COMILLAS
UNIVERSIDAD PONTIFICIA
ICAI

GRADO EN INGENIERÍA EN TECNOLOGÍAS INDUSTRIALES

TRABAJO FIN DE GRADO

Aplicación de técnicas de Machine Learning para mejorar la seguridad en entornos industriales

Autor: Valentín Manuel Moraga Gómez-Olea

Director: Álvaro Jesús López López

Madrid, junio 2020

Declaro, bajo mi responsabilidad, que el Proyecto presentado con el título

Aplicación de técnicas de Machine Learning para mejorar la seguridad en entornos industriales

en la ETS de Ingeniería - ICAI de la Universidad Pontificia Comillas en el

curso académico 2019/2020 es de mi autoría, original e inédito y

no ha sido presentado con anterioridad a otros efectos.

El Proyecto no es plagio de otro, ni total ni parcialmente y la información que ha sido

tomada de otros documentos está debidamente referenciada.

Fdo.: Valentín Manuel Moraga Gómez-Olea Fecha: 19/06/2020

Autorizada la entrega del proyecto

EL DIRECTOR DEL PROYECTO

Fdo.: Álvaro Jesús López López Fecha: 19/06/2020

GRADO EN INGENIERÍA EN TECNOLOGÍAS INDUSTRIALES

TRABAJO FIN DE GRADO

Aplicación de técnicas de Machine Learning para mejorar la seguridad en entornos industriales

Autor: Valentín Manuel Moraga Gómez-Olea

Director: Álvaro Jesús López López

Madrid, Junio 2020

APLICACIÓN DE TÉCNICAS DE MACHINE LEARNING PARA MEJORAR LA SEGURIDAD EN ENTORNOS INDUSTRIALES

Autor: Moraga Gómez-Olea, Valentín Manuel.
Director: López López, Álvaro Jesús
Entidad colaboradora: ICAI – Universidad Pontifica Comillas.

RESUMEN DEL PROYECTO

Palabras clave: Industria 4.0, Ciberseguridad, Aprendizaje automático, Protocolos de comunicación

1. Introducción

La Industria 4.0 está a día de hoy presente en prácticamente cualquier Sistema de Control Industrial (ICS por sus siglas en inglés). Esto es algo que no debería sorprender a nadie. No es un concepto o idea remota que pueda llegar a realizarse en un futuro y muchas empresas ya están aprovechando las numerosas oportunidades que esta nueva revolución industrial ofrece. Según un estudio de la PwC sobre la Industria 4.0, se espera que un 72% de las empresas alcancen altos niveles de integración de ésta y de digitalización hacia finales de este año [1]. Además, la situación causada por la COVID-19 ha acelerado este proceso no ha hecho más que acelerar este proceso fuertemente.

Se ha detectado un cambio de rumbo hacia la inteligencia operacional y prescriptiva en detrimento de técnicas analíticas meramente descriptivas. Esto se puede observar en varias tendencias que continuarán los próximos años en relación a la Industria 4.0 y al *"Industrial Internet of Things (IIoT)"*. Alguna de ellas son la recolección de grandes cantidades de datos para su posterior procesado y evaluación, la monitorización y gestión en remoto o niveles cada vez más altos de integración de datos y toma de decisión autónoma. Estas tendencias tendrán como consecuencia una mayor transparencia, un aumento de la eficiencia y una minimización del tiempo de inactividad de los equipos de fabricación. Todo esto con una repercusión directa sobre los niveles de producción. Por tanto, la adopción de herramientas de aprendizaje automático y de inteligencia artificial están cambiando la manera en la que empresas trabajan con sus sistemas industriales. No obstante, estas ventajas no están exentas de algún aspecto negativo, sobre todo en el ámbito de la seguridad.

Como conclusión se puede decir que el IIot y la Industria 4.0 presentan un nuevo paradigma y nuevas implicaciones, especialmente para la Ciberseguridad de los ICS. Que se analizaran a continuación.

Ahora mismo, el Desarrollo Tecnológico de equipos inteligentes y autónomos, herramientas de cloud computing o conexiones flexibles y ubicuas están empezando a tener un papel cada vez en mayor en los ICS como se ha visto antes. Esto requiere de Nuevas Arquitecturas en Sistemas de Control que cubran diferentes sectores como el de

la integración con servicios externos. Para ello estas arquitecturas tienen que ser dinámicas y distribuidas para tener un alto nivel de fiabilidad en los servicios externos. Todo esto conlleva nuevos desafíos para la Ciberseguridad de ICS al hecho que mayores niveles de conectividad han llevado a un aumento exponencial en la superficie de ataque. Consecuentemente, la probabilidad de ataque sobre estos equipos ha aumentado lo cual conlleva al riesgo de perder el control directo sobre la seguridad de un sistema [2].

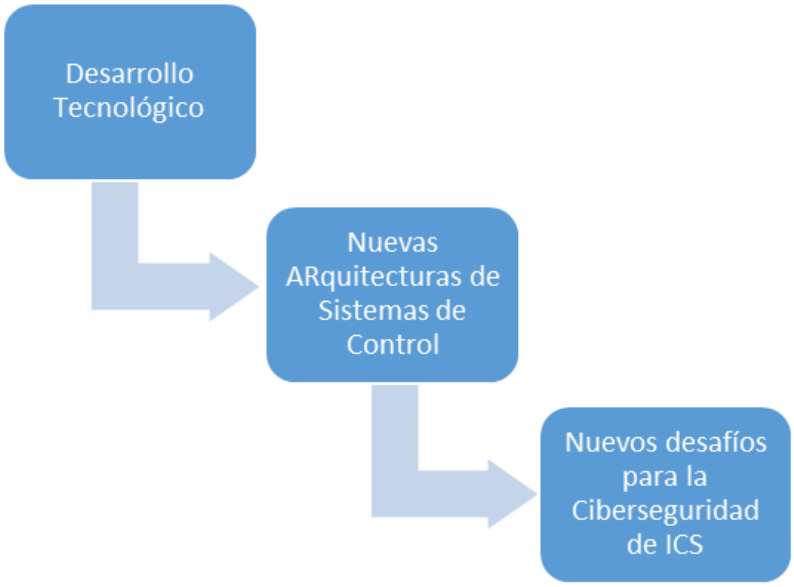


Ilustración 1: Implicaciones para la ciberseguridad industrial del IIoT

2. Estructura y Metodología

La metodología empleada a la hora de desarrollar este trabajo se divide principalmente en dos partes:

En primer lugar, se ha recabado información para poder evaluar a qué riesgos se enfrenta la industria a causa del proceso de transformación digital en el que está sumida. Para ello, se han evaluado cuáles son los protocolos de comunicación industrial más empleados a día de hoy. Éstos, se han analizado para ver qué ventajas e inconvenientes presentan de cara a la nueva era digital. Se han separado en dos grupos: Ethernet Industrial y redes basadas en Fieldbus, se ha analizado la cuota de mercado que tiene cada una, qué características poseen y se han buscado tendencias de cara al futuro. Para obtener la información necesaria se han utilizado estudios de empresas de ciberseguridad sobre cuotas de mercado e informes de entidades como el Instituto Nacional de Ciberseguridad (INCIBE). Finalmente se ha analizado el papel de un protocolo muy concreto, OPC-UA, de cara al futuro para ver si puede ver un buen sustito a los protocolos más tradicionales.

En segundo lugar, se ha llevado a cabo el desarrollo de un clasificador de malware capaz de detectar un ciberataque muy concreto (DDoS) y basado en técnicas de machine learning. Para ello se ha utilizado como lenguaje de programación Python y las diferentes

librearías que ofrece. Además, se ha utilizado un dataset del Instituto Canadiense de Ciberseguridad (CIC) para entrenar y probar el modelo desarrollado. Este set de datos incluía un total de 30GB de información. Cada vector incluye unas 80 características extraídas de la red mediante un *sniffer*. Estos datos se han tenido que reducir para que fueran manejables y analizables con un ordenador. Además, se han tenido que limpiar los datos y preparar las muestras ya que incluían mucha información que no era necesaria o que directamente era errónea. Se han normalizado los datos y se han utilizado para entrenar diferentes algoritmos de clasificación. Finalmente se han evaluado los resultados obtenidos para ver si estos algoritmos pueden ser útiles a la hora de proteger un Sistema de Control Industrial y ser una herramienta útil para compensar el aumento en la superficie de ataque que ha aparecido y de la que se habló antes.

3. Descripción del modelo desarrollado

El código se ha desarrollado en cuatro etapas:

3.1. Generación de muestras

Dado que el dataset contenía un total de 21GB de información, se veía necesario montar en primer lugar una serie de funciones que leyeran los diferentes archivos y tomaran aleatoriamente un número determinado de ataques y los guardaran en archivos más pequeños y manejables. Esto se ha hecho por un lado con vectores malignos, es decir, ataques y por el otro, con vectores benignos para finalmente crear un único archivo que contuviera todas las muestras y que permitiera procesar los datos desde un ordenador con una capacidad limitada.

Entre las variables disponibles para analizar se tienen, entre otras, el tiempo de duración, el número de paquetes enviados y recibidos, flujo de Bytes/s, flujo de Paquetes/s o el IAT (Tiempo entre llegadas). La lista total se adjunta de atributos se adjunta en el Anexo I.

3.2. Data Wrangling

Este paso es necesario dado que los datos, tal y como los almacena el analizador de red o sniffer, no se pueden analizar de manera exacta. Estos datos en su forma natural, a los que se suele referir como “Raw Data”, contienen frecuentemente errores de registro. Se puede dar que haya celdas vacías, celdas con valores que se registran como infinito, problemas al leer formatos como fechas u otros errores. Antes de analizar los datos hay que pre-procesarlos para limpiar, normalizar, y compatibilizarlos para que analizarlos y obtener resultados coherentes.

En este caso se han fijado los formatos de las diferentes columnas ya que al leer el archivo, alguno no se reconocía correctamente. Se han eliminado las columnas que contenían únicamente ceros ya que no aportaban información. Finalmente se han corregido celdas que contenían valores infinitos y que impedían ser procesadas [3]

3.3. Reducción de dimensiones – Análisis de componentes principales

Una vez limpiado y filtrado el dataset, se han pasado de 80 a unas 60 variables lo cual sigue siendo un número bastante elevado. Para ellos se ha recurrido al Análisis de Componentes Principales (PCA) que permite reducir la complejidad del problema reduciendo el número de variables sin apenas perder información. Al final, varias variables pueden medir lo mismo desde diferentes puntos de vista y aportar la misma información. También puede ser que estén fuertemente correladas, es decir, una puede ser combinación lineal de otras. El objetivo es encontrar el número de variables que expliquen la mayor parte posible de la varianza, esto se puede ver a través del "scree plot" que muestra la varianza explicada acumulada al añadir una componente principal más. [4]

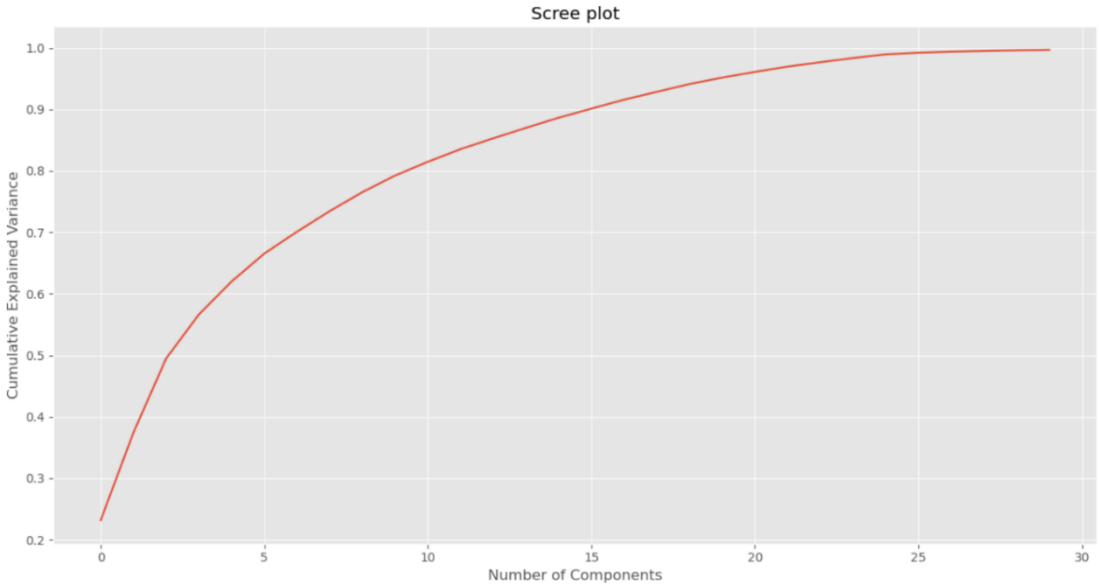


Ilustración 2: Scree plot, Principal Component Analysis

Como se puede observar en la figura, las 60 variables que quedaban, se pueden representar en tan solo diez que aportan más del 80% de la variabilidad del problema. Con 15, que sigue siendo un número bastante inferior, se llega al 90% lo cual es un valor muy bueno. Esto permite entrenar los algoritmos que se van a utilizar con un número notablemente inferior de variables lo cual aligera y simplifica el proceso.

Si se observa cómo se distribuyen las dos primeras componentes principales se observa ya una estructura que permite identificar ataques y vector benignos sin demasiada dificultad.

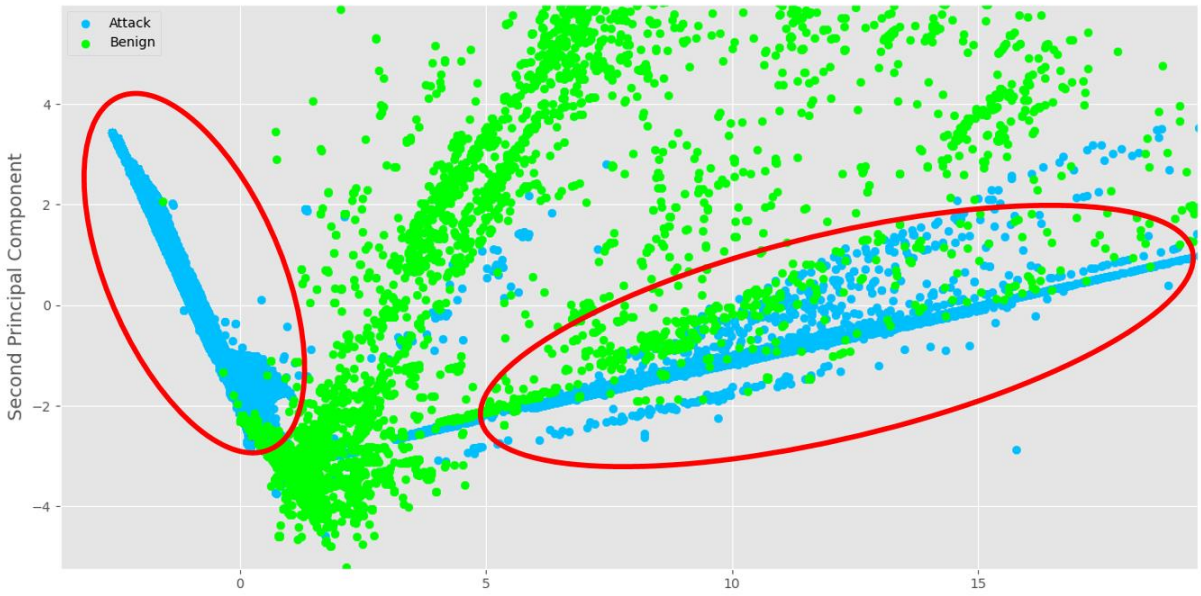


Ilustración 3: Primeras dos componentes principales, Scatter Plot

3.4. Aplicación de algoritmos de Machine Learning

Con los datos filtrados y preparados, se han elegido diferentes algoritmos de Machine Learning para analizar la viabilidad de este tipo de algoritmos a la hora de desarrollar herramientas que permitan proteger sistemas industriales ante ciberataques.

Los algoritmos elegidos han sido:

1. Random Forest
2. Neural Net
3. Adaptive Boost (AdaBoost)
4. Naive Bayes Classificator
5. Quadratic Discriminant Analysis (QDA)

Cada algoritmo se ha entrenado con una parte del dataset que se ha utilizado como Training Set y posteriormente evaluado con el resto que hacía de Test Set. Se ha obtenido de cada algoritmo la matriz de confusión, el score obtenido, el error, la precisión, la exhaustividad o "recall" y el Valor-F o "F1-Score". Los resultados obtenidos han sido los siguientes:

Classifier	Score	Error	Precision	Recall	F1-Score
Nearest Neighbors	99,77%	0,23%	99,84%	99,89%	99,86%
Decision Tree	99,41%	0,59%	99,75%	99,56%	99,65%
Random Forest	97,77%	2,23%	97,69%	99,74%	98,71%
Neural Net	97,11%	2,89%	96,91%	99,80%	98,33%
AdaBoost	99,24%	0,76%	99,62%	99,48%	99,55%
Naive Bayes	88,14%	11,86%	90,04%	96,84%	93,31%
Loigistic Regression	- No ha convergido -				

Número total de ataques: 134175
Número total de vectores benignos: 22823
Número total de muestras: 156998

Tabla 1: Resultados de los diferentes modelos

4. Resultados y conclusiones

El primer resultado al que se ha llegado y que sirve para marcar las líneas de investigación del resto de trabajo es que la industria está sufriendo una transformación profunda y transversal debida a llegada de la Industria 4.0

Estas nuevas tendencias traen consigo numerosas ventajas que permiten que las empresas sean más eficientes en sus procesos, más eficaces y tengan mayores capacidades de computación, cálculo y previsión gracias a la recogida de grandes cantidades de datos de prácticamente cualquier parte del proceso industrial.

No obstante, la transformación digital también trae consigo riesgos. El más importante que se ha detectado es el aumento exponencial en la superficie de ataque al desaparecer el "air gap" que había cuando las máquinas operaban de manera aislada y no estando conectadas a toda la red.

Por ello se hace necesario encontrar soluciones para prevenir y mitigar los peligros que pueden aparecer. Un primer punto de análisis han sido los protocolos de comunicación industriales que son los que se encargan de transmitir información a través de toda la fábrica y que son imprescindibles a la hora de evitar que alguien se infiltre en el sistema y en las comunicaciones.

Para ello se han analizado los principales protocolos utilizados a día de hoy que se pueden clasificar en Ethernet industrial y en redes Fieldbus. Aunque a día de hoy estas dos tecnologías copen en su gran mayoría el mercado, se han detectado principalmente dos puntos débiles que hacen pensar que estas dos familias no están llamadas a ser portadoras e impulsoras de la llamada cuarta revolución industrial. Los puntos son:

a) Son sistemas lentos y rígidos que dificultan en gran medida desarrollar las arquitecturas de los sistemas de control descritas al principio. Flexibilidad y dinamismo son dos características fundamentales que no se van a poder implantar. Esto aplica especialmente a Fieldbus.

b) La ausencia de seguridad impide la conexión con servidores externos. Eso no permite utilizar herramientas como el Cloud-computing que cada vez son más usuales en este tipo de entornos. Esto aplica especialmente a Ethernet.

Por ellos se hace necesario buscar alternativas y preguntarse, si estos protocolos están empezando a quedarse obsoletos, qué protocolo o protocolos pueden estar preparados para encarar esta nueva etapa en la industria. Como candidato se propone el protocolo basado en el estándar abierto OPC, OPC-UA. Los motivos principales que han llevado a pensar que este protocolo está preparado y va a ser bueno son:

- ✓ Utiliza una plataforma compatible con cualquier sistema operativo (OS)
- ✓ Preparada para incorporar tecnologías del futuro y para comunicarse con sistemas antiguos
- ✓ Fácil de configurar y mantener
- ✓ Es una tecnología orientada a los servicios
- ✓ Tiene un mayor alcance de conectividad
- ✓ Tiene un alto rendimiento

Finalmente se van a comentar los resultados obtenidos del análisis del dataset que se ha analizado. Este conjunto de datos provenía del Instituto Canadiense de Ciberseguridad y se centraba en concreto en el ataque DDoS. El objetivo de este análisis era analizar la viabilidad de utilizar técnicas de aprendizaje automático o machine learning para detectar y clasificar correctamente este tipo de ciberataques.

En primer lugar, hay que destacar que, al ser información proveniente de la red, un sniffer va a ser capaz de registrar una gran cantidad de datos que harán que se obtenga un archivo crudo muy grande y con muchísima más información de la necesaria. Por ello, no queda otra que filtrar, limpiar y preparar los datos para que sean manejables.

El haber realizado un análisis de componentes principales ha resultado ser fundamental para entender realmente la información y la estructura que subyacía al conjunto de datos. Se pudee observar como las variables ligadas a tiempos entre sucesos y cantidad de información transmitida son totalmente fundamentales a la hora de realizar un ataque de este tipo. Esto ha quedado recogido en las dos primeras componentes principales.

Respecto de los resultados obtenidos de los modelos entrenados con una parte de los datos y evaluados con el resto, se puede decir que los modelos han desempeñado muy bien todos. Las métricas que se han calculado han dado resultados muy positivos y se han clasificado la mayoría de los ataques y vectores benignos correctamente. Los resultados, como se ha comentado en el apartado correspondiente, son igual hasta demasiado positivos. Esto puede ser debido a que el conjunto de datos es sintético y que por ello los algoritmos no hayan tenido excesivas dificultades para entender su estructura. Sin embargo, aunque un conjunto de datos real hubiera arrojado resultados peores a estos, seguro que hubieran seguido estando dentro de un rango bastante admisible.

5. Futuras líneas de trabajo

Los modelos de aprendizaje automático presentan dos inconvenientes. El primero, que los atacantes también los pueden utilizar para hacer que sus ataques sean cada vez más sofisticados y dificulten su identificación. Además, estos modelos aprenden con observaciones pasadas, es decir, se podrían fabricar datos falsos que hagan que estos modelos aprendan mal. Esta técnica de intentar enseñar modelos a base de engañarlos se llama Adversarial Machine Learning y podría ser una potencial línea de investigación para futuros trabajos.

Otras ideas para continuar desarrollando serían ver más en detalle el funcionamiento de OPC-UA y analizar qué puntos débiles tiene este protocolo.

Respecto al modelo de aprendizaje automático, en un futuro se pretende desarrollar otro algoritmo que en tiempo real sea capaz de percibir cambios que puedan significar que hay un ataque inminente

6. Referencias

[1] PwC, (2016) *Industry 4.0: Building the digital Enterprise, Global Industry 4.0 Survey*, p.11

[2] Secure-by-Design Industrial Internet of Things ARC's Sid Snitkin at June 2015 ARC Industry Forum

[3] Qué es data wrangling – Parte I (12/06/2019) último acceso: 11/06/2020, https://www.gantabi.com/2019/06/12/que-es-el-data-wrangling-parte-i/

[4] Amat, J. (Junio 2019) *Análisis de Componentes Principales (Principal Component Analysis, PCA) y t-SNE*, último acceso: 17/06/2020, https://www.cienciadedatos.net/documentos/35_principal_component_analysis

APPLYING MACHINE LEARNING TECHNIQUES TO IMPROVE SAFETY IN INDUSTRIAL ENVIRONMENTS

Author: Moraga Gómez-Olea, Valentín Manuel.

Director: López López, Álvaro Jesús

Collaborating entity: Universidad Pontifica Comillas.

ABSTRACT

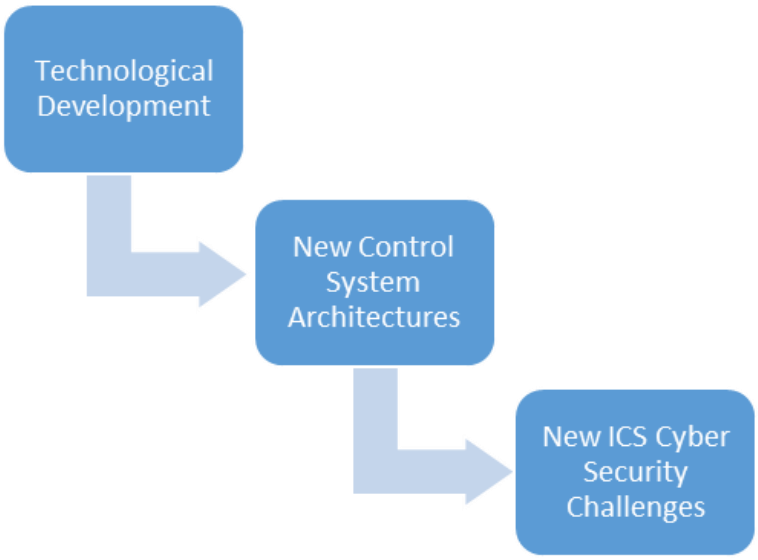
Key words: Industry 4.0, Cybersecurity, Machine Learning, Communication Protocols

1. Introduction

The fact that Industry 4.0 is nowadays present in almost every Industrial Control System (ICS) should not take anybody by surprise. It is not a remote concept that might become true in the future and many companies are already taking advantage of the opportunities that this new industrial revolution offers. According to a survey carried out by PwC on Industry 4.0, 72% companies expect to achieve advanced levels of integration and digitization by 2020 [1]. Furthermore, the current situation due to COVID-19 pandemic has accelerated this process.

There is a shift towards operational intelligence and prescriptive rather than descriptive analytics which can be seen in some trends that will continue into 2020 in regards to the Industrial Internet of Things (IIoT). Some of them are the increased adoption of collecting data for further processing and evaluation, remote monitoring and management or higher levels of data integration and autonomous decision-making. These trends will lead to higher levels of transparency, improvement of efficiency and minimization of downtime. All this with a direct impact in production levels. The process of adopting machine learning and AI techniques in order to manage IIoT devices is changing the way companies and businesses have to deal with their industrial systems. Therefore, IIoT and Industry 4.0 present a new paradigm and new implications, especially regarding ICS Cybersecurity.

Right now, new Technological Developments such as smart autonomous devices, cloud-computing tools or a flexible and ubiquitous connectivity are starting to play a major role in ICS. This requires New Control System Architectures that cover different branches such as integration with external services, dynamic and distributed architectures that have a high level of reliance in external services. All this entails New ICS Cybersecurity Challenges due to the fact that higher levels of connectivity have led to an exponential increase in the attack surface. Hence, the likelihood of attacks has increased and leading to the risk of losing direct control of security [2].



Graph 1: IIoT Implications for Cybersecurity

2. Methodology

The methodology used for this project can be divided into two parts:

First, information has been gattered in order to evaluate what risks is facing the industry due to the digital transformation process it is facing. Therefore, the most commonly used communication protocols have been evaluated and their strengths and weaknesses have been analysed. The focus was constantly set on how well prepared these traditionally used protocols are in order to face the changes in this new digital era. Therefore, they have been split into two different groups: Industrial Ethernet and Fieldbus based networks. The work includes an analysis of the market share of each type of protocol, more specific, how their share has developed in the last five years and what future trends are expected. In order to get the needed information for this analysis, a study carried out by cybersecurity companies and reports from public entities such as the INCIBE (Spanish National Cybersecurity Institute). Finally, the role of an emerging protocol has been taken into account and analysed: OPC-UA. Regarding the future and how we expect it looks like, this protocol could be a could substitute for Ethernet and Fieldbus in case they do not fit the new trends we expect.

Secondly, a malware classifier has been developed. The aim for this classifier was to see if it is able to identify and classify correctly a specific cyberattack. (DDoS). The model is based on machine learning algorithms. The programming language that has been used is Python and the different libraries it offers. Additionally, a dataset from the Canadian Institute For Cybersecurity has been used to train and to test the developed models. This dataset includes a total of 21 GB of information on that cyberattack. Each vector included 86 attributes extracted from the network traffic using a network analyser. These data were reduced in order to make them handy and manageable with a single computer. Otherwise it would have been impossible to process such a huge amount of data. Furthermore, the data have been cleaned, filtered and organised so that they were ready to be used to train models. Finally, the models'

results have been evaluated to see if these type of algorithms are useful in order to protect an Industrial Control System.

3. Description of the developed model

The code has been developed in four stages:

3.1. Samples generation

Since the dataset contained a total of 21GB of information, it was necessary to first set up a number of functions that would read the different files and randomly take a certain number of attacks and save them in smaller and more manageable files. This was done on the one hand with malignant vectors, i.e. attacks, and on the other hand with benign vectors to finally create a single file containing all the samples and allowing the data to be processed from a computer with limited capacity.

Among the variables available for analysis one can find the time duration, the number of packets sent and received, the Byte/s flow, the Packet/s flow or the IAT (Inter Arrival Time) and other. The total list of attributes cn be found in the Annex I.

3.2. Data Wrangling

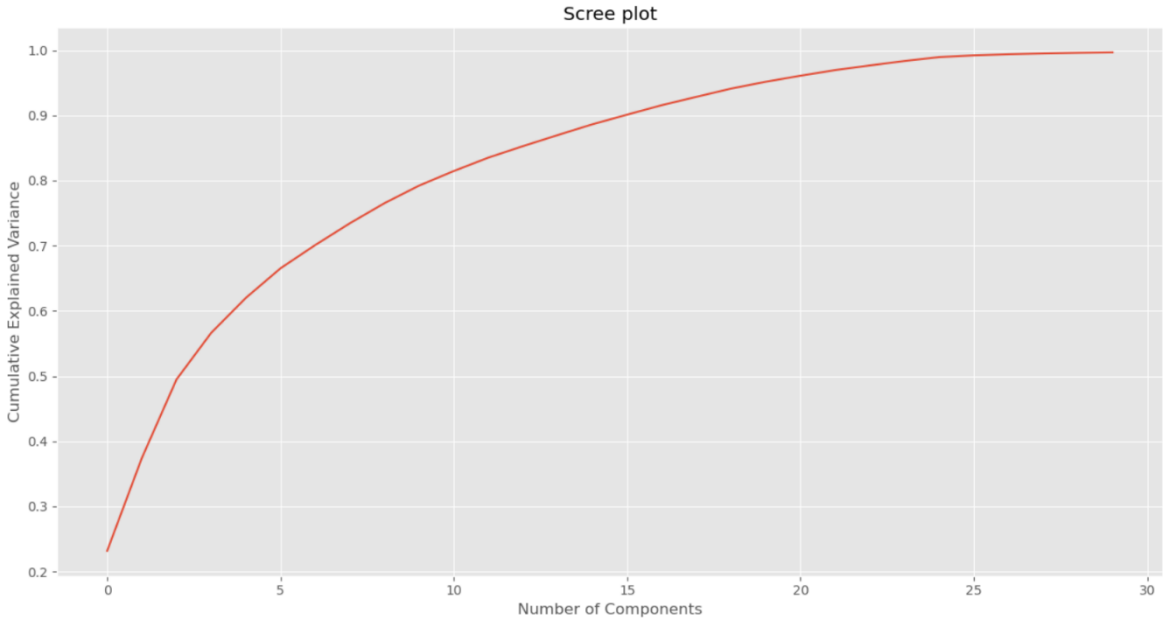
This step is necessary because the data, as stored by the network analyser or sniffer, cannot be analysed correctly. This data in its natural form, often referred to as "raw data", often contains recording errors. There may be empty cells, cells with values that are recorded as infinite, problems reading formats such as dates, or other errors. Before analysing the data, it must be pre-processed to clean, normalize, and match the data so that it can be analysed for consistent results.

In this case, the formats of the different columns have been fixed, since when reading the file, some of them were not recognized correctly. The columns that contained only zeros have been deleted, as they did not provide information. Finally, cells that contained infinite values and that prevented them from being processed have been corrected [3]

3.3. Reduction of dimensions – Principal Component Analysis

Once the dataset has been cleaned and filtered, the number of variables has decreased from 86 to about 60, which is still a fairly high number. For them, the Principal Component Analysis (PCA) has been used to reduce the complexity of the problem by reducing the number of variables without hardly losing any information. In the end, several variables can measure the same thing from different points of view and provide the same information. They may also be

strongly correlated, that is, one may be a linear combination of others. The objective is to find the number of variables that explain the most of the variance, which can be seen through the "scree plot" that shows the accumulated explained variance by adding one more main component.[4]

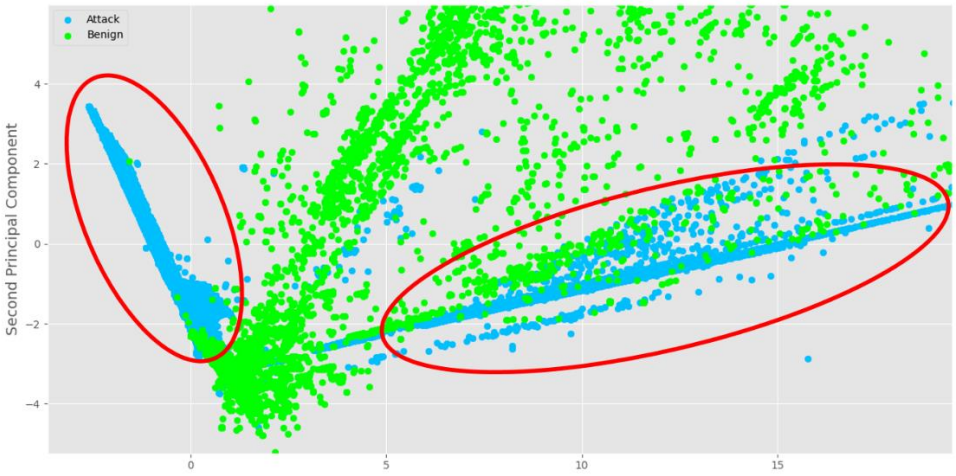


Graph 2: Scree Plot, Principal Component Analysis

As can be seen in the figure, the 60 remaining variables can be represented in only ten that contribute more than 80% of the variability of the problem. With 15, which is still a much lower number, we reach 90%, which is a very good value. This allows training the algorithms with a significantly lower number of variables which eases and simplifies the process.

If we observe how the first two main components are distributed, we can already see a structure that allows us to identify benign attacks and vectors without too much

difficulty.



Graph 3: Scatter plot, 2 first principal components

3.4. Application of Machine Learning algorithms

With the data filtered and prepared, different Machine Learning algorithms have been chosen to analyse the viability of this type of algorithm when developing tools to protect industrial systems against cyber attacks.

The algorithms chosen have been:

1. Random Forest
2. Neural Net
3. Adaptive Boost (AdaBoost)
4. Naïve Bayes Classificator
5. Quadratic Discriminant Analysis (QDA)

Each algorithm has been trained with a part of the dataset that has been used as a Training Set and later evaluated with the rest that was a Test Set. The confusion matrix, the score, the error, the precision, the recall and the F1-Score have been obtained for each algorithm. The results obtained are as follows:

Classifier	Score	Error	Precision	Recall	F1-Score
Nearest Neighbors	99,77%	0,23%	99,84%	99,89%	99,86%
Decision Tree	99,41%	0,59%	99,75%	99,56%	99,65%
Random Forest	97,77%	2,23%	97,69%	99,74%	98,71%
Neural Net	97,11%	2,89%	96,91%	99,80%	98,33%
AdaBoost	99,24%	0,76%	99,62%	99,48%	99,55%
Naive Bayes	88,14%	11,86%	90,04%	96,84%	93,31%
Loigistic Regression	- No ha convergido -				

Número total de ataques: 134175
Número total de vectores benignos: 22823
Número total de muestras: 156998

Table 1: Results after training the different models

4. Results and conclusions

The first outcome that has been found and that serves to mark the lines of research of the rest of the work is that the industry is undergoing a deep and transversal transformation due to the arrival of Industry 4.0

These new trends bring with them numerous advantages that allow companies to be more efficient in their processes, more effective and have greater computing, calculation and forecasting capacities thanks to the collection of large amounts of data from practically any part of the industrial process.

However, the digital transformation also brings with it risks. The most important risk detected is the exponential increase in the attack surface as the "air gap" that existed when machines operated in isolation and not connected to the entire network disappears. It is therefore necessary to find solutions to prevent and mitigate the dangers that may arise. A first point of analysis have been the industrial communication protocols that are responsible for transmitting information throughout the factory and are essential to prevent someone from infiltrating the system and communications. For this purpose, the main protocols used today have been analysed and classified into Industrial Ethernet and Fieldbus based networks. Although today these two technologies are mostly used in the market, we have detected two main weaknesses that make us think that these two families are not called to be carriers and drivers of the so-called fourth industrial revolution. The points are:

a) They are slow and rigid systems that make it very difficult to develop the control system architectures described at the beginning. Flexibility and dynamism are two fundamental characteristics that cannot be implemented. This applies especially to Fieldbus.
b) The lack of security prevents connection to external servers. This does not allow the use of tools such as Cloud-computing, which are increasingly common in this type of environment. This applies especially to Ethernet.

Therefore, it is necessary to look for alternatives and ask oneself, that if these protocols are starting to become obsolete, which protocol or protocols could be prepared to face

this new stage in the industry. As a candidate, the protocol based on the open standard OPC, OPC-UA, is proposed. The main reasons that have led us to believe that this protocol is ready and will be good are:

- ✓ It uses a platform compatible with any operating system (OS)
- ✓ It is ready to incorporate future technologies and to communicate with old systems
- ✓ Easy to set up and maintain
- ✓ It is a service-oriented technology
- ✓ It has a greater range of connectivity
- ✓ It has a high performance

Finally, we will comment on the results obtained from the analysis of the dataset that has been analysed. This dataset came from the Canadian Institute for Cybersecurity and focused specifically on the DDoS attack. The aim of this analysis was to analyse the feasibility of using machine learning techniques to detect and correctly classify this type of cyber attack.

Firstly, it should be noted that, as it is information from the network, a sniffer will be able to record a large amount of data that will result in a very large raw file with much more information than necessary. Therefore, there is no other choice but to filter, clean and prepare the data so that it is manageable.

Having performed a principal component analysis has proven to be critical to truly understanding the information and structure underlying the data set. It can be seen how the variables linked to time between events and the amount of information transmitted are totally fundamental when carrying out an attack of this type. This has been reflected in the first two main components.

Regarding the results obtained from the models trained with a part of the data and evaluated with the rest, it can be said that the models have performed very well all of them. The metrics that have been calculated have given very positive results and most of the attacks and benign vectors have been classified correctly. The results, as commented in the corresponding section, are maybe too positive. This may be due to the fact that the data set is synthetic and therefore the algorithms have not had excessive difficulties to understand its structure. However, even if a real data set would have produced worse results than these, they would surely have remained within a fairly acceptable range.

5. Future lines of work

There are two drawbacks to machine learning models. The first is that attackers can also use them to make their attacks increasingly sophisticated and difficult to identify. In addition, these models learn from past observations, i.e., false data could be fabricated to make these models learn poorly. This technique of trying to teach models by tricking them is called Adversarial Machine Learning and could be a potential line of research for future work.

Other ideas for further development would be to look in more detail at how OPC-UA works and to analyse the weaknesses of this protocol.

Regarding the machine learning model, in the future we intend to develop another algorithm that is capable of perceiving changes in real time that could mean an imminent attack.

6. References

[1] PwC, (2016) *Industry 4.0: Building the digital Enterprise, Global Industry 4.0 Survey,* p.11

[2] Secure-by-Design Industrial Internet of Things ARC's Sid Snitkin at June 2015 ARC Industry Forum

[3] Qué es data wrangling – Parte I (12/06/2019), last accessed: 11/06/2020, https://www.gantabi.com/2019/06/12/que-es-el-data-wrangling-parte-i/

[4] Amat, J. (Junio 2019) *Análisis de Componentes Principales (Principal Component Analysis, PCA) y t-SNE*, last accessed: 17/06/2020, https://www.cienciadedatos.net/documentos/35_principal_component_analysis

Índice de la memoria

Índice de figuras

Índice de tablas

Capítulo 1. Introducción

1.1. MOTIVACIÓN DEL PROYECTO

La Industria 4.0 está a día de hoy presente en prácticamente cualquier Sistema de Control Industrial (ICS por sus siglas en inglés). Esto es algo que no debería sorprender a nadie. No es un concepto o idea remota que pueda llegar a realizarse en un futuro y muchas empresas ya están aprovechando las numerosas oportunidades que esta nueva revolución industrial ofrece. Según un estudio de la PwC sobre la Industria 4.0, se espera que un 72% de las empresas alcancen altos niveles de integración de ésta y de digitalización hacia finales de este año [1]. Además, la actual situación desatada por el COVID-19 seguramente acelere esta transformación.

Se ha detectado un cambio de rumbo hacia la inteligencia operacional y prescriptiva en detrimento de técnicas analíticas meramente descriptivas. Esto se puede observar en varias tendencias que continuarán los próximos años en relación a la Industria 4.0 y al *"Industrial Internet of Things (IIoT)"*. Alguna de ellas son la recolección de grandes cantidades de datos para su posterior procesado y evaluación, la monitorización y gestión en remoto o niveles cada vez más altos de integración de datos y toma de decisión autónoma. Estas tendencias tendrán como consecuencia una mayor transparencia, un aumento de la eficiencia y una minimización del tiempo de inactividad de los equipos de fabricación. Todo esto con una repercusión directa sobre los niveles de producción. Por tanto, la adopción de herramientas de aprendizaje automático y de inteligencia artificial están cambiando la manera en la que empresas trabajan con sus sistemas industriales. No obstante, estas ventajas no están exentas de algún aspecto negativo, sobre todo en el ámbito de la seguridad.

Como conclusión se puede decir que el IIot y la Industria 4.0 presentan un nuevo paradigma y nuevas implicaciones, especialmente para la Ciberseguridad de los ICS. Que se analizaran a continuación.

1.2 IMPLICACIONES PARA LA CIBERSEGURIDAD DE LOS ICS

Ahora mismo, el *Desarrollo Tecnológico* de equipos inteligentes y autónomos, herramientas de cloud computing o conexiones flexibles y ubicuas están empezando a tener un papel cada vez en mayor en los ICS como se ha visto antes. Esto requiere de *Nuevas Arquitecturas en Sistemas de Control* que cubran diferentes sectores como el de la integración con servicios externos. Para ello estas arquitecturas tienen que ser dinámicas y distribuidas para tener un alto nivel de fiabilidad en los servicios externos. Todo esto conlleva n*uevos desafíos para la Ciberseguridad de ICS* al hecho que mayores niveles de conectividad han llevado a un aumento exponencial en la superficie de ataque. Consecuentemente, la probabilidad de ataque sobre estos equipos ha aumentado lo cual conlleva al riesgo de perder el control directo sobre la seguridad de un sistema [2].

1.3 *ESTADO DE LA CUESTIÓN*

Ahora mismo, el *Desarrollo Tecnológico* de equipos inteligentes y autónomos, herramientas de cloud computing o conexiones flexibles y ubicuas están empezando a tener un papel cada vez en mayor en los ICS como se ha visto antes. Esto requiere de *Nuevas Arquitecturas en Sistemas de Control* que cubran diferentes sectores como el de la integración con servicios externos. Para ello estas arquitecturas tienen que ser dinámicas y distribuidas para tener un alto nivel de fiabilidad en los servicios externos. Todo esto conlleva n*uevos desafíos para la Ciberseguridad de ICS* al hecho que mayores niveles de conectividad han llevado a un aumento exponencial en la superficie de ataque. Consecuentemente, la probabilidad de ataque sobre estos equipos ha aumentado lo cual conlleva al riesgo de perder el control directo sobre la seguridad de un sistema [3].

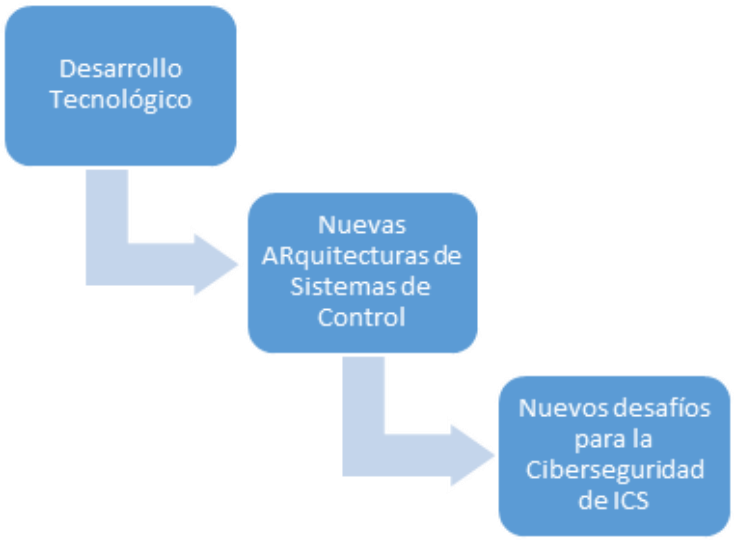


Figura 1: Implicaciones del IIoT para la ciberseguridad indsutrial

1.4 *METODOLOGÍA*

La primera parte del proyecto va a ser de investigación. Se van a consultar diferentes fuentes y trabajos para analizar en detalle qué técnicas se utilizan a día de hoy y cuáles pueden ser tendencias de cara al futuro.
La segunda parte consistirá en desarrollar un código para intentar identificar y prevenir un ciberataque. Para ello se utilizará un *dataset* ofrecido por el CIC (Canadian Institute for Cybersecurity). En primer lugar se va a hacer un análisis de los datos (exploratory data analysis) para entender qué datos y qué información contiene el *dataset*. A continuación se van a limpiar y preparar los datos para poder aplicar algoritmos. A esta
parte se la denomina *Data Wrangling*. Finalmente, se procederá a aplicar diferentes algoritmos de clasificación para desarrollar un detector de ciberataques y se verá cómo de eficientes son los diferentes algoritmos utilizados.

1.5 RECURSOS A EMPLEAR

La parte de desarrollo de este proyecto se va a desarrollar en el lenguaje de programación Python. Se he decidido utilizar éste frente a Matlab debido a que al ser código abierto permite que cualquier persona pueda utilizar las librerías y recursos sin necesidad de disponer de una licencia como ocurre con Matlab. Como el objetivo es que sea un programa adaptable a cualquier entorno, utilizar software de código abierto es más óptimo. Además, se van a utilizar datos del Instituto Canadiense de Ciberseguridad (CIC) y otros recursos disponibles en artículos publicados en internet.

Capítulo 2. Revisión de los protocolos de seguridad en entornos OT

2.1. INTRODUCCIÓN - DEFINICIÓN DE PROTOCOLO DE COMUNICACIÓN

Hablando de una manera más general, en informática y telecomunicación, un protocolo de comunicaciones es un conjunto de reglas y normas que permiten la comunicación entre dos o más sistemas para transmitir cualquier tipo de información entre ellos. Lo que define un protocolo son: la sintaxis, la semántica y la sincronización de la comunicación. Otro aspecto que engloban es la actuación ante errores y formas de recuperar un mensaje que se ha perdido o no llega entero. La forma de implementar un protocolo puede ser bien por hardware, por software o por una combinación de ambos.

2.1.1. PROPIEDADES TÍPICAS

Los protocolos de comunicación pueden variar mucho entre sí. Todo depende de lo sofisticado que sea éste y la robustez que se intente alcanzar. Propiedades típicas que especifican los protocolos de comunicación son, entre otras, las siguientes: [4]

- ✓ Detección de la conexión física subyacente (con cable o inalámbrica)
- ✓ Handshaking (Establecimiento de la conexión)
- ✓ Establecimiento de varias características de la conexión
- ✓ La manera de iniciar y finalizar una conexión
- ✓ Procedimiento en el formateo de un mensaje
- ✓ Detección y corrección de errores (Cómo actuar ante mensajes corruptos)
- ✓ Cómo detectar y recuperar una pérdida inesperada de la conexión
- ✓ Finalización de la conexión
- ✓ Formas de mejorar la seguridad y privacidad (cifrado, autenticación)
- ✓ Construcción de la red física
-

2.1.2. PROTOCOLOS BASADOS EN NIVELES DE ABSTRACCIÓN

Aunque en el campo de las redes informáticas, los protocolos se pueden dividir en varias categorías, uno de las clasificaciones más estudiadas y conocidas es el modelo OSI (Open System Interconnection). Este modelo fue creado en el año 1980 por la Organización Internacional de Normalización y su desarrollo comenzó en 1977 [5].

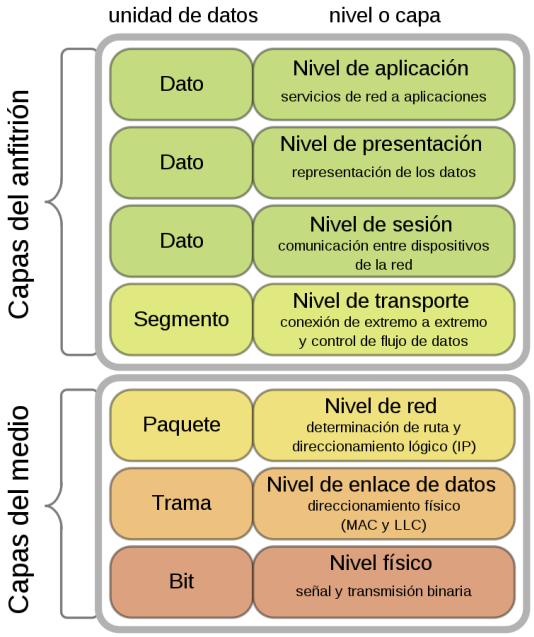


Figura 2: Capas del modelo OSI (1980) [5]

Este estándar tiene por objetivo interconectar distintos sistemas de manera que puedan intercambiar información entre ellos independientemente de los protocolos que utilizaran inicialmente.

Según este modelo, la comunicación entre varios equipos se puede estudiar dividiéndola en 7 niveles representados en la *Figura 2: Capas del modelo OSI (1980) [5]* y con algún ejemplo en la en la Tabla 2*:*

Capas	Niveles	Ejemplos	Categorías
Capa 7	Nivel de aplicación	SNMP, HHTP, FTP, SSH	APLICACIÓN
Capa 6	Nivel de presentación	ASN.1	
Capa 5	Nivel de sesión	NetBIOS, RPC, SSL	
Capa 4	Nivel de transporte	TCP, UDP, SPX	
Capa 3	Nivel de red	IP, IGMP, NetBEUI	TRANSPORTE DE DATOS
Capa 2	Nivel de enlace de datos	ARP, Ethernet, HDLC, CDP	
Capa 1	Nivel físico	Cable coaxial , microondas, fibra óptica	

Tabla 2: Capas modelo OSI con ejemplos

2.2. PROTOCOLOS DE COMUNICACIÓN EN LA INDUSTRIA

2.2.1. INTRODUCCIÓN

- El motivo por el cual aparece el concepto de Protocolo de comunicación industrial o ICP por sus siglas en inglés, es el hecho de que muchas empresas industriales presentan la existencia de islas que operan de manera automatizada y que son casi completamente impedientes entre sí. Sin embargo, muchas veces es necesario que haya comunicación entre las distintas partes que componen un proceso de fabricación e incluso desde el control de la fábrica que monitoriza todo el proceso industrial. Por ese motivo, aparecen unos protocolos de comunicación, similares a los ya existentes pero adaptados específicamente al entorno industrial. Y desde la irrupción de los microcontroladores en la industria, se ha simplificado en gran medida la integración de redes de comunicación [6].

2.2.2. IMPACTO DE LA INDUSTRIA 4.0

Una de las principales ventajas que trae consigo la Industria 4.0 es la capacidad de interconectar aún más todas las etapas que componen los procesos de producción. Como se explicó al principio de este trabajo, una de las consecuencias más directas es que al tener una gran parte de la fábrica conectada a una red, una intrusión en ella implicaría tener acceso y control sobre todas las máquinas y equipos disponibles. No solo se podría acceder a información sensible, sino que además se podrían manipular máquinas para llegar incluso a sabotearlas. Un ejemplo muy claro para esto último un ataque que se realizó sobre una central nuclear iraní. Mediante un gusano, conocido actualmente como Stuxnet, en 2010, se tomó el control de 1000 máquinas implicadas en el proceso de producción de materiales nucleares y les dio la instrucción de autodestruirse. Uno de los equipos infectados eran las centrifugadoras de la central, fundamentales para enriquecer uranio [7]. Por ello, es más importante que nunca revisar y actualizar los protocolos con los que se transmite la información dentro de estos entornos, mucho más si se trata de infraestructuras críticas.

2.2.3. REVISIÓN DE LOS PROTOCOLOS MÁS UTILIZADOS

A nivel industrial se utilizan muchos protocolos distintos. Sin embargo, todos ellos se pueden clasificar en su mayoría entre Ethernet Industrial y redes basadas en Fieldbus.

2.2.3.1. ETHERNET INDUSTRIAL

Estos protocolos aprovechan las ventajas funcionales y la seguridad que ofrece tanto Ethernet como TCP/IP a la hora de transferir información en los sistemas de control industriales. Por ello, intentan encajar la parte de datos del protocolo original parcialmente en una red Ethernet. Estos protocolos se basan en el estándar TCP/IP y utiliza el hardware y software de Ethernet para crear un protocolo que configure, acceda y controle los dispositivos de automatización industrial. Los protocolos más utilizados los últimos 5 años basados en Ethernet son: Profinet, Ethernet/IP, Ethercat, Modbus TCP y Powerlink entre otros [8].

2.2.3.2. *FIELDBUS*

En general, los buses de datos que posibilitan la integración de equipos para facilitar la medición y control de variables de proceso, se llaman buses de campo (Filedbus). Un bus de campo no deja de ser un sistema de transmisión de datos que simplifica en gran medida la instalación, operación y mantenimiento de las máquinas y equipos industriales utilizados. El objetivo de uno de estos buses es el de sustituir las conexiones punto a punto entre los elementos de campo y el equipo de control mediante un lazo de corriente. Normalmente son redes digitales, bidireccionales, multipunto y que sirven para conectar PLCs, sensores, actuadores y equipos de supervisión. Los protocolos más importantes y con mayor presencia en esta área son: Profibus, Modbus, Devicenet, CC-Link, Canopen o ASI entre otros [9].

2.2.4. *EVOLUCIÓN DE LAS CUOTAS DE MERCADO*

Durante los últimos años, el Ethernet Industrial y las redes inalámbricas han ido creciendo y aumentando su presencia en el mercado, mientras que Fieldbus va perdiendo fuerza. Lo que sí que está claro es que cada vez más y más equipamiento industrial está siendo conectado a redes industriales. Según HMS, en 2019 el número de nodos nuevos aumentó un 10%.

Además, se concluyó que en 2019 Ethernet había superado oficialmente por primera vez las redes tradicionales Fieldbus. Se espera que esta tendencia continúe. Esta transición se debe a la necesidad de mayores rendimientos y la necesidad de una mayor integración entre las instalaciones de la fábrica y los sistemas IT o aplicaciones IIoT.

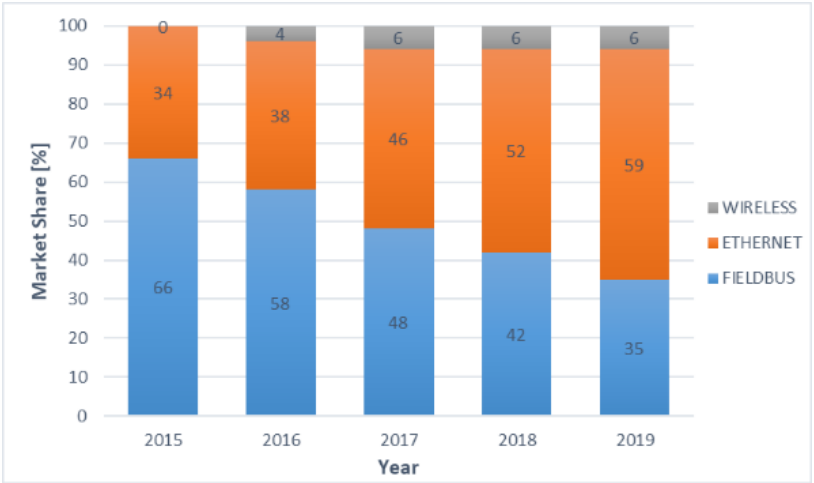


Figura 3: Cuotas de mercado 2015-2019 según HMS [11]

Como se ve en la Figura 3: *Cuotas de mercado 2015-2019 según HMS [11]* elaborada con datos recopilados por HMS durante los últimos años, tradicionalmente Fieldbus ha tenido una mayor cuota en el mercado que Ethernet. Esto es debido a que la tecnología Fieldbus es más robusta y más fiable, dando mayores niveles de seguridad. En 2015, casi el 70% de todas las redes industriales se basaban en Fieldbus y solo un 30% en Ethernet. Sin embargo, debido a la necesidad de conectar cada vez más dispositivos, impulsada por la llegada del IIoT y a su mayor velocidad, el Ethernet Industrial ha experimentado un gran crecimiento. Esta tendencia se aprecia

perfectamente en la gráfica también. En tan solo cinco años, Fieldbus ha vivido una caída del 47% de cuota, mientras que Ethernet ha representado en 2019 3 de cada 5 redes industriales.

Adicionalmente, durante los últimos cuatro años, ha entrado un nuevo competidor en el mercado. Se trata de las redes inalámbricas que se están consolidando en el mercado como alternativas a las redes tradicionales. Todo esto permite pensar que en los próximos años Fieldbus seguirá perdiendo cuota de mercado mientras que Ethernet y las inalámbricas seguirán ganando terreno.

Esta conclusión encaja con las nuevas arquitecturas de sistemas de control que ya se mencionaron con anterioridad. En concreto, hay otro aspecto más que diferencia Ethernet de Fieldbus y que es clave. Un dispositivo con Ethernet se puede utilizar como puente para conectar islas aisladas, pero también como puerta de acceso para conectar sistemas de automatización con sistemas de supervisión o sistemas en la nube. Y justo estos dos últimos aspectos son muy importantes a la hora de hablar de los cambios que están experimentando los sistemas de control industriales debido a la Industria 4.0.

Ahora que se han identificado los dos principales *players* dentro de los ICPs, se va a entrar más en detalle para ver qué protocolos en concreto son los más utilizados tanto en Ethernet como en Fieldbus. Así se puede encontrar un protocolo representante para cada tipo. La idea que subyace es que así, analizando las fortalezas y debilidades de cada uno de esos protocolos representantes, se puedan ver qué amenazas se presentan a una red industrial por utilizar estas tecnologías.

La Tabla 3: *Distribución de protocolos dentro de Ethernet Industrial* muestra el reparto de los diferentes protocolos basados en Ethernet durante los últimos cinco años:

	2015	**2016**	**2017**	**2018**	**2019**
PROFINET	24	21	24	23	24
ETHERNET/IP	24	24	24	29	25
ETHERCAT	15	16	15	13	12
MODBUS TCP	9	11	9	8	8
POWERLINK	6	8	9	8	7
OTHERS	24	21	20	19	24

Tabla 3: Distribución de protocolos dentro de Ethernet Industrial

Como conclusión, se puede ver que entre Profinet y Ethernet/IP representan aproximadamente el 50% de aquellos ICPs que se basan en Ethernet.

Si se hace un análisis similar con Profibus, se obtienen los siguientes resultados reflejados en la Tabla 4: *Distribución dentro de Fieldbus*:

	2015	2016	2017	2018	2019
PROFIBUS	27	29	29	28	28
MODBUS	10	12	12	14	14
DEVICENET	9	8	8	9	8
CC-LINK	9	10	12	14	17
CANOPEN	7	8	10	9	8
ASI	6	5	0	0	0
OTHERS	30	25	27	23	22

Tabla 4: Distribución dentro de Fieldbus

En este caso, los protocolos Profibus, Modbus y CCLink son los más representativos para este tipo de redes. No obstante, hay "otros" protocolos que representan casi una cuarta parte del total.

Hasta el momento, se ha estudiado cuáles son los protocolos de comunicación más habituales dentro de un Sistema de Control Industrial. Se ha descubierto que hay claramente una tendencia hacia Ethernet y redes inalámbricas mientras que la tecnología Fieldbus parece empezar a quedarse atrás ya que pierde de media entre un 4 y un 8% de cuota de mercado y nada hace pensar que esta tendencia se vaya a revertir. Además, esto hace pensar que estas tecnologías también van a ser más vulnerables en el futuro ya que van a tener menos soporte. Además, no van a permitir que un sistema industrial aproveche todas las ventajas que trae consigo la Industria 4.0 lo cual llevará a un sistema ineficiente y no competitivo.

Sin embargo, tomando información de un documento del INCIBE (Instituto Nacional de Ciberseguridad) [11], se descubre que Profinet, Ethercat y Ethernet/IP están ganando importancia, comparten algunas características con Profibus y Modbus que hacen pensar que ni Fieldbus ni Ethernet están llamados a ser el futuro de los ICPs. El problema es que ninguno de ellos lleva un sistema propio de encriptación. Si se le añade el hecho de que no hay sistema alguno de autenticación, se ve que estos protocolos por sí mismos no representan un entorno seguro para un ICSs porque no inlcuyen funciones de seguridad nativas. Claro que hay soluciones para esto: Por ejemplo, Modbus se puede proteger con SSL o encriptación VPN o técnicas de inspección de tráfico como Snort o IPS (Tofino). Y profinet tiene su propia Guía de Seguridad [11]. Sin embargo, la ausencia de autenticación y la falta de seguridad hace necesario que estos sistemas estén aislados de otros componentes de la red y esto significa que el sistema de control industrial no podrá adaptarse a los cambios y la transformación que está viviendo la industria.

2.2.5. RESULTADOS DEL ANÁLISIS

El análisis que se ha llevado a cabo ha permitido identificar dos puntos débiles de los protocolos que se han utilizado tradicionalmente a nivel industrial:

c) Son sistemas lentos y rígidos que dificultan en gran medida desarrollar las arquitecturas de los sistemas de control descritas al principio. Flexibilidad y dinamismo son dos características fundamentales que no se van a poder implantar. Esto aplica especialmente a Fieldbus.
d) La ausencia de seguridad impide la conexión con servidores externos. Eso no permite utilizar herramientas como el Cloud-computing que cada vez

son más usuales en este tipo de entornos. Esto aplica especialmente a Ethernet.

Dado que el resultado del análisis ha sido que los dos principales protocolos de comunicación utilizados a día de hoy no son compatibles realmente con las necesidades de la fábrica del futuro, se ha buscado un protocolo que sí cumpla las características necesarias y que no presente los puntos débiles que se han localizado en Fieldbus y Ethernet.

La conclusión a la que se ha llegado es a que el protocolo de comunicación OPC-UA (OPC Unified Architecture) está emergiendo como una alternativa a los protocolos habitualmente utilizados. A pesar de que a día de hoy no aparece en las estadísticas, su uso está empezando a ser cada vez más y más habitual. De acuerdo con datos del ADC Advisory Group, más de 40 millones de dispositivos a nivel mundial ya utilizan OPC que es el estándar que rige OPC-UA [13]. Además, de acuerdo la Sociedad Internacional de Automatización (IAS), más de 4200 proveedores de automatización producen y comercializan productos que funcionan únicamente con OPC-UA. Este estándar permite la comunicación industrial en la misma máquina, entre distintas máquinas y desde máquinas a otros sistemas unificando IT y OT. Tiene un amplio soporte entre diferentes plataformas gracias lo cual OPC-UA es escalable desde pequeños controladores embebidos hasta grandes infraestructuras en la nube. Adicionalmente, incluye tanto control de acceso como autenticación y encriptación que eran los puntos débiles de Ethernet y Fieldbus. Otra característica importante es su flexibilidad. Por ello, este protocolo se adaptaría muy bien a las nuevas arquitecturas de las que ya se ha hablado. OPC-UA ya se utiliza a día de hoy en diferentes industrias como la automovilística, alimentación y bebida, petróleo y gas, energía y otras. Finalmente se van a listar las ventajas, algunas mencionadas y otras no, que trae consigo este nuevo protocolo de comunicación [12]:

- ✓ Utiliza una plataforma compatible con cualquier sistema operativo (OS)
- ✓ Preparada para incorporar tecnologías del futuro y para comunicarse con sistemas antiguos
- ✓ Fácil de configurar y mantener
- ✓ Es una tecnología orientada a los servicios
- ✓ Tiene un mayor alcance de conectividad
- ✓ Tiene un alto rendimiento

Para concluir esta sección, se puede decir que a medida que la industria avanza hacia el Internet Industrial de las cosas (IIoT) y la Industria 4.0, OPC-UA está bien posicionado para reemplazar protocolos y estándares de comunicación tradicionales y para encarar los nuevos retos a nivel de ciberseguridad que van a aparecer.

Capítulo 3. Ciberataques en la industrial

El objetivo de este capítulo no es otro que hacer una breve recopilación de los ataques más típicos que existen a día de hoy. Como más adelante se va a entrar al detalle en un concreto se ha decidido dar una visión más genérica antes y explicar una posible forma de estructurar las diferentes fases de uno de estos ataques.

3.1. INTRODUCCIÓN

La transformación que está sufriendo la industria está conllevando a que los sistemas industriales tengan mayores posibilidades de ser atacados, tal y como se explicó al principio de este trabajo. Gracias a que todos los equipos están conectados entre sí para transmitir una gran cantidad de datos, la empresa se expone a tener una mayor superficie de ataque lo cual conlleva inevitablemente a un mayor riesgo. Uno de los mayores garantes de seguridad, conocido como "air-gap" desaparece en el momento en el que se conectan absolutamente todas las máquinas y sistemas a una misma red. Esto ha conllevado a un incremento de ciberataques a empresas e industrias. Hay varios datos que alertan sobre la creciente necesidad de invertir cada vez más en proteger cualquier empresa industrial. Un dato curioso es el hecho de que solo tres industrias concentraron en 2016 el 95% de toda la información filtrada a través de ciberataques. Estas industrias fueron gobierno, venta minorista y tecnológicas. Esto no se debe a que estas industrias sean menos competentes a la hora de protegerse, es debido al hecho de que contienen un gran volumen de información de carácter personal. Además, tal y como se refleja en un informe de la asesora Juniperre [14], se estima que el coste por toda la información filtrada por ciberataques en 2020 ascienda a 2 billones de dólares. Otro dato curioso y relevante por lo que se va a analizar más adelante en este trabajo es el hecho de que ataques DDoS a gran escala han aumentado su tamaño en un 500%, según un reporte de Nexusguard [15] en el que se analiza el impacto de este tipo de ataques.

3.2. CIBERATAQUES MÁS COMUNES

A continuación, se van a listar algunos de los ataques más comunes acompañados de una breve descripción de cada uno de ellos con información obtenida del blog de la empresa de ciberseguridad Newtrix [17].

- **Denial of Service (DoS) and Distributed Denial of Service (DDoS)**
    - Este tipo de ataques sobrecargan los recursos de un sistema para que no pueda responder las solicitudes de servicio por parte de un usuario. Un ataque DDoS también es un ataque a los recursos del sistema, la diferencia es que se incida desde un gran número de máquinas anfitrionas que también están infectadas por software malicioso controlado por el atacante.
- A diferencia de ataque cuyo objetivo es conseguir información o acceder a sistemas, la denegación de servicio no produce directamente ningún beneficio para el atacante más allá del hecho que la víctima no puede utilizar sus recursos. Este tipo de ataques tienen sentido cuando la víctima, por ejemplo, pertenece a un competidor comercial.

En este caso el beneficio es claro, sabotear a la competencia para reforzar su posición en el mercado. Otro posible motivo de este ataque puede ser el desconectar un sistema o inutilizarlo con la intención de lanzar más adelante otro ataque diferente. Hay diferentes tipos de ataques Dos y DDoS. Sin embargo, no se va a entrar más en detalle ahora ya que más adelante se va a analizar con lupa este tipo de ataques.

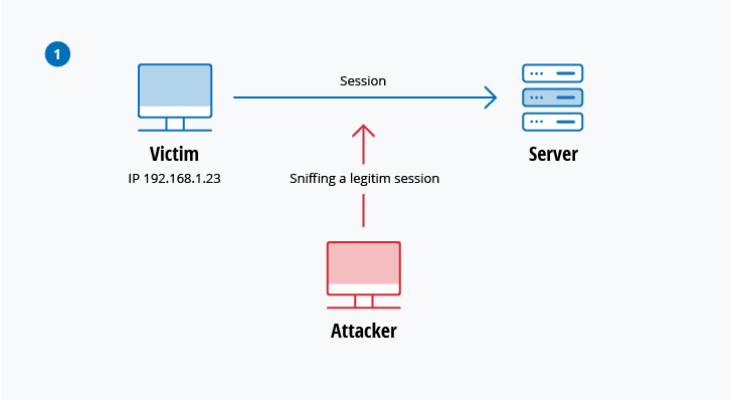
-
- **Man-in-the-middle attack (MitM)**
    - Un ataque de Man in the Middle ocurre cuando un atacante es capaz de infiltrarse entre las comunicaciones de un cliente y un servidor. Al estar el atacante simplemente registrando la información que se transmite de un punto a otro sin alterarla, es muy complicado de detectar este tipo de ataques. Un ejemplo de este tipo de ataques sería el Secuestro de sesión:
- En este tipo de ataque MitM, un atacante secuestra una sesión entre un cliente de confianza y un servidor de red. La máquina del atacante sustituye su dirección IP por la del cliente de confianza mientras el servidor continúa la sesión, pensando que está comunicándose con el cliente. IP Spoofing y Repetición otros dos ejemplos de ataques de hombre en el medio en los que no se va a entrar más en detalle ahora.
- 

Figura 4: Primera etapa de un MitM attack

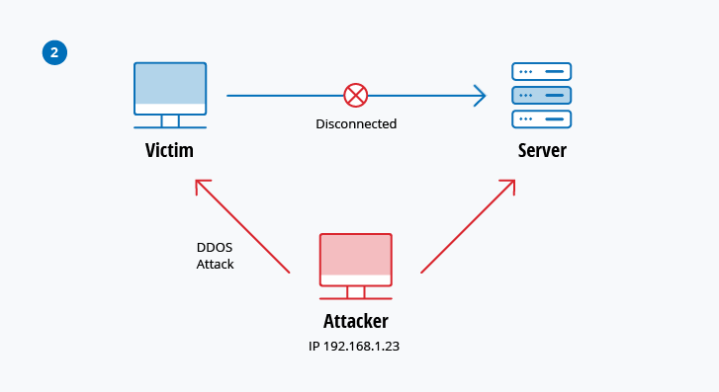
- 

Figura 5: Segunda etapa de un MitM attack

- Tanto la Figura 4como la Figura 5 representan cómo se lleva a cabo este tipo de ataques. En primer lugar, un cliente se conecta a un servidor. El atacante en primera instancia observa la situación hasta que toma el control sobre el cliente. Después

desconecta al cliente del servidor y reemplaza su dirección IP por la del cliente real. De esta manera el cliente ha perdido al servidor y ha sido reemplazado por el atacante sin que el servidor se haya dado cuenta de esto.

- **Phishing attacks**
  - Los ataques de phishing son muy habituales y no solo afectan a nivel industrial sino muchas veces directamente al cliente o consumidor de un servidor de correo electrónico. El ataque consiste en enviar correos electrónicos que imitan fuentes oficiales y siempre confiables. El objetivo es que la víctima entre en algún enlace que contenga otro ataque o directamente proporcione información personal sin darse cuenta de que no es un correo oficial. También puede ser que estos correos incluyan un archivo adjunto cargado con malware. De la misma manera puede contener un enlace a una página que intente introducir malware en el equipo.
- **Drive-by attack**
  - El principal objetivo de este tipo de ataques es el de propagar malware. Los atacantes intentan buscar páginas poco seguras e introducen scripts maliciosos en el código HTTP o PHP de las páginas. A través de ese script, se consigue instalar malware directamente en el ordenador de la víctima con el mero hecho de visitar la página en cuestión. Lo que distingue estos ataques de otros es que no es necesario que el usuario abra o descargue ningún archivo. Simplemente visitando páginas que son conocidas para el usuario pero que no son seguras, el atacante es capaz de introducir el malware en el ordenador.
- **Password attack**
  - A día de hoy, las contraseñas siguen siendo el mecanismo más habitual para autenticar usuarios en un sistema. Es verdad que están surgiendo alternativas biométricas y medidas adicionales como códigos enviados por SMS al teléfono móvil. Pero es verdad que obtener una contraseña sigue siendo una forma muy efectiva de acceder a información de un usuario. La forma de obtener la contraseña consiste en observar y vigilar qué páginas visita la víctima y cual de ella pueden no tener las claves encriptadas. La forma de obtener finalmente la contraseña puede ser bien por fuerza bruta, es decir, probando de manera aleatoria o con “diccionario” de contraseñas utilizadas habitualmente.
- **SQL injection attack**
  - Este ataque se ha convertido en un problema habitual en páginas web basadas en bases de datos. El ataque ocurre cuando un atacante ejecuta una petición SQL a la base de datos a través de la información de un cliente. Una vez accedida a la base de datos se ejecutan comandos que hacen que la base de datos acabe entregando información de todos los clientes que es confidencial.
- **Eavesdropping attack**

- Este tipo de ataques ocurren cuando se intercepta tráfico de red. Al espiar, el atacante puede obtener contraseñas, números de tarjetas de crédito y otra información confidencial que se puede llegar a distribuir por la red. Estos ataques pueden realizarse de manera activa o pasiva:
  - **Eavesdropping activo**: El hacker accede a la información haciéndose pasar por una unidad amiga y enviando consultas a los transmisores. Esto se llama sondeo, escaneo o manipulación.
  - **Eavesdropping pasivo**: El hacker detecta la información escuchando la transmisión de mensajes en la red sin involucrarse activamente.
  -

- **Zero-day vulnerability**
- Estos ataques explotan vulnerabilidades que no han sido anunciadas de manera públicas aún. Es decir, ataques que no se conocen. Al ser desconocida la vulnerabilidad, estos ataques suelen ser muy difíciles de detectar e identificar porque no se tienen antecedentes. Además, se suma el problema de que, aunque el ataque haya sido identificado, hará falta tiempo para desarrollar un parche para dicho problema ya que este aún no existe.

3.3. FASES DE UN CIBERATAQUE

En este apartado se va a describir tomando como base una propuesta del INCIBE, las diferentes fases en las que se puede dividir un ciberataque. Esta propuesta se hace ya que a pesar de que muchas empresas están invirtiendo mucho dinero en ciberseguridad, muchas veces no son capaces de mantener sus sistemas del todo seguros. La clave para detectar, interrumpir y recuperarse de un ciberataque está en comprender y entender el ciclo de vida de éste. Este ciclo de vida es conocido como Cyber Kill Chain. Este concepto es original de los militares que lo empleaban a la hora de definir los pasos que empleaba el enemigo al atacar un objetivo.

Finalmente, antes de entrar en detalle a ver qué siete fases componen un ataque, hay que tener presente que un ataque es un proceso dirigido con una intención bien definida: conseguir unos efectos que pueden variar de ataque a ataque, sobre el objetivo. Esta intención puede ser inutilizar un sistema, robar datos confidenciales o bloquear una máquina para forzar a la víctima a pagar un rescate.

Las fases son las que se detallan en la siguiente figura:

Figura 6: Fases de un ciberataque

3.3.1. RECONOCIMIENTO

En esta primera fase, el atacante recopila información sobre su target. Para ello se suele basar en información y detalles que la organización publica en abierto y busca información sobre qué tecnologías utiliza, así como datos en redes sociales o a través de interacciones por correo electrónico.

Una vez obtenida esta información, el atacante valora qué métodos de intrusión podrían ser efectivos en este caso y qué probabilidad de éxito tendría con cada uno de ellos.

3.3.2. PREPARACIÓN

A continuación, se prepara el ataque de forma específica. Una parte de este proceso podría ser preparar documentos falsos que se van a incluir en un correo electrónico para suplantar la identidad de una persona de confianza.

3.3.3. DISTRIBUCIÓN

En esta fase es cuando se transmite el ataque a través del canal especificado. Por ejemplo, mediante la propagación de un documento infectado.

3.3.4. EXPLOTACIÓN

En esta fase se lleva a cabo la parte más dañina del ataque. Es aquí cuando se accede al sistema que se desea controlar y de este paso depende en gran medida el éxito o fracaso del ataque. Aquí se detona el ataque de manera que se compromete al equipo infectado y a la red a la que pertenezca dicho equipo. Esto suele ocurrir explotando una vulnerabilidad del sistema que muchas veces ya es conocida.

3.3.5. INSTALACIÓN

Fase en la que el atacante instala el malware en la víctima. Hay ocasiones en las que esta fase no implica necesariamente la instalación de ningún software. Por ejemplo hay un ataque conocido como el fraude del CEO que consiste en pedir información o que se lleve a cabo una transacción haciéndose pasar por el CEO de la empresa que está siendo atacada.

3.3.6. COMANDO Y CONTROL

En esa sexta fase, el atacante ya se ha hecho con el control del sistema de la víctima. Ahora ya podrá lanzar sus acciones maliciosas muchas veces sin ser siquiera detectado. Puede sustraer credenciales, tomar información de equipos, sustraer información confidencial, instalar otros programas para acometer otros ataques, conocer más en detalles cómo opera la red de la víctima, etc.

3.3.7. ACCIONES SOBRE LOS OBJETIVOS

Esta ya es la fase final del ataque. En ella, el atacante se hace definitivamente con la información e intenta expandir su acción hacia otros objetivos. Aquí se puede ver que la cadena descrita no es lineal sino cíclica ya que el atacante va a repetir el proceso una y otra vez mientras no sea detectado.

Capítulo 4. Conjunto de datos utilizado - CICDDoS2019

4.1. PROCEDENCIA

El dataset utilizado, se ha obtenido del CIC, Canadian Institute for Cybersecurity. Este instituto depende de Universidad de New Brunswick en Fredericton [17] y es un centro multidisciplinario de entrenamiento en el cual se investiga y desarrolla en el ámbito de ciencias sociales, empresas, computer science, ingeniería y ciencia [18].

4.2. DENIAL OF SERVICE (DOS) Y DISTRIBUTED DENIAL OF SERVICE (DDOS)

4.2.1. DEFINICIÓN

En ciberseguridad, según un artículo publicado en la Revista de la Universidad Autónoma de México [19], un ataque de denegación de servicio (DoS por sus siglas en inglés) busca como objetivo la saturación o explotación de alguna vulnerabilidad en un servicio que opere dentro de una infraestructura, dando como resultado la saturación, caída o fallo del mismo.

El ataque de denegación de servicio distribuido, DDoS, es un ataque basado en DoS el cual consiste en generar un enorme flujo de información desde varios equipos hacia un mismo punto de destino. De esta manera se satura el equipo de destino y se impide que los usuarios hagan uso de ese servicio. Aunque el ataque se pueda realizar de muchas formas, lo habitual es que utilicen los protocolos TCP/IP.

4.2.2. TAXONOMÍA

Según el CIC, ya existen taxonomías para catalogar este tipo de ataques, no obstante, su alcance es limitado. Por ello, proponen una nueva taxonomía a base de analizar nuevos ataques que pueden ser llevados a cabo utilizando protocolos TCP/UDP en la capa de aplicación. Esta forma de clasificar los ataques DDoS consiste en separarlos entre Reflection-based DDoS y Exploitation-based DDoS.

4.2.2.1. Ataques DDoS basados en reflexión (Reflection-based attacks)

Engloban el tipo de ataque en los que la identidad del atacante permanece oculta. Para ello, el atacante envía los paquetes a servidores reflectores con la

dirección IP de origen configurada de manera que señale la dirección IP de la víctima para sobrecargarla con paquetes de respuesta. Estos ataques se realizan a través de protocolos de la capa de aplicación o application layer utilizando la capa de transporte o transport layer. Por ejemplo: Transmission Control Protocol (TCP), User Datagram Protocol (UDP) o una combinación de ambos. Como se muestra en la Figura 7, esta categoría incluye los siguientes ataques basados en TCP: MSSQL, SSDP. Por el otro lado, los que se basan en UDP son: CharGen, NTP y TFTP. Hay algunos ataques que pueden llevarse a cabo tanto mediante TCP como UDP como DNS, LDAP, NETBIOS y SNMP.

4.2.2.2. Ataques DDoS basados en explotación (Exploitation-based attacks)

Al igual que antes, en estos ataques el atacante oculta su identidad enviando los paquetes a servidores. Sin embargo, estos están configurados con la dirección IP de origen establecida en la dirección IP de la víctima para abrumarla con paquetes de respuesta. De nuevo, estos ataques pueden llevarse a cabo mediante los protocolos TCP y UDP. Los ataques basados en TCP incluyen inundación SYN y los ataques basado en UDP incluyen inundación UDP y UDP-Lag. El ataque de inundación UDP se inicia en el host remoto enviando una gran cantidad de paquetes UDP.

Estos paquetes UDP se mandan a puertos seleccionados aleatoriamente en la máquina de destino a una velocidad muy elevada. Esto provoca que el ancho de banda disponible de la red se agote, el sistema se bloquee y el rendimiento se degrade. Además, la inundación SYN consume también recursos del servidor al usar el protocolo de enlace de tres vías TCP. Este ataque se inicia enviando repetidamente paquetes SYN a la máquina de destino hasta que el servidor falla o funciona mal. El ataque UDP-Lag, interrumpe la conexión entre el cliente y el servidor. Este ataque pretende ralentizar o interrumpir el movimiento de otros jugadores para superarlos.

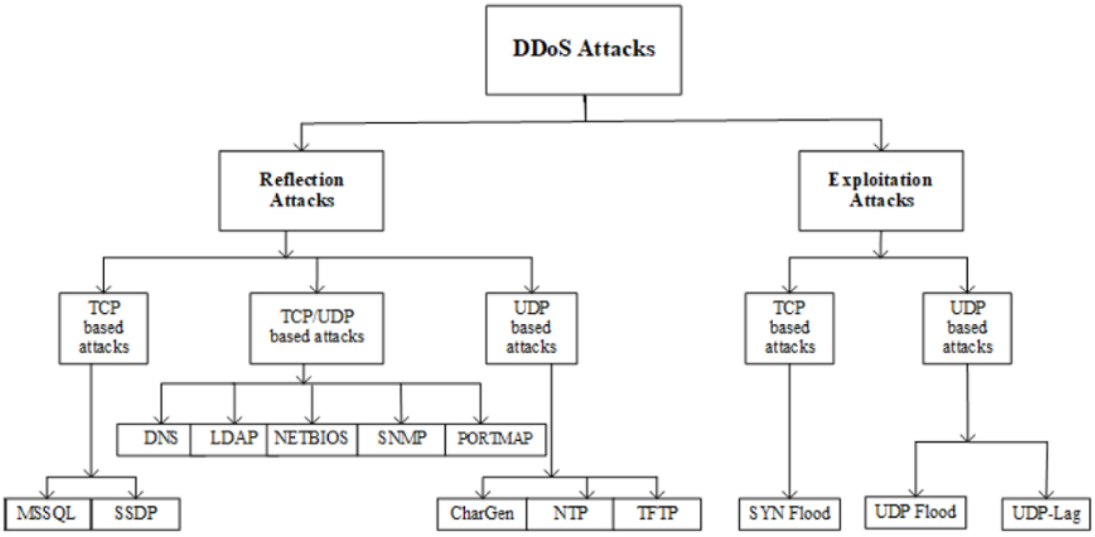


Figura 7: Taxonomía de un ataque DDoS según el CIC

A continuación, se va a detallar uno de los ataques mencionados.

TCP SYN Flood Exploitation Attack:

En este ataque, lo que el atacante hace es explotar el uso del espacio del búfer durante un protocolo de enlace de inicialización de sesión del Protocolo de Control de Transmisión (TCP). De esta manera, el atacante inunda la cola en proceso del sistema de destino con solicitudes de conexión, pero sin responder cuando el sistema de destino (víctima) responde a esas solicitudes. Así, el sistema de destino agota el tiempo de espera mientras la espera la respuesta del dispositivo del atacante, lo que hace que el sistema se bloquee y quede inutilizable. Posibles contramedidas para este tipo de ataques pueden ser colocar los servidores de red detrás de un firewall configurado para detener los paquetes SYN entrantes. Otra opción sería aumentar el tamaño de la cola de conexión y disminuir el tiempo de espera en las conexiones que hay abiertas en ese momento.

4.3. SOBRE EL CONJUNTO DE DATOS

El conjunto de datos CICDDoS2019 contiene ataques DDoS benignos y los ataques más actualizados de DDoS que asemeja a datos reales (PCAPs). También incluyen los resultados de análisis de red utilizando el programa CICFlowMeter-V3 que etiqueta flujos de datos basándose en marcas de tiempo, IP de origen y destino, protocolos y ataques. Estos datos están registrados en archivos CSV. El objetivo a la hora de elaborar este dataset ha sido lograr generar flujos de datos realistas que se asemejen a circunstancias reales. Se ha utilizado un sistema específico (B-Profile system) para perfilar el comportamiento humano y generar trafico benigno natural. Para este conjunto de datos se simulado el comportamiento abstracto de 25 usuarios

basado en HTTP, HTTPS, FTP, SSH y protocolos de email. En la siguiente tabla se representan los diferentes equipos utilizados para simula el tráfico de datos:

Máquina	Sistema Operativo	IPs
Server	Ubuntu 16.04 (Web Server)	192.168.50.1 (first day) 192.168.50.4 (second day)
Firewall	Fortinet	205.174.165.81
PCs (first day)	Win 7	192.168.50.8
	Win Vista	192.168.50.5
	Win 8.1	192.168.50.6
	Win 10	192.168.50.7
PCs (second day)	Win 7	192.168.50.9
	Win Vista	192.168.50.6
	Win 8.1	192.168.50.7
	Win 10	192.168.50.8

Tabla 5: Equipos utilizados para la obtención datos

4.4. ORGANIZACIÓN DEL CONJUNTO DE DATOS

El dataset está organizado por días. Para cada día se han guardado datos crudos incluyendo el tráfico de red (Picaps) y registros de eventos para cada **máquina**. A la hora de extraer características, el CICFlowMeter-V3 ha extraído más de 80 características de tráfico y las ha guardado como un archivo CSV para cada máquina.

A continuación, se ha añadido una lista con algunas de las características extraídas por el analizador de red sin detallar a qué hace referencia cada una de ellas. La lista completa con nombres y explicaciones se ha añadido en el Anexo 1.

0	Unnamed: 0	34	Bwd IAT Std
1	Flow ID	41	Fwd Header Length
2	Source IP	43	Fwd Packets/s
3	Source Port	47	Packet Length Mean
4	Destination IP	48	Packet Length Std
5	Destination Port	58	Down/Up Ratio
6	Protocol	63	Fwd Avg Bytes/Bulk
7	Timestamp datetime64ns]	64	Fwd Avg Packets/Bulk
8	Flow Duration	65	Fwd Avg Bulk Rate
9	Total Fwd Packets	66	Bwd Avg Bytes/Bulk
10	Total Backward Pakets	67	Bwd Avg Packets/Bulk
11	Total Length of Fwd Packets	68	Bwd Avg Bulk Rate
12	Total Legth of Bwd Packets	69	Subflow Fwd Packets
15	Fwd Packet Length Mean	70	Subflow Fwd Bytes
19	Bwd Packet Length Mean	71	Subflow Bwd Packets
20	Bwd Packet Length Std	72	Subflow Bwd Bytes
21	Flow Bytes/s	75	act_data_pkt_fwd
22	Flow Packets/s	76	min_seg_size_forward
23	Flow IAT Mean	77	Active Mean
27	Fwd IAT Total	78	Active Std
29	Fwd IAT Std32 Bwd IAT Total	81	Idle Mean
33	Bwd IAT Mean	87	Label

Tabla 6: Lista con algunas etiquetas extraídas del analizador de red

Como se verá más adelante, muchas de estas variables no han aportado ninguna información por lo que se han eliminado.

Capítulo 5. Código desarrollado

5.1. GENERADOR DE MUESTRAS

Como se mencionó previamente, el dataset contenía 21GB de información y un total de 30 millones de vectores. Lo primero que hubo que hacer fue tomar muestras aleatorias de estos archivos. La función principal lee los archivos disponibles uno a uno y toma un numero dado de vectores creando archivos nuevos más pequeños que se llaman "simple_tipo_de_ataque.csv".

La función que realizaba esto era la siguiente:

```
# Funcion 4: Esta función crea para cada ataque una muestra con
unos 40000 ataques
def f_create_samples():

    print("Creating sample files ... ... ...")
    # Cargamos archivo en dataframe
    for file in filenames:
        train_file = os.path.join(dataset_folder, file)
        sample_file  =  os.path.join(samples_folder,  'sample_'  +
file)
        if os.path.isfile(sample_file):
            a = True
        else:
            df = pd.read_csv(train_file, usecols = column_names)
            # Eliminamos filas benignas
            dfa = df[df[' Label'] != 'BENIGN']

            start = 0;
            stop = dfa.shape[0]-1
            n_samples = 40000
            ind1 = np.random.randint(start, stop, n_samples)
            # = ind_unique(ind1)
            ind = np.asarray(ind1)
            df_samples = dfa.reindex(ind)
            df_samples.to_csv(sample_file)

    print('Sample files created successfully!')
```

El bucle *for* se encarga de leer cada archivo de la ruta especificada y más delante se comprueba si ya existe un archivo de muestras para el archivo que se va a leer. Esto se ha hecho así, dado que este proceso es el más largo y puede ser que se vea interrumpido. Para no empezar siempre desde cero, se comprueba siempre primero si el archivo que se va a muestrear ya contiene.

La aleatoriedad de las muestras tomadas se garantiza al definir el vector de índices así: `ind1 = np.random.randint(start, stop, n_samples.` La función *randint* de la librería numpy implementa un generador de números pseudo-aleatorios generando números uniformemente distribuidos entre [0.0 y 1.0) y mediante el Mersenne Twister [20], Python genera los enteros entre los límites marcados por las variables start y stop. En este caso, start vale 0 ya que en Python el primer vector lleva el índice 0 y llega hasta la longitud del archivo menos uno que es el índice del último vector [21]. Al hacerlo así, se garantiza que independientemente del archivo que se esté tratando, se van a poder coger un número *n_samples* de entre cualquiera de los vectores de ese archivo.

Una vez creados los archivos de muestras, se llama otra función que simplemente coge estos archivos creados y los junta en uno. Intentar hacer esto de un plumazo puede conllevar a que la memoria del ordenador sature ya que el proceso consume mucha RAM. Se muestra a continuación una parte de esta función:

```
def f_create_attack_set():


    print('Creating attack file ... ... ...')

    for file in filenames:
        sample_file      =      os.path.join(samples_folder,
'sample_' + file)
        df_temp   =   pd.read_csv(sample_file,   usecols   =
column_names)

        if (i == 0):
            df_total = df_temp
        else:
            df_total = df_total.append(df_temp)
        i += 1
        df_temp = None
        del df_temp

    for file in filenames:
        sample_file      =      os.path.join(samples_folder,
'sample_' + file)
        os.remove(sample_file)

   print('Attack file created successfully!')
```

Este mismo proceso se repite más adelante pero en vez de coger ataques, se cogen todos los vectores benignos. Como el número de vectores benignos es mucho menor

que el de ataques, en el caso de benignos se han extraído todos los vectores. Las funciones que hacen esto se han definido como:

```
def f_create_benign():
def f_create_benign_set():
```

La idea de crear archivos de malignos y benignos por separado es poder analizar ambos por separado, como para entrenar los modelos hace falta estar juntos, hay una última función que se encarga de juntar ambos archivos para tener un único set reducido con muestras de todos los tipos de ataque y vectores benignos:

```
def juntar_benign_attack():
```

De esta manera se han logrado dos cosas:

- ✓ Conseguir un único set representativo con un número de muestras tomadas de manera aleatoria y de mucho menor tamaño que el original. Esto es importante ya que se reduce considerablemente el tiempo de ejecución a la hora de realizar pruebas y entrenar los modelos. Una vez probado todo, siempre se pueden crear archivos con más muestras para alcanzar una mayor calidad y precisión, pero hasta entonces es imprescindible tener un dataset manejable y ligero.
- ✓ Se ha lidiado con el problema de que los datos están desbalanceados, es decir, que había un número mucho mayor de un grupo que de otro. En este caso, el 99,89% de los vectores eran ataques. El problema es que si esto no se ajusta, cualquier modelo se limitará a clasificar todos los vectores como ataques y ninguno como benigno y alcanzará 99.9% de aciertos. Por eso es necesario compensar la proporción. De esta manera se ha conseguido que la clase minoritaria pase de representar el 0.1% a ser el 15%.

5.2. DATA WRANGLING

5.2.1. INTRODUCCIÓN

La segunda parte consistía en organizar y preparar los datos. Esta práctica, conocida como Data Wrangling puede llegar a ocupar el 80% del tiempo, por lo que es fundamental. El Data Wrangling se refiere al proceso de pasar los datos de un formato crudo a otro formato que permite el procesado de éstos con diferentes funciones o algoritmos. Al final, lo que permite esto es tomar mejores decisiones en menos tiempo y utilizando menos recursos. El hecho de que la información sea captada por un analizador de red conlleva a que se guarden o registren muchos datos que no son relevantes para algunos propósitos o que se guarden datos erróneos.

5.2.2. CÓDIGO DESARROLLADO

En el set de datos analizado se han encontrado principalmente 3 problemas que impedían que los datos fueran aptos para introducirlos en los modelos de ML:

- ✓ Había columnas con valores que se habían registrado como infinito y que, por ejemplo, no permiten calcular una simple media.
- ✓ Otras columnas estaban vacías lo cual hace que el equipo esté destinando más recursos de los necesarios para nada
- ✓ Algunas columnas contenían formatos como fechas que tampoco son aptas para las funciones que se han utilizado

Para solventar estos problemas se ha creado una función llamada:

```
def f_clear_df(df):
```

Se han eliminado las siguientes columnas por no ser de utilidad:

```
borrar = ['Unnamed: 0', 'Flow ID', ' Source IP', ' Source
Port',' Destination IP', ' Destination Port', ' Protocol',
' Timestamp', 'SimillarHTTP']
```

A continuación, se han eliminado columnas vacías o con solo un valor:

```
# Eliminamos columnas vacías o columnas sin más de un valor
distinto
    for col in df.columns:
        a = df[col].value_counts()
        b = len(a)
        if b<2:
            df.drop(col, axis=1, inplace = True)
```

Lo que se ha hecho es iterar todas las columnas una a una. En cada caso se ha guardado en la variable 'a' los diferentes valores que hay en una columna. En b se ha sumado el número de valores distintos que hay en a. Si este número es inferior a 2, significa que esa columna no sirve por lo que en la siguiente línea se elimina.

De esta manera se ha reducido el número de columnas de 86 a unas 60.

Finalmente, se han eliminado en cada columna aquellas filas que contuvieran en algún momento el valor infinito:

```
    # Quitamos valores que valgan infinito
    for col in df.columns:
            with    pd.option_context('mode.use_inf_as_na',
True):
                df = df.dropna(subset=[col], how='all')
    return df
```

5.2.3. CONCLUSIONES

Una vez realizada esta fase, el dataset tiene un tamaño adecuado para ser probado y además no va a haber problemas de incompatibilidad con ningún valor. Adicionalmente, se ha vuelto a reducir el tamaño del set simplemente eliminando columnas vacías lo cual podría permitir repetir el proceso tomando un número mayor de muestras para llegar al tamaño de archivo que se tenía inicialmente.

5.3.ANÁLISIS DE COMPONENTES PRINCIPALES (PCA)

5.3.1. INTRODUCCIÓN

El Análisis de Componentes Principales o Principal Component Analysis (PCA) es una técnica de aprendizaje no supervisado, que suele aplicarse como parte de lo que se llama Análisis exploratorio de los datos. A diferencia de otros algoritmos de aprendizaje supervisado, el objetivo no es predecir una variable y a partir de un grupo de variables $(X = X_1, X_2, \dots, X_p)$. Solo interesa la información que subyace a esas variables, como por ejemplo, si existen subgrupos entre las observaciones.

Una de las aplicaciones de PCA es la reducción de dimensionalidad, perdiendo la menor cantidad de información posible. Para ello, se parte de la base de que existe correlación entre algunas de las variables de manera que unas puedan ser expresadas como combinación lineal de otras. Estas nuevas variables que se obtienen se denominan componentes principales y el objetivo es explicar la mayor parte de la variabilidad posible a través de ellas. Estas nuevas variables, aparte de ser combinación lineal de las originales, son independientes entre sí y no están correladas [22].

En definitiva, el análisis de componentes principales permite resumir o simplificar un set con un número menos de variables representativas que de manera colectiva explican gran parte de la variabilidad del set original.

5.3.2. EXPLICACIÓN DEL ALGORITMO DE PCA

Para obtener las variables nuevas (componentes principales), se calcula de cada variable la matriz de covarianzas una vez se le resta la media. Después se calculan los autovectores y los autovalores de la matriz de covarianza. Una vez obtenidos, se ordenan los autovectores según el valor de su autovalor asociado (de mayor a menor).

El resultado son unas nuevas variables $Z_1, Z_2, \dots Z_m (m \leq p)$ a partir de las originales. Cada componente principal se puede expresar de la siguiente manera:

$$Z_m = \sum_{j=1}^{p} \varphi_{\mathrm{jm}} * X_j$$

$$\sum_{j=1}^{p} \varphi_{j1}^2 = 1$$

Los coeficientes φ son los "scores", es decir, el peso que tiene cada variable original dentro de esa componente principal.

A pesar de que se obtienen tantas componentes principales como variables originales había, no interesan todas. Para ello, se calcula la varianza explicada por cada componente principal (dada por su autovalor) y se cogen tantas hasta que se llegue un valor de varianza explicada acumulada suficientemente alto. Esto se puede observar en el *scree plot* que se mostrará más adelante.

Se tiene en cuenta que:

La varianza total que un set de datos con n observaciones presenta se calcula como:

$$\sum_{j=1}^{p} Var(X_j) = \sum_{j=1}^{p} \frac{1}{n} \sum_{i=1}^{n} x_{ij}^2$$

(Se asume que las variables han sido centradas y que tienen media igual a cero)

La varianza explicada por la componente principal m se obtiene como:

$$\frac{1}{n} \sum_{i=1}^{n} z_{im}^2 = \frac{1}{n} \sum_{i=1}^{n} \left(\sum_{j=1}^{p} \varphi_{jm} * x_j \right)^2$$

Este valor coincide con el autovalor asociado al autovector obtenido.

Finalmente, la proporción de varianza explicada (PVE) por la componente m:

$$\frac{\sum_{i=1}^{n}\sum_{j=1}^{p}\varphi_{jm} * x_j{}^2}{\sum_{j=1}^{p}\sum_{i=1}^{n}x_{ij}^2}$$

Los PVEs, por tanto, son siempre positivos la suma de todos ellos da 1. [23]

5.3.3. CÓDIGO DESARROLLADO

Antes de aplicar la transformación a los datos, como se indica en el apartado anterior, se han normalizado y escalado los datos con la función StandardScaler:

```
# Noramlizamos los datos
    scaler = StandardScaler()
    # Calculamos la media para poder hacer la transformacion
    scaler.fit(dfb)
    # Escalamos y normalizamos datos
    X_scaled = scaler.transform(dfb)
```

Normalizarlos es necesario para obtener valores con media 0 y varianza igual a 1 (condición necesaria para poder aplicar PCA)

Además, se escalan los datos para que no afecte al algoritmo el hecho de que algunas variables tomen valores del orden de decenas y otros de cientos de miles.

A continuación, se ha instanciado el objeto PCA y se ha indicado cuántas componentes principales se desea obtener, siempre y cuando no se supere el número de columnas que hay. Se han tomado 30 variables ya que, a priori no se sabe cuántas van a hacer falta. De manera alternativa se puede instanciar el objeto PCA, pero en vez de indicar el número de variables se puede especificar el mínimo de varianza que se desee y en base a ese valor se obtiene el número de variables mínimas necesarias.

5.3.4. RESULTADOS OBTENIDOS

Una vez obtenidas los scores de las diferentes componentes principales se ha dibujado el scree plot para decidir cuántas componentes se van a utilizar, dentro del código, la

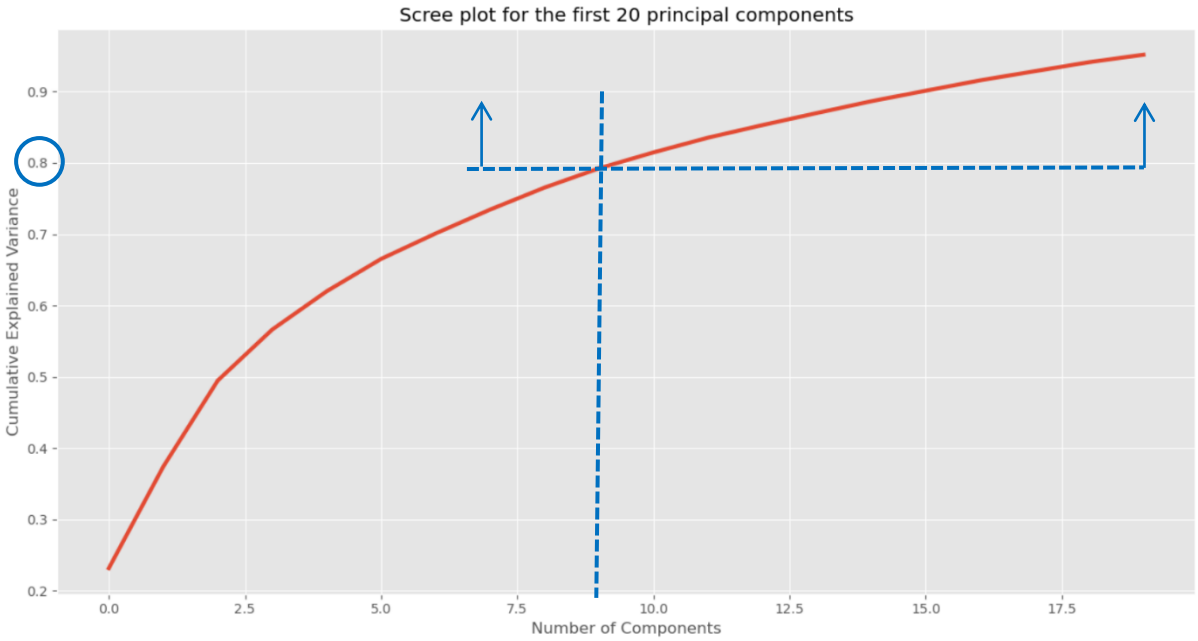


Figura 8: Scree Plot, Análisis de Componentes Principales

función desarrollada para generar esta gráfica es `f_scree_plot(pca, npca)`:

Aunque este scree plot no presente un claro punto de inflexión o codo, se puede ver que con las primeras ocho componentes principales ya se explica un 80% de la varianza total y que con 15 incluso el 90% lo cual es un valor muy razonable.

Tener 15 variables es todavía un número considerable pero la en comparación con las 60 que se tenía antes y las 86 iniciales, se ve que hemos reducido el problema a menos de una quinta parte del original sin perder apenas información. Este proceso es fundamental a la hora de utilizar grandes cantidades de datos.

La forma de la gráfica muestra que la correlación entre variables no es excesivamente alta, aunque sí que hay variables correladas.

En la Tabla 7 se presenta el autovalor de cada componente principal que indica la varianza explicada por ésta:

0.2317	0.1424	0.1206	0.07147	0.0537
0.0454	0.0356	0.03347	0.03086	0.0270
0.0223	0.02061	0.01764	0.0169	0.01655
0.0147	0.0146	0.0128	0.0104	0.0087

Tabla 7: Varianza explicada de las primeras 20 componentes principales

La suma da 0.952, es decir, con las primeras 20 componentes principales calculadas se puede explicar el 95% de la variabilidad en los datos presentados.

Se puede observar que las primeras cuatro aportan más de la mitad (56,6%) pero solo con estas no sería suficiente, ya que se estaría perdiendo demasiada información y nuestro modelo no desempeñaría correctamente.

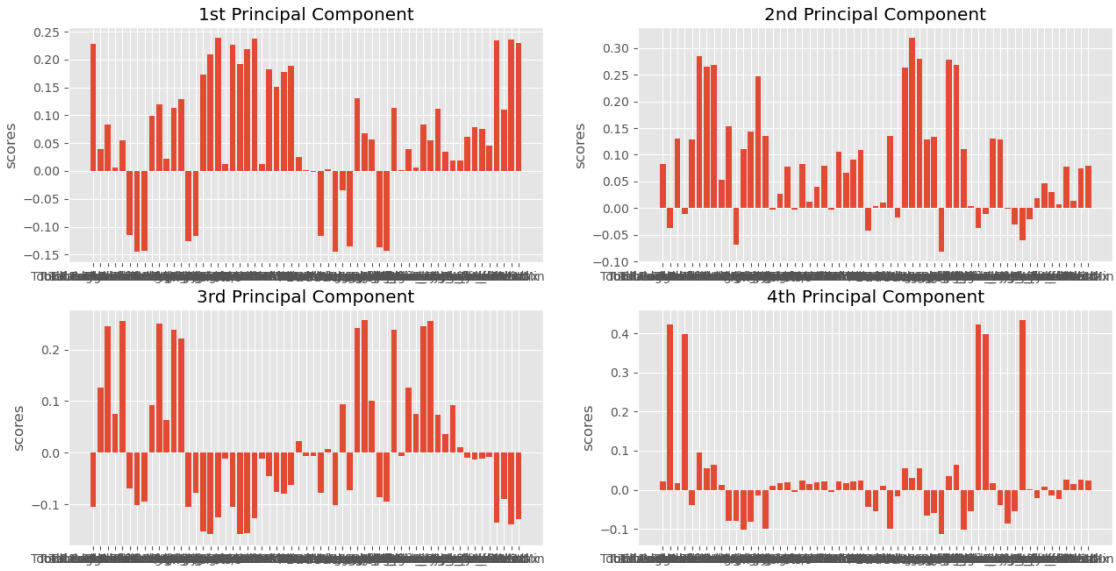


Figura 9: Loadings de las primeras 4 componentes principales

Con los resultados obtenido se han analizado para las 4 primeras componentes principales qué variables originales tienen más peso dentro de cada una de estas como se puede apreciar en la Figura 9.

En la Figura 10 se presenta en detalle las dos primeras componentes principales con el fin de entender que información contienen y qué representan dentro del dataset.

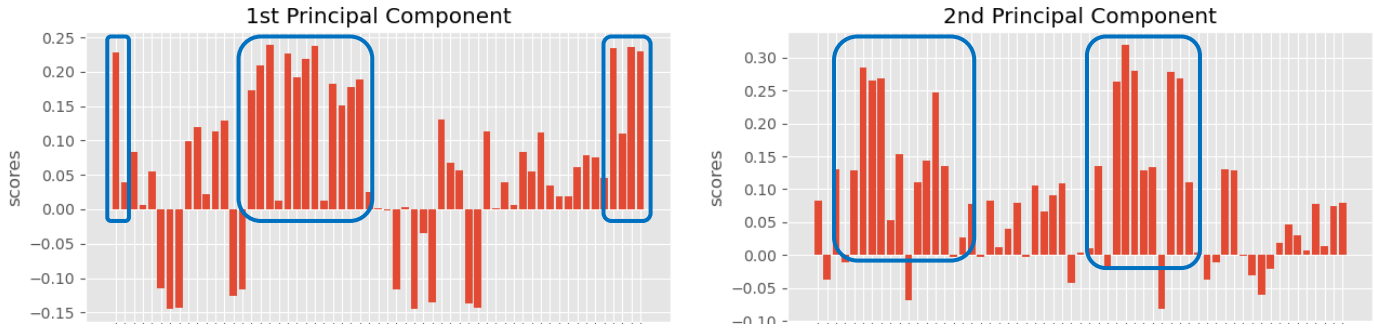


Figura 10: Detalle de las primeras dos componentes principales

Se han marcado en los diagramas las variables que dentro de cada componente tienen un coeficiente más alto respecto de las otras.

En el caso de la primera componente principal, esas variables son: Flow duration, Flow IAT Mean, Flow IAT Std, Flow IAT Max, Fwd IAT Total, Fwd IAT Mean, Fwd IAT Std, Fwd IAT Max, Bwd IAT Total, Bwd IAT Mean, Bwd IAT St, Bwd IAT Max, Idle Mean, Idle Std, Idle Max, Idle Min

En el caso de la segunda, las variables más relevantes son: Fwd Packet Length Max, Fwd Packet Length Min, Fwd Packet Length Mean, Flow Bytes/s, Min Packet Length, Max Packet Length, Packet Length Mean, Average Packet Size, Avg Fwd Segment Size.

Es decir, de una manera la primera componente principal viene a resumir la información contiene un vector relativo a tiempos. Tienen mucho peso el tiempo de flujo de paquetes, el tiempo entre llegadas (IAT o Inter Arrival Time), el tiempo máximo entre llegadas, pero también tiempos de respuestas y de envíos hacia atrás (Backwards Inter Arrival Time). Se puede apreciar claramente que, de manera automática, el análisis de componentes principales ha agrupado este tipo de información que podemos clasificar como “temporal” en una única variable. Es decir, a la hora de analizar este tipo de ataques, es fundamental analizar el tiempo en el que se producen las diferentes comunicaciones. En concreto, parece ser fundamental a la hora de analizar este aspecto, el tiempo que transcurre de una petición a la siguiente. El tiempo que pasa entre que el servidor responde a una petición y el cliente está haciendo la siguiente solicitud. Esto tiene sentido, al final un ataque DDoS se caracteriza por lanzar peticiones de manera masiva en un intervalo muy ajustado de tiempo, lo suficientemente corto como para que el servidor no sea capaz de procesarlas todas y acabe por saturar y dejar de funcionar. Luego esta sencilla técnica ha dado un nuevo enfoque clave a la forma de preparar una defensa eficaz a este tipo de amenazas. Y en parte acaba de constatar algo que de manera intuitiva se podría haber supuesto al principio del análisis. Las etiquetas con información relacionada con “idle” dan una pista sobre el tiempo que un flujo estaba libre o desocupado antes de ponerse activo que también puede tener bastante importancia

Si se analiza con más detalle la segunda componente principal se reconoce de nuevo un patrón. Esta vez la información clave no es relativa al tiempo sino al tamaño de los paquetes y a la cantidad de información que fluye. Esta variable recopila información relativa al máximo tamaño de paquetes que se envía, a la longitud media y máxima de estos y otras variables relacionadas también con el tamaño. Luego, se puede concluir claramente que esta segunda componente principal caracteriza un vector por la cantidad de información que envía. Tiene sentido que este aspecto sea el segundo que aparece, probablemente sea más fácil saturar una banda si se le exige grandes cantidades de información que si solo se piden unos pocos bytes.

Luego, se acaba de resolver gran parte del problema planteado mediante un algoritmo que utiliza como base un concepto tan antiguo como conocido que es el de los autovectores y autovalores. Es fascinante la de información que se ha podido abstraer de un problema con inicialmente 86 variables, un número que a mano no se podría manejar.

Finalmente, se han dibujado todos los vectores en el espacio formado por las dos primeras componentes principales en un biplot, Figura 11. De esta manera, se puede ver cómo se agrupan entre sí y si solo en base a estas dos componentes hay diferencias entre un vector que es ataque y uno que no lo es. Los resultados obtenidos son los siguientes:

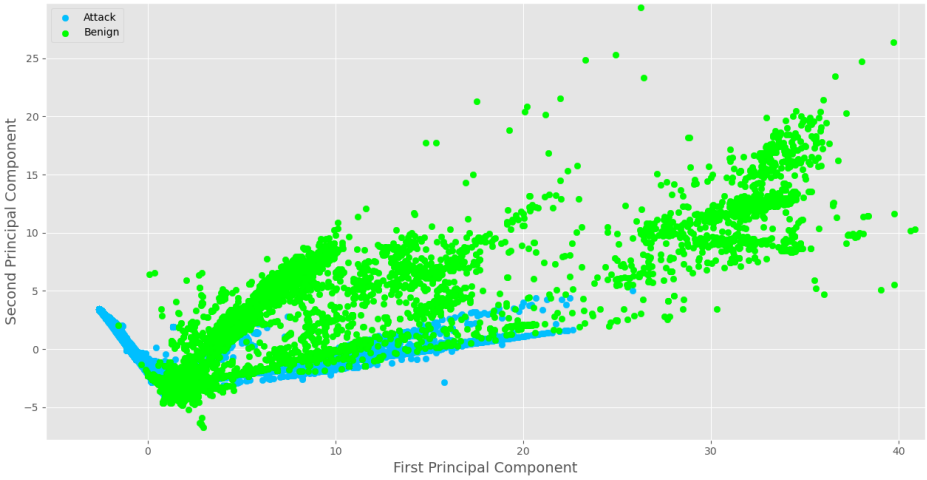


Figura 11: Biplot Plot - Ataque/Benigno

Dada la dispersión de los puntos se ha ampliado la vista:

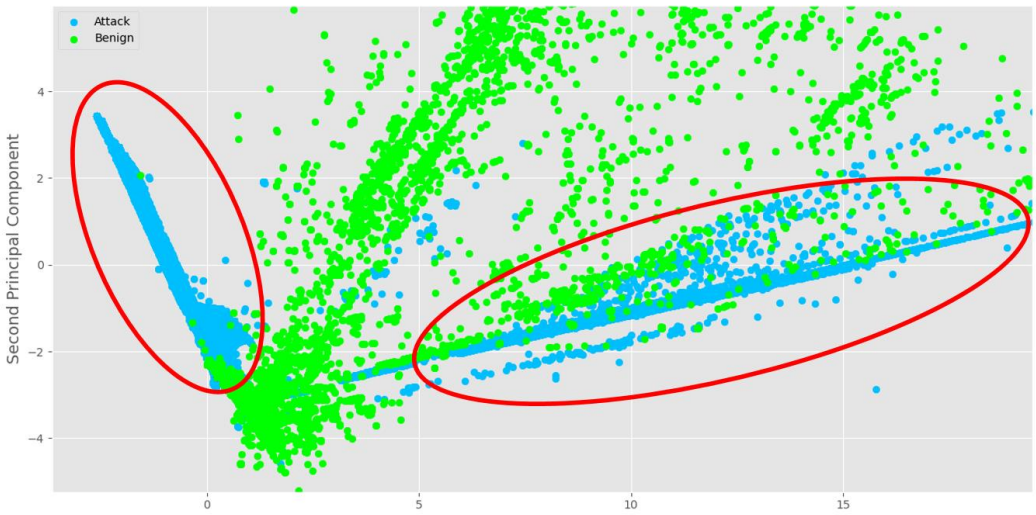


Figura 12: Biplot ampliado

De esta manera se puede apreciar ya cómo los diferentes ataques tienen una estructura similar entre sí mientras que los vectores que son benignos se distinguen de los anteriores y parece que están menos estructurados. Esto es coherente ya que un ataque está fabricado y dependiendo del tipo de ataque tiene sentido que tenga unas características u otras. Se puede ver cómo cuando la primera componente principal toma valores negativos estamos ante un ataque con una muy alta probabilidad, valores muy altos en esta variable también pueden apuntar a que estamos ante un ataque.

Respecto de la segunda componente principal no se puede sacar ninguna conclusión fija solo de observar este biplot.

Se ha añadido la misma gráfica en la cual los ataques se han separado según el tipo de ataque DDoS. Así se puede reconocer mejor la estructura de cada uno de ellos y se entiende mejor las agrupaciones que se han formado. Esta visualización concuerda perfectamente con las dos anteriores lo cual tiene sentido. No va a cambiar las características de un vector por el hecho de que se catalogue como “ATAQUE” o como “DrDoS_MSSQL”, el vector sigue siendo en el mismo y se tienen que distribuir todos de la misma manera. Sin embargo, en este análisis más detallado se aprecia que hay un ataque que tiene unas características bastante propias que es el UDP-Lag. Este ataque se distingue bastante de los demás y que se agrupan en una rama propia por la izquierda del plano.

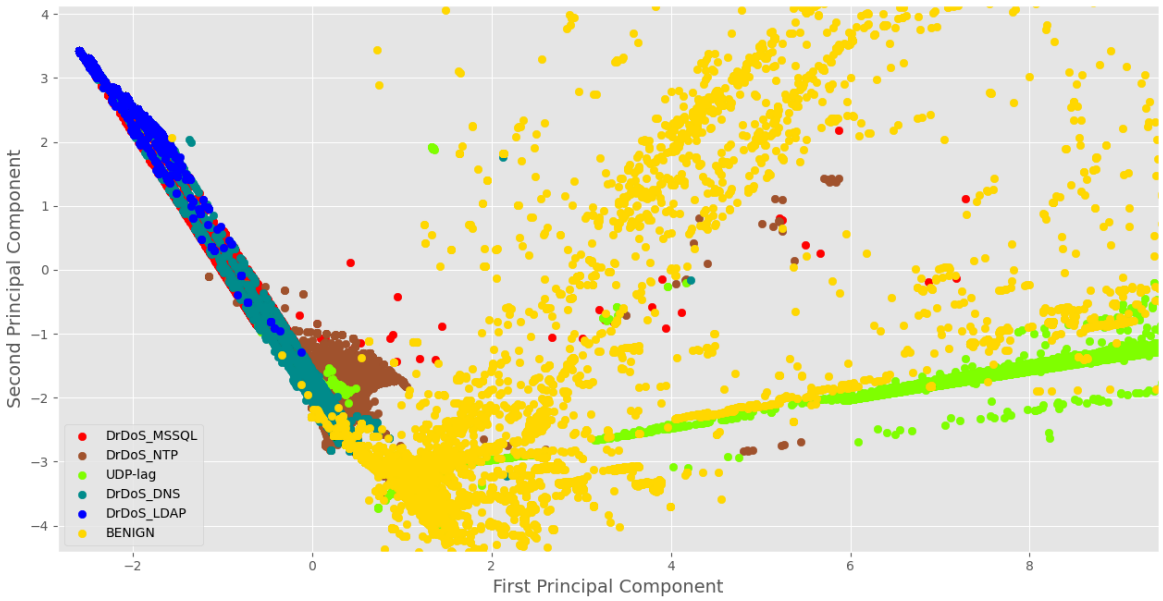


Figura 13: Biplot con ataque separados por colores

5.4. CLASIFICACIÓN Y MODELOS DE APRENDIZAJE AUTOMÁTICO

5.4.1. INTRODUCCIÓN

El aprendizaje automático o machine learning hace referencia a un tipo de algoritmos y procesos muy concretos que aprenden y son capaces de a través de datos obtenidos en el pasado, predecir y tomar decisiones a futuro. Los algoritmos de ML consisten de técnicas matemáticas, implementadas en ordenadores, que son capaces de procesar información, identificar patrones e inferir en base a esos datos.

De una manera muy genérica, se pueden separar en algoritmos supervisados que se tienen un enfoque Bayesiano, utilizando probabilidades de datos evento previamente para inferir las probabilidades de nuevos eventos. Por otro lado, están los algoritmos no supervisado que trazan abstracciones de conjuntos de datos no clasificados y aplican estas a nuevos datos. Ambas familias de algoritmos se pueden aplicar a problemas de clasificación (asignar observaciones a categorías) o de regresión (predecir propiedades numéricas de una observación).

El machine learning forma parte de una rama de la computación más amplia que engloba la inteligencia artificial y el deep learning o aprendizaje profundo. Como se observa en la figura, el machine learning forma el núcleo de lo que es la IA. Por ejemplo, un coche autónomo tiene que identificar muchos objetos como señales de tráfico, personas, árboles, … para luego tomar decisiones como velocidad o sentido de conducción. La primera parte sí que la resuelve el machine learning, sin embargo, para las decisiones más complejas hacen falta otro tipo de algoritmos que exceden el ámbito de lo que es el aprendizaje automático.

El deep learning es otro concepto que engloba ciertos algoritmos de ML que intentan imitar la labor neuronal del cerebro [24].

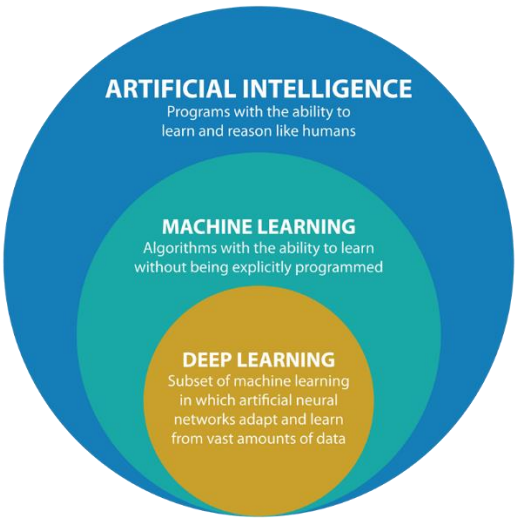


Figura 14: Relación IA, ML, DL

5.4.2. APLICACIONES DEL MACHINE LEARNING EN ENTORNOS INDUSTRIALES

Actualmente, el aprendizaje automático tiene tienen diversas aplicaciones a nivel industrial. Un ejemplo es la detección de anomalías: Se pueden monitorizar y controlar con sistemas SCADA (Supervisory Control And Data Acquisition) máquinas de una fábrica. Estos sistemas dan información acerca de la energía consumida, recursos de CPU que utiliza una máquina y otros datos. Estos, si la máquina hace una actividad que se repite con periodicidad, se pueden utilizar para entrenar algoritmos de aprendizaje automático que avisen y detecten si la máquina está consumiendo más energía de la pensada o si algo del proceso falla. Otra aplicación es a la hora de controlar el acceso de los empleados a diferentes recursos dentro de la empresa. Si un empleado, de media, accede 5 veces al día a cierta información y un día hay una actividad más elevada de lo habitual, puede estarse dando el caso de que alguien esté suplantando a ese empleado para acceder y extraer información relevante. Mediante la detección de anomalías esto se podría evitar. Además, se estima que en esta aplicación en concreto se van a invertir varios miles de millones de dólares dada su importancia.

Otro ejemplo a nivel industrial en el que es útil el machine learning es a la hora de planificar la producción de una empresa. La aplicación de estas técnicas proporciona a la empresa una visión más profunda sobre el funcionamiento de ésta y facilitará en gran medida diseñar estrategias para optimizar la producción. Analizando datos recopilados previamente la empresa puede diseñar y adaptar sus procesos de producción para que sean más eficientes y competitivos.

El control de calidad también es un área dentro de la industria que puede verse beneficiado por estos algoritmos que permiten agilizar y optimizar estos procesos para garantizar la seguridad y calidad de los productos que se producen.

En este caso, se van a aplicar los algoritmos propios de esta rama de la compoutacion para ver si su uso para proteger una empresa o un sistema industrial ante ciberataques puede ser favorable [25]. En este caso, se van a aplicar los algoritmos propios de esta rama de la computación para ver si su uso para proteger una empresa o un sistema industrial ante ciberataques puede ser favorable.

5.4.3. ALGORITMOS EMPLEADOS

Los algoritmos que se han empleado para evaluar los datos que se han preparado son los siguientes:

5.4.3.1. K-Nearest Neighbors (KNN)

- Este algoritmo es una forma simple de clasificar un elemento x a una clase $c_j$calcluando la probabilidad a posterior de que dicho elemento pertenezca a esa clase. El algoritmo se basa en buscar ejemplos cercanos en el espacio de los elementos.

5.4.3.2. Decision Tree

- El árbol de decisión es un modelo predictivo que mapea observaciones sobre un elemento a conclusiones sobre el valor objetivo del elemento. Estos modelos de árbol, en los que la variable destino puede tomar un conjunto finito de valores se denominan árboles de clasificación. Las ramas representan conjunciones de características que conducen a unas etiquetas representadas en las hojas.

5.4.3.3. Random Forest

- El random forest es un algoritmo que combina diferentes árboles de decisión siendo la salida de cada uno de ellos considerada como un ‘voto’. La opción más votada será la respuesta del bosque aleatorio. Se toman k atributos de los m total (con k <= m) y se crea un árbol de decisión con esas características. Después se repite el proceso n veces variando en cada árbol los k atributos que se han elegido. A cada uno de esos n árboles se le entrena y se le dan los datos para que haga una clasificación. Se guardan todos los resultados y finalmente se comparan entre todos los árboles tomando aquél que más veces haya salido.

5.4.3.4. AdaBoost

- AdaBoost es una abreviación de “Adaptive Boosting”, cuyo objetivo es mejorar la eficacia de otros algoritmos más “débiles”. Para ello, toma las salidas de estos algoritmos (‘estudiantes débiles’) y las combina en una suma

ponderada que representa la salida final del clasificador potenciado. – Aunque los alumnos sean algoritmos débiles, se puede demostrar que, mientras el rendimiento de cada uno de ellos sea mejor que adivinar al azar, el modelo final converge a un alumno fuerte.

5.4.3.5. Naive Bayes

- Este algoritmo se basa en el teorema de Bayes para clasificar los elementos. Este tipo de modelos son llamados "inocentes" o "naive" ya que asumen que las variables productoras son independientes entre sí. Esto significa que la presencia de una cierta característica en una serie de datos no tiene nada que ver con la presencia de otra. El algoritmo calcula una probabilidad a posteriori dadas las probabilidades de eventos a priori:
-
- $P(A/B) = \frac{P(B/A)*P(A)}{P(B)}$

5.4.3.6. Regresión logística

- Este modelo analiza los datos distribuidos binomialmente de la forma
- $Y_i \sim B(p_i, n_i), para\ i = 1, \dots, m$
- donde el número de ensayo n_i es conocido mientras que las probabilidades de acierto p_i son desconocidas.
- El modelo se obtiene a partir de la información que proporcione cada ensayo (valor de i) y las diferentes variables independientes acerca de la probabilidad final. Si se toman estas variables como un vector X_j k-dimensional, el modelo toma la forma:
- $p_i = E(\frac{Y_i}{n_i}|X_i)$
- Se modelan los logits de las probabilidades binomiales desconocidas como una función lineal de los X_i.
- $logit(p_i) = \ln\left(\frac{p_i}{1-p_i}\right) = \beta_0 + \beta_1 * x_{1,i} + \cdots + \beta_k * x_{k,i}$
- La función logit es el negativo de la derivada de la función de entropía binaria.

5.4.4. CÓDIGO DESARROLADO

La última parte del programa ha consistido en entrenar los diferentes modelos para ir analizando uno a uno que resultados arrojan y cómo desempeñan. Los modelos que se han utilizado son los descritos en el apartado anterior, no obstante, ha habido alguno, como por ejemplo la regresión logística, que no ha llegado a converger.

Para hacer el código lo más eficiente posible se han creado un vector con todos los objetos de los diferentes modelos que se iban a utilizar, cada uno con los parámetros que requirieran:

```
classifiers = [
        KNeighborsClassifier(5),
        DecisionTreeClassifier(max_depth=5),
        RandomForestClassifier(max_depth=5,
n_estimators=20, max_features=1),
        MLPClassifier(alpha=1, max_iter=1000),
        AdaBoostClassifier(),
        GaussianNB(),
        LinearRegression(max_iter=1000)]
```

Por ejemplo, en el algoritmo del árbol de decisión hay que especificar la profundidad máxima a la que se desea llegar. En otros el número máximo de iteraciones. Manipulando estos parámetros se logra que el algoritmo desempeñe mejor y de mejores resultados. Como no se ha querido evaluar uno en concreto sino varios, se han ajustado los parámetros de manera que todos los algoritmos convergieran, pero no se ha buscado afinar en exceso ya que habría conllevado bastante tiempo y no hubiera mejorado mucho los resultados obtenidos.

A continuación, se han preparado el dataset. Para ello se ha dividido en un training set y un test set. En aprendizaje automático, el training set se implementa para crear un modelo, mientras que test set (o validation set) se utiliza para validar el modelo desarrollado. Para ello, se excluyen los datos utilizados en el entrenamiento del test set. La proporción en la que se divide los datos es arbitraria. Normalmente se utiliza una mayor cantidad de los datos para el entrenamiento, se puede llegar a utilizar un 70%, mientras que para la validación se utiliza el resto, un 30%. Lo que está claro es que cuantos más datos se utilicen para entrenar el modelo, mejor va a desempeñar. Un mismo algoritmo que se entrena con un 50% de las observaciones y se valida con el otro 50% va a dar resultados mucho peores que uno que se entrena con el 90% de datos y se valida con el 10%. Aunque cabe decir que no hay una proporción óptima, ya que cada conjunto de datos es distinto.

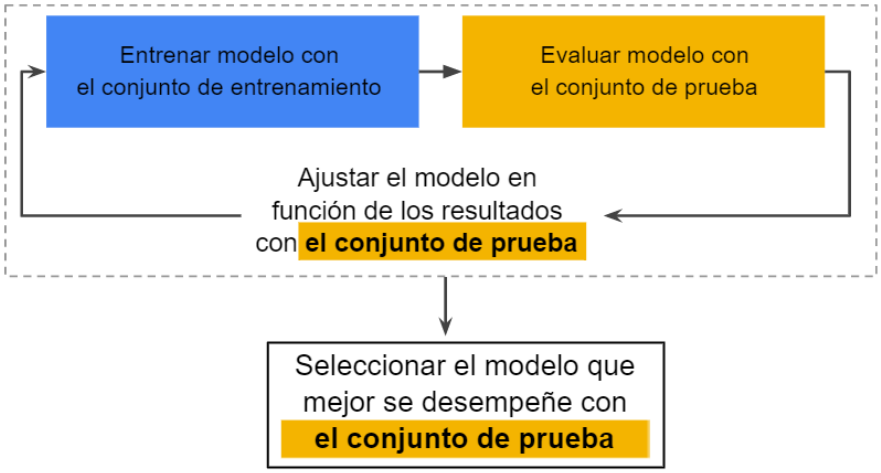


Figura 15: Ilustración Training y Test Set

En este caso se han evaluado diferentes casos, para ver también cómo influye la relación entrenamiento-validación.

Después separar los datos en training y test set se ha hecho un bucle que itera a través de los diferentes objetos instanciados de cada modelo, lo entrena y lo evalúa:

```
for name, clf in zip(names, classifiers):
    #clf=classifiers[2]
    clf.fit(X_train, y_train)
    score = clf.score(X_test, y_test)
    print('SCORE', name, ':', score)

    pred_y = clf.predict(X_test)
    results = confusion_matrix(y_test, pred_y)
    error = zero_one_loss(y_test, pred_y)

    dar_resultados(results, error)
```

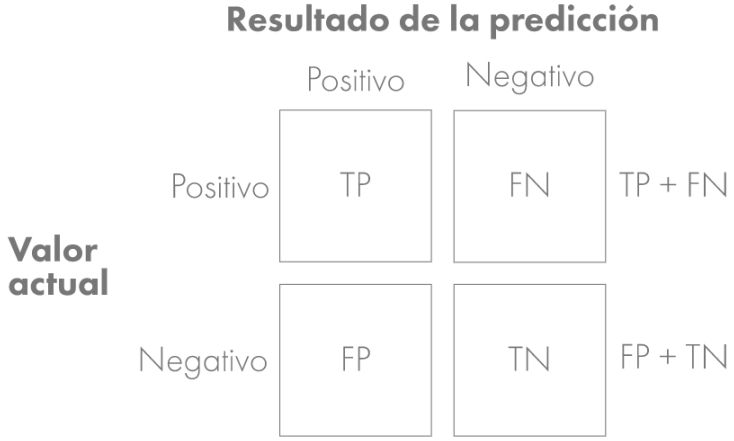
5.4.5. MÉTRICAS

A la hora de evaluar cómo ha desempeñado un algoritmo de aprendizaje automático, hay varios parámetros o métricas que se pueden utilizar. Al final no es uno de ellos solo el que indica si el modelo ha funcionado correctamente o no, sino el conjunto de estos valores el que proporciona la imagen más realista.

5.4.5.1. Matriz de confusión

La matriz de confusión es una herramienta que permite visualizar el desempeño de un algoritmo de aprendizaje supervisado. En las columnas se representan el

número de predicciones de cada clase, es decir la clase en la que se ha clasificado cada elemento. En las filas se representa la clase real a la que pertenece realmente cada elemento.



- ✓ En este caso, un positivo es un ataque y un negativo es un vector benigno.
- ✓ Se define como True Positive (TP) o verdadero positivo un vector maligno que ha sido clasificado como tal.
- ✓ Un False Positive (FP) o falso positivo es un vector benigno que se ha clasificado como ataque. A esto se le denomina además Error tipo I.
- ✓ Un false negative (FN) o falso negativo es un ataque que se ha clasificado como benigno. Este caso es un Error tipo II.
- ✓ True negative (TN) o verdadero negativo es un vector benigno que se ha clasificado como tal.

5.4.5.2. Métrica de exactitud (Accuracy)

Esta métrica es la más utilizada y la primera que se mira para ver cómo ha desempeñado un modelo. Indica el número de elementos que se han clasificado de manera correcta en relación al número total. No obstante, esta métrica tiene una gran limitación: No sirve en casos en los que haya clases desequilibradas ya que en caso de que una clase minoritaria sea clasificada erróneamente, la exactitud no va a reflejar esto igual que otras métricas.

$$accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

5.4.5.3. Métrica de exhaustividad o sensibilidad (Recall)

El recall muestra la cantidad de verdaderos positivos que ha clasificado en relación al número total de valores positivos. La pregunta a la que intenta responder esta métrica es: ¿Qué proporción de los ataques se identificaron correctamente?

$$recall = \frac{TP}{TP + FN}$$

5.4.5.4. *Métrica de precision*

La precisión representa la proporción de verdaderos positivos de todos los vectores clasificados como ataques. Es decir, de todos los elementos que se han clasificado como ataques, cuántos lo eran realmente.

$$precision = \frac{TP}{TP + FP}$$

5.4.5.5. *Puntuación F1 (F1-Score)*

- Esta métrica es una combinación de las dos anteriores, precisión y exhaustividad. Intenta mostrar un compromiso entre ambas medidas y cuanto mejor sean las dos por separadas mejor será la puntuación F1. De la misma manera, cuanto peor sean ambas por separado, peor será esta métrica. El mejor valor que puede tomar es 1 u el peor es 0.

$$F1 = 2 * \frac{precision * recall}{precision + recall}$$

5.4.6. *RESULTADOS*

Los resultados obtenidos para las diferentes métricas se han reflejado en la siguiente tabla:

Classifier	Score	Error	Precision	Recall	F1-Score
Nearest Neighbors	99,77%	0,23%	99,84%	99,89%	99,86%
Decision Tree	99,41%	0,59%	99,75%	99,56%	99,65%
Random Forest	97,77%	2,23%	97,69%	99,74%	98,71%
Neural Net	97,11%	2,89%	96,91%	99,80%	98,33%
AdaBoost	99,24%	0,76%	99,62%	99,48%	99,55%
Naive Bayes	88,14%	11,86%	90,04%	96,84%	93,31%
Loigistic Regression	- No ha convergido -				

Número total de ataques: 134175
Número total de vectores benignos: 22823
Número total de muestras: 156998

Tabla 8: Resultados obtenidos de los distintos modelos

5.4.7. *CONCLUSIONES*

Como se puede ver, los resultados en general de todos los modelos han sido muy buenos. Probablemente sean incluso demasiado buenos. Esto puede deberse a que el dataset del CIC sea sintético y que por ello el algoritmo no tenga demasiada dificultad a la hora de clasificar. Además, se veía en el apartado de análisis de componentes que los

ataques se agrupaban bastante bien solo viendo las dos primeras componentes principales. Es decir, no parece ser excesivamente complicado en este tipo de problemas identificar correctamente una intrusión o un ataque sobre un sistema. Por ello, en un caso real, los resultados serán algo inferiores, pero no por ello malos.

Por lo general, se ve que el que ha obtenido un score más alto ha sido el de vecinos cercanos con un 99.77% de aciertos y el que peor el de Naive Bayes con un 88,14% de aciertos. Esto se puede deber a que este clasificador asume una distribución binomial dentro de los elementos y no hay garantía de que en este caso sea así.

En base a los resultados obtenidos se puede concluir que utilizar técnicas y algoritmos de aprendizaje automático para aumentar la seguridad en entornos industriales es una opción viable y eficaz.

Finalmente, se quería tener en cuenta la eficacia que ha mostrado el análisis de componentes principales ya que ha permitido entender mejor los datos que se tenían y ha sacado estructuras y patrones que tenían los vectores y que sin este tipo de análisis no se hubieran podido detectar ni analizar.

CAPÍTULO 6. CONCLUSIONES Y FUTURAS LÍNEAS DE TRABAJO

Las conclusiones que se pueden extraer de este trabajo son varias, debido a los dos análisis que se han realizado.

Como conclusión principal, se ha podido constatar que gracias a la digitalización y a la Industria 4.0, las empresas van a aumentar sus productividad y eficiencia. La posibilidad de interconectar todos los equipos y máquinas que forman un proceso industrial permite tener mucho más control en todo momento. Además, con la posibilidad de extraer grandes cantidades de datos para su posterior análisis, se abre la puerta a llevar a niveles máximos la automatización de la industria.

No obstante, en contrapartida, se ha detectado que todas estas ventajas conllevan varios riesgos, uno de ellos especialmente relevante. Al tener todo el proceso enlazado, se puede acceder desde cualquier punto de la fábrica a cualquier equipo, es decir, una brecha en cualquier equipo pasa a no solo comprometer ese quipo sino a comprometer todo el sistema. Hasta ahora, máquinas que operaban de manera aislada sin conexión a la red contaban con una ventaja fundamental a nivel de seguridad, la brecha de aire o *air gap*.

Por este motivo, era necesario revisar y ver hasta qué punto los protocolos de ciberseguridad que se han empleado tradicionalmente son fiables y seguro a día de hoy. El análisis llevado a cabo a concluido en que este tipo de protocolos, con el Ethernet Industrial y el Fieldbus como máximos exponentes, presentan carencias a nivel de seguridad. Análogamente, se ha identificado el protocolo OPC-UA como candidato a liderar esta transformación ya que cuenta con características muy potentes a nivel de seguridad y compatibilidad que le colocan en una posición ventajosa.

A parte de este análisis sobre la madurez de protocolos en entornos OT, se ha desarrollado un clasificador de ciberataques con el objetivo de aprender y entender un ataque muy concreto, el DDoS. Las conclusiones de esta parte son varias: Hay técnicas de aprendizaje automático con potencial para detectar una amenaza de este tipo. Además, estos algoritmos están accesibles para todos ya que están programados en Python que código abierto y de la misma manera que lo son todas sus librerías.

Además, se ha descubierto que a través del análisis de componentes principales se puede entender mucho mejor la estructura que subyace a un conjunto de datos. No solo reduce la cantidad de variables que se tienen que manejar. En este caso ha permitido vislumbrar que a la hora de detectar un ataque hay dos factores que son fundamentales: Por un lado, todas las variables que registran tiempos son muy relevantes. Ya sea el tiempo entre llamadas al servidor o el tiempo entre una petición y una respuesta. El tiempo es una variable clave en este entorno. Por otro lado, se ha detectado la importancia del tamaño de los paquetes. Junto con el tiempo, ha sido muy importante analizar la cantidad de información que se transmite de un punto a otro para ver si ese movimiento es o no es una potencial amenaza.

Finalmente, los resultados obtenidos en los modelos de aprendizaje automático confirman que este tipo de herramientas pueden ser muy útiles a la hora de prevenir ataques en un sistema OT. Los resultados han sido muy favorables con apenas errores a la hora de

clasificar los vectores. Se puede pensar que, si esto se hiciera en entornos reales probablemente el rendimiento de los algoritmos fuera peor, pero seguro que seguirían siendo valores muy elevados en todas las métricas que se han analizado.

Cabe mencionar algunos aspectos más: Uno de los principales problemas detectados durante el análisis de datos, es que estaban completamente desbalanceados, es decir, había un 99,99% de ataque y apenas un 0.01% de vectores benignos. Esto hay que tenerlo muy en cuenta ya que sino el modelo puede aprender a clasificar todo como ataque y va a tener un desempeño casi del 100% por lo que desaparece la clase benigna. Por ello, no solo se puede mirar una única métrica, sino que es el conjunto de varias de ellas el que nos permite juzgar adecuadamente si un modelo ha aprendido bien o no.

Finalmente mencionar que los modelos de aprendizaje automático presentan dos inconvenientes. El primero, que los atacantes también los pueden utilizar para hacer que sus ataques sean cada vez más sofisticados y dificulten su identificación. Además, estos modelos aprenden con observaciones pasadas, es decir, se podrían fabricar datos falsos que hagan que estos modelos aprendan mal. Esta técnica de intentar enseñar modelos a base de engañarlos se llama Adversarial Machine Learning y podría ser una potencial línea de investigación para futuros trabajos.

Otras ideas para continuar desarrollando serían ver más en detalle el funcionamiento de OPC-UA y analizar qué puntos débiles tiene este protocolo.

Respecto al modelo de aprendizaje automático, en un futuro se pretende desarrollar otro algoritmo que en tiempo real sea capaz de percibir cambios que puedan significar que hay un ataque inminente.

Referencias

[1] PwC. (2016). Building the digital Enterprise. *Global Industry 4.0 Survey*, 11.

[2] Labs, W. (8 Noviembre 2017), último acceso: 03/03/2020,*'Industry 4.0' network architecture relies on interconnectivity,* https://www.foodengineeringmag.com/articles/97066-industry-40-network-architecture-relies-on-interconnectivity

[3] Snitkin, S. (Junio 2015) *Secure-by-Design Industrial Internet of Things*, ARC Industry Forum

[4] Protocolos de comunicaciones, último acceso: 10/06/2020, https://es.wikipedia.org/wiki/Protocolo_de_comunicaciones

[5] Zimmerman, H. (April 1980), *OSI Reference Model . The ISO Model of Architecture for Open System Interconnection*, IEEE Transactions on Communications Vol. 4, p.425

[6] Protocolos de comunicaciones industrials, último acceso: 11/06/2020 http://www.aie.cl/files/file/comites/ca/articulos/agosto-06.pdf

[7] BBC (11 Octubre 2015), *El virus que tomó control de mil máquinas y les ordenó autodestruirse*, último acceso: 14/06/2020, https://www.bbc.com/mundo/noticias/2015/10/151007_iwonder_finde_tecnologia_virus_stuxnet

[8] Protocolos de comunicación industriales (17 Junio 2019), último acceso: 14/06/2020, https://www.logicbus.com.mx/blog/protocolos-de-comunicacion-industriales/#:~:text=Muchas%20veces%20se%20ha%20escuchado,entre%20varios%20dispositivos%20que%20forman

[9] Protocolos de comunicaciones industrials, último acceso: 11/06/2020 http://www.aie.cl/files/file/comites/ca/articulos/agosto-06.pdf

[10] Herrero, M., López, A., *Protocols and Network security in ICS infrastructures*, INCIBE

[11] Profibus Nutzerorganisation, (Octubre 2014), *PROFINET System Description – Technology and Application*

[12] *OPC UA (Arquitectura Unificada)*, último acceso: 20/06/2020 https://www.matrikonopc.es/opc-ua/index.aspx

[13] Resnick, C. (28/09/2017) *OPC Technology Well-positioned for Further Growth in Tomorrow's Connected World*, último acceso: 20/06/2020, https://www.arcweb.com/blog/opc-technology-well-positioned-further-growth-tomorrows-connected-world

[14] Juniper Research, último acceso: 20/06/2020, https://www.juniperresearch.com/home

[15] DDoS Threat Report 2018 Q2, último acceso: 20/06/2020, https://blog.nexusguard.com/threat-report/ddos-threat-report-2018-q2

[16] Mellnick, J. (10/03/2020), Top 10 Most Common Types of Cyber Attacks, último acceso: 20/06/2020, https://blog.netwrix.com/2018/05/15/top-10-most-common-types-of-cyber-attacks/

[19]Ruíz, J.A. *DDOS ACTUALIDAD, TAXONOMÍA Y CONTRAMEDIDAS*, último acceso: 22/06/2020, https://revista.seguridad.unam.mx/numero-13/ddos-actualidad-taxonom%C3%AD-y-contramedidas

[20] *Mersenne Twister*, último acceso: 22/06/2020, https://es.wikipedia.org/wiki/Mersenne_twister

[21] *Generate pseudo-randome numbers*, último acceso: 22/06/2020, https://docs.python.org/3/library/random.html

[22] Gil, C. (06/2018) Análisis de componentes Principales, último acceso: 23/06/2020, https://rpubs.com/Cristina_Gil/PCA

[23] Eugenio Sánchez Úbeda – Principal Components Analysis, Estadística II, February 2019

[24] ExperiencIA Oracle (14/09/2018) Diferencias entre la Inteligencia Artificial y el Machine Learning, último acceso: 15/06/2020, https://medium.com/@experiencIA18/diferencias-entre-la-inteligencia-artificial-y-el-machine-learning-f0448c503cd4

[25]*Utilización del Machine Learning en la industria 4.0*, último acceso: 16/06/2020, https://geinfor.com/business/utilizacion-del-machine-learning-en-la-industria-4-0/

Anexo A: LISTA DE LAS 87 ETIQUETAS DEL ANALIZADOR DE RED EXPLICADAS (EN INGLÉS)

Flow.ID
a flow identifier following the next format:

Source.IP
The source IP address of the flow.

Source.Port
The source port number

Destination.IP
The destination IP address.

Destination.Port
The destination port number.

Protocol
The transport layer protocol number identification (i.e.,TCP = 6, UDP = 17).

Timestamp
The instant the packet was captured stored in the next date format: Dd/mm/yyyy HH:MM:SS

Flow.Duration
The total duration of the flow

Total.Fwd.Packets
The total number of packets in the forward direction.

Total.Backward.Packets
The total number of packets in the backward direction.

Total.Length.of.Fwd.Packets
The total quantity of bytes in the forward direction obtained from all the flow (all the packets transmitted). This is obtained from the Total Length field stored on the packets header.

Total.Length.of.Bwd.Packets
The total quantity of bytes in the backward direction obtained from all the flow (all the packets transmitted). This is obtained from the Total Length field stored on the packets header.

Fwd.Packet.Length.Max
The maximum value in bytes of the packets length in the forward direction.

Fwd.Packet.Length.Min
The minimum value in bytes of the packets length in the forward direction.

Fwd.Packet.Length.Mean
The mean value in bytes of the packets length in the forward direction.

Fwd.Packet.Length.Std
The standard deviation in bytes of the packets length in the forward direction.

Bwd.Packet.Length.Max
The maximum value in bytes of the packets length in the backward direction.

Bwd.Packet.Length.Min
The minimum value in bytes of the packets length in the backward direction.

Bwd.Packet.Length.Mean
The mean value in bytes of the packets length in the backward direction.

Bwd.Packet.Length.Std
The standard deviation in bytes of the packets length in the backward direction.

Flow.Bytes.s
The number of bytes per second in the flow.

Flow.Packets.s
The number of packets per second in the flow.

Flow.IAT.Mean
The mean value of the inter-arrival time of the flow (in both directions).

Flow.IAT.Std
The standard deviation of the inter-arrival time of the flow (in both directions).

Flow.IAT.Max
The maximum value of the inter-arrival time of the flow (in both directions).

Flow.IAT.Min
The minimum value of the inter-arrival time of the flow (in both directions).

Fwd.IAT.Total
The total Inter-arrival time in the forward direction.

Fwd.IAT.Mean
The mean inter-arrival time in the forward direction.

Fwd.IAT.Std
The standard inter-arrival time in the forward direction

Fwd.IAT.Max
The maximum value of the inter-arrival time in the forward direction

Fwd.IAT.Min
The minimum value of the inter-arrival time in the forward direction

Bwd.IAT.Total
The total Inter-arrival time in the backward direction.

Bwd.IAT.Mean
The mean inter-arrival time in the backward direction.

Bwd.IAT.Std
The standard inter-arrival time in the backward direction.

Bwd.IAT.Max
The maximum value of the inter-arrival time in the backward direction.

Bwd.IAT.Min
The minimum value of the inter-arrival time in the backward direction

Fwd.PSH.Flags
The number of times the packets sent in the flow had the pushing flag bit set as 1 in the forward direction. The Pushing flag allows to send information immediately without filling all the buffer size from a packet, notifying the receptor to pass the packet to the application at once, it is very useful for real time applications.

Bwd.PSH.Flags
The number of times the packets sent in the flow had the PSH (pushing) flag bit set as 1 in the backward direction.

Fwd.URG.Flags
The number of times the packets sent in the flow had the URG (Urgent) flag bit set as 1 in the forward direction. The URG flag is used to inform a receiving station that certain data within a segment is urgent and should be prioritized. If the URG flag is set, the receiving station evaluates the urgent pointer, a 16-bit field in the TCP header. This pointer indicates how much of the data in the segment, counting from the first byte, is urgent.

Bwd.URG.Flags
The number of times the packets sent in the flow had the URG (Urgent) flag bit set as 1 in the backward direction.

Fwd.Header.Length
The header length of the packets flow in the forward direction.

Bwd.Header.Length
The header length of the packets flow in the backward direction.

Fwd.Packets.s
The number of packets per second in the forward direction.

Bwd.Packets.s
The number of packets per second in the backward direction.

Min.Packet.Length

The minimum length of the packets registered in the flow (both forward and backward directions).

Max.Packet.Length
The maximum length of the packets registered in the flow (both forward and backward directions).

Packet.Length.Mean
The mean value of the length of the packets registered in the flow (both forward and backward directions).

Packet.Length.Std
The standard deviation of the length of the packets registered in the flow (both forward and backward directions).

Packet.Length.Variance
The variance of the length of the packets registered in the flow (both forward and backward directions).

FIN.Flag.Count
The number of times the packets sent in the flow had the FIN flag bit set as 1. In the normal case, each side terminates its end of the connection by sending a special message with the FIN (finish) bit set. This message, sometimes called a FIN, serves as a connection termination request to the other device, while also possibly carrying data like a regular segment. The device receiving the FIN responds with an acknowledgment to the FIN to indicate that it was received. The connection as a whole is not considered terminated until both sides have finished the shutdown procedure by sending a FIN and receiving an ACK.

SYN.Flag.Count
The number of times the packets sent in the flow (in both directions) had the SYN (Synchronize) flag bit set as 1. The SYN (Synchronize) flag is the TCP packet flag that is used to initiate a TCP connection. A packet containing solely a SYN flag is the first part of the "three-way handshake" of TCP connection initiation. It is responded to with a SYN-ACK packet. Packets setting the SYN flag can also be used to perform a SYN flood and a SYN scan.

RST.Flag.Count
): The number of times the packets sent in the flow (in both directions) had the RST (Reset) flag bit set as 1 - (An RST says reset the connection. It must be sent whenever a segment arrives which apparently is not intended for the current connection - FIN says, "I finished talking to you, but I'll still listen to everything you have to say until you're done" (Wait for an ACK) RST says, "There is no conversation. I am resetting the connection!").

PSH.Flag.Count
The number of times the packets sent in the flow (in both directions) had the PSH (Pushing) flag bit set as 1.

ACK.Flag.Count

The number of times the packets sent in the flow (in both directions) had the ACK (Acknowledged) flag bit set as 1. To establish a connection, TCP uses a three-way handshake. Before a client attempts to connect with a server, the server must first bind to and listen at a port to open it up for connections: this is called a passive open. Once the passive open is established, a client may initiate an active open.

URG.Flag.Count
The number of times the packets sent in the flow (in both directions) had the URG (Urgent) flag bit set as 1.

CWE.Flag.Count
The number of times the packets sent in the flow (in both directions) had the CWR (Congestion Window Reduced) TCP flag set as 1. During the synchronization phase of a connection between client and server, the TCP CWR and ECE (Explicit Congestion Notification - Echo) flags work in conjunction to establish whether the connection is capable of leveraging congestion notification. In order to work, both client and server need to support ECN (Explicit Congestion Notification). To accomplish this, the sender sends a SYN packet with the ECE and CWR flags set, and the receiver sends back the SYN-ACK with only the ECE flag set. Any other configuration indicates a non-ECN setup.

ECE.Flag.Count
The number of times the packets sent in the flow (in both directions) had the ECE (Explicit Congestion Notification Echo) TCP flag set as 1.

Down.Up.Ratio
Download and upload ratio.

Average.Packet.Size
The average size of each packet. It is important to notice that Packet Length specify the size of the whole packet including the header, trailer and the data that send on that packet. But Packet Size specify only the size of the header on the packet.

Avg.Fwd.Segment.Size
The average segment size observed in the forward direction. A TCP segment is the Protocol Data Unit (PDU) which consists of a TCP header and an application data piece which comes from the upper Application Layer. Transport layer data is generally named as segment and network layer data unit is named as datagram but when UDP is used as transport layer protocol the data unit is called UDP datagram since the UDP data unit is not segmented (segmentation is made in transport layer when TCP is used).

Avg.Bwd.Segment.Size
Average Segment size observed in the backward direction.

Fwd.Header.Length.1
The header length of the packets flow in the forward direction. This attribute has the exact same values than the attribute Fwd Header Length, hence it can be a bug on the CICFlowmeter software.

Fwd.Avg.Bytes.Bulk

The average number of bytes bulk rate in the forward direction. Bulk data transfer is a software-based mechanism designed to move large data file using compression, blocking and buffering methods to optimize transfer times.

Fwd.Avg.Packets.Bulk
Average number of packets bulk rate in the forward direction.

Fwd.Avg.Bulk.Rate
Average number of bulk rate in the forward direction.

Bwd.Avg.Bytes.Bulk
Average number of bytes bulk rate in the backward direction.

Bwd.Avg.Packets.Bulk
Average number of packets bulk rate in the backward direction.

Bwd.Avg.Bulk.Rate
Average number of bulk rate in the backward direction.

Subflow.Fwd.Packets
The average number of packets in a subflow in the forward direction. The core idea of multipath TCP is to define a way to build a connection between two hosts and not between two interfaces (as standard TCP does). In standard TCP, the connection should be established between two IP addresses. Each TCP connection is identified by a four-tuple (source and destination addresses and ports). Given this restriction, an application can only create one TCP connection through a single link. Multipath TCP allows the connection to use several paths simultaneously. For this, Multipath TCP creates one TCP connection, called subflow, over each path that needs to be used. The detailed protocol specification is provided in RFC 6824

Subflow.Fwd.Bytes
The average number of bytes in a subflow in the forward direction.

Subflow.Bwd.Packets
The average number of packets in a subflow in the backward direction.

Subflow.Bwd.Bytes
The average number of bytes in a subflow in the backward direction.

Init_Win_bytes_forward
The total number of bytes sent in the initial window in the forward direction. TCP uses a sliding window flow control protocol. In each TCP segment, the receiver specifies in the receive window field the amount of additionally received data (in bytes) that it is willing to buffer for the connection. The sending host can send only up to that amount of data, before it must wait for an acknowledgment and window update from the receiving host.

Init_Win_bytes_backward
The total number of bytes sent in the initial window in the backward direction.

act_data_pkt_fwd
Count of packets with at least one byte of TCP data payload in the forward direction.

min_seg_size_forward
Minimum segment size observed in the forward direction.

Active.Mean
The mean time a flow was active before becoming idle.

Active.Std
Standard deviation time a flow was active before becoming idle.

Active.Max
Maximum time a flow was active before becoming idle.

Active.Min
Minimum time a flow was active before becoming idle.

Idle.Mean
Mean time a flow was idle before idle before becoming active.

Idle.Std
Standard deviation time a flow was idle before becoming active.

Idle.Max
The maximum time a flow was idle before becoming active.

Idle.Min
The minimum time a flow was idle before becoming active.

Label
The state of the flow (benign or malign).

L7Protocol
This attribute represents the code number of the layer 7 protocol as obtained from nDPI in Ntopng application. It is a number that varies from 0 to 226 (e.g., 0 is labeled as Unknown application).

ProtocolName
This attribute is the objective class of the dataset. It holds the application name following the code number stored in the L7Protocol attribute (e.g., YouTube, Yahoo, Facebook, etc.).1

[1] Fuente: https://www.kaggle.com/jsrojas/ip-network-traffic-flows-labeled-with-87-apps, último acceso: 13/06/2020

ANEXO B: ALINEACIÓN DEL PROYECTO REALIZADO CON LOS ODS Y LA AGENDA 2030

Ilustración 4: Objetivos de Desarrollo Sostenible

Según se explica en la página de la Organización de las Naciones Unidas (ONU), los Objetivos de Desarrollo Sostenible (ODS) "constituyen un llamamiento universal a la acción para poner fin a la pobreza, proteger el planeta y mejora las vidas […] de las personas en todo el mundo" (ONU, 2015)[2]. Para ello, se aprobaron en 2015 17 Objetivos como parte de la Agenda 2030[3], en la cual se detalla el plan de acción para alcanzar estos Objetivos en 15 años.

Éstos objetivos se pueden englobar en tres bloques: Uno de ámbito económico, uno relacionado con lo social y otro que engloba aspectos relacionados con la biosfera. La intersección de estos tres conceptos es la que permite que tengamos un desarrollo sostenible, es decir, que las generaciones actuales puedan "satisfacer [sus] necesidades sin comprometer la capacidad de las futuras generaciones para satisfacer sus propias necesidades"[4]

Se ha detectado una fuerte relación entre este trabajo y el Objetivo de Desarrollo Sostenible 9: Industria, Innovación e Infraestructura. Este ODS pertenece al grupo de objetivos que engloban el ámbito económico.

Este objetivo, sostiene que la industrialización inclusiva y sostenible, junto con la innovación y la infraestructura, son elementos claves para tener economías dinámicas y competitivas capaces de generar empleo y riqueza. Para ello es necesario introducir nuevas tecnologías y permitir hacer un uso eficiente de recursos. Según dice la ONU en su página, "la innovación y el progreso tecnológico son claves para descubrir soluciones duraderas"[5]. Para ello recalca la importancia de invertir en Investigación y Desarrollo (I+D).

Dentro de los distintos "targets" o metas de este objetivo se pueden distinguir dos pilares sobre los cuales radican. Por un lado, invertir en infraestructuras fiables y sostenibles que permitan el desarrollo de la industria en las distintas regiones. Sin embargo, estas infraestructuras no sólo tienen que ser a nivel de edificios o transporte. Para que una región pueda desarrollar una industria fuerte y sostenible tiene que desarrollar una infraestructura de red o digital muy potente también con todo lo que ello conlleva: Acceso a comunicaciones, pero también a que esas comunicaciones se produzcan de manera segura. Si un edificio se construye para que no se caiga o un tren para que no descarrile, una red de comunicaciones tiene que permitir el acceso y el intercambio de información de manera segura, sin comprometer información personal o ningún proceso de producción. Y eso es lo que se ha estado buscando en parte con este trabajo, lograr que la industria avance a esta nueva etapa basada en la automatización de procesos pero de una manera segura y sin poner en peligro nada.

[2] https://www.un.org/sustainabledevelopment/es/development-agenda/
[3] https://www.un.org/ga/search/view_doc.asp?symbol=A/RES/70/1&Lang=S
[4] "Nuestro Futuro Común", 1987, Comisión Mundial sobre el Medio Ambiente
[5] https://www.un.org/sustainabledevelopment/es/infrastructure/

Este trabajo, dentro de estas metas, está relacionada de manera muy estrecha con la 9.5 que habla de "aumentar la investigación científica y mejorar la capacidad tecnológica"[6]. Al final, la industria está avocada a implementar cada vez más herramientas basadas en aprendizaje automático e inteligencia artificial. Aquí está el desarrollo. Como estas oportunidades tienen que ser accesibles para todos, se ha apostado en este trabajo de manera muy fuerte por código abierto y librerías que son accesibles para cualquier persona con acceso a internet. Si se hubiera utilizado en vez de Python, por ejemplo, Matlab que tiene unas librerías y "toolboxes" muy potentes, se habrían obtenido seguramente muchas ventajas, pero se estaría privando a mucha gente que no pueda permitirse pagar una licencia de Matlab de poder aplicar estas técnicas que entre otras cosas buscan garantizar la seguridad en entornos industriales.

La ciberseguridad es cada vez más importante en cualquier industria y es una pieza clave para el desarrollo. Si una fábrica pretende a día de hoy producir de manera eficiente y sostenible, inevitablemente, acabará encontrándose con la necesidad de interconectar y automatizar procesos. Si esto no va acompañado de un desarrollo de una infraestructura de ciberseguridad robusta, estará excesivamente expuesta a cualquier intento de ataque que haría colapsar todo el sistema. Entonces, tendría que prescindirse de cualquier intento de digitalizar la industria y esto iría en contra del desarrollo industrial y la innovación marcadas por el Objetivo 9 de los ODS.

[6] https://www.un.org/sustainabledevelopment/es/infrastructure/