Diseño de Sistemas de Detección de Intrusión en Redes Definidas por Software: revisión basada en Machine Learning

Graciela Becci[1], Miguel Morandi[1], Luis Marrone[2]

[1] Universidad Nacional de San Juan, Av Libertador San Martín Oeste 1109, San Juan, J5400ARL Argentina
[2]Universidad Nacional de La Plata, Facultad de Informática, Calle 50 y Av 120, La Plata, Buenos Aires, B1900ASF Argentina
gbecci@unsj.edu.ar, morandi@unsj.edu.ar, lmarrone@linti.unlp.edu.ar

Resumen. El paradigma de Redes Definidas por Software SDN propone el control centralizado de la red, desacoplando los dispositivos de enrutamiento e independizando la red del software propietario. Esto ofrece la ventaja al controlador de tener una visión integral de la topología, y flexibilidad para generar reglas que gobiernen el comportamiento de toda la red. No obstante en una red SDN se agrega el desafío a la seguridad debido a su naturaleza distribuida: cada nodo es un punto de vulnerabilidad, y el controlador es pasible de ataques del tipo man-in-the-middle, en su conexión con el plano de aplicaciones, y con el plano de datos de los enrutadores. En seguridad de redes una de las herramientas básicas es el Sistema de Detección de Intrusión IDS, siendo la detección de amenazas un problema doble: identificación de datos sospechosos, y clasificación del flujo en normal o anómalo. Para aportar al diseño de un IDS es que en este artículo se presenta un resumen de los tipos de IDS con aplicabilidad en SDN y con el enfoque de Machine Learning, para dotar al IDS de inteligencia, mejorando la performance, y aprovechando las características de programabilidad del controlador SDN.

Palabras clave: Redes Definidas por Software SDN, Sistemas de Detección de Intrusión IDS, Machine Learning

1 Introducción

Una **Red Definida por Software - Software Defined Network SDN** es un paradigma de red alternativo a la red tradicional. A cambio del control distribuido, embebido en cada dispositivo de la red tradicional, SDN propone un esquema de control centralizado y separado de los dispositivos de enrutamiento, independiente del software propietario. SDN propone una arquitectura de red compuesta de tres planos, donde el plano central es el plano de control, encargado de la administración de la red. El plano de control se conecta con el plano de datos, compuesto por los dispositivos de red mediante lo que se denomina conexión southbound, y se conecta con el plano de aplicaciones mediante la conexión northbound. El controlador SDN es programable y configurable, los dispositivos de red reciben y redirigen el flujo de datos según las instrucciones del controlador. Desde el punto de vista de seguridad de la red, el controlador centralizado SDN ofrece la ventaja de una visión integral de la red y la flexibilidad de programar mediante reglas el comportamiento de toda la red. Adicionalmente a las múltiples amenazas a la seguridad que tiene una red estándar, el desa-

fío en temas de seguridad de una red SDN está en su naturaleza distribuida. Cada nodo es un punto vulnerable, el controlador representa un punto de falla individual y puede ser el blanco de ataques del tipo man-in-the-middle, en ambas direcciones southbound and northbound, en su conexión con el plano de datos y aplicaciones respectivamente.

Una de las herramientas fundamentales en seguridad de redes es el **Sistema de Detección de Intrusión - Intrusion Detection System IDS** que se localiza, según la estrategia de seguridad, en el borde de la red, en servidores o en hosts. La detección de un amenaza a la seguridad de la red es un problema doble, primero la identificación de flujo sospechoso, y segundo la clasificación del flujo en normal o anómalo.

El diseño de un IDS debe resolver cuestiones como dónde localizar el IDS, cómo detectar las amenazas, qué tipo de amenazas puede detectar, qué herramientas de detección son más apropiadas para detectar las amenazas seleccionadas, etc. Para responder algunas de estas preguntas, este artículo presenta en la **Sección 2** un resumen de los tipos de IDS, según el enfoque de detección; también se describen problemas comunes del diseño y modelos de aprendizaje relacionados con Machine Learning. En la **Sección 3** se presenta una revisión de herramientas de IDS haciendo foco en Machine Learning, agrupadas según el modelo de aprendizaje. El artículo concluye con recomendaciones de diseño generales basadas en la literatura revisada.

2 IDS: Diseño

La primera parte del diseño de un IDS debe considerar el tipo de amenaza a la red, que está directamente relacionada con el método de detección y la herramienta de clasificación usada para determinar si el flujo de datos es normal o anómalo. En particular los IDS inteligentes basados en algoritmos de **Aprendizaje Inteligente o Machine Learning ML** necesitan un conjunto de datos de muy buena calidad, que significa un número elevado de datos de muestreo lo suficiente como para dividir el conjunto de datos en datos para entrenamiento, testeo y validación del modelo inteligente, además de estar los datos identificados o rotulados para caracterizar correctamente el flujo de datos.

Problema de dimensionalidad. La caracterización del flujo de datos puede realizarse con pocos o muchos parámetros, mientras más parámetros identifiquen el flujo de datos, mejor será la descripción del flujo. El costo de la alta definición del flujo implica el crecimiento exponencial del conjunto de datos. Hay que considerar que no todas las características del flujo son significativas para la detección de un ataque a la red. Algunas características del flujo de datos cambian significativamente cuando la red está bajo ataque, como es demostrado en [1] indicando que algunas características tienen mayor capacidad de detección, como es el caso de dirección IP origen, puerto origen/destino, tipo de protocolo, etc. Las características del flujo de datos pueden ser descriptas por parámetros estadísticos como la tasa de ocurrencia de paquetes TCP. También pueden ser ponderadas por la entropía o contenido de información, dando lugar al análisis de la entropía por ejemplo, del puerto origen/destino, dirección IP origen, o la entropía condicional, por ejemplo entropía de la dirección IP origen dado el puerto destino, etc. Este método ayuda a reducir el número de características de interés y por consiguiente el número de datos. Los diferentes enfoques en la selección del conjunto de datos se resumen en la siguiente Sección, y llevan a clasificar al IDS según el grado de información que contiene de la amenaza, en amenazas conocidas o desconocidas, y por el modo en que la amenaza es detectada, mediante anomalías en el flujo de datos, o mediante abuso de la red [2].

Dataset de amenazas. Para el entrenamiento, test y validación de los modelos inteligentes es necesario el uso de gran cantidad de datos con información de cómo ocurren los ataques a la red. Dado que es muy difícil registrar datos de los ataques, existen variados proyectos tendientes a diseñar métodos para recolectar este tipo de datos.

KDD-99 Knowledge Discovery and Data Mining es una competencia destinada a la creación de un IDS de red y la simulación de varios ataques en una red militar [3]. Los ataques principales registrados corresponden a cuatro categorías. El ataque DoS Denial-of-Service destinado a agotar recursos del equipo y ancho de banda, impidiendo el servicio normal. El ataque de Probe o Scanning destinados a obtener el reconocimiento de equipos para luego ejecutar ataques de mayor impacto. El ataque User-to-Root (U2R) son un conjunto de exploits diseñados para lograr el acceso root de un equipo desde un usuario local sin privilegios. El ataque Remote-to-Local User (R2L) usado para ganar acceso local a la máquina vulnerable desde la red, y desde esa máquina comprometida lograr tener acceso a la red interna mediante una escalada de privilegios.

KDD-99 y el subset NSL-KDD son dos de las bases de datos de ataques más usadas para diseñar y comparar modelos de IDS, a pesar que los ataques fueron generados en entorno artificial. Algunos autores sostienen que las características de ataque de KDD-99 no son suficientes para distinguir un ataque de otro, pero pueden usarse como indicadoras de la presencia de un ataque [4].

Un estudio comparativo de diversas bases de datos de ataques incluye comparativos de performance no tan solo de modelos de IDS sino también de la validez de las bases de datos de ataques [3]. KDD-99 pertenece al tipo de base de datos que contienen características del tráfico. Otras bases de datos tales como las del proyecto CAIDA y DARPA, contienen rastros de datos dejados en la red por los ataques, y otras bases de datos usan técnicas adversarias como CCRC para el diseño de IDS de red o NIDS.

2.1 Tipos de IDS según el grado de conocimiento de las amenazas

Las amenazas a la red se pueden clasificar en conocidas o desconocidas según el grado de información que se tenga. Las amenazas desconocidas son por ejemplo ataques del día cero (zero-day attacks), ataques indefinidos, que en general no están asociados a vectores claramente identificados, que puedan ser detectados por el comportamiento del flujo. Por el contrario, las amenazas conocidas pueden ser detectadas por medio del análisis de especificaciones, datos estadísticos, análisis de protocolos, y por búsqueda en base de datos especializadas. Mishra et al. en su investigación de ML aplicada a IDS presentan cuatro categorías de IDS basados en ML [4]. Estas categorías están formadas por clasificadores individuales y clasificadores múltiples, utilizando el set de datos completo e incompleto respectivamente. Las categorías múltiples permiten el uso de más de un modelo de detección, dando lugar a combinaciones de modelos de detección de anomalías y abuso de la red, basados en diferentes algoritmos de ML. Según los autores este enfoque mejora la tasa de detección y reduce los falsos positivos.

Amenazas Desconocidas

IDS para detección de anomalías y mal uso de la red. Las anomalías en el flujo de la red pueden ser detectadas mediante la presencia de outliers o datos fuera del rango, o mediante patrones fuera de lo normal que impliquen un comportamiento inusual de la red. En ámbito de ML el comportamiento normal es registrado durante la fase de aprendizaje para extraer el perfil del flujo, que luego es comparado con el flujo están-

dar de la red para determinar la presencia de amenazas. Este enfoque tiene muy buena performance en la detección de anomalías pero da una alta tasa de falsos positivos [1]. La detección y selección de características particulares puede realizarse mediante Auto-codificadores - Autoencoders, redes neuronales artificiales de tres capas consideradas Deep Neural Networks por la gran cantidad de nodos que pueden contener, capaces de aprender en forma no supervisada las características principales del flujo de datos [5] [6]. Este método es útil cuando el conjunto de características es grande, complejo y desconocido.

Amenazas Conocidas

IDS basado en la firma - Signature-based IDS busca patrones relacionados con amenazas registradas en bases de datos denominadas listas negras, o en registros históricos de ataques [7]. Este método presenta una buena tasa de detección de ataques bien conocidos y definidos, pero no detecta ataques desconocidos o no definidos en las listas.

IDS basado en especificaciones busca características del flujo que indiquen anomalías de algún protocolo en particular [8]. Por ejemplo la búsqueda de anomalías relacionadas con el uso de direcciones IP o puertos de origen/destino, rótulos de flujo, payload y características en general que puedan indicar un flujo sospechoso [9].

IDS basado en estadísticas busca variaciones en los parámetros del flujo que se asocien con un comportamiento anormal [1]. Este método es útil en la detección de ataques en el perímetro de la red, con mejor performance en la observación focalizada en una dirección o puerto en particular, de lo contrario puede dar falsos positivos.

IDS basado en el análisis de protocolos stateful, donde se mantiene registro de los estados de la sesión, busca anomalías en las sesiones del protocolo [10]. Este tipo de análisis pormenorizado implica una carga extra para el IDS y su administración, no obstante ofrece un muy buen entendimiento del comportamiento de la red y de las causas de mal funcionamiento, ayudando a discriminar entre amenazas reales y falsas. En la práctica el diseño de un IDS requiere de conocimiento y análisis de protocolos, como conocimiento básico, que puede ser combinado con otros métodos de detección más prácticos.

2.2 Modelos de Aprendizaje: enfoque en Machine Learning

Machine Learning construye un modelo matemático basado en modelos probabilísticos y algoritmos para llevar a cabo la clasificación del flujo de datos y la predicción de comportamiento futuro sin requerir instrucciones explícitas. La máquina es entrenada con datos de muestra para encontrar los parámetros que la definen, a esto se le denomina aprendizaje. Las tres categorías principales de aprendizaje son aprendizaje supervisado, no supervisado y aprendizaje por refuerzo o reinforcement learning.

Modelos de aprendizaje

Aprendizaje Supervisado - Supervised Learning SL. Mediante este método se entrena a la máquina para aprender una función, que mapea la entrada en la salida de la red neuronal, basada en muestras de datos de los pares entrada-salida. Este método requiere suficiente cantidad de datos identificados o rotulados para el aprendizaje, de lo contrario no se garantiza una buena detección.

Aprendizaje No-Supervisado - Unsupervised Leanring UL. Basado en el aprendizaje asociativo o aprendizaje de Hebbian, el aprendizaje no supervisado se usa cuando los datos no están identificados o rotulados, y permite encontrar patrones en el conjunto de datos, aprendiendo la relación probabilística existente entre ellos [11]. El aprendizaje no-supervisado sigue un proceso de auto-organización para construir un modelo de densidad de probabilidad de la entrada. Este método tiene la desventaja de encontrar mínimos locales, impidiendo la generalización y resultando en una pobre detección de anomalías.

Aprendizaje por Refuerzo - Reinforcement Learning RL. RL asocia situaciones con acciones y el proceso de aprendizaje consiste en la búsqueda y ejecución de acciones que maximicen una recompensa acumulativa [12]. La recompensa resultante de una acción en particular afecta las acciones futuras que se tomen. En general, las dos características fundamentales de RL son prueba y error, y recompensa futura. A diferencia del aprendizaje supervisado, RL no necesita pares de entrada/salida para el entrenamiento de la red. Comparado con el aprendizaje no-supervisado, RL no requiere acciones sub-óptimas para ser mejoradas. Una variante la ofrece el aprendizaje por refuerzo profundo, Deep Reinforcement Learning, que soluciona el problema de convergencia y complejidad, mediante la selección de una arquitectura con un número ilimitado de capas de determinado tamaño, heterogéneas en su diseño, de forma tal que cada capa aporte un grado más de refinamiento de la solución al problema.

Aprendizaje por Refuerzo basado en la Teoría de Juegos - Game theory-based RL. Ofrece un enfoque diferente, donde los elementos de la red de datos y los potenciales atacantes son considerados jugadores, con información incompleta respecto a las intenciones del resto de jugadores y compitiendo por recompensas o puntaje.

Paradigmas de Aprendizaje: Modelo Generativo vs Modelo Discriminativo

Es importante entender los dos enfoques principales de aprendizaje de los modelos en relación con la tarea a desempeñar, ya sea identificación o clasificación de amenazas, y el conocimiento que se tenga de las relaciones probabilística de los datos. Los dos paradigmas de aprendizaje fundamentales son el modelo generativo y el discriminativo, según la dependencia probabilística entre la variable observada o de entrada X y la variable rótulo clasificador o de salida Y [13].

Para emplear un modelo generativo es necesario tener conocimiento explícito del espacio de datos, su estructura y la relación probabilística entre los datos. El clasificador generativo usa la probabilidad conjunta de X e Y: $P(X,Y)$, y realiza las predicciones basado en la regla de Bayes para calcular la probabilidad condicional de Y dado X: $P(Y/X=x)$, para luego seleccionar el rótulo de mayor probabilidad. Estas reglas son un buen descriptor de este tipo de problema y ayudan a obtener datos más refinados a partir del set original, por lo que este modelo es útil para generar nuevas muestras [14]. El modelo discriminativo usa directamente la probabilidad condicional de la salida Y dada la entrada X: $P(Y/X=x)$ para obtener el conjunto de probabilidades de los rótulos. Esto hace que el modelo del clasificador discriminativo llegue al resultado en forma directa, evitando errores del modelado intermedio, por lo que está optimizado para discriminar y predecir.

Ambos modelos presentan diferente performance, el modelo de aprendizaje discriminativo presenta menor error asintótico, por lo que puede usarse con muy buena performance con un número elevado de datos. Por otro lado el modelo generativo alcanza el error asintótico más pronto, por lo tanto su mejor uso es para un set pequeño de datos [13].

Ejemplos de modelos usados en la generación de anomalías es el modelo Hidden Markov o Markov oculto, redes Bayesianas (e.g. Naive Bayes, modelo Autoregresi-

vo), máquina de Boltzmann (e.g. máquina restricta de Boltzmann, Deep belief Network), redes adversarias generativas. Ejemplos de modelos discriminativos son k-Nearest Neighbours, regresión Logística, Support Vector Machine, modelos de Markov de máxima entropía, redes neuronales, etc.

En la siguiente Sección se analizan los principales modelos de aprendizaje con ejemplos aplicados en la implementación de Sistemas de Detección de Intrusión en entornos de redes SDN.

3 IDS: revisión según el modelo de aprendizaje

3.1 IDS & Aprendizaje Supervisado

El problema de detección de intrusión tiene dos partes, como ya se mencionara, la selección de características representativas del flujo de datos y la discriminación entre flujo normal o anómalo. El aprendizaje supervisado es de fundamental ayuda en la etapa de selección de características, donde dada la observación de un set de datos se llega a una conclusión acerca de su clasificación. Para ello es necesario contar con un set de datos de entrenamiento, y tener un conocimiento bastante detallado del set de datos en estudio. A continuación se detallan algunos de los métodos supervisados empleados en la clasificación de flujo de datos.

k-vecinos más cercanos - k-Nearest Neighbour k-NN. Método usado para clasificación supervisada y regresión encontrando relaciones entre el dato de entrada y el conjunto de k-vecinos [8]. El resultado de la clasificación es la asignación a cada dato de entrada de una función miembro de una clase definida por los k-vecinos más cercanos. Debido a la característica de seleccionar k-vecinos, el método es sensible a la estructura de datos y al ruido producido por datos irrelevantes. Para sortear este problema es que se suele ponderar a los vecinos en forma inversamente proporcional a la distancia.

Support Vector Machine SVM. Método de aprendizaje supervisado usado para clasificación y regresión. La estrategia de SVM consiste en encontrar hiper-planos para separar la entrada en diferentes categorías, definidas por sus respectivas funciones de centro o funciones de kernel. Estas funciones estiman el margen o distancia de cada categoría a los respectivos hiper-planos. El objetivo es encontrar el algoritmo que maximice el margen entre los planos para una efectiva separación de clases. SVM es aplicado tanto a problemas de clasificación lineales como no-lineales [15], y en caso de una difícil separación de los respectivos kernels, el método permite el uso de los denominados límites blandos o soft-boundaries. SVM aplicado en la detección de intrusión en redes, es eficiente en la detección de ataques dado que se trata de un problema con dos clases, ataque o ausencia de ataque, separados por un solo hiper-plano. No obstante la performance de SVM decrece en la medida en que la dimensión del espacio aumenta, como es el caso de extracción de características del flujo de entrada donde, si se considera un gran número de características, el algoritmo falla al encontrar los hiper-planos adecuados [16].

Árbol de Decisión - Decision Tree DT. Dada una característica de entrada, el Árbol de Decisión resuelve la clase a la cual pertenece mediante inferencia [17]. Siguiendo una conjunción de reglas, que constituyen las ramas del árbol, se toman decisiones para llegar a las clases a las cuales pertenece la entrada, definidas por los nodos hoja del árbol. En particular en el caso de clasificación de tráfico es necesario contar

con un conocimiento detallado del tráfico para definir la estructura del árbol capaz de identificarlo, es un problema del tipo white box. En base a las reglas de identificación del tráfico se van construyendo las ramas del árbol que llevarán a la clasificación final. Esta estructura detallada del árbol es la causa fundamental para que los DT tiendan a sobre-adaptarse al problema (overfit), presentando inexactitudes en el reconocimiento de nuevas variaciones en el tráfico. Otras de las desventajas de los DT es que son inestables, pequeños cambios en la entrada implican grandes cambios en la estructura del árbol para que pueda identificar con exactitud la entrada, esto implica agregar complejidad al árbol. Para mejorar la exactitud de los DT se usa Random Forest de árboles de decisión.

Random Forest RF. Random Forest es un conjunto de árboles de decisión trabajando como clasificadores y compitiendo por la maximización de votos. RF son resilientes al ruido y a elementos fuera de rango. A diferencia de los árboles de decisión, tienden a no sobre-adaptarse al problema en particular, presentan varianzas grandes y sesgos o bias pequeños. Cuando se usan RF promediando múltiples árboles de decisión, la varianza disminuye a expensas de un incremento pequeño en el sesgo. El entrenamiento de RF a través de bagging es un caso de promediación del modelo, dado un conjunto de entrenamiento, el bagging genera un nuevo set de datos mediante el muestreo con reemplazo [8]. Esto genera n-muestras que ingresan a RF creando n-diferentes árboles. En materia de IDS, RF tiene menor exactitud comparado con SVM, no obstante tiene mejor precisión en la detección, debido a que se puede incrementar la profundidad de los árboles [16].

Extreme Learning Machine ELM. Una Máquina de Aprendizaje Extremo es una red neuronal feed-forward compuesta por una o múltiples capas. Los parámetros de las capas internas son asignados en forma aleatoria o aprendidos en un solo paso y nunca actualizados [18]. El método de entrenamiento es costoso desde el punto de vista computacional pero más rápido que el empleado en backpropagation, que retropropaga los errores de la salida hacia capas anteriores. Estas características hacen que ELM sea un buen método para generalizar con capacidad de manejar gran cantidad de datos en la entrada, dando resultados comparativamente superiores a Support Vector Machine y Random Forest [16].

Comparando métodos supervisados empleados para la clasificación de flujo de datos, un estudio muestra el resultado de un IDS en redes SDN donde se compararon los métodos Neural Networks NN, Linear Discriminant Analysis LDA, Decision Tree DT, Random Forest RF, Linear Support Vector Machine SVM, k-Nearest Neighbour k-NN, Naive Bayes NB, Extreme Learning Machine ELM, AdaBoost, Random Undersampling Boost RUSBoost, LogitBoost and BaggingTrees [8]. Según el estudio, Random Forest y Decision Trees son los métodos con mejor performance en términos de exactitud, presentando RF menor tasa de falsos positivos, cuando se usó el total de 41 características de la base de datos de prueba. El tercer método sin boosting con mayor performance fue k-NN. También se realizaron pruebas con un grupo reducido pre-seleccionado de seis características y el método de mayor exactitud de detección fue RUSBoost, y el de menor tasa de falsos positivos fue RF. Estos resultados demuestran que la detección mejora cuando se combinan con técnicas de muestreo con reemplazo como boosting o bagging (Bootstrap Aggregating) para mejorar la estabilidad y exactitud [19].

3.2 IDS & Aprendizaje No-supervisado

Cuando los datos de prueba no están clasificados o rotulados, no es posible encontrar una correspondencia entre el dato y su respectivo rótulo clasificador. En estos casos el objetivo del aprendizaje no-supervisado es encontrar estructuras comunes en el conjunto de datos y agruparlos por características comunes. Esta es la forma de encontrar los valores más significativos para identificar el flujo de datos, y de reducir el tamaño del conjunto de datos. La detección de elementos fuera de rango o outliers puede indicar un posible ataque a la red, como en el caso de algunos ataques de fuerza bruta. Cuando las amenazas no pueden identificarse como outliers, es necesario detectar patrones de comportamientos anómalos, que es la base de los métodos de clustering como k-means, y de redes neuronales entrenadas en forma no-supervisada, como Self-Organizing Map, Autoencoders, Deep-Belief Networks y Generative Adversarial Networks.

k-means. Es un método de clustering que permite cuantizar vectores, donde los datos son divididos en clusters agrupados por vecindad al valor medio más cercano. Los valores medios iniciales del cluster pueden ser elegidos en forma aleatoria y se van actualizando iterativamente hasta converger a un valor final. Los clusters resultantes son similares en extensión, forma y convergencia al valor óptimo local. Los datos de muestra no necesitan estar identificados por rótulos, el método de k-means clustering es sensible a ruidos y datos fuera de rango, dado que estos valores se consideran en el cálculo del próximo valor medio en la iteración. El método se usa para detectar anomalías asociadas con amenazas a la red tales como DDoS. Para acelerar la convergencia del método se pueden pre-seleccionar mediante heurística los valores medios iniciales [1]. Esto lo convierte en un método semi-supervisado, además de la convergencia más rápida tiene mayor exactitud comparado con la forma de seleccionar los valores medios en forma aleatoria.

Self-organizing Map SOM. SOM es un mapa de neuronas ordenadas topológicamente, es decir valores de entrada cercanos son mapeados juntos en la salida para conservar la interrelación existente [20]. Este mapa permite reducir la dimensionalidad de los datos de entrada mediante una red de nodos entrenados toman los valores de mejor aproximación - Best Matching Unit BMU, que representa la vecindad del nodo al valor de entrada. Este entrenamiento competitivo es más eficiente que otros entrenamientos de redes neuronales basados en la corrección de errores, que pueden quedar atrapados en valores mínimos locales. SOM con pocos nodos se comporta en forma similar a k-means, no obstante la gran ventaja es en espacios de alta dimensionalidad, con gran cantidad de nodos.

En el campo de seguridad de redes, SOM ofrece una reducción de dimensionalidad del espacio de entrada, simplificando el análisis ulterior clasificando el tráfico en normal o anómalo. Por otro lado, el mapa de BMU simplifica la visualización de los datos del tráfico de la red, ayudando a identificar comportamientos fuera de los patrones normales [20].

Para la detección de DDoS en redes SDN, se entrenó una red neuronal SOM para simplificar el análisis de flujo de datos, usando un mapa de neuronas cuyos nodos están definidos por vectores de peso de cinco dimensiones [20]. Estas dimensiones están constituidas por cinco variables seleccionadas por su entropía o contenido de información, observadas tanto en tráfico normal como bajo ataque. Estas variables son, dirección IP origen, puerto origen y destino, tipo de protocolo y número de paquetes. Estos vectores de entrenamiento representan el estado de la red en un período determinado. En la fase de clasificación binaria del tráfico, tráfico normal o DDoS, se usaron dos métodos priorizando diferentes aspectos. El primer método usó k-NN para

priorizar la exactitud, y el segundo priorizó la agilidad de selección mediante el análisis de la distancia existente entre los vectores de entrada a un vector representativo del flujo normal de datos, calculado en base a los valores medios de cada variable. Ambos métodos mostraron una alta tasa de detección y agilidad debido a la pre-selección de las variables de interés. En este caso particular cada instancia de tráfico se analizó en forma independiente removiendo la dependencia cronológica, que en algunos tipos de ataques puede ser de gran valor en la detección.

La arquitectura de las redes SOM presenta dos desventajas fundamentales, la primera es que la estructura de la red es fija y debe ser conocida de antemano en base a la experiencia del caso. La segunda desventaja es que no refleja jerarquías en la estructura de datos a representar, por lo que SOM no es flexible para representar diferentes niveles de profundidad en los datos que mapea [21]. En el caso particular de sistemas de detección de intrusión en SDN, las reglas de clasificación de paquetes del controlador se presentan en diferentes grados de abstracción o profundidad para poder definir qué acción se debe tomar. Para solucionar el problema de jerarquías de información en la estructura de datos es que se diseñó la arquitectura Growing-Hierarchical Self-Organized Maps GH-SOM [21]. Esta arquitectura forma un árbol de clusters partiendo de los clusters formados en el SOM inicial.

El SOM inicial, que es una representación neuronal de baja dimensionalidad del espacio de entrada, se descompone en clusters. Cada cluster del SOM inicial se analiza para determinar si satisface las condiciones para adoptar una regla de clasificación del controlador SDN [22]. Si hace falta mayor información, cada cluster se descompone sucesivamente en un nuevo SOM con mayor detalle y es analizado nuevamente para determinar la correspondencia con la reglas. Este procedimiento de descomposición pormenorizada de cada cluster termina cuando se encuentran las reglas necesarias y suficientes para que el controlador SDN tome una acción.

De esta manera GH-SOM resuelve el problema de encontrar una estructura adecuada de antemano, pues va generando nuevos SOMs de los clusters según sea necesario. Por otro lado GH-SOM refleja jerarquías de información como son las reglas de clasificación de paquetes del controlador SDN.

Autoencoder. Es una red neuronal de tres capas, entrenada de forma no-supervisada La capa de entrada aprende la representación del set de datos ignorando el ruido, por lo tanto reduciendo la dimensionalidad de la entrada, que se ve reflejada en la capa oculta. La salida de la red se encarga de reconstruir la entrada de una forma bastante aproximada, en base a su representación reducida de la capa oculta.

Estas características del Autoencoder han sido aprovechadas para realizar un estructura de stacked Autoencoders para la detección de amenazas, entrenando los Autoencoders con datos o eventos considerados como normales [23]. Los Autoencoders al reproducir de forma fidedigna los eventos normales, detectan fácilmente flujo anómalo de datos por desviación de la normalidad. La función de stacked-Autoencoder es lograr niveles de resolución de datos mayores. Esto facilita la detección de amenazas con mayor precisión, en forma automática y aliviando las tareas de análisis de alertas por parte de personal especializado.

3.3 IDS & Aprendizaje por Refuerzo - Reinforcement Learning RL

La estrategia de Aprendizaje por Refuerzo es aplicable en redes SDN como parte del entrenamiento de un orquestador para detectar y prevenir ataques [24]. Se obtienen métricas de la red para agruparlas en perfiles de comportamiento asociados a un conjunto acciones que contrarrestan las anomalías. Estas acciones son valorizadas por un sistema de recompensas para ser luego seleccionadas por su efectividad mediante

aprendizaje por refuerzo. Las acciones más efectivas definen las políticas del controlador que son almacenadas en una tabla de estado-acción. Cuando se detecta una anomalía, el orquestador y el controlador activan las correspondientes políticas en defensa. Este sistema muestra buena performance en el manejo de problemas, pero el número de entradas en la tabla de estado-acción se incrementa significativamente ante pequeños incrementos en el tamaño de la topología SDN, considerando que cada dispositivo que se agrega tiene asociado un número considerable de reglas.

3.4 IDS & Deep Learning

Restricted Boltzmann Machine RBM. Máquina de Boltzmann restringida es un modelo clasificador generativo de aprendizaje no-supervisado, basado en una red neuronal recurrente y estocástica, compuesto de dos capas: la capa de entrada visible, y la capa de salida oculta. Cuando los datos ingresan en la capa de entrada, el valor de los nodos de la capa oculta es definido por la función de distribución de probabilidad de energía de Boltzmann [25] [26]. A diferencia de las máquinas de Boltzmann estándar, donde todos los nodos están completamente conectados, en RBM tanto los nodos de la capa visible como los de la capa oculta están conectados a todos los nodos de la otra capa, pero no conectados entre sí. El aprendizaje en una máquina RBM implica encontrar la distribución de Boltzmann que mejor represente los datos de entrada. Dada la naturaleza del aprendizaje de la máquina RBM, por medio de la función de distribución de probabilidad, el ruido y los valores fuera de rango son eliminados debido a su baja probabilidad. Esta característica da a la RBM muy buena performance en aplicaciones de extracción de características [9] [11], y en tareas de discriminación entre flujo normal y amenazas [25].

Deep Belief Network DBN. Red de Creencia Profunda basada en un modelo generativo de aprendizaje no-supervisado, es una clase de Boltzmann Machine. La DBN está compuesta por una secuencia de RBM en la entrada para la extracción de características, y una red neuronal entrenada con backpropagation en la salida como clasificador, conformando un modelo particular BP-DBN [27]. RBM extrae las características de la entrada mediante un aprendizaje no-supervisado, luego estas características forman la entrada de la red Backpropagation Netowrk BPN, entrenada en forma supervisada. El algoritmo de backpropagation ajusta los pesos de los nodos de la red neuronal mediante la retropropagación de errores en forma sucesiva, a las capas anteriores en la red RBM, hasta alcanzar los pesos óptimos. La performance de la red DBN aumenta con el número de interacciones propias de la red RBM. El aprendizaje no-supervisado de la red DBN hace que este tipo de red sea apropiada para manejar gran cantidad de variables en la entrada, por lo que la red DBN ofrece muy buena performance en espacios de gran dimensión cuando se trata de extracción de características del flujo de la red [6]. La red neuronal de Memoria Asociativa DBN, un modelo alternativo de DBN, ofrece similares características, donde la red neuronal de salida es entrenada mediante memoria asociativa, resultando con buena performance como clasificador de flujo en presencia de un espacio de grandes dimensiones [28].

3.5 IDS & Teoría de Juegos

Los nodos en una red neuronal pueden ser considerados como jugadores compitiendo entre sí. Estos nodos no poseen un conocimiento completo del estado del resto de nodos, al menos que dicho estado sea comunicado al resto de los nodos, incluso en este caso la actualización de la información no siempre es en tiempo real. Por lo tanto si los nodos desconocen el estado del resto de nodos, el juego no es cooperativo, de-

nominándose juego no-cooperativo [29]. La mejor estrategia recibe el puntaje más alto, en el contexto de IDS algunos jugadores son defensores y otros atacantes de la red. Por lo tanto, en un sistema de defensa, el nodo que detecta y rechaza un ataque recibe la puntuación máxima.

La estrategia para detectar un ataque depende del conocimiento que cada nodo tenga de su propio estado y de los estados pasados, además del estado del resto de nodos y los respectivos valores pasados. El sistema de puntaje se forma en base al conjunto de estados del sistema, políticas, y acciones derivadas de dichas políticas. Las acciones son evaluadas de acuerdo al resultado de la transición de estado que producen [30]. Una función computa el puntaje con el objetivo de maximizar la recompensa a largo plazo. Este resultado es almacenado en una tabla conteniendo estados, acciones y puntaje. Este escenario de información incompleta, acciones y retribuciones conforman un caso de conocimiento adaptivo, que se resuelve mediante aprendizaje por refuerzo, denominado **teoría de juego mediante aprendizaje por refuerzo**.

La teoría de juegos ofrece ventajas para modelar un IDS en entorno SDN, como es el caso de redes SDN virtualizadas con múltiples controladores [31]. En este caso los jugadores son: hipervisores, redes SDN virtualizadas, controladores virtuales, dispositivos de enrutamiento y atacantes. El hipervisor a cargo de vigilar y defender la red SDN virtualizada tiene recursos limitados e información incompleta referente al resto de nodos. En la jerga de la teoría de juegos ese conocimiento limitado es lo que se denomina "creencia" del jugador y es modelada por una función de probabilidad. La regla de Bayes es una de las funciones que mejor se adapta al problema de una red de datos, debido a que el escenario cambia de un instante a otro, evolucionando dinámicamente. Cuando un switch en una red SDN es comprometido por un ataque, el flujo comprometido es enviado a los controladores, y eventualmente el ataque se transmite al resto de los controladores amplificando el ataque original. Estas condiciones se adjuntan al sistema de puntaje original que refleja el estado de los nodos. El resultado del juego es la recomendación de la distribución de recursos que mejor ayude a la tarea de monitoreo realizada por el hipervisor. El puntaje final indica la vulnerabilidad de los switches asociada a la probabilidad de un ataque de intrusión a la red. Este escenario es modelado por un juego Bayesiano dinámico no-cooperativo, usado para asignar dinámicamente los recursos de la red para mejorar el sistema defensivo [31].

La teoría de juego ha sido usada también para el caso de orquestación de una red SDN, por ejemplo desarrollando un juego de planificación o scheduling dinámico basado en un juego del tipo Bayesian-Stackelberg [32]. En este tipo de juego los jugadores son de diferente tipo (atacantes y defensores), y la incertidumbre en las estrategias de los jugadores se modela mediante sus respectivas funciones de probabilidad. La estrategia Stackelberg se da cuando el atacante elegido aleatoriamente, observa los movimientos de los defensores, para aplicar una estrategia de ataque que maximice sus beneficios. La propuesta de defensa es implementar un scheduler que indique la rotación de diferentes controladores heterogéneos para dificultar el efecto de ataques remotos. Modelos teóricos de juego se pueden desarrollar para el análisis cuantitativo de los ataques a la seguridad [33], o para evaluar el riesgo de la seguridad [34].

Modelos Adversarios. Una revisión comparativa de modelos adversarios usando ML y empleados en la detección de intrusión y malware, presenta las estrategias más comunes empleadas en el diseño de detección de intrusión [35]. Estas estrategias de detección incluyen por ejemplo la inclusión de muestras de ataques en el set de datos de entrenamiento; el uso de una red neuronal para entrenar otras con la opción de agregar ruido en el entrenamiento, que permita ampliar la capacidad de detección de la defensa; uso de características comprimidas en el caso de set de datos de gran dimensionalidad, que permita una definición de buena calidad del ataque, favoreciendo la tasa de verdaderos positivos, al mismo tiempo haciendo manejable la dimensionalidad de los datos de entrenamiento. También se presentan estrategias como el enmas-

caramiento de gradiente, que elude la predicción que pueda hacer un atacante para determinar la pendiente de entrenamiento del sistema de defensa. El atacante interroga al clasificador para estimar el gradiente y producir nuevos ataques que no sean detectados. Esta característica contrarresta una de las vulnerabilidades de los sistemas de defensa que usan técnicas de ML, cuyo aprendizaje, precisamente por estar basado en datos, son susceptibles a ataques maliciosos [36]. En este artículo también se encuentra la dificultad de comparar resultados ante la disparidad de parámetros de performance usados en los artículos revisados, salvando este inconveniente mediante el listado de trabajos y la performance usada en cada caso.

Redes Adversarias Generativas - Generative Adversarial Networks GAN. En una red GAN hay dos redes compitiendo entre sí, la generativa y la discriminativa. Mientras que el objetivo de la red generativa es engañar a la red discriminativa presentando datos similares a los reales, el objetivo de la red discriminativa es determinar qué datos son los reales [37]. La red generativa continúa produciendo nuevos datos tratando de maximizar el error cometido por la red adversaria discriminativa.

En el campo de detección de anomalías se emplean redes GAN dada su capacidad de manejar gran cantidad de datos complejos. Para ello se ha propuesto el método de detección de anomalías Adversarially Learned Anomaly Detection ALAD basado en redes GANs, destinado a derivar características aprendidas en forma competitiva y que son usadas para la detección de anomalías [38]. Este método ha alcanzado buena performance en el campo de análisis de imágenes y datos tabulados, quedando abierto el camino para su uso en otras áreas.

IDSGAN. Dado que los registros estándard de datos de amenazas a la red no son suficientes para reflejar variedad y cantidad de ataques, necesarios para entrenar una red neuronal, es que se hace uso de las ventajas de una red GAN con sus dos redes, generadora y discriminadora. La red generadora transforma el tráfico malicioso y el ruido en ataques a la red, generando de este modo variaciones de ataques. La red discriminadora que simula un IDS, compite detectando los ataques con un IDS black-box, esta arquitectura se denominó IDSGAN [39]. El black-box IDS fue entrenado con diferentes algoritmos de ML: Support Vector Machine, Naive Bayes, Multilayer Perceptron, Logistic Regression, Decision Tree, Random Forest y K-Nearest Neighbor. El feedback del IDS black-box ayuda al generador a mejorar la performance de los ataques. La performance de IDSGAN fue superior al resto de IDS black-box y fue mejorada en la medida en que se modificó un mayor número de características de los ataques.

Cycle-GAN. En el caso particular de redes GAN, hace falta para su entrenamiento pares de datos que guarden una correspondencia entre sí. En los casos en que no es posible conseguir estos pares de datos, se puede usar una red Cycle-GAN, que se entrena en base a datos que no se correspondan. El principio de una red Cycle-GAN es capturar ciertas características de la entrada y mapearlas en un dato de salida que no guarde correspondencia con el dato de origen, salvo determinada relación subyacente entre ambos dominios [40]. No obstante hay muchas trasformaciones que cumplirían con esta premisa, por lo que se impone la restricción que se satisfaga el ciclo inverso, dado el dato de salida obtener el dato original de entrada mediante la aplicación de mapeo inverso. Esta restricción se penaliza mediante la denominada pérdida de consistencia del ciclo, que motiva a las redes a competir para minimizar esta pérdida y el resto de pérdidas de la competencia en redes GAN.

La ausencia de correspondencia entre datos de entrada y salida que maneja una red Cycle-GAN ha servido de inspiración para generar anomalías sintéticas a partir del tráfico estándar de la red [41]. Estas anomalías combinadas son la entrada de un IDS

basado Multilayer Perceptron MLP. De este modo, el IDS entrenado con anomalías sintéticas mostró una mejora sustancial comparada con el IDS entrenado con tráfico desbalanceado, incrementando la tasa de detección o recall de ataques del tipo que dejan poca evidencia, o small footprint attacks.

3.6 IDS & Aprendizaje Inteligente Híbrido

Modelo híbrido: Restricted Boltzmann Machine-Support Vector Machine RBM-SVM.
RBM, basado en un modelo generativo, ejecuta la extracción de características usando un algoritmo de aprendizaje supervisado. Estas características más significativas son la entrada o "alimentan" al algoritmo supervisado de SVM, modelo discriminativo, para lograr mejor performance en la detección o recall [11].

Modelo híbrido: Restricted Boltzmann Machine-Deep Belief Network RBM-DBN.
Ambas máquinas se basan en modelos estadísticos generativos. RBM es usado para la extracción de características mediante un algoritmo no-supervisado [11]. Los parámetros generados por RBM son la entrada de DBN, que es entrenada mediante el método de aprendizaje supervisado usando el algoritmo de backpropagation, para mejorar la performance de la red neuronal.

Modelo híbrido: Principal Component Analysis-Backpropagation PCA-BP.
La red neuronal BP se basa en un modelo estadístico discriminativo [11]. PCA es una herramienta estadística con buenas características para explorar el espacio de datos. No obstante no es la mejor herramienta para tareas discriminativas en dominios complejos, o con un elevado número de componentes principales. Por lo tanto este modelo híbrido usa PCA para extraer las principales características no correlacionadas, que son usadas para entrenar la red neuronal mediante el método de backpropagation en forma supervisada.

Modelo híbrido: k-NN / NB / SVM – SOM.
En un estudio destinado a evaluar la performance de métodos combinados, se comparó la performance de métodos con aprendizaje supervisado para clasificar flujo de datos y detectar anomalías en redes SDN, tales como k-Nearest Neighbour k-NN, Naive Bayes NB, Support Vector Machine SVM y Self-Organizing Map SOM y la combinación de cada uno de ellos con SOM [42]. Los métodos híbridos mostraron mejor performance que los métodos individuales, con mayor exactitud en la detección de tráfico anómalo, con mayor tasa de detección o recall, y menor porcentaje de falsos positivos.

Modelo híbrido: wrapper – DT / SVM / KNN / NB.
Un método híbrido de clasificación de tráfico en redes SDN, usó la combinación de filtros basados en parámetros de selección tales como correlación, Chi-square, ANOVA, experiencia, y wrappers; empleando ML para encontrar los datos de mayor significancia que identifiquen el flujo de manera robusta [17]. En este estudio el filtro provee de datos al wrapper seleccionando los de mayor relevancia. Los métodos supervisados usados en el wrapper fueron: Decision Tree C4.5, Support Vector Machine SVM, k-Nearest Neighbour (k-NN) y Naïve Bayes (NB). Los métodos del wrapper de mayor performance fueron DT y k-NN, considerando que este último consumió el mayor tiempo de ejecución de todos los métodos probados.

La **Tabla 1** muestra un resumen comparativo de los modelos de IDS presentados en este artículo, comparando la performance de los modelos según la exactitud, tasa de falsos positivos FPR, y el conjunto de datos usados para el entrenamiento de los modelos.

Tabla 1. Métodos de IDS: resumen de valores de performance

- FPR: tasa de falsos positivos FPR (False Positive Rate)

IDS Objetivos	IDS Modelo	Exactitud %	Exhaustividad %	FPR %	Datos de Entrenamiento
DDoS [1] [1] Hadoop-based hybrid feature selection	SKM-HFS semi-supervised weighted k-means + Hybrid Feature Selection (Hadoop-based FS)	no disponible	99.59	no disponible	1. DARPA DDoS 2. CAIDA ``DDoS attack 2007'' 3. CICIDS ``DDoS attack 2017''
DoS, R2L, U2R, Probing [8] [6] best performance full set	DT RF BaggingTrees RUSBoost AdaBoost LogitBoost KNN …	**99.70** 99.70 99.33 99.19 99.03 98.95 98.14 < 93.00	no disponible	0.31 **0.29** 0.81 0.96 1.03 0.94 1.92 > 4.50	NSL-KDD
DoS, R2L, U2R, Probing [8] [6] best performance reduced set 6 features	RUSBoost AdaBoost BaggingTrees LogitBoost DT RF KNN ELM …	**99.68** 99.56 99.54 99.38 98.37 98.09 98.23 93.16 < 90.00	no disponible	**0.29** 0.38 0.47 0.43 0.31 **0** 3.18 2.25 > 2.5	NSL-KDD
SDN DDoS [9] [7]	Deep-Autoencoder-K-means average cluster values	99.20	no disponible	0.79	KDD99
DoS, R2L, U2R, Probing [11] [9]	PCA-BPNN RBM-SVM RBM-DBN	92.26 96.31 **97.16**	no disponible	0.43 **0.40** 0.48	KDDCup99
Deep Packet Inspection Features FWFS Filter-Wrapper Feature Selection [17] [15]	FWFS wrapper-C4.5 wrapper-k-NN wrapper-NB wrapper-SVM	**99.87** 99.39 98.34 96.71 94.65	no disponible	no disponible	Cambridge Dataset
DDoS [20] [18]	k-NN SOM-k-NN SOM distributed-neurons d=0.95 SOM distributed-center d=0.95	no disponible	**99.05** 98.24 88.23 97.28	2.74 **2.14** 5.08 22.36	NSL-KDD

IDS Objetivos	IDS Modelo	Exactitud %	Exhaustividad %	FPR %	Datos de Entrenamiento
DoS, R2L, U2R	Autoencoder	**65.29**	**98.85**	no dis-	NSL-KDD
Probing	Stacked-	61.13	98.19	ponible	KDDTest
[23]	Autoencoder				
[21]					
DoS, R2L, U2R,	DBN1-RBM L1	74.22	no disponible	3.15	KDDCup99
Probing	DBN2-RBM L2	82.65		2.72	
	DBN3-RBM L3	90.07		1.42	
[27]	DBN4-RBM L4	**93.49**		**0.76**	
[25]	SVM	86.82		1.96	
	NN	82.30		2.31	
DDoS	KNN	81.41	82.24	18.54	NSL-KDD
	NB	83.49	84.03	15.30	
[42]	SVM	86.49	87.55	11.80	
[38]	SOM	93.24	93.37	7.30	
	KNN-SOM	92.49	91.54	8.19	
	NB-SOM	97.19	96.55	3.13	
	SVM-SOM	**98.12**	**97.15**	**2.71**	

4 Conclusiones y Recomendaciones de Diseño

El diseño de un IDS en entorno SDN tiene dos problemas fundamentales a evaluar, el tipo de amenaza para la cual se diseña y la herramienta a emplear en la defensa de la red. La detección de amenazas y su selección requiere un entendimiento profundo del comportamiento de la amenaza y suficiente cantidad de datos que describan el evento. Esto ayuda a seleccionar la herramienta apropiada para la clasificación del flujo de datos en normal o anómalo.

Dataset de Amenazas. Contar con un gran número de muestras de buena calidad es fundamental en ML, en el aprendizaje se requieren diferentes conjuntos de datos para entrenamiento, validación y test. La cantidad de datos ayuda a que el modelo pueda generalizar sin quedar atrapado en valores mínimos locales. Realizar un registro de amenazas no es trivial dada su difícil detección, poca variación y baja tasa de ocurrencia en períodos razonablemente cortos, lo que determina conjuntos de datos desbalanceados. En estos casos las bases de datos de amenazas como KDD-99 identifican un rango de amenazas y son muy útiles como punto de partida en el diseño de un IDS basado en ML. No obstante en el caso de amenazas desconocidas o de comportamiento complejo son mejor modeladas usando aprendizaje automático como Reinforcement Learning, Deep Learning y teoría de juego. La teoría de juego, particularmente redes adversarias generativas, ofrece una forma de generar amenazas sintéticas y un amplio campo a explorar en temas de seguridad de redes.

Modelos de Aprendizaje. Los modelos de aprendizaje generativo y discriminativo son conceptos básicos que ayudan a determinar las herramientas adecuadas de selección y detección de un IDS. La selección del modelo depende del objetivo, generar valores nuevos, dado un conjunto de observaciones o discriminar unos valores de otros. El modelo generativo requiere un conocimiento detallado de los datos para extraer relaciones probabilísticas. El modelo discriminativo requiere del conocimiento de la probabilidad condicional para la clasificación de datos. La performance en relación al número de datos de entrenamiento es otro indicativo a tener en cuenta: entrenamiento con pocos datos, el modelo generativo tiene mejor performance; por el contrario, entrenamiento con gran cantidad de datos, los modelos discriminativos presentan menor error asintótico.

Tendencias en seguridad de SDN. La arquitectura abierta de redes SDN favorece en gran medida el desarrollo de mecanismos de seguridad de redes de datos. Los

avances de ML junto a estrategias como por ejemplo en el campo de blockchain y aprendizaje federado son sólo algunos ejemplos de la variedad de soluciones a problemas complejos de seguridad de redes, y son campos en constante evolución.

Referencias

[1] Y. Gu, K. Li, Z. Guo, and Y. Wang, "Semi-Supervised K-Means DDoS Detection Method Using Hybrid Feature Selection Algorithm," *IEEE Access*, vol. 7, pp. 64351–64365, 2019, doi: 10.1109/ACCESS.2019.2917532.

[2] D. Jankowski and M. Amanowicz, "A method of network workload generation for evaluation of intrusion detection systems in SDN environment," in *2016 International Conference on Military Communications and Information Systems (ICMCIS)*, May 2016, pp. 1–7, doi: 10.1109/ICMCIS.2016.7496575.

[3] I. Homoliak, K. Malinka, and P. Hanacek, "ASNM Datasets: A Collection of Network Attacks for Testing of Adversarial Classifiers and Intrusion Detectors," *IEEE Access*, vol. 8, pp. 112427–112453, 2020, doi: 10.1109/ACCESS.2020.3001768.

[4] P. Mishra, V. Varadharajan, U. Tupakula, and E. S. Pilli, "A Detailed Investigation and Analysis of Using Machine Learning Techniques for Intrusion Detection," *IEEE Commun. Surv. Tutor.*, vol. 21, no. 1, pp. 686–728, Firstquarter 2019, doi: 10.1109/COMST.2018.2847722.

[5] M. Yousefi-Azar, V. Varadharajan, L. Hamey, and U. Tupakula, "Autoencoder-based feature learning for cyber security applications," in *2017 International Joint Conference on Neural Networks (IJCNN)*, May 2017, pp. 3854–3861, doi: 10.1109/IJCNN.2017.7966342.

[6] M. Z. Alom and T. M. Taha, "Network intrusion detection for cyber security using unsupervised deep learning approaches," in *2017 IEEE National Aerospace and Electronics Conference (NAECON)*, Jun. 2017, pp. 63–69, doi: 10.1109/NAECON.2017.8268746.

[7] S. Nanda, F. Zafari, C. DeCusatis, E. Wedaa, and B. Yang, "Predicting network attack patterns in SDN using machine learning approach," in *2016 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN)*, Nov. 2016, pp. 167–172, doi: 10.1109/NFV-SDN.2016.7919493.

[8] M. Latah and L. Toker, "Towards an efficient anomaly-based intrusion detection for software-defined networks," *IET Netw.*, vol. 7, no. 6, pp. 453–459, 2018, doi: 10.1049/iet-net.2018.5080.

[9] A. Dawoud, S. Shahristani, and C. Raun, "A Deep Learning Framework to Enhance Software Defined Networks Security," in *2018 32nd International Conference on Advanced Information Networking and Applications Workshops (WAINA)*, May 2018, pp. 709–714, doi: 10.1109/WAINA.2018.00172.

[10] D. Mudzingwa and R. Agrawal, "A study of methodologies used in intrusion detection and prevention systems (IDPS)," in *2012 Proceedings of IEEE Southeastcon*, Mar. 2012, pp. 1–6, doi: 10.1109/SECon.2012.6197080.

[11] X. Zhang and J. Chen, "Deep learning based intelligent intrusion detection," in *2017 IEEE 9th International Conference on Communication Software and Networks (ICCSN)*, May 2017, pp. 1133–1137, doi: 10.1109/ICCSN.2017.8230287.

[12] R. S. Sutton and A. G. Barto, *Introduction to reinforcement learning*, vol. 135. MIT press Cambridge, 1998.

[13] A. Y. Ng and M. I. Jordan, "On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes," in *Advances in neural information processing systems*, 2002, pp. 841–848.

[14] T. (Tony S.) Jebara, “Discriminative, generative, and imitative learning,” Thesis, Massachusetts Institute of Technology, 2002.
[15] M. M. Adankon and M. Cheriet, “Support Vector Machine,” in *Encyclopedia of Biometrics*, S. Z. Li and A. Jain, Eds. Boston, MA: Springer US, 2009, pp. 1303–1308.
[16] I. Ahmad, M. Basheri, M. J. Iqbal, and A. Rahim, “Performance Comparison of Support Vector Machine, Random Forest, and Extreme Learning Machine for Intrusion Detection,” *IEEE Access*, vol. 6, pp. 33789–33795, 2018, doi: 10.1109/ACCESS.2018.2841987.
[17] F. A. Md. Zaki and T. S. Chin, “FWFS: Selecting Robust Features Towards Reliable and Stable Traffic Classifier in SDN,” *IEEE Access*, vol. 7, pp. 166011–166020, 2019, doi: 10.1109/ACCESS.2019.2953565.
[18] G. Huang, S. Song, J. N. D. Gupta, and C. Wu, “Semi-Supervised and Unsupervised Extreme Learning Machines,” *IEEE Trans. Cybern.*, vol. 44, no. 12, pp. 2405–2417, Dec. 2014, doi: 10.1109/TCYB.2014.2307349.
[19] C. Strobl, J. Malley, and G. Tutz, “An Introduction to Recursive Partitioning: Rationale, Application and Characteristics of Classification and Regression Trees, Bagging and Random Forests,” *Psychol. Methods*, vol. 14, no. 4, pp. 323–348, Dec. 2009, doi: 10.1037/a0016973.
[20] T. M. Nam *et al.*, “Self-organizing map-based approaches in DDoS flooding detection using SDN,” in *2018 International Conference on Information Networking (ICOIN)*, Jan. 2018, pp. 249–254, doi: 10.1109/ICOIN.2018.8343119.
[21] A. Rauber, D. Merkl, and M. Dittenbach, “The growing hierarchical self-organizing map: exploratory analysis of high-dimensional data,” *IEEE Trans. Neural Netw.*, vol. 13, no. 6, pp. 1331–1341, Nov. 2002, doi: 10.1109/TNN.2002.804221.
[22] S.-C. Hung, N. Iliev, B. Vamanan, and A. R. Trivedi, “Self-Organizing Maps-Based Flexible and High-Speed Packet Classification in Software Defined Networking,” in *2019 32nd International Conference on VLSI Design and 2019 18th International Conference on Embedded Systems (VLSID)*, Delhi, NCR, India, Jan. 2019, pp. 545–546, doi: 10.1109/VLSID.2019.00128.
[23] L. Ferrado and M. Cuenca Acuna, “Filtrando eventos de seguridad en forma conservativa mediante deep learning,” presented at the Simposio Argentino de Inteligencia Artificial (ASAI 2016) - JAIIO 45 (Tres de Febrero, 2016)., Nov. 2016, Accessed: Jun. 16, 2020. [Online]. Available: http://sedici.unlp.edu.ar/handle/10915/56884.
[24] L. S. R. Sampaio, P. H. A. Faustini, A. S. Silva, L. Z. Granville, and A. Schaeffer-Filho, “Using NFV and Reinforcement Learning for Anomalies Detection and Mitigation in SDN,” in *2018 IEEE Symposium on Computers and Communications (ISCC)*, Jun. 2018, pp. 00432–00437, doi: 10.1109/ISCC.2018.8538614.
[25] S. Seo, S. Park, and J. Kim, “Improvement of Network Intrusion Detection Accuracy by Using Restricted Boltzmann Machine,” in *2016 8th International Conference on Computational Intelligence and Communication Networks (CICN)*, Dec. 2016, pp. 413–417, doi: 10.1109/CICN.2016.87.
[26] U. Fiore, F. Palmieri, A. Castiglione, and A. Santis, “Network anomaly detection with the restricted Boltzmann machine,” *Neurocomputing*, vol. 122, pp. 13–23, Dec. 2013, doi: 10.1016/j.neucom.2012.11.050.
[27] N. Gao, L. Gao, Q. Gao, and H. Wang, “An Intrusion Detection Model Based on Deep Belief Networks,” in *2014 Second International Conference on Advanced Cloud and Big Data*, Nov. 2014, pp. 247–252, doi: 10.1109/CBD.2014.41.

[28] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A Fast Learning Algorithm for Deep Belief Nets," *Neural Comput.*, vol. 18, no. 7, pp. 1527–1554, Jul. 2006, doi: 10.1162/neco.2006.18.7.1527.

[29] E. C. Amadi, G. E. Eheduru, F. U. Eze, C. Ikerionwu, and K. C. Okafor, "Anti-DDoS firewall; A zero-sum mitigation game model for distributed denial of service attack using Linear programming," in *2017 IEEE 4th International Conference on Knowledge-Based Engineering and Innovation (KBEI)*, Dec. 2017, pp. 0027–0036, doi: 10.1109/KBEI.2017.8324996.

[30] J. Xie *et al.*, "A Survey of Machine Learning Techniques Applied to Software Defined Networking (SDN): Research Issues and Challenges," *IEEE Commun. Surv. Tutor.*, vol. 21, no. 1, pp. 393–430, Firstquarter 2019, doi: 10.1109/COMST.2018.2866942.

[31] R. A. Niazi and Y. Faheem, "A Bayesian Game-Theoretic Intrusion Detection System for Hypervisor-Based Software Defined Networks in Smart Grids," *IEEE Access*, vol. 7, pp. 88656–88672, 2019, doi: 10.1109/ACCESS.2019.2924968.

[32] Z. Lu, F. Chen, G. Cheng, and J. Ai, "A secure control plane for SDN based on Bayesian Stackelberg Games," in *2017 3rd IEEE International Conference on Computer and Communications (ICCC)*, Dec. 2017, pp. 1259–1264, doi: 10.1109/CompComm.2017.8322745.

[33] C. Qi, J. Wu, H. Chen, H. Yu, H. Hu, and G. Cheng, "Game-Theoretic Analysis for Security of Various Software-Defined Networking (SDN) Architectures," in *2017 IEEE 85th Vehicular Technology Conference (VTC Spring)*, Jun. 2017, pp. 1–5, doi: 10.1109/VTCSpring.2017.8108542.

[34] W. He, C. Xia, H. Wang, C. Zhang, and Y. Ji, "A Game Theoretical Attack-Defense Model Oriented to Network Security Risk Assessment," in *2008 International Conference on Computer Science and Software Engineering*, Dec. 2008, vol. 6, pp. 498–504, doi: 10.1109/CSSE.2008.1651.

[35] N. Martins, J. M. Cruz, T. Cruz, and P. Henriques Abreu, "Adversarial Machine Learning Applied to Intrusion and Malware Scenarios: A Systematic Review," *IEEE Access*, vol. 8, pp. 35403–35419, 2020, doi: 10.1109/ACCESS.2020.2974752.

[36] M. Usama, J. Qadir, A. Al-Fuqaha, and M. Hamdi, "The Adversarial Machine Learning Conundrum: Can the Insecurity of ML Become the Achilles' Heel of Cognitive Networks?," *IEEE Netw.*, vol. 34, no. 1, pp. 196–203, Jan. 2020, doi: 10.1109/MNET.001.1900197.

[37] I. J. Goodfellow *et al.*, "Generative Adversarial Nets," 2014.

[38] H. Zenati, M. Romain, C.-S. Foo, B. Lecouat, and V. Chandrasekhar, "Adversarially Learned Anomaly Detection," in *2018 IEEE International Conference on Data Mining (ICDM)*, Nov. 2018, pp. 727–736, doi: 10.1109/ICDM.2018.00088.

[39] Z. Lin, Y. Shi, and Z. Xue, "IDSGAN: Generative Adversarial Networks for Attack Generation against Intrusion Detection," *ArXiv*, 2018.

[40] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks," *ArXiv170310593 Cs*, Nov. 2018, Accessed: Jun. 16, 2020. [Online]. Available: http://arxiv.org/abs/1703.10593.

[41] M. Salem, S. Taheri, and J. S. Yuan, "Anomaly Generation Using Generative Adversarial Networks in Host-Based Intrusion Detection," in *2018 9th IEEE Annual Ubiquitous Computing, Electronics Mobile Communication Conference (UEMCON)*, Nov. 2018, pp. 683–687, doi: 10.1109/UEMCON.2018.8796769.

[42] V. Deepa, K. M. Sudar, and P. Deepalakshmi, "Design of Ensemble Learning Methods for DDoS Detection in SDN Environment," *2019 Int. Conf. Vis. Emerg. Trends Commun. Netw. ViTECoN*, 2019, doi: 10.1109/ViTECoN.2019.8899682.