido sobre la red para saber cuántos bots ha perdido durante la etapa anterior.

4.2 Análisis de trafico de red

El análisis de trafico de red es una poderosa herramienta que permite visualizar las condiciones en las que se encuentra la red (en un determinado periodo de tiempo), para garantizar la seguridad, rendimiento e integridad de la red. Éste permite evidenciar el tráfico producido por los diferentes nodos de procesamiento que se encuentran conectados a la red, a través de cualquier dispositivo que permita la transmisión y recepción de datos (por ejemplo: una tarjeta de red).

Este análisis permite visualizar no solo los paquetes que se transmiten, sino también los protocolos de comunicación que actualmente están siendo utilizados, tales como: TCP, UDP, HTTP; direcciones ip de los emisores y receptores, tamaño del paquete, entre otras. Generalmente para realizar este tipo de análisis, se hace uso de herramientas de "sniffing", las cuales permiten interceptar el flujo de paquetes que están atravesando la red.

Mediante el tráfico de red es posible evidenciar si una red presenta tráfico benigno o maligno, ya que los diferentes tipos de paquetes pueden dar una simple evidencia forense de comunicaciones producidas por malware o aplicaciones normales. "algunas de las anomalías son indicios de cuellos de botella en el rendimiento, los cuales pueden ser causados por muchas razones tales como flash crowds, DDoS o fallos de componentes de la red" (Kim, Sim, Tierney, Suh, & Kim, 2019).



Grafica 2: Análisis de trafico de red mediante Wireshark

Actualmente existen 2 formas para realizar análisis de trafico de red:

- **El análisis de paquetes** permite analizar una muestra de paquetes, permitiendo realizar capturas completas del tráfico de la red. Está, provee suficientes detalles para analizar y plantear soluciones de problemas en la red. Algunas de las desventajas de esta variante son cuando se presenta tráfico encriptado y el excesivo consumo de recursos que conlleva realizar un análisis de este tipo. (Minarik, 2016). Un ejemplo de este tipo de análisis es mediante herramientas de Sniffing como Wireshark, donde en la *Grafica 2* se puede evidenciar un ejemplo de captura de paquetes a una red local.

- **El Flow data o análisis de Netflows** es definido por el tráfico que tiene la misma dirección ip de origen y destino, protocolo, puerto de origen y destino. Cuando se produce un cambio en alguna de estas variables, un nuevo flujo es definido. (NetFort Technologies Limited, 2014)

4.4 DoS y DDoS

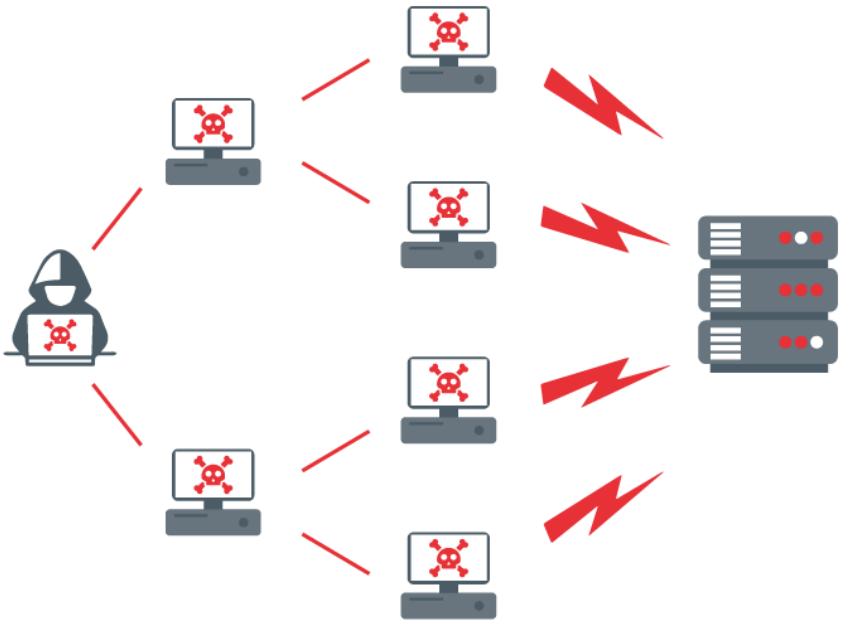
Los ataques de denegación de servicio distribuidos (DDoS) son unos de los ciberataques más populares en nuestros días. Años atrás solo se conocían los ataques de tipo DoS, los cuales eran dirigidos con el fin de deshabilitar la disponibilidad de un recurso, impidiendo el acceso a los usuarios durante un periodo de tiempo. Los recursos a los cuales un ataque DoS puede dirigirse abarcan desde nodos de procesamiento únicos o grupos de dispositivos hasta redes normales o empresariales. (Muhammad & Syed Mustafa, 2019)

Estos ataques pueden ser llevados a cabo contra diferentes capas del modelo OSI o TCP/IP. El tipo de ataque puede variar dependiendo la técnica usada, el software utilizado, el protocolo a atacar y la naturaleza de la víctima (servidor, red o dispositivos). (Muhammad & Syed Mustafa, 2019)

Una vez los cibercriminales se dieron cuenta de que necesitaban más poder computacional para lanzar un ataque DoS, especialmente cuando se necesita generar tráfico masivo en un corto lapso, decidieron distribuir los ataques de tal forma que se permitiera utilizar muchos más equipos para enviar más tráfico hacia un objetivo (Muhammad & Syed Mustafa, 2019).

Los cibercriminales lograron tomar control de otras máquinas, incluso sin el consentimiento de sus propios dueños, con el fin de incrementar el daño producido por un ataque de este tipo. Estos dispositivos generalmente eran vulnerados mediante bugs o brechas de seguridad en sus plataformas. (Muhammad & Syed Mustafa, 2019)

En la *Grafica 3* podemos evidenciar un ejemplo de la naturaleza de un ataque DDoS, donde el cibercriminal sincroniza y controla los nodos computacionales con el fin de denegar el servicio a la víctima, que en este caso puede ser por ejemplo un servidor localizado en una red empresarial.



Grafica 3: Ejemplo de Ataque de denegación de servicio distribuido (DDoS)

Hoy en día existen diferentes tipos de ataques de denegación de servicio, tales como SYN flood, UDP flood, HTTP flood, ping of death, smurf attack, fraggle attack, slowloris, Application level attacks, NTP amplification, APDoS, Zero Day DDoS attacks. (Poremba, 2017)

4.5 Machine Learning

"Machine Learning is an application of artificial intelligence (AI) that provides systems the ability to automatically learn and improve from experience without being explicitly programmed" (Expert System, 2017)

Esta rama de las ciencias computacionales ha permitido a la humanidad generar innumerables cambios en ámbitos sociales, económicos o políticos. A su vez, va muy ligada a la ciencia de datos, ya que estos permiten obtener los insumos necesarios para desarrollar modelos, métricas y experimentos conforme a las necesidades del sector, con el fin de detectar patrones que permitan tomar las mejores decisiones posibles.

Machine Learning puede ser de diferentes tipos, entre los cuales tenemos:

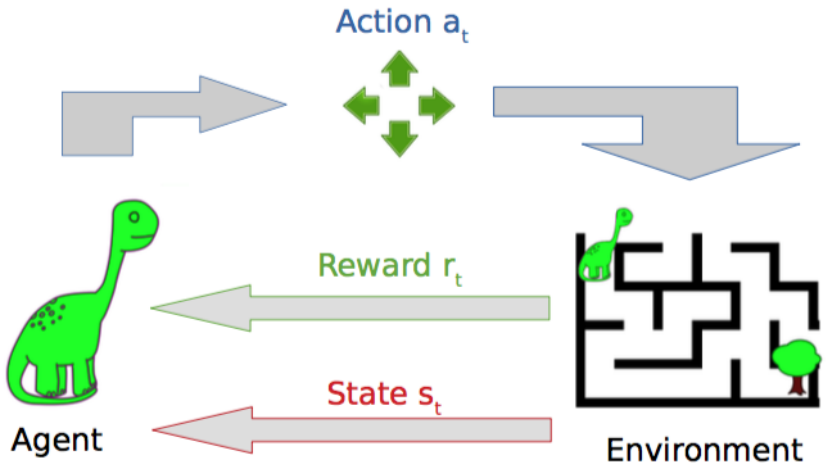
1. ***Algoritmos supervisados:*** Requiere de datos previamente etiquetados, con el fin de prever futuros eventos. También permite al modelo proveer variables objetivo, los cuales pueden ser comparados con salidas reales con el fin de revisar si los valores de las métricas son correctos. (Expert System, 2017). En la *Grafica 4* se puede evidenciar un ejemplo de datos etiquetados, donde en ese caso se quiere predecir si una persona tiene carro o no, dependiendo de los valores presentes en las variables edad e ingresos

Grafica 4: Ejemplo de Aprendizaje Supervisado, tomada del Curso de analítica de Datos (Pregrado), Universidad Icesi

2. ***Algoritmos no supervisados:*** No requiere uso de datos previamente etiquetados. Permite estudiar cómo el sistema infiere una función para describir estructuras o patrones ocultos en el set de datos. (Expert System, 2017). En la *Grafica 5* podemos evidenciar un ejemplo de escenarios de tipo no supervisado, donde en este caso es el mismo ejemplo de la *Grafica 4*, pero no existe la variable objetivo "tiene carro". El algoritmo por si solo tendrá la capacidad de clasificar dependiendo ciertas similitudes, los perfiles de las personas y con esto se podrá inferir si un grupo en particular tiene carro o no.

Grafica 5: Ejemplo de Aprendizaje NO Supervisado , tomada del Curso de analítica de Datos (Pregrado), Universidad Icesi

3. ***Algoritmos de aprendizaje por refuerzo:*** Es un método de aprendizaje que interactúa con el entorno, lo que permite producir acciones y descubrimientos a partir de la tasa de errores o el éxito que tenga el modelo. (Expert System, 2017). En la *Grafica 6* podemos evidenciar un ejemplo de aprendizaje por



refuerzo, donde el agente es "recompensado" si realiza una acción que le permita llegar hasta el final del escenario, o en ese caso un "castigo" si la acción realizada no es la adecuada.

Grafica 6: Ejemplo de aprendizaje por refuerzo

4.6 Ciberseguridad y Ciencia de datos

Ciberseguridad es la acción de poner bajo protección sistemas, redes, programas y datos de posibles ataques maliciosos que provienen de personas denominadas cibercriminales (Kaspersky, s.f.). Puede dividirse en varias categorías:

- **Seguridad de la red:** Es la práctica de asegurar una red de computadores de intrusos

- **Seguridad en las aplicaciones:** Se enfoca en mantener el software y los dispositivos libres de amenazas, ya que una aplicación comprometida puede proveer acceso a datos sensibles.

- **Seguridad de la información:** Protege la integridad y privacidad de los datos, tanto en almacenamiento como en tránsito.

- **Seguridad en las operaciones:** Incluye los procesos y decisiones para manejar y proteger los data assests (bienes)

- **Recuperación ante desastres y continuidad del negocio:** Define como una organización responde ante un accidente que compromete su seguridad informática y otros eventos que pueden ocasionar un paro en las operaciones o la perdida de datos.

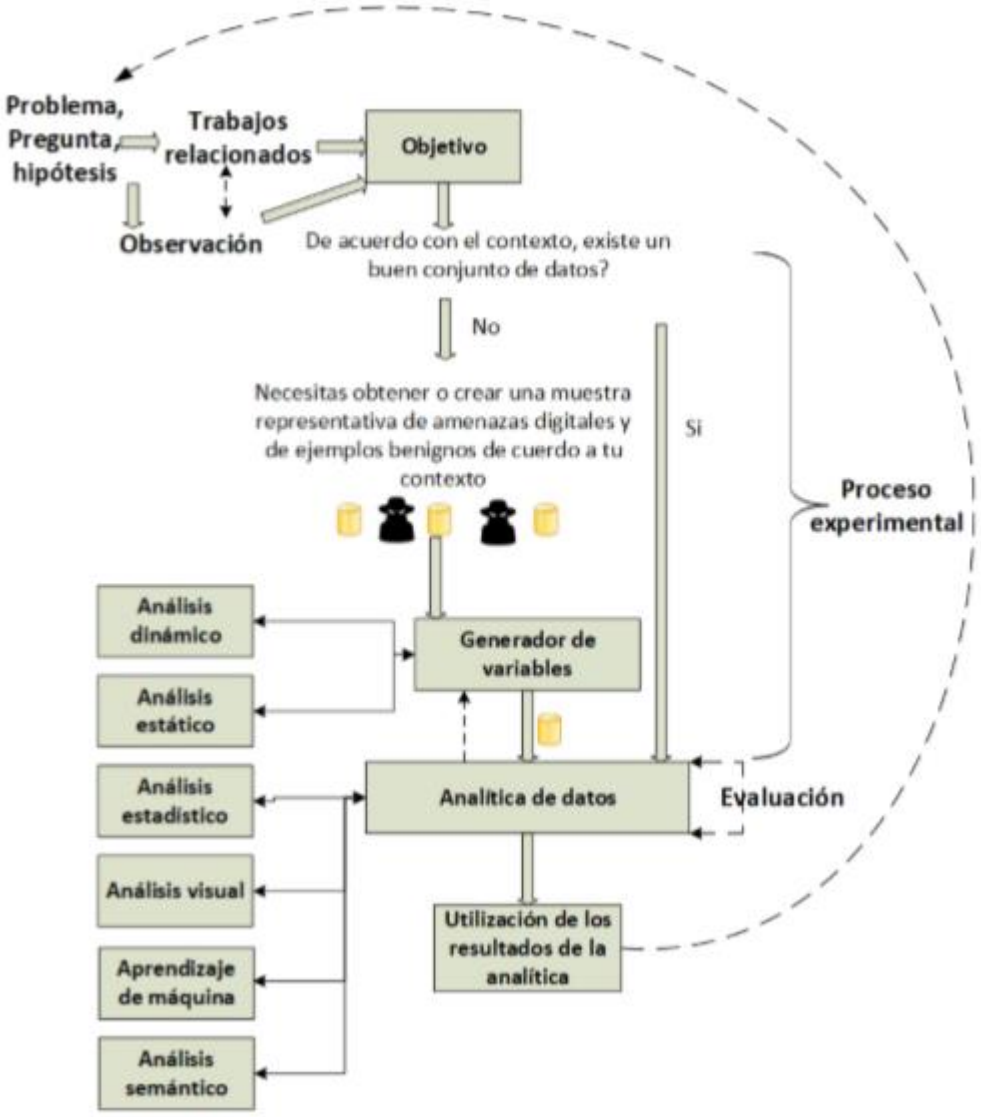
- **Educación del usuario:** Asegura el factor de seguridad informática más impredecible: las personas. Cualquiera puede accidentalmente introducir un virus por no seguir las buenas prácticas de seguridad. Enseñarles a seguir estas prácticas es vital para la seguridad de la empresa.

"La ***Ciencia de Datos*** tiene como objetivo obtener elementos de valor de distintas fuentes de información, a través de técnicas y herramientas que incluyen métodos de estadística, minería de datos, machine Learning y visualización" (Urcuqui , García Peña, Osorio Quintero, & Navarro Cadavid, 2018). En la ciencia de datos existen 2 enfoques de análisis:

- **EDA (Exploratory Data Analysis)**: Tiene como finalidad conocer las relaciones o patrones que existen en los datos, aplicable cuando de antemano no se tiene una hipótesis o entendimiento de ellos.

- **CDA (Confirmatory Data Analysis)**: Asume que el científico tiene una hipótesis acerca de la información y tiene como objetivo probarla o descartarla a partir de los modelos creados.

"La ciencia de datos puede llegar a ser una herramienta importante para el desarrollo de la ciberseguridad" (Urcuqui , García Peña, Osorio Quintero, & Navarro Cadavid, 2018). Mediante este análisis, se pueden proponer enfoques que permitan a los investigadores seguir un conjunto de pasos, como el evidenciado en libro *Ciberseguridad: Un enfoque desde la ciencia de datos*, donde se presenta un framework para el entrenamiento de modelos de machine Learning aplicados a la detección de amenazas cibernéticas. Este Framework puede evidenciarse en la *Grafica 7*.



Grafica 7: Framework propuesto en "Ciberseguridad: Un enfoque desde la ciencia de datos"

Estado del Arte

5.1 Identifying, Modeling and Detecting Botnet Behaviors in the Network

Sebastián García realizó una investigación donde recopiló una serie de características de investigaciones anteriores relacionadas con botnets, donde propuso categorizarlos dependiendo 4 dimensiones globales:

- **Detection sources**: Fuente principal de la información usada para la detección.
- **Detection features**: Clasificación de las características usadas para la detección.
- **Detection techniques**: Técnicas usadas para la detección y cuáles son los mejores caminos para la detección de botnets.
- **Detection algorithms**: Clasificación de los algoritmos usados para obtener los resultados.

Una vez categorizadas estas investigaciones previas en las 4 dimensiones anteriores, presento un cuadro comparativo como el siguiente:

Survey	Detection Sources	Detection Features	Detection Techniques	Detection Algorithms
Michael Bailey, Evan Cooke, Farnam Jahanian, Yunjing Xu, and Manish Karir. A survey of botnet technology and defenses	Network packets, DNS logs, darknet and traffic flows	Not included	Behavior (attack and cooperative) and signature	Not included
Jing Liu, Yang Xiao, Kaveh Ghaboosi, Hongmei Deng and Jingyuan Zhang. Botnet : Classification , Attacks, Detection, Tracing and preventive measures	Honeynets and network packets	Not included	Signature and anomaly based	Not included
S. Y. Lim and A. Jones. Network Anomaly Detection System: The state of Art of Network Behavior Analysis	Not included	Not included	Anomaly models (model, rule and statistical based) and specification models (protocol, state and transaction based)	Data mining, neuronal networks pattern matching, expert systems, Bayesians, covariance, matrices, chi-squared
Amit Kumar Tyagi and G Aghila. A Wide Scale Survey on Botnet	Honeynets and network traffic	Not included	Behavior, DNS and data mining based	Not included
Mohammad Jorjor Shooshtaru Zadeh,	Honeynets and IDS	Not included	Signatura and anomaly	Not included

Hossein Rouhani Zeidanloo, M. Safari, Mazdak Zamani and Payam Vahdani Amoli. A Taxonomy of Botnet Detection Techniques			(host and network based)	

Tabla 1: Cuadro comparativo entre las diferentes investigaciones (Garcia, 2014)

Como podemos observar en el cuadro comparativo anterior, muchas de las investigaciones evaluadas por García, utilizan técnicas de mitigación tales como Honeynets, Honeypots, que en nuestra investigación no utilizaremos. Además, las que realmente utilizan técnicas de machine Learning en su investigación, no mencionan cuales fuentes de detección utilizaron (como análisis de trafico de red, por ejemplo).

En esta investigación, García menciona que existen algunas problemáticas en cuanto a las investigaciones previas sobre botnets:

En primer lugar, tenemos la reproducibilidad de la investigación. Esta característica define si la investigación es capaz de aportar datos significativos, con el fin de que en investigaciones posteriores se pueda volver a realizar los mismos pasos que los investigadores anteriores llevaron a cabo. Lastimosamente muchas de estas no pueden ser reproducidas nuevamente (Garcia, 2014)

En segundo lugar, tenemos el número de host, donde el diseño de la investigación depende de esta característica. Muchas investigaciones necesitan un gran número de host y paquetes para obtener buenos resultados, pero lastimosamente juega un rol negativo, ya que es difícil en algunas ocasiones acceder a la misma cantidad de host utilizados. (Garcia, 2014)

En tercer lugar, tenemos las características que surgen cuando los investigadores realizan el pre-proceso de los datos antes de la extracción de características. Muchas tienden a sobrefiltrar los datos y generan los algoritmos con base en estos. (Garcia, 2014)

Por último, los experimentos y las métricas son de suma importancia para las investigaciones. Lastimosamente la mayoría de las investigaciones no computan el set completo de métricas o no definen los experimentos correctamente. Esto imposibilita la comparación entre investigaciones. (Garcia, 2014)

En suma a todo lo anterior, García propone trabajar por un lado con análisis de tráfico de red, porque es el mejor camino para proteger miles de computadores al mismo tiempo y por el otro lado basado en comportamiento, porque parece el mejor camino que representa las acciones complejas de las botnet (2014).

Su metodología consistía en lo siguiente: Crearon una instalación de captura de malware para obtener un tráfico de malware bueno, etiquetado y real. Desarrollaron una herramienta denominada SimDetect para detectar similitudes en el tráfico botnet. Una vez aprendieron de sus errores, desarrollaron BClus, método que automáticamente realizaba clusters y detectaba grupos similares en cuanto a comportamiento sobre las botnets. Finalmente, con una mejor comprensión de los comportamientos de las botnets, desarrollaron CCDetector, que permite detectar los patrones de comportamiento de los canales de C&C de las botnets, usando un novel state based model del tráfico de red. Además, BClus y CCDetector fueron comparados con otros métodos, verificando sus resultados con técnicas del estado del arte. (Garcia, 2014)

Las ventajas de esta investigación es que nos ayuda a entender los comportamientos de las botnets analizando tráfico real. Además, los algoritmos y los pcaps generados por esta investigación son open source, los cual nos permitiría (llegado el caso), reproducir la investigación. El único problema es que a pesar de que permiten detectar patrones de comportamiento, no enfatizan en la extracción de ciertas variables de trafico de red que puedan ser de suma importancia para análisis y detección en tiempo real. Tampoco utilizan un módulo de notificación para avisar al usuario de que será un posible objetivo de la botnet y no hacen uso de modelos de machine Learning con el fin de predecir lo anterior.

5.2 Botnet Command Detection using Virtual Honeynet

Por otra parte, una investigación llevada a cabo en Cyber Security Technology Division, CDAC Mohali, INDIA 160071 por los autores J.S. Bhatia, R.K. Sehgal y Sanjeev Kumar muestra una perspectiva un poco diferente de las investigaciones anteriores. "There are mainly two approaches for botnet detection and tracking. One approach is based on setting up honeynets. The other approach for botnet detection is based on passive network traffic monitoring and analysis" (Bhatia, Sehgal, & Kumar, 2011)

Ellos desarrollaron un algoritmo basado en el comportamiento de la red en tiempo de ejecución de los bots y a la secuencia de comandos utilizada en la conversación que estos tienen en la fase de C&C. La única desventaja es que a pesar de que crearon el prototipo, desarrollar una infraestructura escalable y robusta para alcanzar sus objetivos era un problema (Bhatia, Sehgal, & Kumar, 2011).

Esta investigación es muy proactiva, ya que permite visualizar a gran escala una serie de comandos que cualquier botnet podría utilizar en la fase de C&C. La diferencia con nuestra investigación es que a pesar de que nosotros utilizamos un análisis de trafico de red con el fin de utilizar una serie de variables extraídas de los datos, ellos utilizan un prototipo completo basado en Honeynets (que en nuestra investigación no utilizaremos), con el fin de hacer correlaciones entre comportamiento y nombres de comandos utilizados en la fase de C&C. Además, no utilizan un módulo de notificación al usuario para avisarle de que será un posible objetivo de la botnet, ni tampoco técnicas de machine Learning con el fin de predecir estos comportamientos.

Metodología

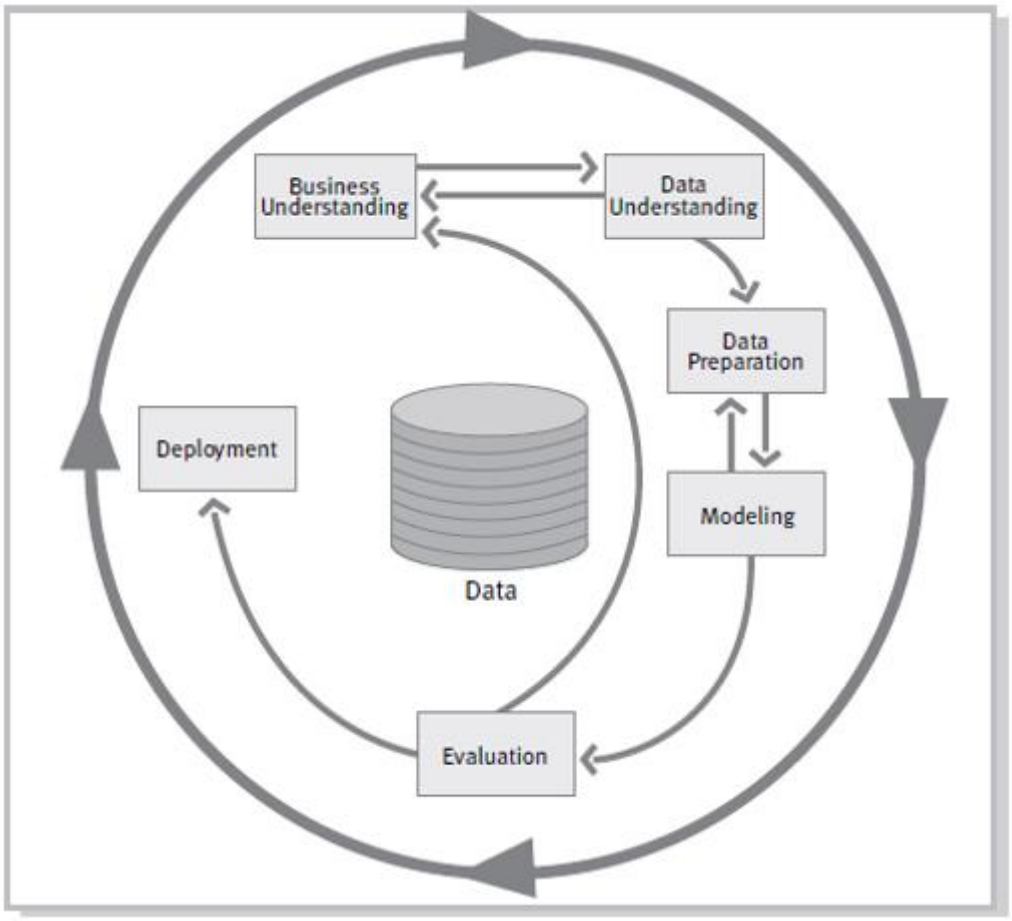
Antes de definir nuestra metodología, es necesario analizar los 2 principales componentes fundamentales que componen nuestro proyecto:

El primer componente estará directamente relacionado con ciencia de datos. Esto, debido a la necesidad de generar un dataset a partir de los pcaps obtenidos en el objetivo específico número uno. Ese conjunto de datos debe ser limpiado, realizando una exploración de los datos con el fin de detectar datos atípicos, anómalos, escalas y otros factores que se identifican en el EDA (Exploratory Data Analysis). Una vez procesados los datos, el siguiente paso es realizar el entrenamiento y evaluación de los modelos de Machine Learning (se mencionarán más adelante).

El segundo componente contemplara el desarrollo de un software que permita notificar al conjunto de usuarios si hacen parte de la estructura de la botnet (zombies). La herramienta será desarrollada bajo el esquema incremental, donde se realizará una etapa de elicitación de requerimientos, diseño y desarrollo del prototipo de la solución, teniendo en cuenta que debe tener un módulo de notificación asociado a nuestros modelos anteriormente entrenados y un posible modulo que permita la recopilación de trafico de red.

6.1 CRISP-DM

CRISP-DM (Cross-Industry Estándar Process for Data Mining), es la guía de referencia más ampliamente utilizada en el desarrollo de proyectos de Data Mining. Una de sus ventajas es la posibilidad de replicar proyectos relacionados con Machine Learning y Data Mining, por ello es una de las más utilizadas en el mundo.



Grafica 8: CRISP_DM

6.1.1 Entendimiento del negocio

En esta sección se realizó un estudio previo a las investigaciones presentes en el estado del arte.

6.1.2 Entendimiento de los datos

Antes de comenzar nuestra investigación fue necesario indagar acerca de la naturaleza de los datos que íbamos a usar para el desarrollo de nuestro aplicativo. Fue necesario consultar en la página de Stratosphere Lab la forma en la que se obtuvieron los registros que se encontraban en el. Los datos benignos que ellos disponían fueron registros de peticiones HTTP a varios sitios web registrados en una página denominada ALEXA. Por el contrario, los datos malignos evidenciaban capturas de malware donde quedaban registrados comportamientos propios del ciclo de vida de Comando y Control de las botnets que estaban estudiando.

Antes de continuar, es necesario entender que es una ventana de tiempo. **Una ventana de tiempo** es un periodo de tiempo (generalmente 5 minutos), en donde se condensa un conjunto de comunicaciones evidenciadas a través de la red, donde estas conexiones son representadas en forma de Netflows (resumen de la comunicación desde un nodo hacia otro).

Una vez transformados los pcaps a ventanas de tiempo, generamos un aproximado de 258.697 registros extraídos por medio de un algoritmo implementado en la fase 2 de la presente investigación. Las ventanas de tiempo generadas se encuentran divididas de la siguiente forma:

- 519 ventanas de tiempo Benignas
- 258.178 ventanas de tiempo Malignas

Debido al desbalance presente en la cantidad de datos, en una primera instancia aplicamos una técnica denominada **OVERSAMPLING**, donde intentamos generar

una mayor cantidad de datos Benignos con nuestro laboratorio. El resultado fue la generación de 649 nuevas ventanas de tiempo benignas.

Sin embargo, igualar la cantidad de datos Malignos era una tarea difícil, por lo que decidimos aplicar la técnica de **UNDERSAMPLING**, donde se tomó una muestra de datos malignos para poder tener un dataset balanceado.

Finalmente, es necesario entender las variables presentes en los Datasets. Algunas de las variables fueron generadas directamente con la herramienta de NFDUMP presenten en el laboratorio de investigación. Las demás fueron propuestas por la presente investigación, debido a que era necesario darle un enfoque particular a nuestra propuesta . Las variables presentes en los Datasets

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: Percentil**: 25% de todas las duraciones en la ventana de tiempo
7. **P2_d: Percentil**:50% de todas las duraciones en la ventana de tiempo
8. **P3_d: Percenti**l:75% 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**: Percentil 25% de todas las entradas de paquetes en la ventana de tiempo
26. **P2_ip**: Percentil 50% de todas las entradas de paquetes en la ventana de tiempo
27. **P3_ip**: Percentil 75% de todas las entradas de paquetes en la ventana de tiempo
28. **P1_ib**: Percentil 25% de todas las entradas de bytes en la ventana de tiempo
29. **P2_ib**: Percentil 50% de todas las entradas de bytes en la ventana de tiempo
30. **P3_ib**: Percentil 75% de todas las entradas de bytes en la ventana de tiempo
31. **Type**: Tipo de ventana de tiempo (Maligna/Benigna)

6.1.3 Preparación de los datos

En esta fase se realizó la exploración y limpieza de los datos, donde se procedió a:

1. Eliminar los valores faltantes de las columnas
2. Cambiar el tipo de algunas columnas (Object-Int, Float-Int)
3. Eliminar Outliers
4. Seleccionar las variables más representativas para el conjunto de datos a través de una técnica denominada **feature importance** por cada experimento realizado.
5. Realizar un escalamiento (Normalización) de los datos para facilitar su manejo.

6.1.4. Modelado

En esta fase de modelado se entrenaron los siguientes algoritmos clasificatorios:

1. Gaussian Naive Bayes
2. Decision Tree Classifier
3. KNN
4. Logistic Regression
5. Random Forest Classifier

Se realizo un **Holdout** para obtener un 70% de los datos dirigidos al entrenamiento de los modelos y un 30% de los datos para pruebas. En cada uno de los modelos se aplicó una técnica denominada **GridSearchCV**, el cual realiza una búsqueda exhaustiva de cuáles son los mejores parámetros especificados para un modelo en específico

6.1.5. Evaluación

En la fase de evaluación se tuvieron en cuenta métricas como:

1. **Precision**: Es el valor de la predicción positiva

$$Precisión = \frac{VP}{VP + FP}$$

2. **Recall**: También conocido como sensibilidad. Menciona que proporción de todos los positivos reales pudo identificar el algoritmo.

$$Recall = \frac{VP}{VP + FN}$$

3. **F1-Score**: Es el promedio armónico.

$$F1_{score} = 2 * \frac{precision * recall}{precision + recall}$$

4. **Accuracy**: Es la tasa de correctitud. Es representada por :

$$Accuracy = \frac{VP + VN}{VP + VN + FP + FN}$$

5. **Kappa**: Es el coeficiente de concordancia entre las predicciones y las clases reales. Sustrae el efecto de concordancia por suerte (AC) del valor del accuracy (concordancia observada-OA)

Estas métricas fueron almacenadas en un archivo CSV para realizar una comparación entre los modelos presentes en cada experimento. Además, Al final de cada prueba realizada por cada uno de los modelos, se desplegaba una Matriz de confusión con el fin de evidenciar la distribución de las predicciones categorizadas en VP, VN, FP, FN

6.2 Ciclo de Vida Incremental

El modelo de ciclo de vida incremental permite tener un desarrollo basado en incrementos, lo que permite tener un mayor control sobre el flujo del proyecto. También permite estructurase junto al cronograma planteado, logrando así alcanzar un crecimiento progresivo de la funcionalidad global del proyecto.

6.2.1 Fase 1

En esta fase se procedió a descargar los pcaps Benignos y Malignos del laboratorio Stratosphere Lab (https://www.stratosphereips.org/). Se almaceno en un archivo CSV las URL de descarga de cada uno de los pcaps, el nombre que tenía y el nuevo nombre que se le dio al archivo, permitiendo generar esquemas similares a la *Grafica 9*

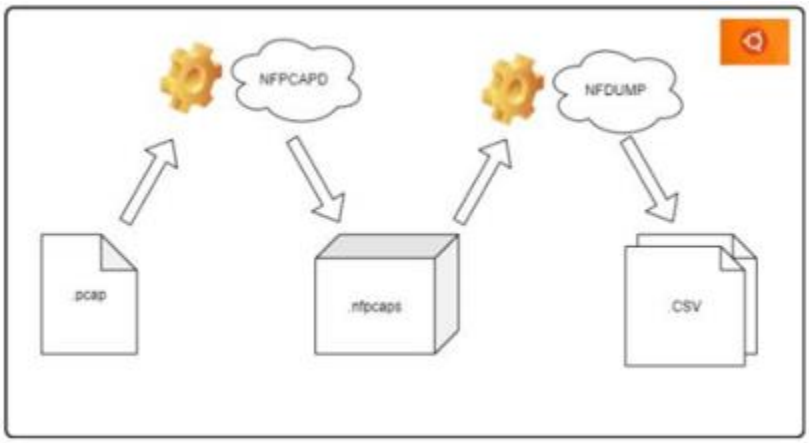
	Ref	Download	Pcap Name
0	1	https://mcfp.felk.cvut.cz/publicDatasets/CTU-M...	2018-05-03_win12
1	2	https://mcfp.felk.cvut.cz/publicDatasets/CTU-M...	162.222.213.28
2	3	https://mcfp.felk.cvut.cz/publicDatasets/CTU-M...	2018-04-04_win16
3	4	https://mcfp.felk.cvut.cz/publicDatasets/CTU-M...	2018-04-03_win12
4	5	https://mcfp.felk.cvut.cz/publicDatasets/CTU-M...	2018-04-03_win11
...	...	...	...
75	76	https://mcfp.felk.cvut.cz/publicDatasets/CTU-M...	2017-06-24_win8
76	77	https://mcfp.felk.cvut.cz/publicDatasets/CTU-M...	2017-06-24_win7
77	78	https://mcfp.felk.cvut.cz/publicDatasets/CTU-M...	2017-06-24_win6
78	79	https://mcfp.felk.cvut.cz/publicDatasets/CTU-M...	2017-06-24_win5
79	80	https://mcfp.felk.cvut.cz/publicDatasets/CTU-M...	2017-06-24_win4

80 rows × 3 columns

Grafica 9: Ejemplo de CSV Generado en Fase 1

6.2.2 Fase 2

En esta fase se procedió a realizar el montaje del laboratorio de investigación. Para ello, se utilizó una distribución Ubuntu (Linux) y las herramientas mencionadas en la *Grafica 10*



Grafica 10: Montaje del Laboratorio de Investigación

Las funcionalidades que nos permite realizar el laboratorio son:

1. Transformar un archivo PCAP a un conjunto de archivos NFPCAP

2. Renombrar el conjunto de archivos NFPCAPS a una secuencia de números consecutivos

3. Transformar el conjunto de archivos NFPCAPS en archivos CSV

4. Comprimir los CSV en un solo archivo a partir de un algoritmo generador, el cual permitiera generar las variables descritas en el entendimiento de los datos de CRISP-DM

6.2.3 Fase 3

En esta fase se realizó todo lo relacionado con la analítica de datos, evidenciado en la metodología CRISP-DM mencionada anteriormente.

6.2.4 Fase 4

En esta fase se procedió a implementar una aplicación web, la cual contemplaba los unos requerimientos funcionales y un diagrama de deployment creado a partir de ellos.

6.2.4.1 Requerimientos Funcionales

1. La aplicación debe permitir tener un módulo de minería de datos, el cual realizara las siguientes funcionalidades:

    a. Permitir correr un proceso de NFDUMP con el fin de extraer datos de la red local.
    b. Convertir el archivo PCAP generado cada determinada cantidad de paquetes, en un conjunto de archivos NFPCAPS
    c. Convertir el conjunto de archivos NFPCAP a archivos CSV.
    d. Transformar los archivos CSV a un solo archivo CSV que presente las variables mencionadas la fase de entendimiento de los datos de la metodología CRISP-DM.
    e. Eliminar todos los archivos generados que ya hayan sido analizados por el módulo de ciencia de datos

2. La aplicación debe permitir tener un módulo de ciencia de datos, el cual permitirá generar una predicción (cada determinada cantidad de paquetes), con los datos generados por el módulo de minería.

3. La aplicación debe permitir enviar una notificación al correo electrónico ingresado, en caso de que la predicción realizada por el módulo de ciencia de datos sea maligna.

4. La aplicación debe permitir persistir en una base de datos no relacional:

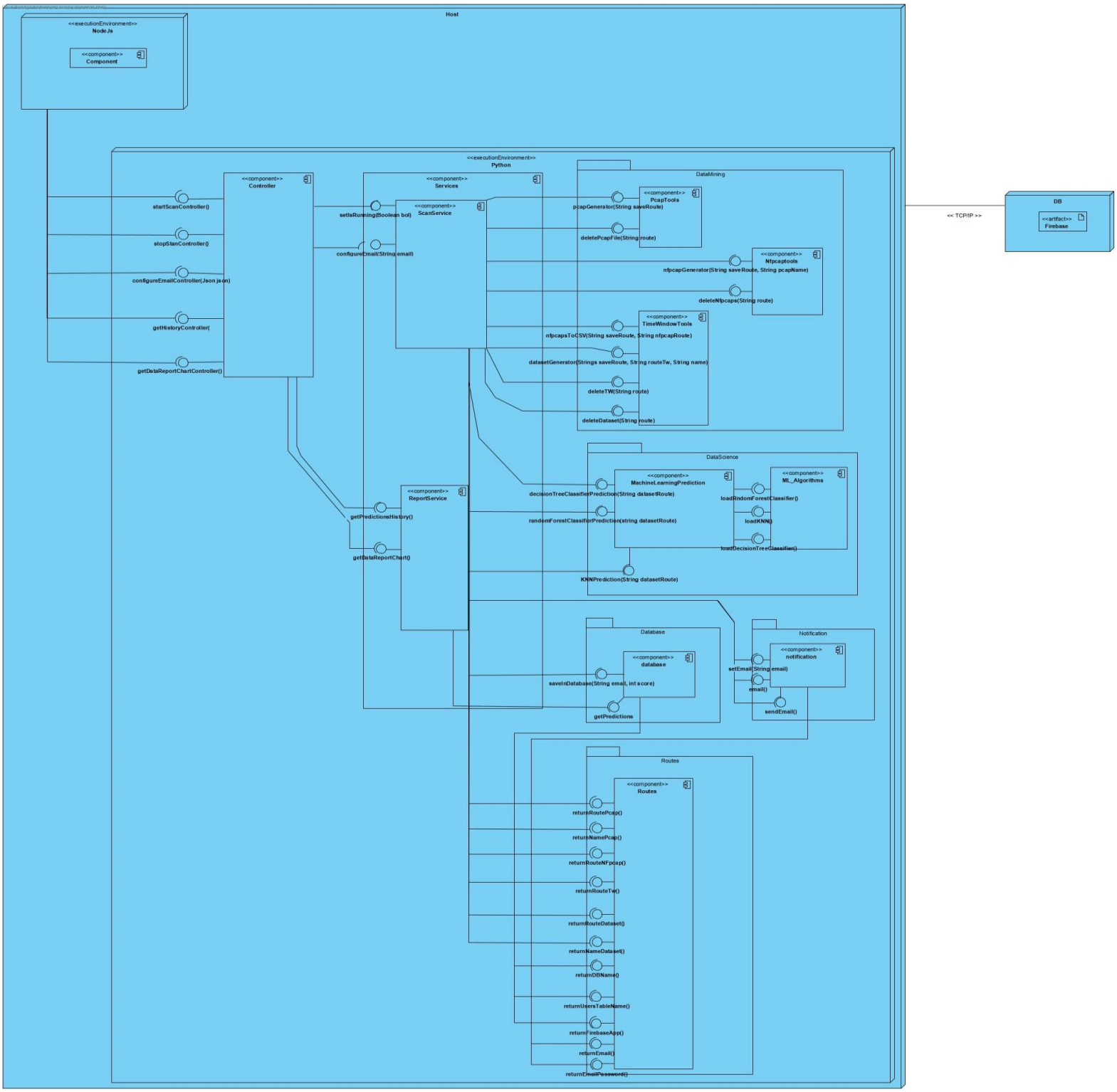
    a. Fecha y Hora
    b. Correo Electrónico
    c. Predicción
    d. Mensaje

5. La aplicación debe permitir por medio de servicios REST:
    a. Consultar el historial de predicciones
    b. Iniciar el servicio
    c. Detener el servicio
    d. Ingresar el correo electrónico de la persona que recibirá las notificaciones
    e. Generar un reporte del historial de predicciones

6.2.4.2 Diagrama de Deployment

El diagrama de deployment que contemplaba las interacciones entre los diferentes módulos de programación es el representado en la *gráfica 11*. Para mayor claridad puede consultarlo en la siguiente dirección:

- https://github.com/i2tResearch/Ciberseguridad



Grafica 11: Diagrama de Deployment

Resultados y Experimentos

7.1 Tecnologías Empleadas

7.1.1 Analítica de datos

Para el desarrollo de la fase relacionada con la analítica de datos, se utilizó la herramienta denominada **Jupyter Notebook** la cual nos permitió crear un conjunto de **Notebooks** que permitían registrar el proceso realizado durante las fases del proyecto.

7.1.2 Front-end aplicativo

Para el desarrollo del Front del aplicativo web, se utilizó **Vue Js** como framework. Algunas de las dependencias usadas en conjunto con el framework mencionada fueron:

1. Axios
2. Vuetify
3. Vuex
4. ApexChart

7.1.3 Back-end aplicativo

Para el desarrollo del Back del aplicativo web, se utilizó **FLASK.** Esté es un framework que nos permite crear aplicaciones web, integrando servicios REST por medio de lenguaje Python.

7.2 Experimentos

7.2.1 Experimento I

En el desarrollo de este experimento se utilizó la siguiente distribución de datos:

1. 519 ventanas de tiempo benignas de Stratosphere Lab
2. 800 ventanas de tiempo malignas de Stratosphere Lab

Una vez realizada la limpieza de los datos se procedió a aplicar la técnica **Feature Importance**. Esta técnica nos permitió extraer las variables más representativas, entre las cuales tenemos:

1. First_sp
2. Avg_bps
3. P1_ib
4. Duration
5. Number_dp
6. Bytes
7. Number_sp
8. First_Protocol
9. P2_ib
10. First_dp
11. P3_ib
12. Netflows
13. P3_d
14. Second_Protocol
15. Type

El paso siguiente fue realizar la separación de los datos de entreno y testeo utilizando la técnica de Holdout (70,30). Se procedió a evaluar los modelos de ML mencionados en la sección de modelado de CRISP-DM.

Se creo un archivo donde se puede evidenciar en orden descendente el Accuracy de los modelos, con el fin de poder seleccionar más fácilmente los mejores. Los resultados se pueden evidenciar en la *Tabla 2 y 3*

Metrics in TRAIN of EXPERIMENT I			
index	**Model**	**Accuracy Value**	**CV**
1	KNN	99,78%	10
2	Random Forest Classifier	99,78%	10
3	Decision Tree Classifier	99,67%	10
4	Logistic Regression	98,04%	10
5	Gaussian Naive Bayes	93,93%	10

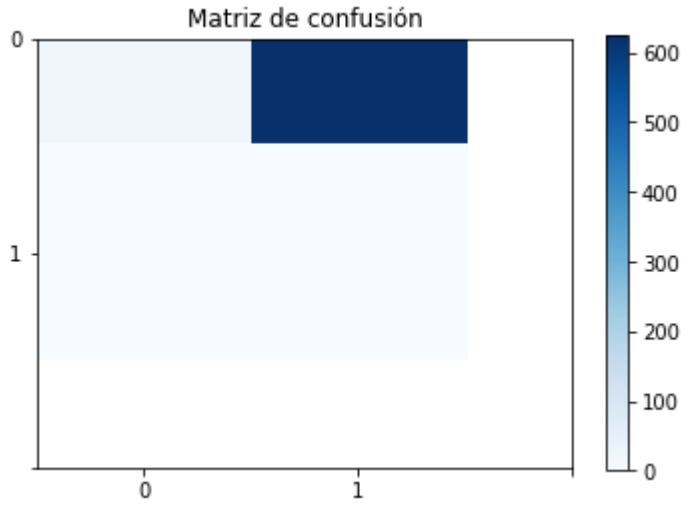
Tabla 2: Métricas en el entrenamiento de los modelos del Experimento I

Metrics in TEST of EXPERIMENT I				
index	**Model**	**Accuracy Value**	**Kappa**	**CV**
1	Random Forest Classifier	99,74%	99,45%	10
2	KNN	99,49%	98,91%	10
3	Decision Tree Classifier	98,98%	97,83%	10
4	Gaussian Naive Bayes	95,45%	90%	10
5	Logistic Regression	90,90%	79,55%	10

Tabla 3: Métricas en el testeo de los modelos del Experimento I

Los 2 mejores modelos de este experimento fueron el Random Forest Classifier con un Accuracy del 99,74% y KAPPA del 99.45% con una configuración de *criterion*="Gini", *max_depth*="5", *n_stimators*="64" y el KNN con un Accuracy del 99.49% y un KAPPA del 98.91% con una configuración de *metric*="Manhattan", *K*=5 y *weights*="distance" (Las anteriores configuraciones fueron extraídas por medio de la técnica de GridSearchCV)

Una vez obtenidos los mejores modelos, procedimos a probar ambos modelos con la muestra benigna destinada a realizar el intento de OVERSAMPLING (recuerde que optamos por utilizar UNDERSAMPLING debido a la cantidad de datos malignos que teníamos). El resultado no fue el esperado y se puede evidenciar a continuación:



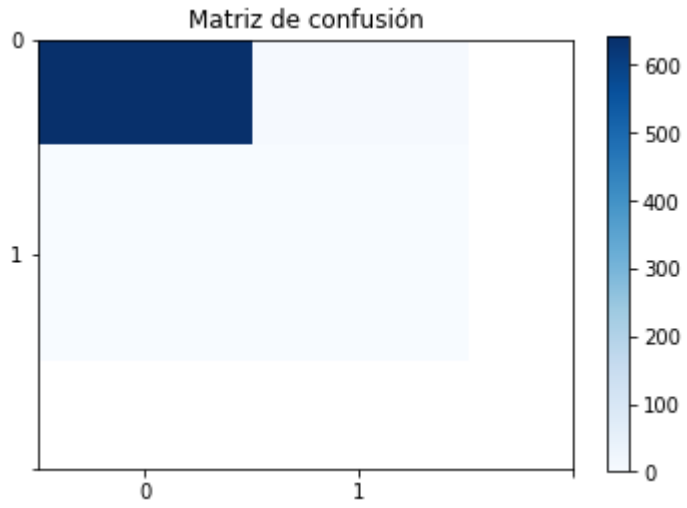
Grafica 12: Matriz de Confusión KNN Classifier Experimento I

En la *Grafica 12* podemos observar que hubo una gran cantidad de Falsos positivos (FP) predichos con el modelo KNN. De 649 datos etiquetados como benignos, el modelo predijo que tan solo 24 eran Benignos y los otros 625 eran malignos. Las métricas se pueden observar en la *Grafica 13*

	precision	recall	f1-score	support
0	1.00	0.04	0.07	649
1	0.00	0.00	0.00	0
accuracy			0.04	649
macro avg	0.50	0.02	0.04	649
weighted avg	1.00	0.04	0.07	649

Grafica 13: Métricas KNN Classifier Experimento I

Respecto al Random Forest Classifier, las predicciones no fueron tan desacertadas. De 649 datos benignos, el modelo predijo que 641 eran benignos y 8 eran malignos. En la *Grafica 14* podemos observar que existe una alta concentración de VP.



Grafica 14: Matriz de Confusión Random Forest Classifier Experimento I

Las métricas obtenidas con este modelo se pueden observar en la *Grafica 15*

	precision	recall	f1-score	support
0	1.00	0.99	0.99	649
1	0.00	0.00	0.00	0
accuracy			0.99	649
macro avg	0.50	0.49	0.50	649
weighted avg	1.00	0.99	0.99	649

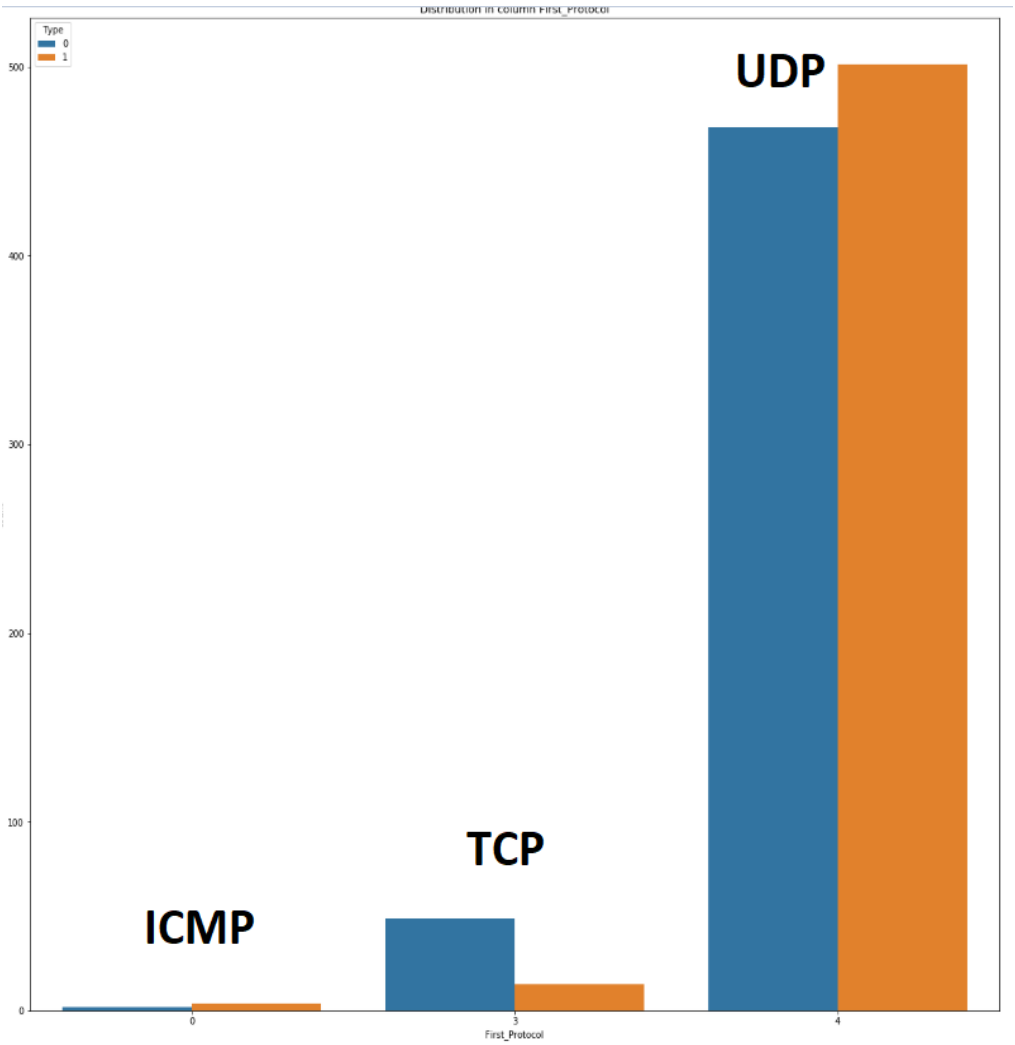
Grafica 15: Métricas Random Forest Classifier Experimento I

Como podemos observar, Random Forest mantuvo un Accuracy bastante elevado (98,76%) a diferencia del KNN (4%). Decidimos indagar por qué las métricas disminuyeron al probar ambos modelos con datos generados en nuestro laboratorio. La comparación la explicaremos en la siguiente sección.

7.2.2 Comparación datos Benignos (Stratosphere vs Our Lab)

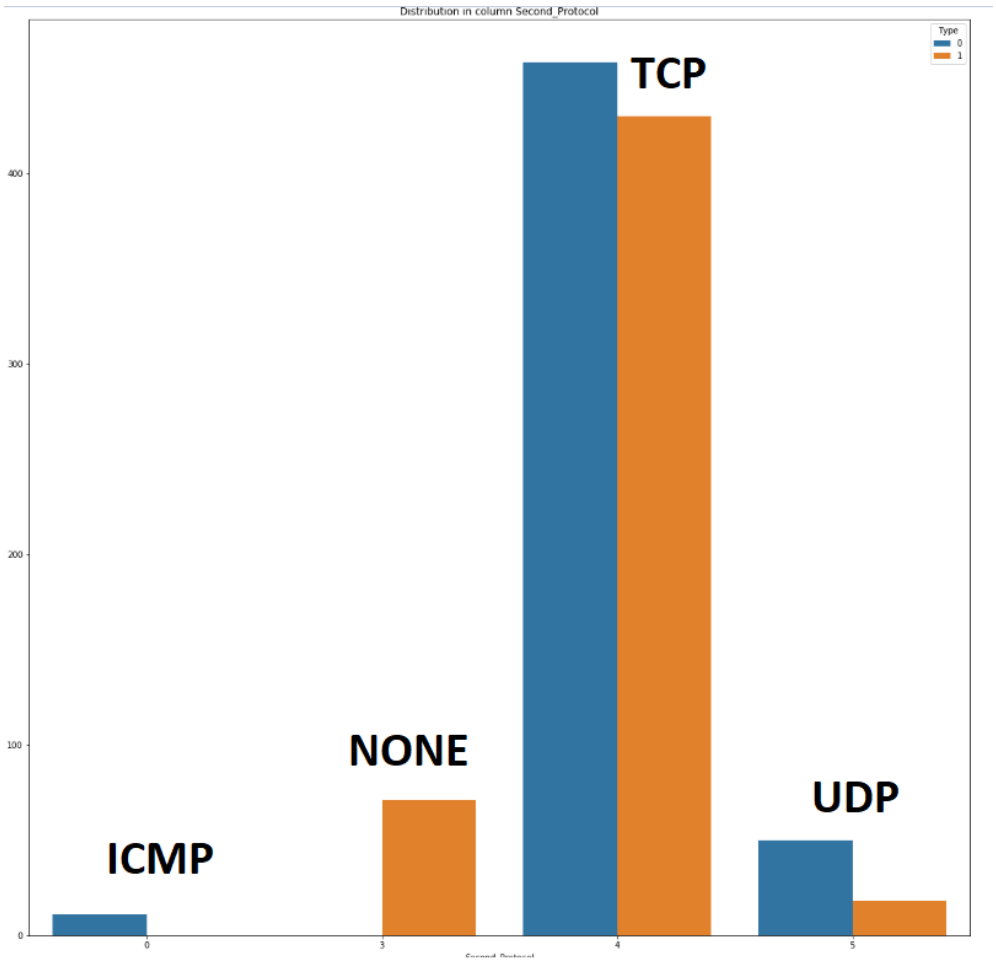
Producto de los resultados obtenidos en el experimento I, decidimos comparar nuestros registros benignos con los obtenidos por medio de Stratosphere Lab en cantidades iguales (519 por cada uno). Comparamos las variables filtradas por medio del Feature Importance y pudimos observar en algunas de las variables lo siguiente:

- La variable **First_Protocol** que representa el protocolo más usado (*Grafica 16*), se observa que existen protocolos donde se comparten ciertas similitudes. Podemos observar que el protocolo UDP fue uno de los protocolos más utilizados y tanto en los datos de Stratosphere (*naranja*) como en los nuestros (*azul*), se presenta en grandes cantidades. Sin embargo, el protocolo UDP tiene más registros en los datos de Stratosphere y TCP tiene más registros en los datos generados por nuestro laboratorio.



Grafica 16: Comparación Datasets en la variable First_Protocol

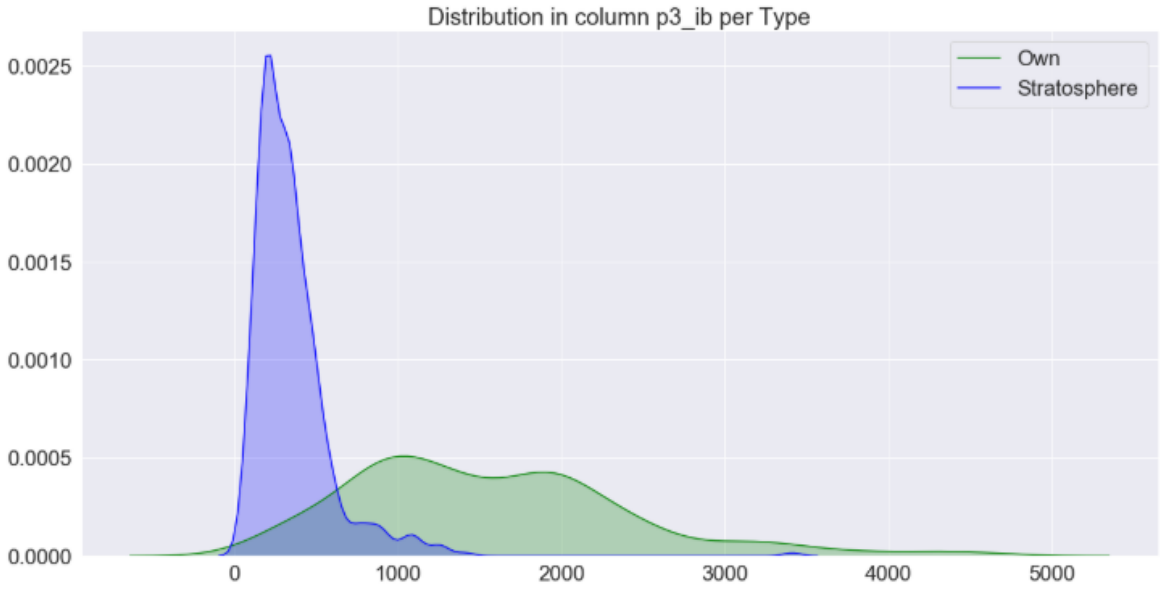
- En la variable **Second_Protocol** que representa el segundo protocolo más evidenciado en las ventanas de tiempo (*Grafica 17*), se pueden observar unas diferencias más claras. Podemos observar que existe una alta concentración de datos en el protocolo TCP. Sin embargo, nuestros datos (*azul*) presentan más registros TCP que Stratosphere (*naranja*). De igual forma sucede en los protocolos UDP e ICMP. Sin embargo, Stratosphere presenta más registros de datos en un protocolo no evidenciado en nuestros registros, el cual es el protocolo None. None es un protocolo no identificado por la herramienta de NFDUMP, el cual es categorizado como un protocolo desconocido.



Grafica 17: Comparación Datasets en la variable Second_Protocol

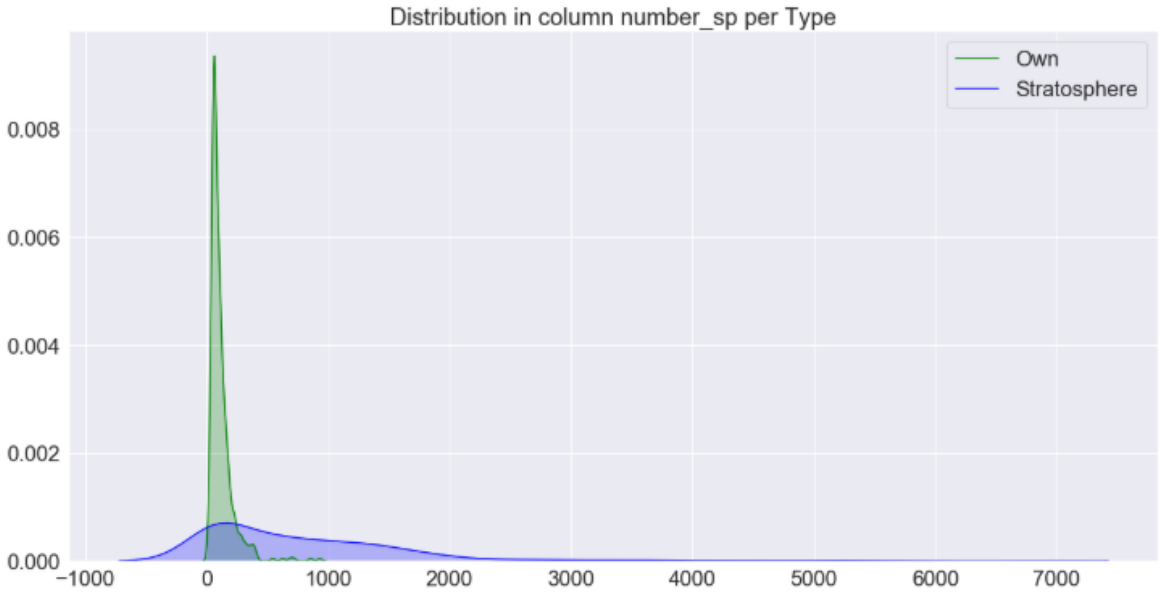
- En algunas variables como **p3_ib**, **number_sp**, **number_dp**, las cuales representan el percentil 3 (75%) de bytes de ingreso, numero de puertos de origen y puertos de destino involucrados respectivamente, se puede evidenciar diferencias muy claras.

    - En la variable **p3_ib** (*Grafica 18*)**,** podemos ver que el rango de los datos de Stratosphere (*azul*) oscilan entre 0 y casi 3500 bytes de entrada. En cambio, los datos generados en nuestra investigación (*verde*), oscilan entre 0 y casi 5000 bytes de entrada

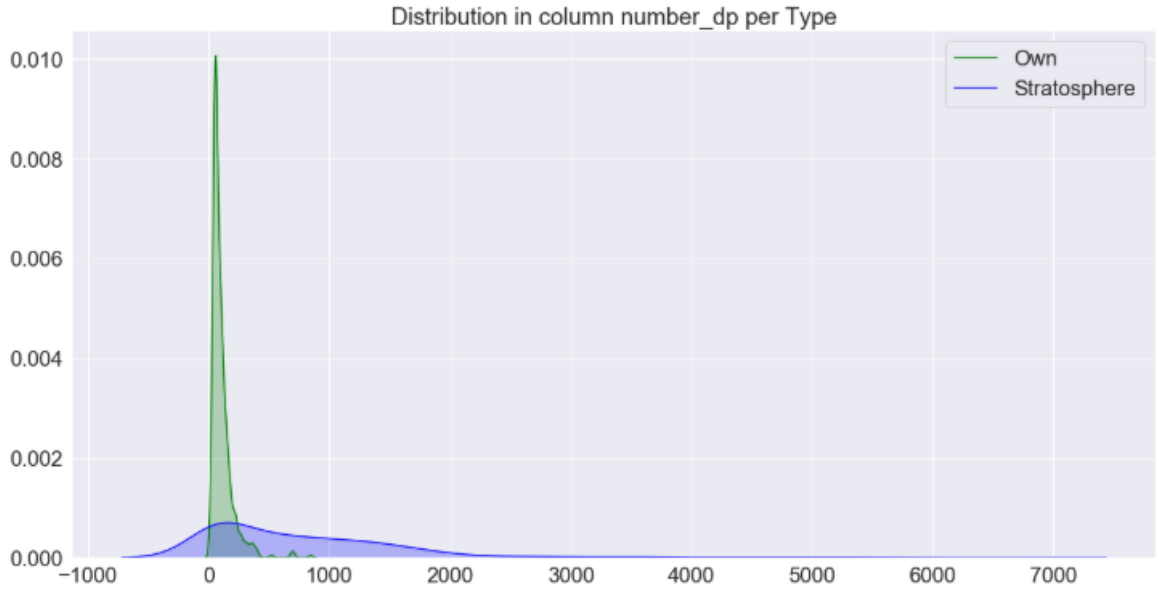


Grafica 18: Comparación Datasets en la variable p3_ib

- En las variables **number_sp** (*Grafica 19*) y **number_dp** (*Grafica 20*)**,** podemos observar que el rango de los datos generados por Stratosphere Lab (*azul*) oscilan entre 0 y casi 7000 puertos de origen. En cambio, nuestros datos (*verde*) oscilan entre 0 y casi 1000 puertos de origen aproximadamente.



Grafica 19: Comparación Datasets en la variable number_sp



Grafica 20: Comparación Datasets en la variable number_dp

En conclusión, si existen diferencias entre ambos Datasets Benignos. En un siguiente experimento incluiremos registros de ambos laboratorios con el fin de complementar aún más los modelos anteriormente generados.

7.2.3 Experimento II

En el desarrollo de este experimento se utilizó la siguiente distribución de datos:

1. 519 datos Benignos de Stratosphere Lab
2. 649 datos benignos de nuestro laboratorio de investigación
3. 1200 datos Malignos de Stratosphere Lab

Una vez realizada la limpieza de los datos se procedió a aplicar nuevamente la técnica **Feature Importance**. Esta técnica nos permitió extraer las variables más representativas, entre las cuales tenemos:

1. Avg_bps
2. Avg_pps
3. Bytes
4. P2_ib
5. Number_sp
6. First_Protocol
7. Number_dp
8. Duration
9. First_sp
10. Netflows
11. P3_ib
12. P3_d
13. Type

Como podemos ver, algunas de las variables que fueron representativas para el experimento I ya no lo son y por ende el número de variables contempladas ahora es menor.

Nuevamente se realizó la separación de los datos de entreno y testeo utilizando la técnica de Holdout. Se procedió a evaluar los modelos de ML mencionados en la sección de modelado de CRISP-DM por medio del GridSearchCV.

Se creo un archivo donde se puede evidenciar en orden descendente el Accuracy de los modelos, con el fin de poder seleccionar más fácilmente los mejores.

Metrics in TRAIN of EXPERIMENT II			
index	**Model**	**Accuracy Value**	**CV**
1	Random Forest Classifier	99,69%	10
2	KNN	99,51%	10
3	Decision Tree Classifier	99,45%	10
4	Logistic Regression	98,85%	10
5	Gaussian Naive Bayes	88,14%	10

Tabla 4: Métricas en el entrenamiento de los modelos del Experimento II

Metrics in TEST of EXPERIMENT II				
index	**Model**	**Accuracy Value**	**Kappa**	**CV**
1	KNN	99,85%	99,71%	10
2	Random Forest Classifier	99,85%	99,71%	10
3	Logistic Regression	99,29%	98,58%	10
4	Decision Tree Classifier	98,73%	97,45%	10
5	Gaussian Naive Bayes	89,42%	78,68%	10

Tabla 5: Métricas en el testeo de los modelos del Experimento II

Como podemos observar en la *Tabla 5*, Random Forest Classifier y KNN siguen siendo los mejores modelos de clasificación, ambos con un accuracy del 99,85% y un KAPPA del 99,71%. Sin embargo, le siguen otros modelos como el Decision Tree Classifier y el Logistic Regression.

Respecto al KNN, los parámetros arrojados por el GridSearchCV fueron:

- n_neighbors = 1
- Metric = "Manhattan"
- Weights = "Uniform"

Respecto al random forest classifier:

- n_estimator = 64
- max_features = "auto"
- max_depth = 6
- criterion = "Gini"

7.2.4 Experimento III

En este experimento se realizó un Testing de una serie de Datasets inyectados directamente en el aplicativo.

Por el lado de los Benignos, generamos unas muestras con diferentes cantidades de paquetes. Los resultados se encuentran en la *Tabla 6*.

Test Benign Data				
Index	Packets	Decision Tree	Random Forest	Logistic Regression
1	5000	Good	Good	Good
2	10.000	Good	Good	Good
3	20.000	Good	Bad	Good
4	500.000	Good	Good	Good

Tabla 6: Resultados Experimento III con las muestras Benignas

En los Malignos, La mayoría fueron extraídos del laboratorio de Stratosphere (garantizando que nunca pasaron por el proceso de entrenamiento de los modelos). Sin embargo, decidimos testear un Dataset proveniente de la Universidad de New Brunswick, la cual fue el resultado de la investigación "*Towards effective feature selection in machine Learning-based botnet detection approaches*". Los resultados se presentan en las *Tablas 7 y 8*

Test Malign Stratosphere				
Index	Source	Decision Tree	Random Forest	Logistic Regression
1	https://mcfp.felk.cvut.cz/publicDatasets/CTU-Malware-Capture-Botnet-204-1/	Good	Good	Good
2	https://mcfp.felk.cvut.cz/publicDatasets/CTU-Malware-Capture-Botnet-195-1/	Good	Good	Good
3	https://mcfp.felk.cvut.cz/publicDatasets/CTU-Malware-Capture-Botnet-192-1/	Good	Good	Good
4	https://mcfp.felk.cvut.cz/publicDatasets/CTU-Malware-Capture-Botnet-179-1/	Good	Good	Good
5	https://mcfp.felk.cvut.cz/publicDatasets/CTU-Malware-Capture-Botnet-169-1/	Good	Good	Good

Tabla 7: Resultados Experimento III con las muestras Malignas de Stratosphere

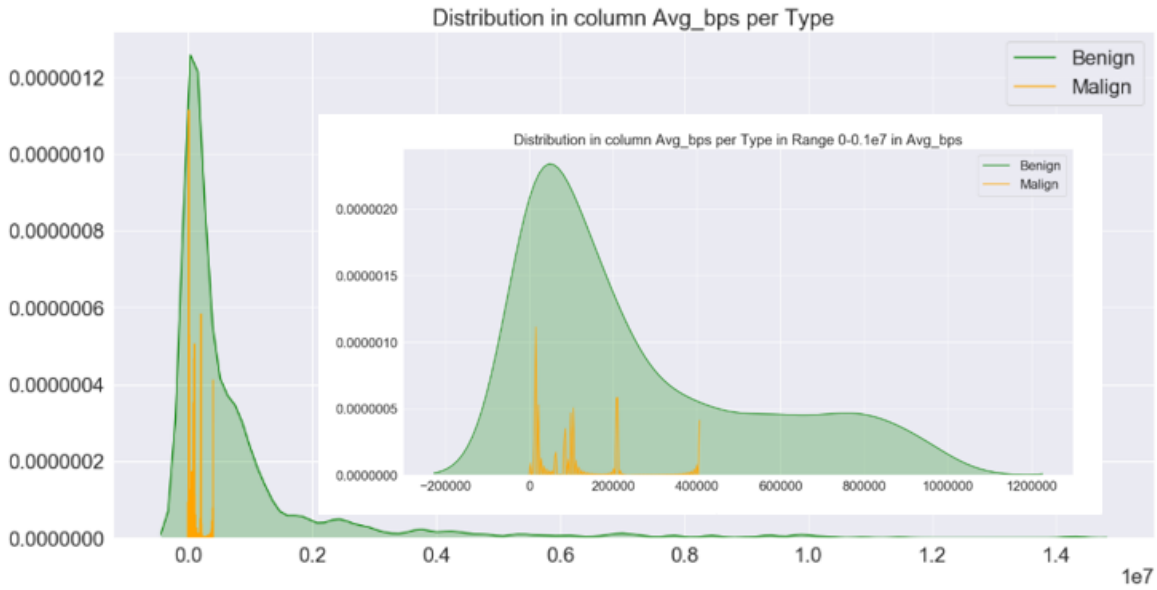
Test Malign New Brunswick University				
Index	Source	Decision Tree	Random Forest	Logistic Regression
1	http://205.174.165.80/CICDataset/ISCX-Bot-2014/Dataset/	Good	Good	Good

Tabla 8: Resultados Experimento III con las muestras Malignas de la Universidad de New Brunswick

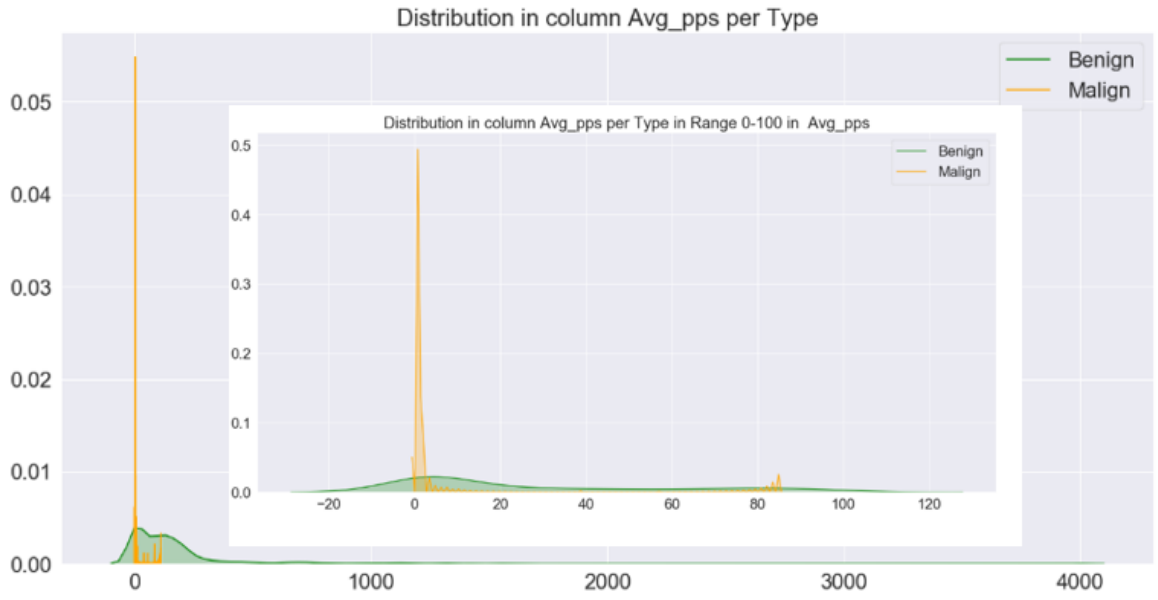
7.3 Análisis de las Variables obtenidas en el Experimento II

Producto de los resultados obtenidos en el experimento II, se decidió hacer un análisis más detallado de las variables obtenidas por medio del Feature Importance. A continuación, analizaremos los patrones de comportamiento evidenciados en cada una de ellas:

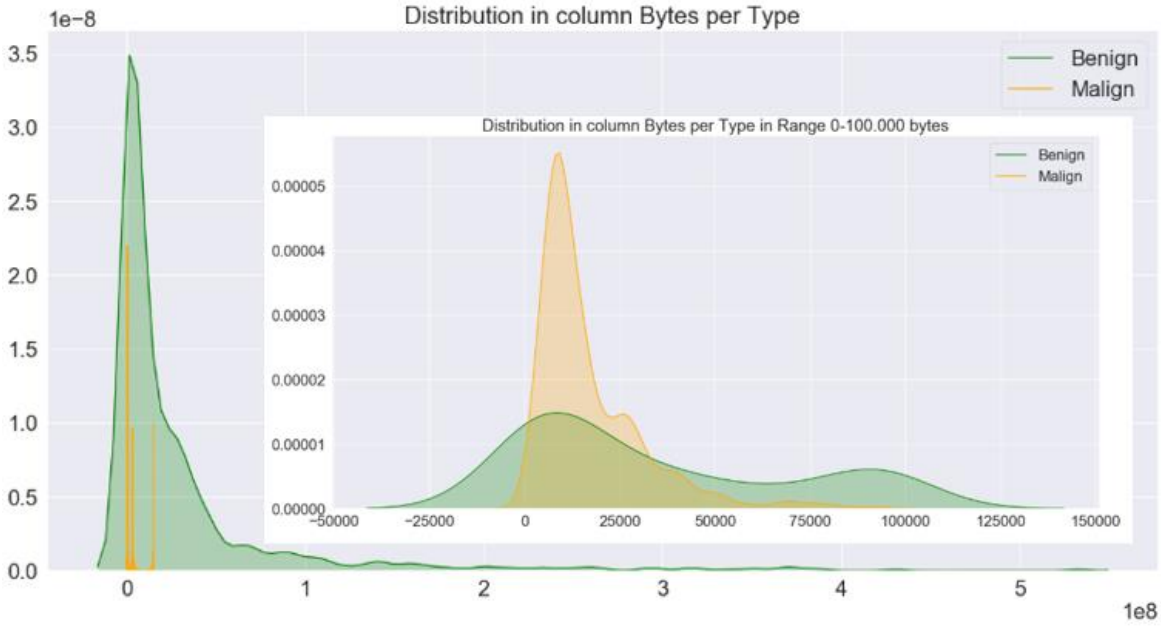
- En las variables **Avg-bps** (Average Bits per Second) (*Grafica 21*), **Avg_pps** (Average packets per Second) (*Grafica 22*) y **Bytes** (*Grafica 23*), se puede evidenciar que los datos benignos poseen un rango mucho mayor que los malignos. Podemos explicar este comportamiento en que las comunicaciones producidas por servidores de C&C, tienden a enviar periódicamente comandos de sincronización, pero no desean mantener conexiones que perduren en el tiempo, a diferencia de las comunicaciones normales que si requieren mantener un flujo constante de datos.



Grafica 21: Comparación Datasets en la variable Avg_bps

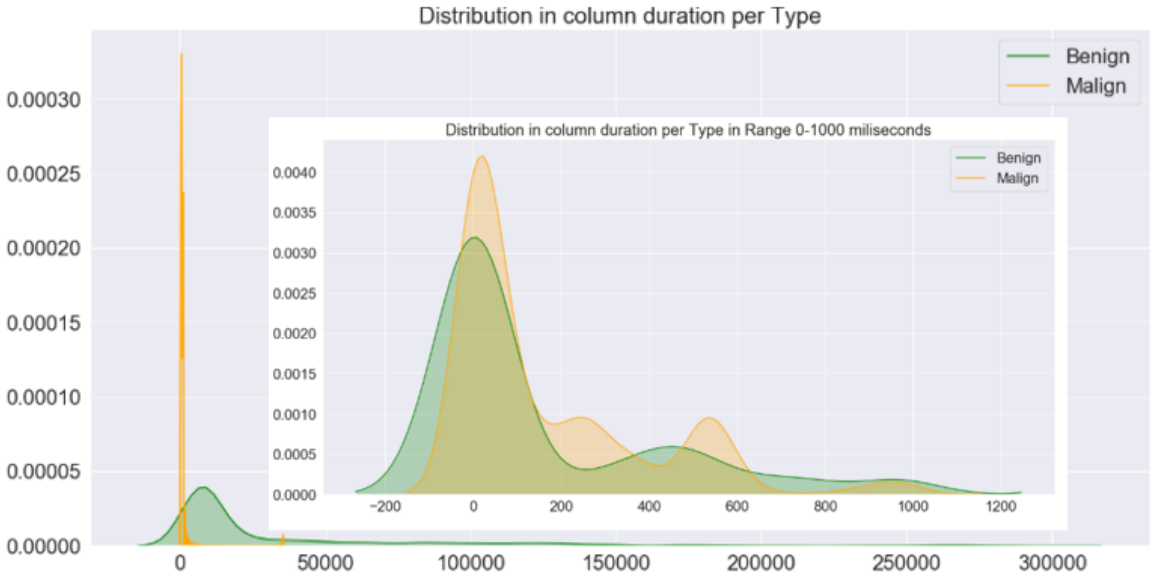


Grafica 22: Comparación Datasets en la variable Avg_pps

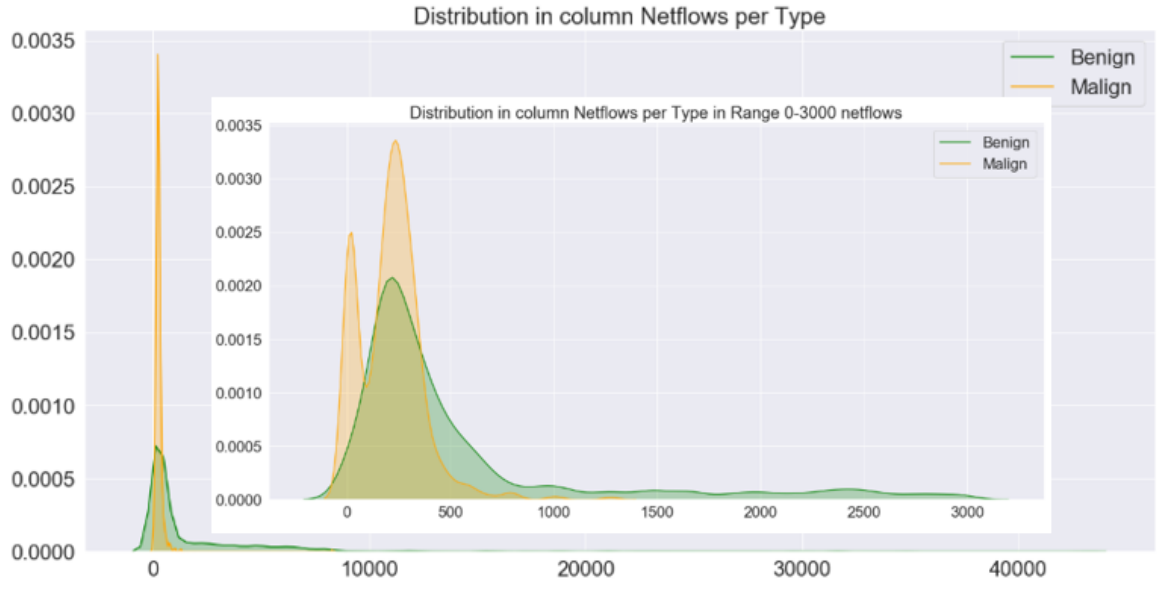


Grafica 23: Comparación Datasets en la variable Bytes

- En suma a lo anterior, variables como **duración** (*Grafica 24*) y numero de **Netflows** (*Grafica 25*), poseen el mismo comportamiento de las variables mencionadas con anterioridad.

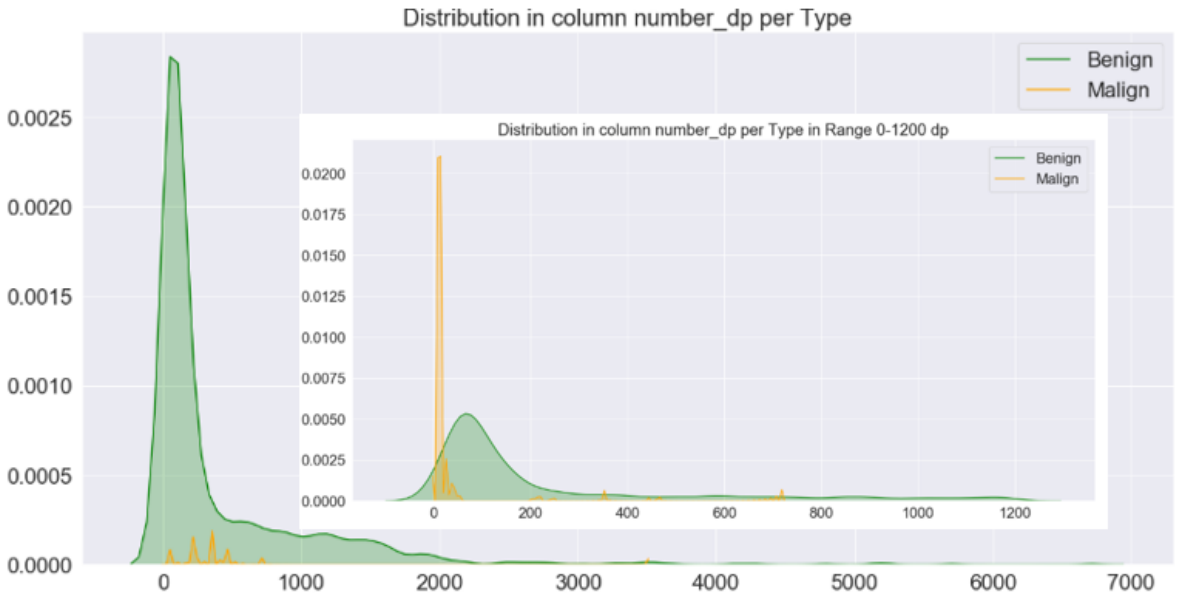


Grafica 24: Comparación Datasets en la variable Duration

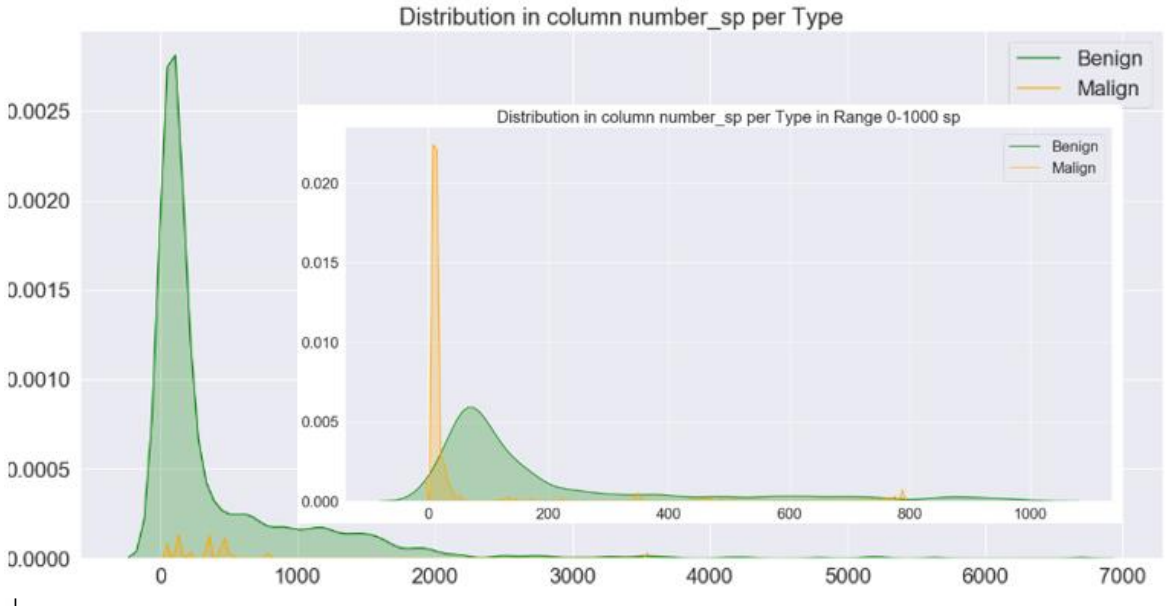


Grafica 25: Comparación Datasets en la variable Netflows

- En las variables que representan el **número de puertos** totales utilizados en las ventanas de tiempo (**origen y destino**), podemos observar que los flujos normales de datos utilizan un rango mucho más amplio de puertos, definidos en el tipo de comunicaciones y protocolos que desean utilizar. En cambio, las comunicaciones malignas utilizan un numero de puertos mucho más bajo, debido a que los cibercriminales de por sí ya conocen cuales puertos son los más vulnerables y los exploits están diseñados para trabajar sobre ellos

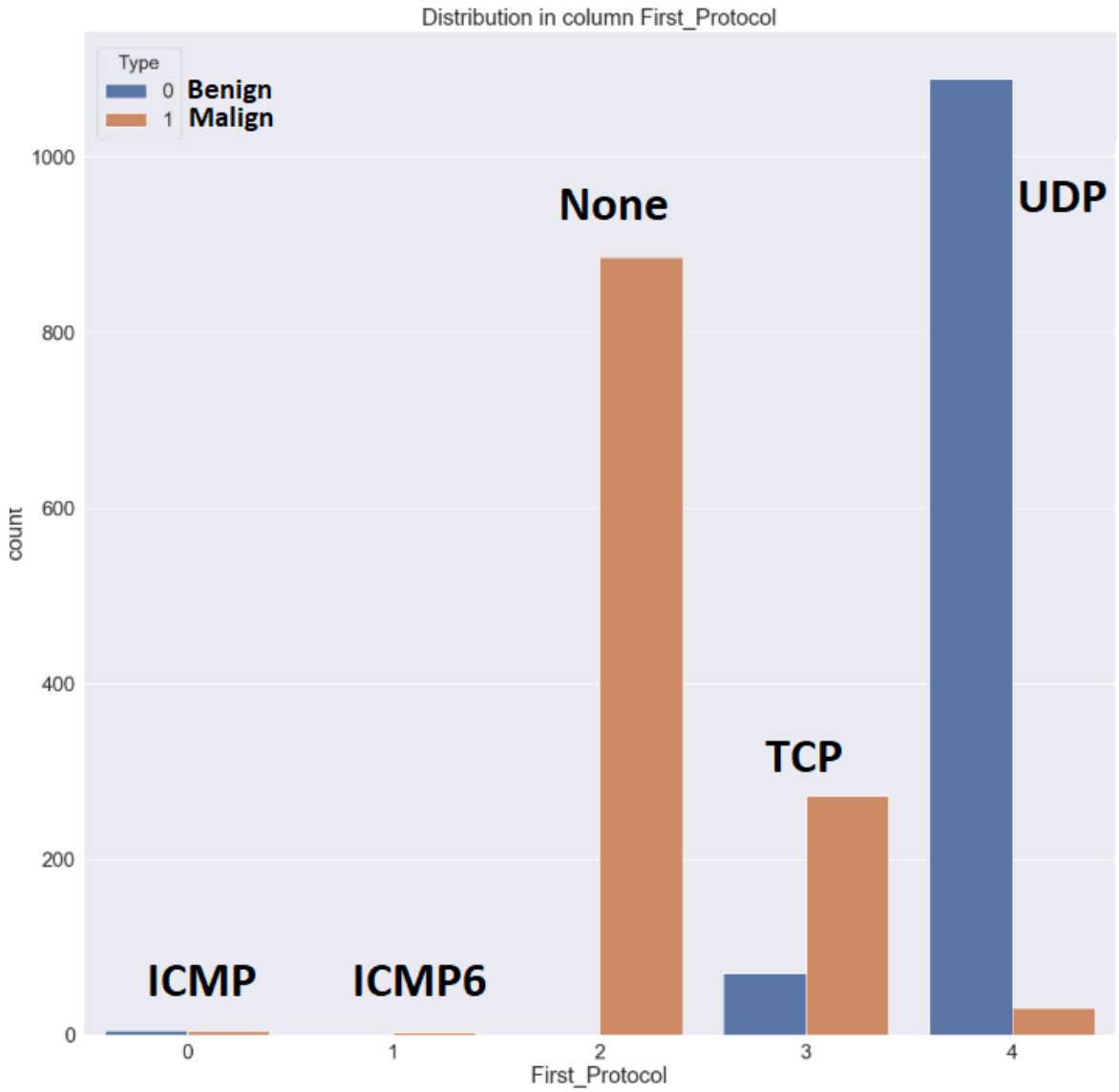


Grafica 26: Comparación Datasets en la variable number_dp



Grafica 27: Comparación Datasets en la variable number_sp

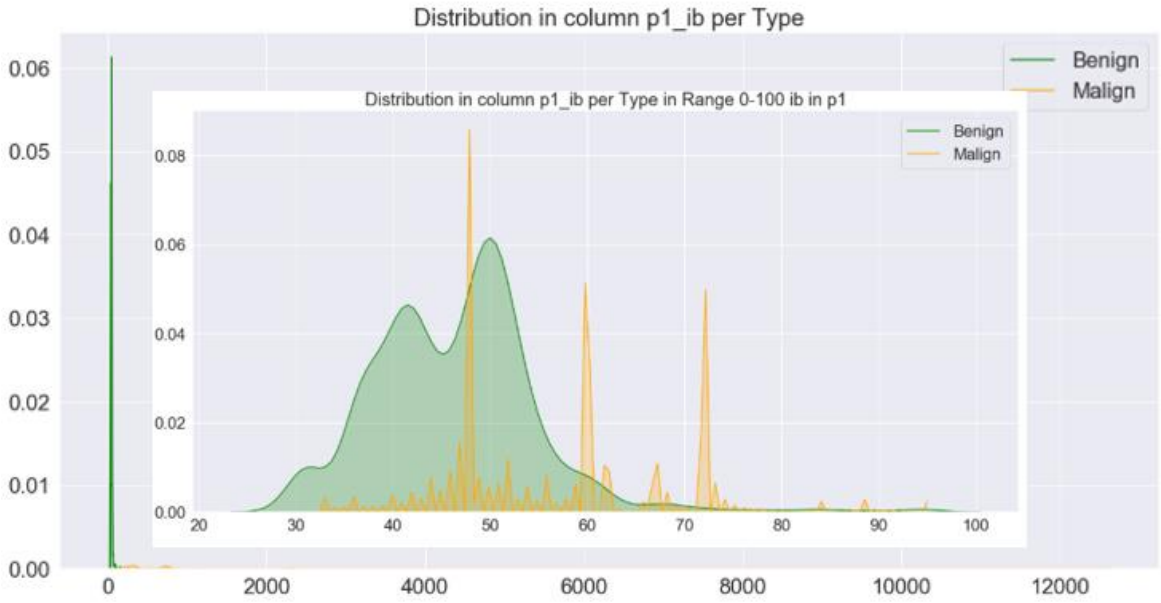
- En la variable **First_Protocol**(*Grafica 28*), podemos evidenciar que los datos benignos (*azul*) utilizan protocolos como UDP, TCP e ICMP (este último en pocas cantidades) para la transmisión de datos. Las comunicaciones malignas (*naranjas*) pueden transmitir su flujo de datos por medio de estos mismos protocolos, pero la diferencia radica en que también hacen uso de protocolos como ICMP6 y un protocolo no reconocido por la herramienta de NFDUMP, en la cual tenemos una hipótesis que explicaremos más adelante.



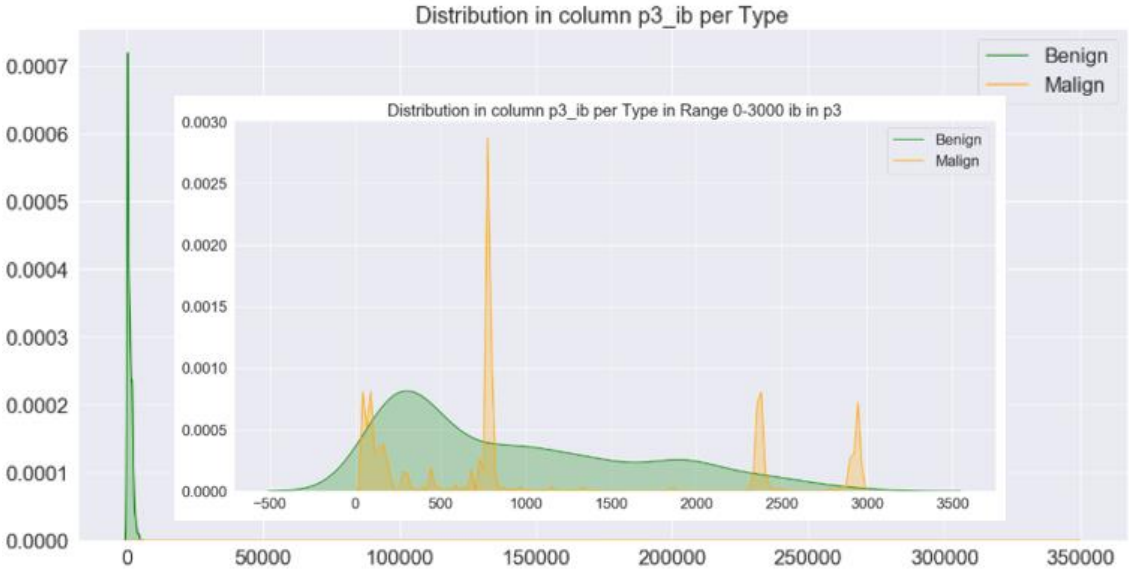
Grafica 28: Comparación Datasets en la variable First_Protocol

- En las variables **p1_ib** (percentil 1 de todos los bytes de entrada presentes en la ventana de tiempo) (*Grafica 29*), **p3_ib** (percentil 3 de todos los bytes de entrada presentes en la ventana de tiempo) (*Grafica 30*), podemos evidenciar que las comunicaciones malignas presentan un rango mucho mayor que los flujos normales de datos. Podemos pensar que esto se debe a que los servidores de C&C periódicamente pueden obligar a los "zombies" a realizar

escaneos a las redes subyacentes, minar datos en el caso que fueran utilizados para minado de criptomonedas, transmitir información entre los nodos con el fin de sincronizarse, descargar nuevos archivos binarios, entre otras.



Grafica 29: Comparación Datasets en la variable p1_ib



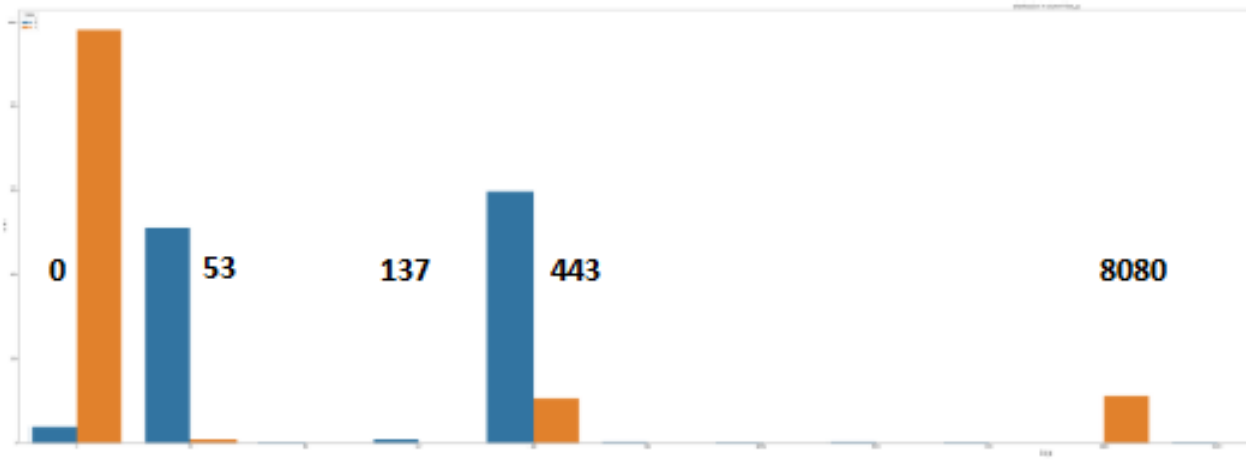
Grafica 30: Comparación Datasets en la variable p3_ib

- En la variable **First_sp** (Puerto de origen más utilizado) (*Grafica 31*), podemos evidenciar que los datos benignos (*azul*) utilizan frecuentemente puertos como el 53 (DNS), 443 (HTTPS), 80 (HTTP) y otros puertos relacionados con protocolos TCP/UDP. Sin embargo, podemos observar que en los datos malignos (*naranja*) existe un alto uso del puerto 0, siguiéndole el 443 (HTTPS) y el 8080 (HTTP).

  El puerto 0 es una forma abreviada de referirse a paquetes sin un encabezado L4 como TCP/UDP (Jones, 2013). Un ejemplo común de este tráfico es ICMP o cuando existe un tráfico IP fragmentado como una respuesta DNS que excede el tamaño máximo de 512 bytes.

  La explicación del puerto 0 lo hace realmente efectivo para los ataques de agotamiento de ancho de banda DDoS (Jones, 2013). Si la victima intenta bloquear el puerto 0, el equipo de reenvió de red puede rechazar la ACL como referencia a un puerto no legítimo y en este caso podríamos pensar que los servidores de C&C lo utilizan por esa misma justificación.

  Sin embargo, algunas aplicaciones pueden utilizar el puerto 0 para realizar un **bind()** de la conexión, el cual les permitirá obtener un puerto automáticamente designado por el sistema operativo que se encuentre disponible. (Speedguide, s.f.).

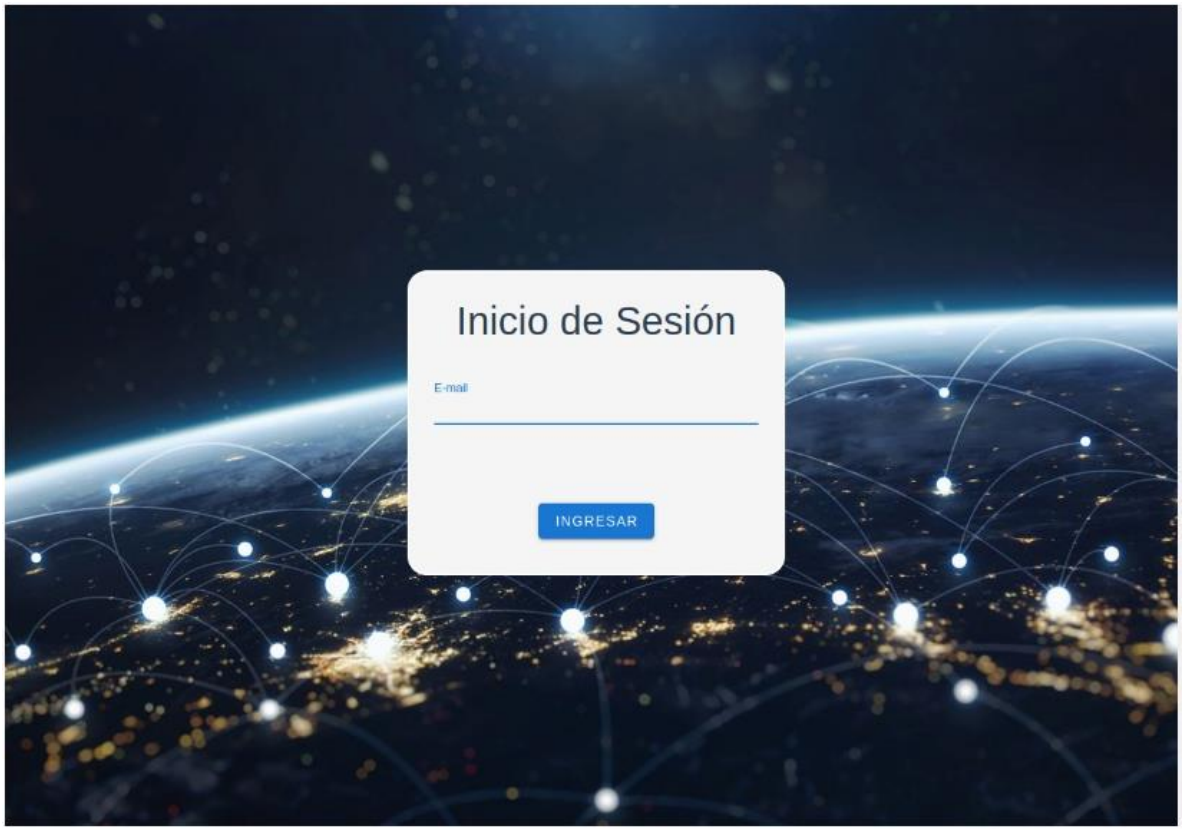


Grafica 31: Comparación Datasets en la variable First_sp

7.4 Aplicativo Web

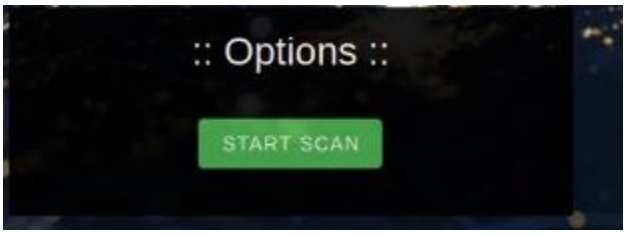
El aplicativo web una herramienta que permitirá a los usuarios:

1. Iniciar Sesión con su correo electrónico (*Grafica 32*). Este correo será utilizado con el fin de notificarle en caso de que un escaneo evidencia comportamiento anómalo.



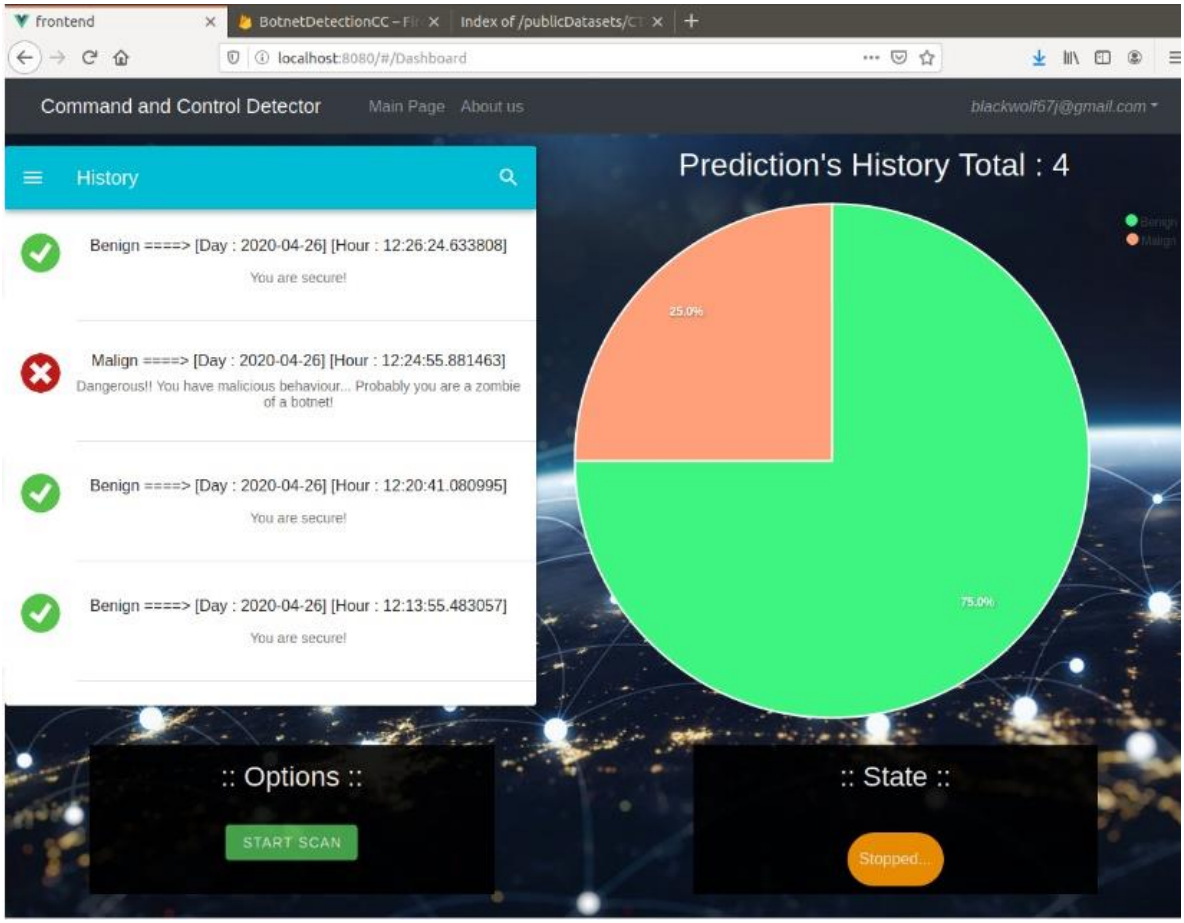
Grafica 32: Login Aplicativo Web

2. Iniciar y detener el escaneo personalizado (*Grafica 33*).



Grafica 33: Opción de Inicio de escaneo personalizado

3. Visualizar el historial de predicciones en una lista y un gráfico de pastel (*Grafica 34*)



Grafica 34: Dashboard aplicativo web

Análisis de Riesgos

Algunos de los riesgos y limitaciones que puede presentar nuestro proyecto son:

1) **Descontento por parte de agentes externos**: Como ya lo habíamos mencionado antes, existen grupos que pueden estar en desacuerdo con el desarrollo del proyecto, ya que estos se pueden ver directamente perjudicados en una posible solución del inconveniente plasmado en este documento.

Conclusiones

El uso de las 259.346 ventanas de tiempo, las cuales tuvieron una duración de 5 minutos cada una, permitió reducir el almacenamiento y procesamiento de los datos, a diferencia de haber trabajado con Netflows, lo cual nos permitió entrenar modelos de Machine Learning con una detección que oscila entre 90.4% y 99.8%.

El experimento II nos permitió llegar a unos modelos más robustos, ya que a través del entrenamiento de estos, y mediante el uso de 1.168 ventanas de tiempo benignas y 1.200 ventanas de tiempo malignas, se logró una tasa de detección que oscila entre el 89.4% y 99.8%. En suma a lo anterior, se llegó a la conclusión de que los mejores modelos fueron el KNN y el RANDOM FOREST, los cuales poseían una precisión del 99.8% y un coeficiente Kappa del 99.7%

A través del uso de las 13 variables presenciadas en el experimento II, las cuales comprometían características como promedio de bits por segundo, número total de bytes, duración de la ventana de tiempo, número de netflows, protocolo más usado, puerto origen más usado y número total de puertos origen y destino usados**,** pudimos visualizar que las conexiones malignas:

- Conocen los puertos más vulnerables.
- No perduran en el tiempo.
- Evidencian el envío de tráfico sin encabezado a través del puerto 0 como técnica para poder hacer ataques DDoS.

Todo lo anterior permite tener diferencias muy claras entre los patrones de comportamientos malignos y normales.

Trabajos a futuro

Actualmente el aplicativo web únicamente funciona para sistemas con distribución Linux, debido a que el laboratorio utiliza la herramienta NFPCAPD presente en NFDUMP, para realizar la transformación de archivos PCAP a ventanas de tiempo. Como trabajo futuro se pueden buscar o desarrollar herramientas que permitan realizar esta misma conversión en sistemas operativos como Windows y Mac OS, lo cual permitiría abarcar aún más la cantidad de equipos empresariales o personales.

En la presente investigación se hizo uso del aprendizaje supervisado para la predicción. Se recomienda a futuro plantear y realizar experimentos usando aprendizaje no supervisado, con el fin de encontrar similitudes y diferencias entre los patrones de comportamiento evidenciados anteriormente.

Para el futuro puede resultar útil analizar una mayor cantidad de datos, permitiendo así obtener nuevos y mejores resultados a la hora de detectar patrones de comportamiento en nuestro aplicativo, así como puede ser bastante interesante el uso de ventanas de tiempo más amplias con el fin de mejorar el entrenamiento de los modelos de machine Learning.

Por todo lo anterior, es necesario que se sigan realizando investigaciones de este tipo, debido a que los ciberdelincuentes no se detienen y cada día están buscando nuevas vulnerabilidades en los sistemas operativos, los cuales podrían usar a su favor para generar nuevos exploits.

Contribución

Nuestra contribución ira principalmente al área de ciberseguridad informática. Se han realizado muchas investigaciones acerca de cómo detectar y mitigar las botnet, pero muchas de ellas quedan obsoletas con el tiempo debido a la naturaleza dinámica de estos mecanismos. Confiamos que nuestra investigación permita aportar un granito de arena a todo el conocimiento que se ha generado durante cada una de estas investigaciones, con el fin de que algún día puedan ser mitigadas por completo.

Los resultados del proyecto permitirán a un administrador de la red o a un investigador evaluar si un nodo computacional está presentando un comportamiento propio de la fase de C&C en las botnets Malignas. Esto le permitirá contrarrestar el ataque salvaguardando así la seguridad e integridad tanto de los usuarios involucrados como de la red en general.

En suma a lo anterior, nuestra investigación ayudara a mitigar los ataques producidos por las botnets y esto permitiría ayudar a disminuir el número de dispositivos raptados y usados para generar ataques DDoS.

Por todo lo anterior, las fases de CRISP-DM y el aplicativo web generado en esta investigación serán de carácter **Open Source**. El proyecto se podrá encontrar en el siguiente enlace de GitHub: https://github.com/i2tResearch/Ciberseguridad

La ventaja de la presente investigación es el desarrollo del aplicativo web, el cual permitiría a un investigador reemplazar los modelos de machine Learning vistos anteriormente con algunos nuevos modelos, entrenados con datos actuales por ejemplo, lo cual permitirá reutilizar las herramientas y evitando que quede obsoleta en el transcurso del tiempo.

Bibliografía

Bhatia, J., Sehgal, R., & Kumar, S. (2011). *Botnet command detection using virtual honeynet*. ResearchGate.

Expert System. (07 de 3 de 2017). *Expert System*. Obtenido de Expert System: https://expertsystem.com/machine-learning-definition/#targetText=Machine%20learning%20is%20an%20application,use%20it%20learn%20for%20themselves.

Garcia, S. (Enero de 2014). *Researchgate*. Obtenido de archgate.net/publication/271204142_Identifying_Modeling_and_Detecting_Botnet_Behaviors_in_the_Network/download

Jain, D. (18 de junio de 2019). *NSFOCUS*. Obtenido de https://nsfocusglobal.com/nsfocus-shares-botnet-trends-new-2018-insights-report/

Jiménez, J. (07 de 8 de 2019). *Redes Zone*. Obtenido de Redes Zone: https://www.redeszone.net/2019/08/07/nueva-variante-echobot-mas-exploits/

Jones, T. (27 de August de 2013). *Endace*. Obtenido de Endace: https://blog.endace.com/2013/08/27/ddos-attacks-on-port-0-does-it-mean-what-you-think-it-does/

Kaspersky. (s.f.). *Kaspersky*. Obtenido de Kaspersky: https://www.kaspersky.com/resource-center/definitions/what-is-cyber-security

Kim, J., Sim, A., Tierney, B., Suh, S., & Kim, I. (2019). *Multivariate network traffic analysis using clustered patterns*.

Kupreev, O., Badovskaya, E., & Gutnikov, A. (5 de Agosto de 2019). *Kaspersky*. Obtenido de https://securelist.lat/ddos-report-q2-2019/89325/

Minarik, P. (2016). *How to analyze and understand your network*.

Muhammad, A., & Syed Mustafa, A. (2019). *DDoS attack detection with feature engineering and machine learning : the framework and performance evaluation*. International journal of information security.

Mushtaq, A. (18 de 7 de 2012). *FIREEYE*. Obtenido de FIREEYE: https://www.fireeye.com/blog/threat-research/2012/07/grum-botnet-no-longer-safe-havens.html

NetFort Technologies Limited. (2014). *Flow Analysis Versus Packet Analysis. What should you choose?*

Poremba, S. (4 de 5 de 2017). *eSecurity Planet*. Obtenido de eSecurity Planet: https://www.esecurityplanet.com/network-security/types-of-ddos-attacks.html

SPAMHAUS. (2019). *Botnet Threat Report 2019*.

Speedguide. (s.f.). *Speedguide*. Obtenido de SpeedGuide: https://www.speedguide.net/port.php?port=0

Urcuqui , C. C., García Peña, M., Osorio Quintero, J. L., & Navarro Cadavid, A. (2018). Ciberseguridad: un enfoque desde la ciencia de datos. En C. C. Urcuqui, M. García Peña, J. L. Osorio Quintero, & A. Navarro Cadavid, *Ciberseguridad: un enfoque desde la ciencia de datos* (pág. 86). Cali: Universidad Icesi.