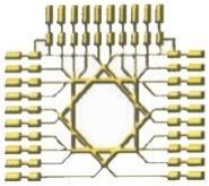
الجمهوريّة العربيّة السـوريّة
المعهد العالي للعلوم التّطبيقيّة والتّكنولوجيا
قسم الاتّصالات



أُعدَّت هذه الأطروحة لنيل
درجة الماجستير في شبكات الاتّصالات

تحسين أداء نظم كشف الاختراق في الشّبكات المعرّفة برمجيّاً باستخدام تعلّم الآلة

إعداد

م. عقبه عبّاس

إشراف

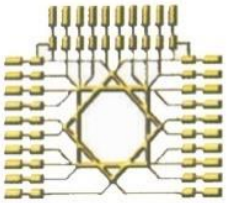
د. خلدون خرزم **د. محمّد عصّورة**

دمشق، تشرين الثّاني 2019

Syrian Arab Republic

Higher Institute for Applied Sciences and Technology

Telecommunication Department



Prepared to get Master's degree in
Telecommunication Networks

Enhance Intrusion Detection Systems Performance in Software Defined Networks, using Machine Learning

By
Eng. Oqbah Abbas

Supervisors
Prof. Khaldoun Khorzom **Prof. Mohammed Assora**

Damascus, 2019

المعهد العالي للعلوم التطبيقية والتكنولوجيا
Higher Institute for Applied Sciences and Technology

المعهد العالي للعلوم التطبيقية والتكنولوجيا مؤسسة حكومية للتعليم العالي أحدثت بموجب المرسوم التشريعي رقم /24/ لعام 1983، وذلك بهدف إعداد كوادر علمية متميزة من مهندسين وباحثين للإسهام الفاعل في عملية التطوير العلمي والتنمية في الجمهورية العربية السورية.

يمنح المعهد العالي درجة الإجازة في الهندسة في الاتصالات والمعلوماتية والنظم الإلكترونية والميكاترونيكس وعلوم وهندسة المواد وهندسة الطيران. يقبل المعهد العالي لدراسة هذه الاختصاصات شريحة منتقاة من المتفوقين في الشهادة الثانوية من الفرع العلمي. يتيح المعهد العالي أيضاً برامج ماجستير أكاديمي في نظم الاتصالات وفي التحكم والروبوتيك وفي نظم المعطيات الكبيرة ونظم المعلومات ودعم القرار وفي علوم وهندسة المواد وعلوم وهندسة البصريات. وأخيراً، يمنح المعهد العالي درجة الدكتوراه في الاتصالات والمعلوماتية ونظم التحكم والفيزياء التطبيقية. تُحدث في المعهد العالي اختصاصات جديدة بحسب متطلبات سوق العمل وتوجهات البحث والتطوير المحلية والعالمية.

إلى جانب النشاط التعليمي، يمارس المعهد العالي عبر جهود أطره وفعالياته العلمية المختلفة نشاطاً حثيثاً في البحث والتطوير، إذ ينفذ مشاريع ودراسات واستشارات متنوعة لصالح الجهات العامة والخاصة في القطر، كما يتعاون مع جهات خارج القطر في بعض المشاريع البحثية والتطويرية. يسعى المعهد أيضاً، عبر دورات تدريبية نظرية وعملية متاحة للقطاعين العام والخاص وللأفراد، إلى إفادة أوسع فئة من المهتمين من إمكانيات أطره العلمية ومختبراته. واستكمالاً لدوره الرائد في مجال التعليم ونشر العلم، ينشر المعهد العالي كتباً علمية عالية المستوى من نتاج أطره، منها ما هو تدريسي يوافق المناهج في المعهد العالي ويفيد شريحة واسعة من الطلاب الجامعيين عموماً، ومنها ما هو علمي ثقافي. يتيح المعهد العالي بعضاً من منشوراته على موقعه على الشابكة، كما يتيح إمكانية الاطلاع على رسائل الماجستير والدكتوراه المنفذة في المعهد العالي وعلى بعض منشورات طلابه وأطره من المقالات العلمية.

المعهد العالي للعلوم التطبيقية والتكنولوجيا، الجمهورية العربية السورية، دمشق، ص.ب 31983

Higher Institute for Applied Sciences and Technology – HIAST

P. O. Box 31983, Damascus, Syrian Arab Republic

هاتف 00963115123819 – فاكس 00963115140761

بريد إلكتروني contact@hiast.edu.sy

موقع إلكتروني www.hiast.edu.sy

إلى عائلتي وأصدقائي

أتقدّم بالشّكر إلى كل من

- الدّكتور خلدون خرزم والدّكتور محمّد عصّورة الذان لم يدّخرا جهداً أو يبخلا بمساعدّة أو نصيحة بهدف إنجاح هذا العمل.
- المهندس محمد سميط لما قدّمه من مساعدات ونصائح.
- المهندس محمد بشار دسوقي لمساعداته القيّمة.
- كل من المهندسين: جودت هارون ونضال الشّاطر ومحمّد مؤيّد منصور وأسامة السّعدي وحسن معرّيني ومحمّد علي الغنطاوي وطه درويش.
- المهندسَين أحمد شنشو وهشام سعد الدّين.
- الدّكتور عبد النّاصر العاسمي.
- جميع الدّكاترة و المهندسين والعاملين في قسم الاتّصالات.
- كل من ساهم في إنهاء هذا العمل.

الملخّص

يهدف المشروع إلى تصميم وتنفيذ نظام كشف اختراق يعمل ضمن الشّبكات المعرّفة برمجيّاً، بحيث يتمّ الاستفادة من قدرة هذه الشّبكات على تأمين مجموعة من الإحصائيّات عن الدّفق المار عبر الشّبكة، واستخدام هذه الإحصائيّات ضمن خوارزميّات تعلّم الآلة لبناء نظام انتخاب قادر على دراسة سلوك المستخدم وتوقّع محاولات الاختراق.

ما يميّز هذا النّظام هو عدم الحاجة لإضافة أجهزة إلى الشّبكة، حيث يتمّ بناؤه كتطبيق برمجي ضمن المتحكّم الخاص بالشّبكة، بالإضافة إلى عدم تسبّبه بعبء إضافي على الشّبكة، إذ أنّه لا يحتاج إلى إرسال طرود جديدة عبر الشّبكة فلا يزيد من التّأخير ضمن الشّبكة أو يقلّل من عرض حزمتها.

Abstract

We aim in this project to design and implement and intrusion detection system within software defined networks, to benefit from its ability to provide several statistical features about any flow that passes the network. Then pass these statistics to several machine learning algorithms to build a voting system, which will be able to study the behavior of the network’s users and predict any possible intrusion.

The best feature about this system is that it does not require any additional devices, as it could be built as an app within the network’s controller, and it does not add any extra load to the network as it only depends on the statistics provided by the network, so it does not increase the latency of the network, nor decrease its bandwidth.

مشكلة البحث

أصبحت الشّبكات المعرّفة برمجيّاً مقصداً لكبرى شركات المعلومات ومراكز البيانات، ممّا جعلها محط أنظار منفّذي الهجمات، فكان لا بد من تطوير طرق كشف ومكافحة لمحاولات الاختراق بما يتناسب مع هذه الشّبكات، فنظراً لعرض الحزمة الكبير الذي تتعامل معه هذه الشّبكات لا فائدة من استخدام الطّرق التقليدية التي تعتمد على قراءة محتوى كل طرد يمر عبر الشّبكة واستخلاص معلومات من هذه الطّرود.

بناءً على ذلك قمنا باقتراح نظام كشف اختراق يستفيد من قدرة الشّبكات المعرّفة برمجيّاً على تأمين إحصائيّات عن كل دفق يمر عبر الشّبكة دون التأثير على أداء الشّبكة.

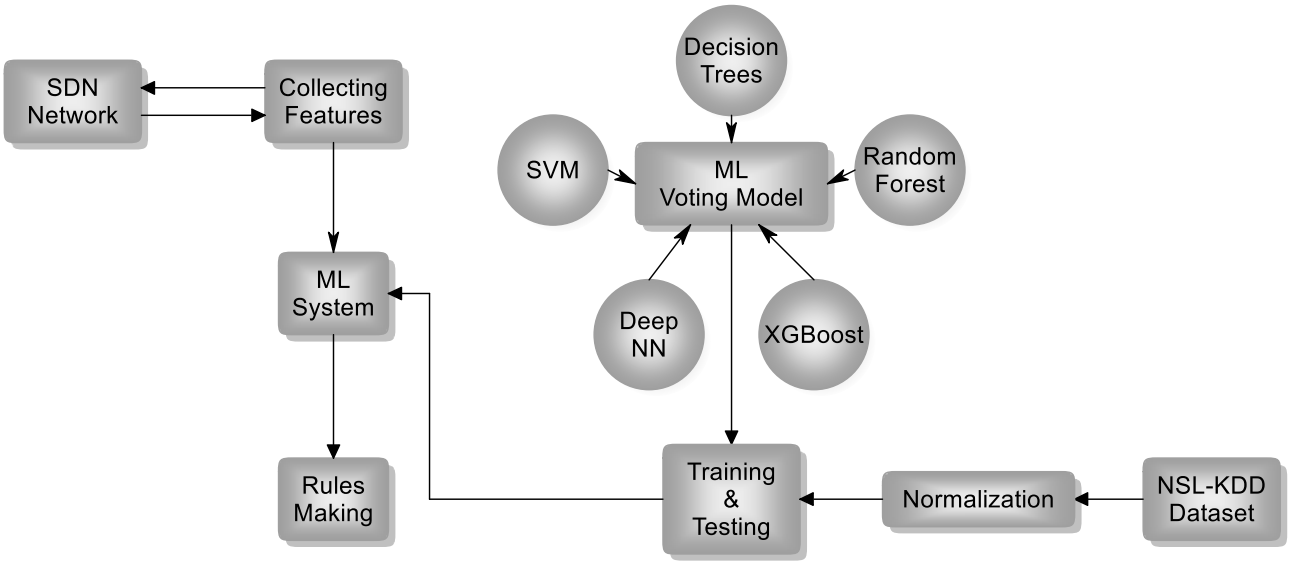
ونظراً لكون كميّة المعطيات التي تؤمّنها الشّبكات المعرّفة برمجيّاً أقل من الكميّة التي تتعامل معها نظم كشف الاختراق عادةً، تمّ اعتماد خيار الشّبكات العصبونيّة وخوارزميّات تعلّم الآلة نظراً لقدرتها على اكتشاف أنماط –حتّى مع كمية معطيات قليلة– لا تستطيع خوارزميّات أخرى اكتشافها.

وبهذا يكون الجزء الأكبر من مشكلة البحث هو اختيار أفضل خوارزميّات تعلّم الآلة لحل مشكلة تصنيف ثنائيّة، الهدف منها هو توقّع ما إذا كان الدّفق المار عبر الشّبكة سليماً أو مرتبطاً بمحاولة اختراق.

وكان الحل المقترح هو تطوير نظام كشف اختراق يعتمد على الإحصائيّات التي توفّرها الشّبكات المعرّفة برمجيّاً، وتدريب مجموعة من خوارزميّات تعلّم الآلة لبناء نظام انتخاب، لتحديد ما إذا كان دفق ما سليماً أو مرتبطاً بمحاولة اختراق، وبهدف إعطاء النّظام القدرة على صدّ محاولات الاختراق تمّت إضافة جزء قادر على إضافة قواعد تمّنع مرور أي دفق يتمّ توقّع ارتباطه بمحاولة اختراق ممّا يحوّل النّظام المقترح إلى نظام منع اختراق.

المخطّط الصّندوقي للنّظام المقترح

يبيّن المخطط التّالي بنية النّظام المقترح لحل مشكلة البحث:



ويتألّف هذا النّظام من:

- جزء خاص بطلب واستخلاص الإحصائيّات من متحكّم الشّبكة المعرّفة برمجيّاً.
- نظام انتخاب يعتمد على مجموعة من خوارزميّات تعلّم الآلة.
- مرحلة إضافة قواعد تمّنع مرور أي دفق مرتبط بمحاولة اختراق.

خطة البحث

تمّ اعتماد المراحل التّالية:

- دراسة مرجعيّة عن الشّبكات المعرّفة برمجيّاً لمعرفة بنيتها ومزاياها.
- دراسة مرجعيّة عن نظم كشف الاختراق لمعرفة الخيارات المتاحة واختيار الأفضل.
- دراسة مرجعيّة عن خوارزميّات تعلّم الآلة لمعرفة آلية عملها واختيار أفضل الطّرق لحل مسائل التّصنيف.
- اقتراح نموذج لنظام كشف اختراق، وإجراء محاكاة لعمله ضمن البيئات البرمجيّة المناسبة.

المساهمات

- بناء نظام منع اختراق يعمل ضمن الشّبكات المعرّفة برمجيّاً ويعتمد على خوارزميات تعلّم الآلة.
- استخدام الخوارزميّة XGBoost لأوّل مرّة ضمن نظام كشف اختراق يعتمد على السّمات التي تؤمّنها الشّبكات المعرّفة برمجيّاً.
- مقارنة أداء الخوارزميّات (Decision Tree, Random Forest, XGBoost, SVM and Deep Neural Networks) عند استخدامها في مجال كشف الاختراق ضمن الشّبكات المعرّفة برمجيّاً.
- إجراء مقارنة بين مجموعتيّ المعطيات KDDCup99 وNSL-KDD.
- إعداد ورقة بحثيّة عن العمل الذي تمّ في هذا البحث بعنوان Machine Learning based IDS for Software Defined Networks.

فهرس المحتويات

قائمة الأشكال

قائمة الجداول

الرّموز والاختصارات

SDN	Software Defined Network/Networking
VLAN	Virtual Local Area Network
IP	Internet Protocol
MAC	Media Access Control
API	Application Program Interface
AT&T	American Telephone and Telegraph
QoS	Quality of Service
IDS	Intrusion Detection System
HIDS	Host-Based Intrusion Detection System
NIDS	Network-Based Intrusion Detection System
MIDS	Mixed Intrusion Detection System
PIDS	Protocol-Based Intrusion Detection System
WIDS	Protocol-Based Intrusion Detection System
TP	True Positive
FP	False Positive
TN	True Negative
FN	False Negative
CR	Classification Rate
DR	Detection Rate
FPR	False Positive Rate
PR	Precision
ReLU	Rectified Linear Unit
KDD	Knowledge Discovery and Data-Mining
SVM	Support Vector Machine
DNN	Deep Neural Network
RBF	Radial Basis Function
DT	Decision Tree
RF	Random Forest
KNN	K-Nearest Neighbors

TPR	True Positive Rate
PPV	Positive Predictive Value
R2L	Root to Local
DoS	Denial of Service
U2R	User to Root
NSL-KDD	Network Security Laboratory Knowledge Discovery and Data-Mining
ID3	Iterative Dichotomies 3
CART	Classification and Regression Trees
XGBoost	Extreme Gradient Boosting
NN	Neural Network/s
RNN	Recurrent Neural Network/s
ML	Machine Learning
DARPA	The Defense Advanced Research Projects Agency
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
ICMP	Internet Control Message Protocol
Sklearn	Scientific Kit for Learning
DNN	Deep Neural Network
GNS	Graphical Network Simulator

مقدّمة عامّة

تزامن التّطور الكبير في أجهزة الحاسب وأنظمة المعلومات مع تطوّر ممّاثل في شبكات الاتّصالات وسرعة نقل المعلومات، ممّا زاد الحاجة لحماية هذه المعلومات وحماية الشّبكات التي تنتقل عبرها المعلومات.

ولطالما كانت هناك محاولات لإيجاد حلول أمنية مناسبة لحماية الشّبكات، تنوعت هذه الحلول بين نظم كشف الاختراق وجدران ناريّة وبرامج مكافحة البرمجيّات الخبيثة، لكن تكمن المشكلة في أنّ التّطور المتسارع لأنظمة المعلومات والشّبكات وظهور مفاهيم جديدة كالشّبكات المعرّفة برمجيّاً وتبنّي الشّركات الكبرى ومراكز البيانات لهذه المفاهيم ترافق مع تطوّر وتنوّع كبير في أنواع التّهديدات والمخاطر، فأصبحت الطرق التّقليديّة لحماية الشّبكة غير كافية وغير مرضية للقائمين على الشّبكة.

وبالرّغم من كثرة الأبحاث في هذا المجال إلا أن العاملين فيه ما زالوا يتطلّعون لنتائج أفضل لمنع أي محاولة تطفّل أو هجوم أو اختراق، لذلك كان لابدّ من إيجاد طرق جديدة أو تطوير الطّرق القديمة بحيث تصبح الجدران الناريّة وأنظمة كشف الاختراق أكثر قدرة على كشف وإيقاف أيّ تهديد، فلم يعد كافياً كشف الهجوم في مراحله الأولى واتّخاذ الإجراءات اللازمة لإيقافه، بل أصبح من الضروريّ توفّر القدرة على دراسة سلوك مستخدمي الشّبكة وتوقّع الهجوم قبل بدايته.

تعتبر خوارزميّات تعلّم الآلة والشّبكات العصبونيّة أفضل الطّرق في هذا المجال، وذلك لقدرتها على استخلاص معلومات جديدة عند أي تفاعل للمستخدم مع الشّبكة، بحيث يمكن دراسة سلوك المستخدم ممّا يسهّل عمليّة اكتشاف الهجمات قبل بدايتها، ويزيد القدرة على اكتشاف أنواع الهجمات الجديدة ومكافحتها.

لكن تكمن المشكلة في أن استخدام هذه الأنواع من الخوارزميّات مع شبكات ضخمة يتطلب كلفة إضافية من ناحية شراء أجهزة جديدة وإضافتها إلى الشّبكة، ومن ناحية الحمل الزائد على الشّبكة لاستخلاص المعلومات عن سلوك المستخدمين.

وبهدف بناء نظام كشف اختراق فعّال لا يضيف أي عبء على الشّبكة ولا يتطلّب إضافة تجهيزات جديدة، لجأنا إلى إنشاء نظام كشف اختراق يعمل ضمن الشّبكات المعرّفة برمجيّاً للاستفادة من قدرتها على تأمين إحصائيّات عن المعلومات التي تمّر عبر الشّبكة لدراسة سلوك المستخدمين، مع القدرة على تحويله إلى نظام منع اختراق من خلال منحه صلاحيّات إضافة قواعد تمّنع مرور أي دفق تمّ توقّع ارتباطه بمحاولة اختراق.

الفصل الأول

الشّبكات المعرّفة برمجيّاً

Software Defined Networks

نقدّم في هذا الفصل مقدّمة عن الشّبكات المعرّفة برمجيّاً، بالإضافة إلى الدّافع من ظهورها، وبنيتها، ونستعرض بعض فوائدها والتحدّيات التي تواجهها.

1-1. الدّافع وراء ظهور الشّبكات المعرّفة برمجيّاً

عندما ظهرت الهواتف المحمولة في تسعينيّات القرن الماضي كان الهاتف مزود بنظام تشغيل يحوي مجموعة من التّطبيقات، وفي حال وجود خلل في نظام التّشغيل أو أحد التّطبيقات كان لا بدّ من انتظار إصدار النّسخة التّالية من هذه التّطبيقات لإصلاح هذا الخلل، لكن مع ظهور الهواتف الذّكية أصبح بإمكان المستخدمين تطوير تطبيقات خاصّة بهم تناسب متطلّباتهم، وتنصيب هذه التّطبيقات على هواتفهم.

بشكل مشابه لذلك، كان يمكن الحصول على الأجهزة الشبكيّة (مبدّلات ومسيّرات) بحيث تدعم مجموعة محدّدة من البروتوكولات تحدّدها الشّركة المصنّعة دون القدرة على استخدام بروتوكولات أخرى، لكن مع ظهور الشّبكات المعرّفة برمجيّاً SDN أصبح بإمكان المطوّرين تطوير بروتوكولات خاصّة بهم واختبارها على الأجهزة الشبكيّة الخاصّة بهم دون الرّجوع للشّركة المصنّعة.

وكان الدّافع الرّئيسي لظهور شبكات SDN هو الزّيادة الكبيرة في عدد الأجهزة المرتبطة بالشّبكة ممّا يؤدّي إلى ضرورة التّعامل مع حجم معطيات كبير جداً، بالإضافة إلى ظهور الخدمات الغماميّة (cloud services) والشّبكات الافتراضيّة ممّا ساهم أيضاً بزيادة كميّة الخدمات التي تقدّمها الشّركات وبالتّالي زيادة كبيرة في حجم المعطيات، كل هذا أدّى إلى إعادة التفكير بقدرة الشّبكات التقليديّة على تلبية هذه المتطلّبات.[1]

إنّ تلبية الشّبكات التّقليدية لمتطلّبات السّوق الحاليّة يكاد يكون مستحيلاً، إذ أنّ الحجم الكبير في كمية المعطيات والخدمات يتطلّب توسعة الشّبكة بشكل مستمر ممّا ينعكس على كلفة تشغيل الشّبكة، إذ يجب إدارة وتحديث كل من الأجهزة الشبكيّة بشكل مستقل عند أي عمليّة صيانة للشّبكة أو تغيير في سياسات الشّركة، بالإضافة إلى الكلفة الإضافيّة التي تترافق مع ضرورة تزويد الشّبكة ببعض الأجهزة التي توفّر تطبيقات الحماية وإدارة تدفّق المعطيات (traffic engineering).

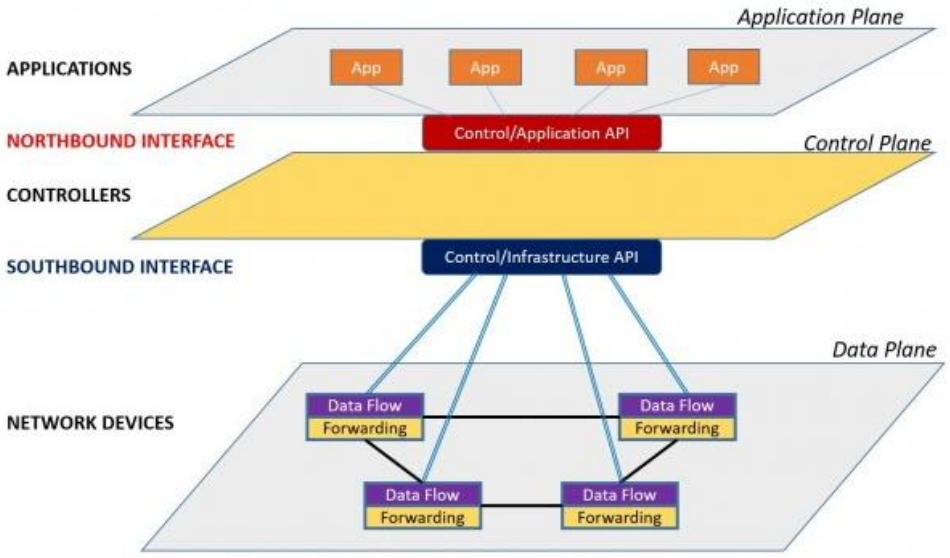
ونذكر فيما يلي بعض المشاكل الإضافيّة التي تعاني منها الشّبكات التقليديّة: [2]

- الارتباط بالشّركة المصنّعة: تأمين التّوافق بين تجهيزات الشّبكة يتطلّب في أغلب الأحيان أن تكون كافّة التّجهيزات من نفس الشّركة ممّا يحدّ من إمكانيّة الحصول على تطبيقات جديدة لا تؤمّنها الشّركة المصنّعة.
- قابليّة التوسّع: تأمين سرعة نقل أعلى عبر الشّبكة يتطلّب دراسة وتنبّؤ لنوع الدّفق الذي يمر عبر الشّبكة، ثمّ القيام بإعادة ضبط مكوّنات الشّبكة لتأمين الزّيادة في سرعة النّقل، لكن مع ظهور مراكز المعطيات أصبح من المستحيل التّنبؤ بنمط المعطيات، ممّا يحبط إمكانية زيادة سرعة النّقل بشكل فعّال.
- سياسات الشّبكة: أي تغيير في سياسات الشّبكة يتطلّب إعادة ضبط كافة مكوّنات الشّبكة بهدف المحافظة على مستوى معيّن من الأمن وجودة الخدمة.

لكن مع تقدّم البحث في مجال شبكات SDN ظهرت حلول لمعظم هذه المشاكل.

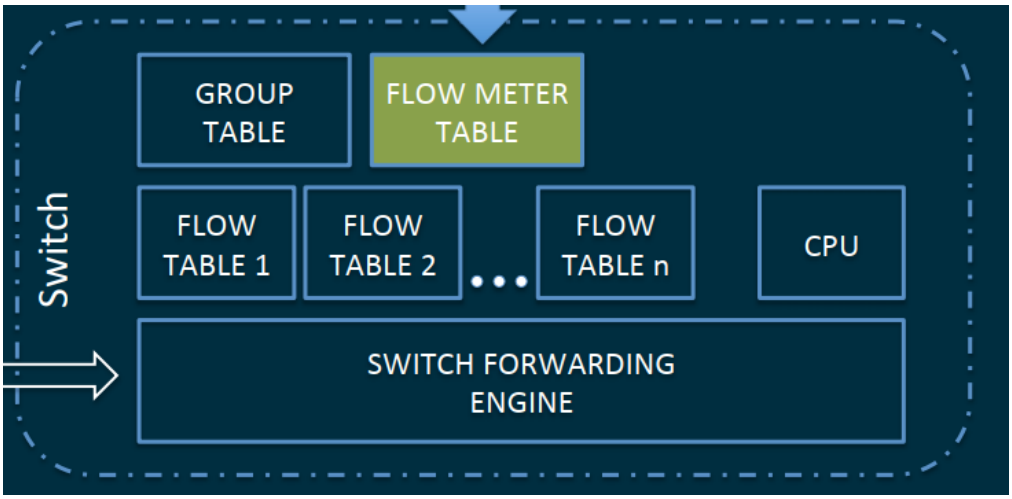
2-1. بنية شبكات SDN

تتألف شبكة SDN من ثلاثة مستويات، مستوي المعطيات الذي يحوي التّجهيزات الشبكيّة المسؤولة عن عمليّات التّمرير، ومستوي التّحكم الذي يتمّ برمجة التّجهيزات الشّبكيّة من خلاله، ومستوي التّطبيقات الذي يتمّ من خلاله إضافة خدمات وتطبيقات جديدة للشبكة [4]، ويبيّن الشّكل التّالي بنية شبكة SDN: [21]



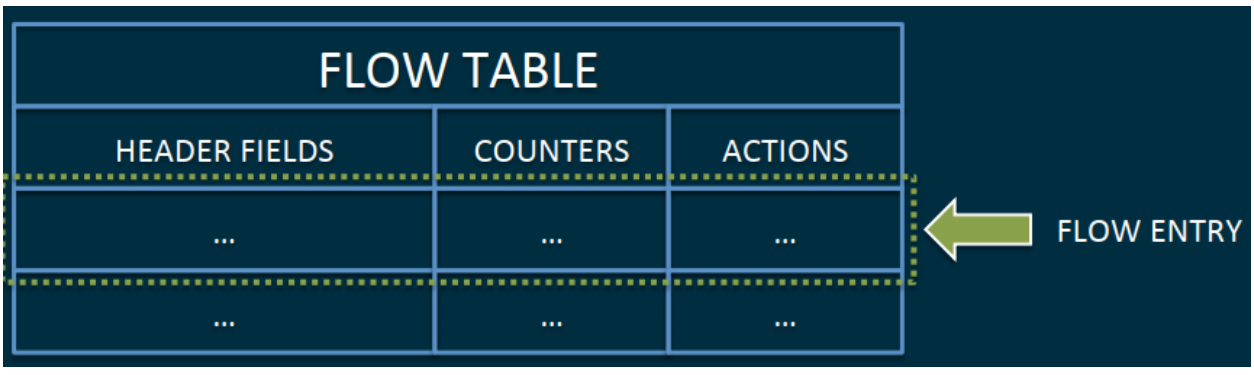
الشّكل **1-1**: بنية شبكة **SDN**

- مستوي المعطيات: يتألّف هذا المستوي من مجموعة المبدّلات (SDN-Switches) المسؤولة عن علميّات التّمرير (forwarding)، وتحوي كل من هذه المبدّلات على مدخل للتّواصل مع المتحكّم الموجود في مستوي التّحكم، ويتمّ عن طريق هذا المدخل إرسال معلومات تساعد المتحكّم على الحصول على نظرة عامّة عن طبولوجيا الشّبكة، بالإضافة إلى إحصائيّات عن دفق المعطيات الذي يمرّ عبر الشّبكة، كما يتمّ عبر هذا المدخل استقبال جداول التّسيير التي تتمّ عمليّات التّمرير بناءً عليها، ويبيّن الشّكل التّالي بنية الـ SDN-Switch الموافقة لبروتوكول OpenFlow1.3: [22]



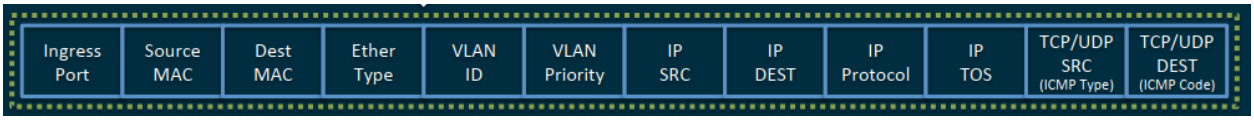
الشّكل **2-1**: بنية الـ **SDN-Switch** الموافقة لبروتوكول **OpenFlow1.3**

ونلاحظ أن المبدّل يتألّف من محرّك لعمليّات التّمرير مهمّته استقبال الطّرود الواردة من إحدى بوابّات الدّخل وإخراجها من البوّابة المناسبة، بالإضافة إلى Group Table مهمّته تجميع عدّة أنواع من الدّفق ومعاملتها بشكل مشابه لـ VLAN، كما ويحتوي على Flow Meter Table الذي يتحكّم بمجموعة من المعاملات الخاصّة بكل دفق (مثلاً معدّل الطّرود الخاصّة بدفق معيّن)، ويحتوي أيضاً على الـ flow tables التي تحوي قواعد عمليّات التّمرير، حيث يتمّ مقارنة كل طرد وارد إلى المبدّل مع السّجلات الموجودة في الـ flow tables وعند التّطابق يتمّ تمّرير الطّرد عبر البوّابة التي ينصّ عليها السّجل، وفي حال عدم التّطابق يتمّ رفع الطّرد إلى المتحكّم ليقرر كيفيّة التّعامل معه، مع العلم أن عمليّة مقارنة الطرد الوارد تتمّ بالتّتالي انطلاقاً من الـ flow table الأوّل، ويبيّن الشّكل التّالي بنية الـ flow table:



الشّكل **3-1**: بنية الـ **flow table** في الـ **SDN-Switch**

ونلاحظ من الشّكل السّابق أن كل سجل من الـ flow table يحوي التّرويسة التي ستتمّ على أساسها مقارنة الطّرود الواردة، ومجموعة من العدّادات الخاصّة بالجدول وعدّادات خاصّة ببوّابات المبدّل وعدّادات خاصّة بالدّفق وهي عدد الطّرود الواردة من هذا الدّفق وعدد البايتات الواردة من هذا الدّفق والمدّة الزّمنيّة الفاصلة بين أوّل طرد ورد من الدّفق وحتّى هذه اللحظة، كما يحتوي كل سجل على العمليّة التي سيتمّ تطبيقها على الطّرد الوارد والذي تطابق مع هذا السّجل (إخراج من بوابة معيّنة أو إهمال الطّرد أو إرساله إلى المتحكّم لتتمّ دراسته أو إرساله إلى جدول آخر ليتمّ إجراء عمليّات مقارنة إضافيّة)، ويبيّن الشّكل التّالي حقول التّرويسة التي تتمّ على أساسها مقارنة الطّرود الواردة مع السّجلات في الـ flow tables:



الشّكل **4-1**: حقول الترويسة في الـ **flow table** في الـ **SDN-Switch**

ويبيّن الشّكل السابق أن المطابقة تتمّ على أساس العنوان المنطقي (IP) والعنوان الفيزيائي (MAC) والبوّابة (Port) بالإضافة لحقول أخرى بنفس الوقت، على عكس الشّبكات التّقليدية حيث تتمّ المقارنة على أساس العنوان المنطقي في الـ Network Layer، وعلى أساس العنوان الفيزيائي في الـ Data Link Layer، وعلى أساس الـ Port في الـ Transport Layer.

- مستوي التّحكم: تحتوي هذه الطّبقة على المتحكّم الذي يعتبر نقطة التّحكم المركزيّة للشّبكة، حيث يقوم بالإشراف على عمليّات التّمرير، وتوليد الـ flow tables ونقلها إلى المبدّلات عبر الواجهة الجنوبيّة (Southbound Interface)، كما أن المتحكّم يتّصل مع مستوي التّطبيقات عبر الواجهة الشّماليّة (Northbound Interface) لتأمين بعض المعلومات التي تحتاجها الخدمات في مستوي التّطبيقات.
- مستوي التّطبيقات: يحتوي هذا المستوي على مجموعة التّطبيقات التي يتفاعل معها المستخدمين (غالباً مدير الشّبكة)، ويمكن لهذه التّطبيقات أن تتلاعب بعمليّات التّمرير التي تتمّ في مستوي المعطيات من خلال التّعديل على الـ flow tables التي يتمّ توليدها في المتحكّم ثمّ نقلها إلى المبدّلات في مستوي المعطيات

كما يحتوي على واجهتين تصلان بين المستويات الثلاثة:

- الواجهة الجنوبيّة (Southbound Interface): تصل بين مستوي المعطيات ومستوي التّحكم، ومهمّتها نقل الـ flow tables من المتحكّم إلى المبدّلات، ونقل معلومات عن طبولوجيا الشّبكة وإحصائيّات عن الدّفق المار عبر الشّبكة من المبدّلات إلى المتحكّم، ويتمّ ذلك من خلال بروتوكول OpenFlow.

- الواجهة الشّماليّة (Northbound Interface): تصل بين مستوي التّحكم ومستوي التّطبيقات، ومهمّتها نقل أوامر تعديل الـ flow tables من مستوي التّطبيقات إلى المتحكّم، ونقل المعطيات التي تهم الخدمات من المتحكّم إلى الخدمات، ويتمّ ذلك من خلال واجهات تخاطب برمجية (API) لتسهيل تعامل مطوّر الخدمة مع المتحكّم.

1-3. مراحل ظهور شبكات SDN

مرّت شبكات SDN بعدّة مراحل حتّى وصلت إلى شكلها الحالي:

- Active Networks (mid 1990s – beginning of 2000s): وفيها أصبحت العقد قابلة للبرمجة لتقوم بعمليّات معيّنة على المعطيات التي تمّر من خلالها لتحقيق هدف معيّن.
- فصل مستوي المعطيات عن مستوي التّحكم (2001 - 2007): ممّا ساهم بفتح واجهات تخاطب open interfaces) بين مستوي التّحكم ومستوي المعطيات.
- تطوير بروتوكول OpenFlow: ممّا ساهم باستغلال الـ open interfaces بشكل أكبر. [2]

1-3-1. Active Networks

تمّ في هذا النّوع من الشّبكات طرح مفهوم جديد يعتمد على إضافة واجهة جديدة في العقد تجعلها قابلة للبرمجة، ممّا جعل العقد قادرة على إجراء مجموعة من العمليّات على المعطيات التي تمّر عبرها بحيث تختلف هذه العمليّات باختلاف الخدمة المراد تقديمها.

الدّافع وراء هذا المفهوم الجديد كان جعل العقد الجديدة القابلة للبرمجة قادرة على إضافة خدمات جديدة للشّبكة دون الحاجة للتّخلص من العقد الموجودة مسبقاً، لأن طرح مفهوم جديد أو تكنولوجيا جديدة –تحتاج لتغيير جذري– في شبكة كبيرة كالانترنت سيحتاج للعديد من السّنوات لتطبيقه بشكل كامل.

على الرغم من الانتشار المحدود لهذا النّوع من الشّبكات فقد تبنّت شبكات SDN المفاهيم التي قامت عليها هذه الشّبكات.[1]

2-3-1. فصل مستوي التّحكّم عن مستوي المعطيات

جرت عدّة محاولات لفصل مستوي التّحكّم عن مستوي المعطيات، أوّلها كان من قبل AT&T، ثمّ تلاها عدّة محاولات أخرى كان الهدف منها الحصول على وحدة تحكّم مركزيّة قادرة على برمجة العقد في مستوي المعطيات ممّا يسهّل إدارة الشّبكة.

وبسبب اعتماد هذا التوجه أصبحت الشّبكة من متحكّم ومبدّلات (SDN-Switches)، بعد أن كانت مسيّرات ومبدّلات في الشّبكات التقليدية

1-3-3. تطوير بروتوكول OpenFlow

قبل ظهور بروتوكول OpenFlow كان من الصّعب تطبيق فكرة الشّبكات المعرّفة برمجيّاً بشكل عملي، فقد كان من الصّعب إعادة برمجة الشّبكة أثناء عملها بحيث يتمّ تنصيب خدمة جديدة أو يتمّ تغيير سلوك الشّبكة.

مهمة بروتوكول OpenFlow هي تأمين التّواصل بين المتحكّم في مستوي التّحكّم والمبدّلات في مستوي المعطيات عبر قناة آمنة باستخدم اتصال TLS، ويتمّ ذلك عن طريق مجموعة من الرّسائل: [23]

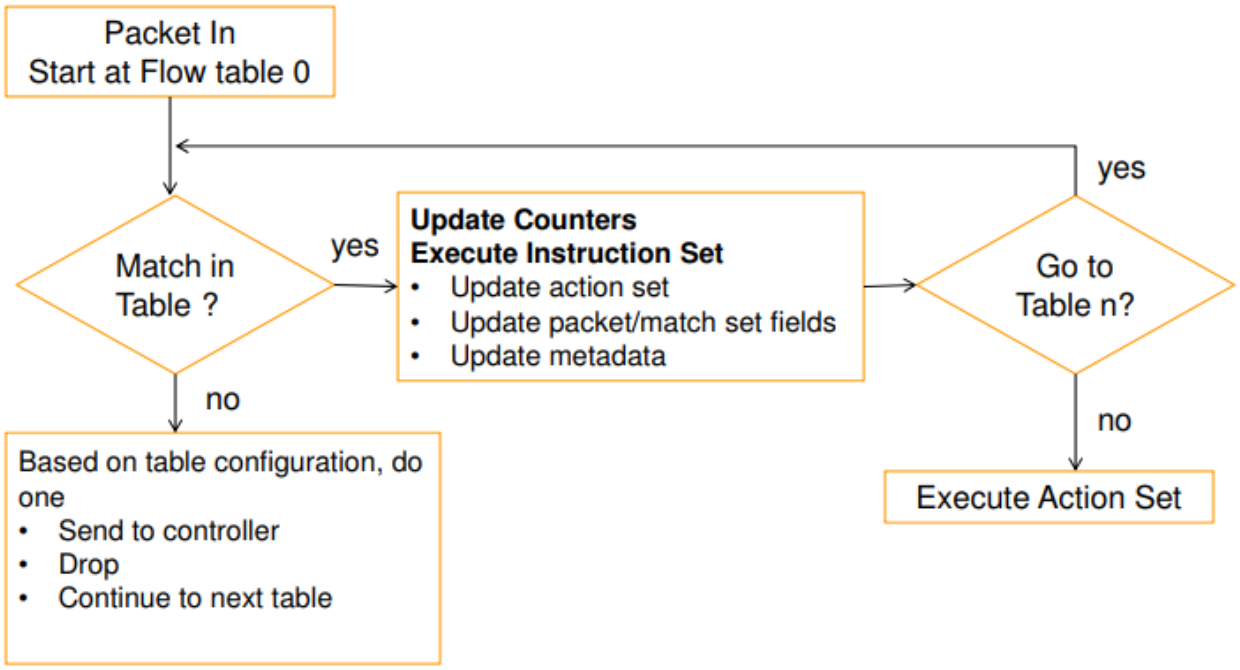
- Controller to Switch: يرسلها المتحكّم إلى المبدّل إمّا لطلب معلومات أو لإرسال معلومات ومنها:
  - Features: لطلب معلومات عن الدّفق الذي يمر عبر الشّبكة من خلال feature_req، يرد عليها المبدّل بـ feature_rep.
  - Modify_state: لإضافة أو تعديل أو حذف الـ flow tables
  - Read_state: للاستعلام عن الـ flow tables.
- Asynchronous-Asynchronous: من المبدّل إلى المتحكّم دون أن يطلبها المتحكّم ومنها:
  - Packet_in: في حال لم يعرف المبدّل كيف يتعامل مع أحد الطّرود بسبب عدم تطابقه مع أي من السّجلات في الـ flow table.
  - Flow_removed: عند حذف أحد السّجلات من الـ flow table نتيجة عدم استخدامه لفترة زمنية معيّنة.
- Symmetric-Symmetric: يمكن أن ترسل في من المتحكّم للمبدّل أو بالعكس دون أن يطلبها المتحكّم ومنها:
  - Hello: عند إنشاء الوصلة ليستدل كلّ من المتحكّم والمبدّل على الآخر.
  - Echo: ليتأكد المتحكّم أو المبدّل أن الاتّصال مع الطّرف الآخر لا زال قائماً، بالإضافة للحصول على معلومات عن التّأخير وعرض الحزمة.

وهكذا يساهم بروتوكول OpenFlow بتأمين نظرة شاملة لمستوي التّحكّم عن طبولوجيا الشّبكة، بالإضافة للسّماح بتغيير برمجة العقد ممّا يغيّر من سلوكها تبعاً لنمط المعطيات المتدفّقة عبرها أو نوع الخدمات المراد تقديمها.

4-1. طريقة عمل شبكة SDN

عند بداية عمل الشّبكة أو عند حصول أي تعديل في الطّبولوجيا الخاصّة بها يتعرّف المتحكّم على بنية الشّبكة من خلال رسائل التّعارف المتبادلة بين المتحكّم والمبدّلات، وبهذا يحافظ المتحكّم على نظرة شاملة عن طبولوجيا الشّبكة، وبالاستفادة من هذه الطّبولوجيا ومعرفة المتحكّم بالخدمات المراد تطبيقها ضمن الشّبكة يضيف بعض القواعد إلى flow table الخاص بكل مبدّل، ثمّ يقوم بإرسال هذه الجداول إلى المبدّلات التي تصبح جاهزة لاستقبال الطّرود.

عند ورود طرد ما إلى أحد المبدّلات يتمّ التّعامل معه وفقاً للمخطّط التدفّقي المبيّن في الشّكل التّالي: [23]



الشّكل **1-5**: المخطّط التّدفقي لخوارزميّة التّعامل مع الطّرود الواردة

يبيّن المخطّط السّابق أنّه عند استقبال طرد جديد يتمّ في البداية مقارنته مع السّجلات الموجودة في الـ flow table رقم 0، وفي حال عدم التّطابق مع أي سجل يتمّ اتباع الإعدادات الموجودة في الـ flow table (إهمال الطّرد أو إرساله إلى المتحكّم ليتمّ اتّخاذ قرار بشأنه أو الانتقال إلى جدول معيّن)، وفي حال التّطابق مع أحد السّجلات يتمّ تعديل قيم العدّادات الخاصّة بهذا السّجل، وفي حال وجود أمر بالانتقال لـ flow table آخر لإجراء عمليّات مقارنة إضافية يتمّ إعادة العمليّات السّابقة، وفي حال عدم وجود أمر بالانتقال لجدول آخر يتمّ تنفيذ الأمر الموجود في السّجل الحالي ليتمّ الانتهاء من التعامل مع هذا الطّرد.

1-5. فوائد شبكات SDN

ذكرنا سابقاً أن الهدف من شبكات SDN هو الحصول على شبكة ذات بنية تحتيّة قابلة للبرمجة ممّا يسهّل إدارتها ويسمح لمدير الشّبكة بتغيير سلوكها، وبالبناء على هذا المفهوم أصبحت شبكات SDN قادرة على تقديم قيمة مضافة في عدّة مجالات:

- ضمان Quality of Service: تعتبر سهولة التلاعب بعمليّة تدفّق المعطيات عبر الشّبكة من أبرز ميّزات شبكات SDN، إذ يمكن تغيير أولويّة نوع معيّن من المعطيات ممّا يساعد في رفع جودة الخدمة (ضمان QoS) في حالات مثل Voice over IP أو إرسال بيانات الوسائط المتعدّدة.[4]
- سهولة إدارة الشّبكة: أصبح بإمكان مدير الشّبكة إضافة خدمات وتطبيقات جديدة بسرعة وفعاليّة، كما أنّ شبكة SDN تؤمّن لمدير الشّبكة رؤية شاملة عن طبولوجيا الشّبكة ومواردها وحالة كل من هذه الموارد ممّا يساعد في متابعة مصادر الأخطاء وإصلاحها.[4]
- خدمات الحماية Security Services: أصبح بالإمكان إضافة تطبيقات خاصّة بأمن الشّبكة دون الحاجة لإضافة أجهزة إضافية، بحيث تستخدم هذه التّطبيقات خوارزميّات ذات وثوقيّة عالية ورد فعل سريع، وعند اكتشاف محاولة اختراق يتمّ إنشاء قواعد تمرير جديدة لإيقاف محاولة الاختراق ويتمّ توزيع هذه القواعد على أجهزة التّمرير لتتابع عملها بالاعتماد على القواعد الجديدة.[6]
- كلفة تشغيل أقل: كلفة أقل عند إضافة خدمات جديدة، بالإضافة لانخفاض كلفة إدارة موارد الشّبكة واستخدامها بشكل أكثر فعاليّة وسهولة تتبّع مصادر الأخطاء.[6]
- ومن أهم فوائد شبكات SDN أن الخدمات التي تتمّ إضافتها إلى الشّبكة يتمّ تشغيلها على حاسب ما بموارد معيّنة دون الحاجة لشراء أجهزة تضاف للشّبكة والتي غالباً ما تكون ذات موارد مخصّصة لخدمة واحدة فقط، أي أنّ كافّة الخدمات يمكن أن تتشارك نفس الموارد، وفي حال الحاجة لموارد إضافية يمكن شراء موارد جديدة ومشاركتها مع كافّة الخدمات.[6]

1-6. التحديات

على الرّغم من الفوائد التي تقدّمها شبكات SDN إلا أنّها تواجه بعض التّحديات:[5]

- أداء المتحكّم: من الممكن أن يحتوي مستوي التّحكّم عدّة متحكّمات بناءً على طبولوجيا الشّبكة، فإذا لم يتمّ التّنسيق بين هذه المتحكّمات قد يحدث عطل في أحد أجزاء الشّبكة ويؤثّر على أداء الشّبكة ككل.
- شبكات SDN الهجينة: وهي شبكة تقليديّة تحوي أجهزة تقليديّة وأخرى تعمل بمفهوم SDN، من المهم دمج شبكة SDN مع الشّبكة التقليديّة بشكل مناسب للحصول على أفضل أداء ممكن.
- الحماية: بما أن المتحكّم يعتبر أهم أجزاء الشّبكة سيصبح الهدف الأساسي للاختراقات.

الفصل الثاني

نظم كشف الاختراق

Intrusion Detection Systems

نعرض في هذا الفصل بنية وأنواع نظم كشف الاختراق وآلية عمل كل منها، بالإضافة إلى عرض مجموعة من الأبحاث التي تمّت في مجال نظم كشف الاختراق.

2-1. مقدّمة وتعاريف

في العقدين الأخيرين ازداد عدد الهجمات على الأجهزة المتصلة بالانترنت بشكل كبير، فوفقاً لموقع cybintsolutions.com يحدث هجوم على أحد الأجهزة المتصلة بالانترنت بمعدل مرة كل 39 ثانية، هذه الزيادة الكبيرة بعدد الهجمات الالكترونية أدت إلى الحاجة لزيادة الدراسات والأبحاث في مجال كشف الاختراق ومن ثمّ إيجاد الطّريقة الأمثل للتعامل مع هذه الهجمات.[13]

قبل أن نقدّم تعريفاً لنظم كشف الاختراق يجب أن نوضح مفهوم الاختراق الذي يعبر عن أي محاولة متعمدة للوصول إلى شبكة أو نظام حاسوبي بشكل غير مصرح به، بهدف الحصول على معلومات معيّنة أو التلاعب بها لتخريبها أو منع الوصول إليها.[14]

بينما يعرف نظام كشف الاختراق على أنه عمليّة مراقبة الأحداث التي يتعرض لها نظام حاسوبي أو شبكة ما بهدف تحليلها، ومحاولة اكتشاف أي هجوم محتمل يهدف لخرق سياسات الأمان في هذا النّظام أو الشّبكة.[15]

حالياً لا تعتبر نظم كشف الاختراق الوسيلة الوحيدة لمكافحة الهجمات ضد الشّبكات والأنظمة الحاسوبية، حيث يوجد نظم كشف الاختراق ونظم منع الاختراق بالإضافة إلى الجدران النارية والتّطبيقات المضادة للفيروسات، إلا أنها تختلف عن بعضها من حيث الوظيفة التي تؤدّيها، فنظم كشف الاختراق تهتمّ بمراقبة سلوك الشّبكة أو النّظام الحاسوبي لكشف أي محاولة دخول غير مصرح به، بينما تتألف نظم منع الاختراق من جزئين، الجزء الأول مشابه لنظام كشف الاختراق بينما يهتمّ الجزء الثاني بتنفيذ تدابير معيّنة لمنع حدوث الاختراق، أما مضادات الفيروسات فمهمتها الأساسية هي اكتشاف ما إذا كان ملف أو تطبيق معيّن ضمن نظام حاسوبي يحوي رمازاً خبيثاً فيتمّ منع تنفيذ هذا الرماز، أما بالنسبة للجدران النارية فيتمّ تنصيبها على نظام حاسوبي لتقوم بمنع مرور مجموعة معيّنة من الطّرود وتسمح بمرور البقيّة.[16]

ورغم تنوع أساليب مكافحة الاختراقات وطرق إلحاق الضرر بالشّبكات والأنظمة الحاسوبية إلا أن تأمين مستوى أمان كافٍ قد يتطلب تطبيق عدّة أساليب في نفس الوقت.

كما نلاحظ أن نظام كشف الاختراق يقوم فقط بمراقبة سلوك الشّبكة أو النّظام الحاسوبي دون أي تأثير عليهما، ويطلق على هذه الطّريقة اسم الطّريقة السلبية (Passive) لأنها لا تتدخل في عمل الشّبكة أو النّظام بأي شكل، أما بالنسبة لنظم منع الاختراق والجدران النارية ومضادات الفيروسات فيطلق عليها اسم الطّريقة الفعالة (Active).

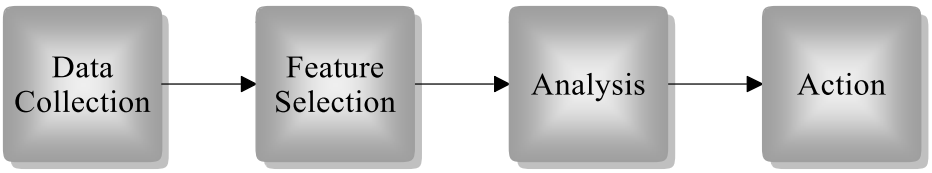
2-2. لمحة تاريخيّة عن نظم كشف الاختراق

بدأت فكرة نظم كشف الاختراق من قبل القوات الجوية الأميركية حيث قام الباحث James Anderson بنشر بحث باسم "Computer Security Threat Monitoring and Surveillance" اقترح فيه نظام كشف اختراق يتمّ فيه مقارنة الدّفق الذي يمر عبر الشّبكة مع مجموعة من الهجمات ذات السلوك المعروف مسبقاً، وتمّ اعتبار هذا البحث نقطة البداية في أتمّتة نظم كشف الاختراق.

وفي تسعينيات القرن الماضي تمّ اقتراح طريقة جديدة يتمّ فيها البحث عن الطّرود التي تمّر عبر الشّبكة وتحمل سلوكاً غريباً [17]، ومنذ تلك الفترة شهد العالم تزايداً في حجم المعطيات المتبادلة، ويزداد هذا الحجم عاماً بعد عام بشكل كبير، ممّا أدى إلى استخدام أنواع جديدة من التكنولوجيا في عمليّة كشف الاختراق كخوارزميّات تعلّم الآلة وخوارزميّات التنقيب عن المعطيات.

2-3. بنية نظام كشف الاختراق

يبيّن المخطط التّالي البنية العامة لنظم كشف الاختراق:



الشّكل **2-1**: البنية العامة لنظم كشف الاختراق

حيث يتألف نظام كشف الاختراق من الأجزاء التّالية:

- Data Collection: يتمّ في هذا الجزء تجميع المعلومات وتخزينها ضمن ملفات ليتمّ تحليلها لاحقاً (في حالة نظام كشف اختراق يعمل ضمن نظام حاسوبي تكون هذه المعلومات عبارة عن إحصائيات عن حالة النّظام، وفي حال كان نظام كشف الاختراق يعمل ضمن شبكة تكون المعلومات عبارة عن طرود تمّر ضمن الشّبكة).
- Feature Selection: يتمّ فيها اختيار جزء محدد من المعلومات المخزنة لتكون دخلاً لجزء التحليل، بدلاً من إدخال كافة المعلومات المخزنة إلى جزء التحليل.
- Analyseis: يتمّ في هذا الجزء اتّخاذ القرار فيما إذا كان يوجد هجوم أو لا، حيث يتمّ مقارنة هذه المعطيات مع معطيات من هجمات معروفة، أو يتمّ مقارنتها مع أنماط معيّنة تقابل سلوكاً معيّناً للشبكة أو النّظام الحاسوبي.

- Action: وهو الجزء الذي يتمّ فيه تنبيه مدير النّظام بحدوث اختراق ليتمّ التعامل بالشّكل المناسب. [18]

4-2. أنواع نظم كشف الاختراق

يوجد العديد من أنواع نظم كشف الاختراق نذكر منها ما يلي:

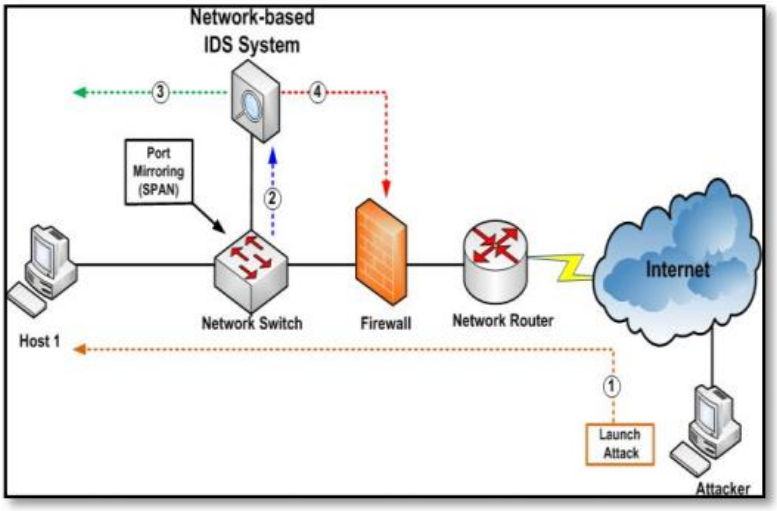
- Host-Based Intrusion Detection System (HIDS)

يتوضع هذا النّظام ضمن الحاسب ويقوم بمراقبة أداء هذا الحاسب (معدل استخدام المعالج والذواكر، والتّطبيقات المستخدمة، وسجلات النّظام، والطّرود الواردة والصادرة إلى هذا الحاسب)، ويركز بشكل أساسي على نقاط الضعف في صلاحيات الوصول إلى المعلومات الموجودة ضمن الحاسب، ويتطلب هذا النوع من نظم كشف الاختراق تنصيب مجموعة من التّطبيقات على نظام التشغيل لمراقبة أداء النّظام وتوليد تقارير بذلك، ويتميّز بكونه أكثر دقّة من بقيّة أنواع نظم كشف الاختراق لأنه يتعامل مع سجلات النّظام التي تحوي كافة الأحداث التي يتعرض لها نظام التشغيل وبالتّالي يكون قادراً على إنشاء تقارير بالهجمات التي حدثت حتّى بعد انتهاء الهجمات، ولكن مشكلة هذا النوع أنه يستهلك جزء كبير من الموارد العتادية للحاسب المضيف.[19]

- Network -ased Intrusion Detection System (NIDS)

يتوضع هذا النّظام ضمن الشّبكة ويقوم بمراقبة الطّرود التي تمّر من النقطة التي يتوضع فيها، ويركز بشكل أساسي على نقاط الضعف في الشّبكة التي تسمح للمخترق بالوصول إلى الحواسب المرتبطة بالشّبكة، وبالمجمل يحتاج هذا النوع إلى موارد أقل من نظام HIDS، إلا أنه قادر فقط على تحليل الطّرود التي تمّر عبر النقطة التي يتوضع فيها، كما أنه غير قادر على تحليل المعطيات المشفرة.

يبيّن الشّكل التّالي مراحل عمليّة كشف الاختراق:



الشّكل **2-2**: مراحل كشف الاختراق في نظام **NIDS**

حيث يصل إلى النّظام نسخة من كافة الطّرود التي تمّر عبر النقطة التي يتوضع فيها في الشّبكة، وفي حال اكتشاف وجود اختراق يتمّ إرسال تنبيه بذلك.[19]

- Mixed Intrusion Detection System (MIDS)

يجمع بين النوعين السابقين فيستفيد من مزايا كل منهما ليعطي قرار أكثر دقّة إلا أنه يتطلب وقتاً أطول لتحليل المعطيات واتّخاذ القرار.[19]

- أنواع أخرى من نظم كشف الاختراق

يوجد مجموعة أخرى من أنواع نظم كشف الاختراق إلا أنها لا تحظى بشعبية النظم المذكورة سابقاً، نذكر منها:

- Protocol-Based Intrusion Detection System (PIDS): مشتق من نظام NIDS ويهتمّ بمراقبة أداء بروتوكول معيّن (HTTP مثلاً).
- Wireless Intrusion Detection System (WIDS): مشتق أي من نظام NIDS ويهتمّ باكتشاف محاولات الاختراق ضمن الشّبكات اللاسلكية.
- Database Intrusion Detection System: يهتمّ بمحاولة اكتشاف الهجمات التي تتعرض لها قواعد المعطيات.[19]

يبيّن الجدول التّالي محاسن ومساوئ أنواع النظم التي تمّ ذكرها:

جدول **2-1**: محاسن ومساوئ نظم كشف الاختراق

النّظام	المحاسن	المساوئ
HIDS	• قادر على معالجة المعطيات المشفرة • قادر على تحديد ما إذا نجح الهجوم أو فشل • سهولة تنصيبه فهو لا يحتاج عتاد إضافي	• يتوقف إذا حدث عطل في نظام التشغيل • يحتاج لكمية موارد كبيرة
NIDS	• يتوضع خارج الأجهزة المرتبطة بالشّبكة فلا يؤثر على أدائها	• لا يمكنه تحديد ما إذا نجح الهجوم أو فشل • لا يمكنه التعامل مع المعطيات المشفرة • لديه معرفة محدودة عن أجزاء الشّبكة
MIDS	• يمتلك مزايا النوعين السابقين • أكثر فعالية	• يتطلب كمية موارد أكثر من النوعين السابقين
PIDS	• أكثر سهولة في اكتشاف الهجمات نتيجة التركيز على بروتوكول واحد	• لا يكتشف الهجمات التي تحدث في طبقة تحت طبقة التّطبيقات
WIDS	• قدرة على التعامل مع خصوصية الشّبكات اللاسلكية	• العقد غالباً لديها قدرة حسابية أقل وسعة محدودة للبطاريات

2-5. تقنيّات كشف الاختراق

يوجد تقنيتين أساسيتين تستخدمان في نظم كشف الاختراق

2-5-1. كشف الشّذوذ Anomaly Detection

تعتمد هذه الطّريقة على معرفة السلوك الطبيعي للشبكات والأنظمة الحاسوبية، وأي سلوك بعيد نوعاً ما عن السلوك الطبيعي يتمّ اعتباره محاولة اختراق، ويتمّ تحديد السلوك من خلال مراقبة الشّبكة أو النّظام الحاسوبي وجمع مجموعة من الإحصائيّات التي تعبر عن حالة النّظام ثمّ تتمّ المقارنة مع تلك الإحصائيّات التي تمّ جمعها في لحظات عدم التعرض لمحاولات اختراق، وفي حال التشابه يتمّ اعتبار السلوك الحالي سليماً ولا يتمّ اعتبار وجود محاولة اختراق.[18]

لكن سلوك النّظام قد يتغير بشكل سريع ممّا قد يؤدّي لحدوث بعض الأخطاء التي تتضمن عدم التحسس لوجود محاولة اختراق، أو إرسال تنبيه في لحظة كان السلوك فيها سليماً.

وتتميّز هذه التقنية بالقدرة على اكتشاف هجمات جديدة لا وجود لأي معلومات مسبقة عنها على عكس تقنية الكشف المعتمد على التواقيع.

2-5-2. الكشف المعتمد على التّواقيع Signature-Based Intrusion Detection

كل نوع من الهجمات يتمّيز بنمط معيّن لا يحيد عنه، ممكن أن يكون هذا النمط عبارة عن قيم معيّنة لمجموعة إحصائيات عن نظام التشغيل أو تتالي معيّن من الطّرود التي تمّر عبر الشّبكة، ويمكن تشبيه هذا النمط بتوقّع خاص لهذا الهجوم.

تعتمد تقنية الكشف المعتمد على التواقيع على مقارنة السلوك الحالي للشبكة أو النّظام الحاسوبي مع مجموعة من التواقيع الخاصة بمجموعة من الهجمات مخزنة في قاعدّة بيانات مرتبطة بنظام كشف الاختراق، ويتمّ إرسال إنذار بوجود محاولة اختراق في حال تشابه السلوك الحالي مع أحد التواقيع. [18]

تتمّيز هذه النقنية بأنها ذات دقّة أعلى في كشف محاولات الاختراق، لكنها غير مجدية في حالات الهجمات الجديدة التي لا يملك النّظام معلومات عن سلوكها، بالإضافة إلى أنها تتطلب وقت أو موارد أكثر لاتّخاذ القرار.

6-2. وظائف نظم كشف الاختراق

يقوم نظام كشف الاختراق بالعديد من الوظائف أهمها:[46]

- التعرف على الأنماط: حيث تقوم نظم كشف الاختراق بالتعرف على محاولات الاختراق التي تقابل هجمات معرفة لديها مسبقاً.
- إعداد تقارير عن الاختراقات: تقوم نظم كشف الاختراق بإعداد تقارير عن محاولات الاختراق التي تمّ كشفها، ممّا يساعد المسؤول عن أمن النّظام في تحليل نقاط الضعف واتّخاذ الإجراءات الكفيلة بحمايته.
- مراقبة حركة البيانات: تقوم نظم كشف الاختراق بتحليل حركة البيانات ضمن النّظام ورصد نشاط المستخدمين لرصد أي تصرف غير طبيعي.
- تنبيه مسؤول أمن النّظام: تقوم نظم كشف الاختراق بإرسال تنبيهات إلى المسؤولين عن أمن الشّبكة عند حدوث أي محاولة اختراق.
- قابلية كشف الهجوم في مراحله الأولى أثناء قيام المهاجم بفحص الشّبكة لمعرفة نقاط ضعفها كي يستغلها في هجومه.
- تتبع أداء المستخدمين: تعمل نظم كشف الاختراق على تطبيق سياسات الأمان في النّظام وتقوم بتنبيه مسؤولي أمن النّظام عن أي محاولة لانتهاك هذه السياسات.

7-2. محدوديّة نظم كشف الاختراق

تعاني أنظمة كشف الاختراق من عدّة مشاكل تحد من أدائها، نذكر منها: [46]

- الحاجة الدائمة لتحديث قواعد المعطيات الخاصة بها بغية التعرف على الهجمات الجديدة وأنماطها.
- عدم القدرة على التعامل مع تقنيات هجوم جديدة مبتكرة من قبل المهاجمين وبالتّالي عدم القدرة على استباق الهجوم قبل حدوثه.
- الحاجة لاستهلاك كمية كبيرة من موارد النّظام لمراقبة حركة المعطيات وتفاعل المستخدمين ضمن النّظام وإعداد تقارير بذلك.

8-2. تقييم نظم كشف الاختراق

عند تقييم أداء نظم كشف الاختراق غالباً ما يتمّ البدء بحساب مصفوفة الالتباس التي تعبر عن أداء نظام كشف الاختراق، وتحوي المعلومات التّالية: [20]

- الصواب الموجب True Positive (TP): عدد محاولات الاختراق التي تمّ كشفها بشكل صحيح.
- الخطأ الموجب False Positive (FP): عدد التنبيهات بوجود محاولات اختراق علماً أن سلوك النّظام كان سليماً.
- الصواب السالب True Negative (TN): عدد المرات التي تمّ اعتبار سلوك النّظام سليماً عندما كان السلوك سليماً.
- الخطأ السالب False Negative (FN): عدد الهجمات التي وقعت على النّظام والتي لم يتمّكن نظام كشف الاختراق من اكتشافها.

ومن المعاملات السّابقة يمكن حساب معاملات جديدة تعبر بشكل أفضل عن أداء نظام كشف الاختراق وهي:[20]

- معدل التّصنيف Classification Rate (CR): ويسمى في بعض المراجع (Accuracy)، يعبر عن دقّة نظام كشف الاختراق في تصنيف الحالات التي يخضع لها النّظام، ويعطى بالعلاقة:

$$CR = \frac{TP + TN}{TP + FP + TN + FN} \qquad (1)$$

- معدل الكشف Detection Rate (DR): وسيمى في بعض المراجع الحساسية (sensitivity)، يعبر عن عدد محاولات الاختراق التي تمّ اكتشافها بالنسبة لعدد محاولات الاختراق الكلي، ويعطى بالعلاقة:

$$DR = \frac{TP}{TP + FN} \qquad (2)$$

- معدل الخطأ الموجب False Positive Rate (FPR): يعبر عن عدد حالات السلوك السليم التي تمّ اعتبارها محاولة اختراق إلى العدد الكلي لحالات السلوك السليم، ويعطى بالعلاقة:

$$FPR = \frac{FP}{FP + TN} \qquad (3)$$

- الدقّة Precision: هو نسبة عدد محاولات الاختراق التي تمّ اكتشافها إلى عدد الحالات التي تمّ اعتبارها محاولة اختراق، ويعطى بالعلاقة:

$$PR = \frac{TP}{TP + FP} \qquad (4)$$

2-9. دراسات في مجال نظم كشف الاختراق

قدّم [7] دراسة عرض فيها نظام لكشف الاختراق في الشّبكات التقليدية يعتمد على شبكة عصبونية عميقة، وتمّ دراسة أثر تغيير معاملات الشّبكة العصبونيّة وتبين أن استخدام تابع ReLU كتابع تفعيل (Activation function) للشبكة العصبونيّة يعطي نتائج أفضل من استخدام توابع أخرى مثل sigmoid و hyperbolic tangent التي تبين أنها عرضة لمشكلة vanishing gradient حيث تصبح تغيرات تابع الكلفة للشبكة العصبونيّة صغيرة جداً بحيث يصبح من غير المفيد زيادة تدريب الشّبكة.

تمّ تغيير عدد العصبونات في كل طبقة بين 2 و 1280 وتبين أن تغيير عدد العصبونات لا يعطي تحسناً ملحوظاً في النّتائج، ثمّ تمّت دراسة أثر تغير معدل تعلّم الشّبكة وتبين أن القيمة0.01 تعطي أفضل نتيجة.

وبتغيير عدد الطبقات المخفية وبمقارنة النّتائج مع مجموعة من خوارزميّات تعلّم الآلة بعد استخدام KDDCup-99 dataset للتدريب والاختبار، وأظهرت النّتائج أن شكبة عصبونية عميقة بثلاث طبقات مخفية أعطت أفضل أداء. حيث تمّ الحصول على دقّة 93%.

أما [8] فقد قدّم مقترح لنظام كشف اختراق يعمل ضمن شبكات SDN، قام فيه بمقارنة أداء مجموعة من خوارزميّات تعلّم الآلة ضمن شبكات SDN حيث يتمّ الاعتماد على الإحصائيّات التي تؤمّنها شبكة SDN دون الحاجة لمعاينة كل packet على حدة.

تمّ محاكاة شبكة SDN تتألف من SDN-Controller و SDN-Switch بحيث يقوم المتحكّم بطلب إحصائيات من المبدّل ضمن فواصل زمنية ثابتة ويتمّ تمّرير هذه الإحصائيّات على مجموعة من خوارزميّات تعلّم الآلة.

تمّ تجميع الـ dataset على مرحلتين:

- المعطيات الغير مرتبطة بهجمات: بالاعتماد على معطيات من مجموعة من الـ domains الخالية من الاختراقات وفقاً لموقع Alexa.
- المعطيات المرتبطة بهجمات: تمّ إجراء محاكاة لشبكة وإجراء مجموعة هجومات عليها وتجميع المعطيات الخاصة بهذه الهجمات.

وبينت النّتائج أن خوارزميّة Random Forest أعطت أفضل نتائج حيث تمّ الحصول على كشف صحيح للصفوف السليمة بمعدل 96.3% ومعدل خطأ إيجابي 0.009.

بشكل مشابه لـ [8] قدّم [9] مقترح لنظام يعمل ضمن شبكات SDN، ويتألف هذا النّظام من 4 أجزاء:

- الجزء الأول مسؤول عن تجميع إحصائيات عن الدّفق الذي يمر عبر الشّبكة.
- الجزء الثاني مسؤول عن الحصول على إحصائيات جديدة انطلاقاً من الإحصائيّات التي تمّ الحصول عليها من الجزء الأول.
- الجزء الثالث يقوم بتمّرير الإحصائيّات على مجموعة من خوارزميّة تعلّم الآلة لتصنيف الدّفق المرتبط بهذه الإحصائيّات.
- الجزء الرابع عبارة عن متحكّم يراقب عمل بقيّة الأجزاء.

تمّ محاكاة شبكة SDN توفّر مجموعة من الخدمات وتتألف من SDN-Controller و SDN-Switch ومجموعة Clients ترسل معطيات عبر الشّبكة، وللحصول على هجمات على الشّبكة تمّ الاستعانة بمجموعة من الأدوات Telnet, Nmap, Hydra, Hping3.

وأظهرت النّتائج أن خوارزميّة Hierarchical Learning Vector Quantization (Hierarchical LVQ1) أعطت أفضل معدل كشف ناجح لكل أنواع الهجمات وأقل معدل false alarm لكل الهجمات باستثناء U2R.

وقدّم [10] مقترح لنظام كشف اختراق يعمل ضمن شبكات SDN، هذا النّظام قادر على استخلاص إحصائيات دورية عن الدّفق الذي يمر ضمن الشّبكة ومن ثمّ استنتاج إحصائيات إضافية وتمّريرها إلى خوارزميّة Support Vector Machine لتصنيف الدّفق الموافق لهذه الإحصائيّات.

بهدف تخفيف الضغط عن خوارزميّة التّصنيف تمّ حساب المعلومات المتبادلة بين كل زوج من الإحصائيّات واختيار الإحصائيّات الأكثر استقلالاً فيما بينها.

تمّ التّدريب والاختبار باستخدام dataset تمّ تجميعها محلياً، وقد تمّ حساب النّتائج في 4 حالات:

- إدخال كافة الإحصائيّات إلى خوارزميّة التّصنيف واستخدام 33% من الـ dataset للتدريب.
- إدخال كافة الإحصائيّات إلى خوارزميّة التّصنيف واستخدام 50% من الـ dataset للتدريب.
- إدخال الإحصائيّات الأكثر استقلالاً فيما بينها إلى خوارزميّة التّصنيف واستخدام 33% من الـ dataset للتدريب.
- إدخال الإحصائيّات الأكثر استقلالاً فيما بينها إلى خوارزميّة التّصنيف واستخدام 50% من الـ dataset للتدريب.

- ويبيّن الشّكل التّالي أن حالة إدخال كافة الإحصائيّات مع استخدام 50% من الـ dataset للتدريب أعطى أفضل معدل كشف صحيح وأقل معدل false alarm.

يقدّم [11] مقترح لنظام كشف اختراق يعمل ضمن شبكات SDN، ويتألف هذا النّظام من جزئين:

- Snort Intrusion Detection System وهو نظام كشف اختراق من النوع Signature based
- Anomaly based Intrusion Detection System يعتمد على شبكة عصبونية بطبقة مخفية وحيدة.

كل الدّفق الذي يعبر الشّبكة يمر عبر الجزء الأول من نظام كشف الاختراق، كما يقوم المتحكّم ضمن شبكة SDN باستخلاص مجموعة من الإحصائيّات عن الدّفق الذي يمر عبر الشّبكة ويمررها إلى الجزء الثاني من نظام كشف الاختراق حيث يتمّ تصنيف الدّفق.

تمّ الاعتماد على NSL-KDD dataset في عمليّة تدريب الشّبكة العصبونيّة، وتمّ اختبار عمل النّظام بالاعتماد على الجزء الخاص بالتّدريب من NSL-KDD dataset بالإضافة إلى توليد مجموعة من الهجمات باستخدام الأداة Parrot، وبالمحصلة تمّ الحصول على كشف بدقّة وصلت لـ 97.4%.

في حين يقدّم [12] دراسة مشابهة لنظام كشف اختراق يعمل ضمن شبكات SDN، ويعتمد على شبكة عصبونية عميقة تحتوي 3 طبقات مخفية.

يقوم النّظام باستخلاص مجموعة من الإحصائيّات عن الدّفق الذي يمر ضمن الشّبكة ويمررها إلى الشّبكة العصبونيّة التي تقوم بتحديد ما إذا كان الدّخل سليماً أو مرتبطاً بهجوم.

لتدريب واختبار النّظام تمّ استخدام NSL-KDD dataset التي تحتوي مجموعة من الدّفق، يرتبط بكل دفق معطيات مجموعة من الإحصائيّات عددها 41، يمكن لشبكة SDN استخلاص 6 من هذه الإحصائيّات.

وبمقارنة أداء النّظام الذي يأخذ الإحصائيّات الـ 6 التي تؤمّنها شبكة SDN مع مجموعة من خوارزميّات تعلّم الآلة التي تأخذ كافة الإحصائيّات الـ 41 التي تؤمّنها الـ dataset وأعطى النظام المقترح دقة مقبولة وصلت إلى 75.75%

وبمقارنة أداء النّظام مع بقيّة الخوارزميّات في حال كانت تأخذ الإحصائيّات الـ 6 التي تؤمّنها شبكة SDN كانت نتيجة النّظام المقترح هي الأفضل

بالنظر إلى النّتائج التي قدّمها [8] و [9] و [10] نجد أن خوارزميّات مثل Random Forest و SVM و Hierarchical LVQ1 تعطي نتائج ممتازة بالمقارنة مع خوارزميّات أخرى، إلا أن القيم التي تمّ عرضها تبقى محل شك على اعتبار أن هذه المراجع لم تستخد dataset معيارية في عمليّات التّدريب والاختبار.

وبالنظر إلى النّتائج المعروضة في [7] و [11] و [12] التي استخدمت dataset معيارية نجد تفوق الشّبكات العصبونيّة العميقة على خوارزميّات تعلّم الآلة، ونجد أن [11] قدّم دقّة كشف وصلت إلى 97%، إلا أن هذه النتيجة محل شك أيضاً بسبب الاعتماد على مجموعة من الهجومات المعروفة لدى الـ signature based IDS الموجود ضمن النّظام المقترح، بالإضافة إلى اختبار أداء النّظام بالاعتماد على الجزء من الـ dataset المستخدم في عمليّة التّدريب دون الجزء المفترض استخدامه في عمليّة الاختبار.

الفصل الثالث

تعلّم الآلة

Machine Learning

نقدّم في هذا الفصل مقدّمة عن مفهوم تعلّم الآلة، وملخصّاً عن أساليب تطبيقه، بالإضافة إلى شرح عن مجموعة من الخوارزميّات المستخدمة في مشاكل التّصنيف.

1-3. مقدّمة وتعاريف

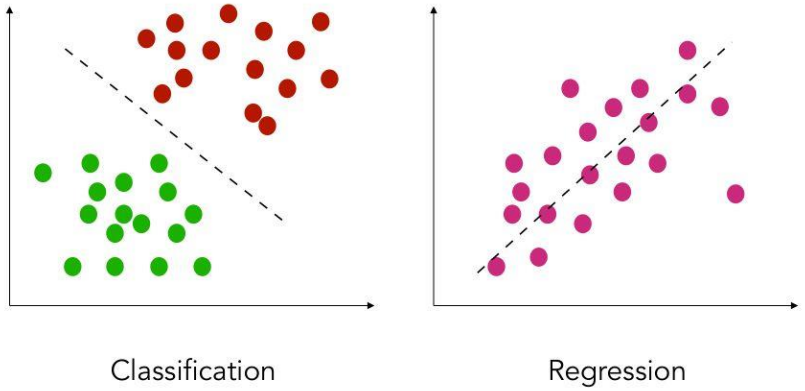
قبل تقديم تعريف لمفهوم تعلّم الآلة لا بدّ من تعريف الذّكاء الاصطناعي، الذي عرفه John McCarthy في عام 1956 على أنّه دراسة وتصميم أنظمة قادرة على فهم وإدراك البيئة المحيطة واتّخاذ الإجراءات المناسبة لزيادة فرص تحقيق هدف معيّن [24]، ويعتبر تعلّم الآلة أحد فروع الذّكاء الاصطناعي، مسؤول عن جعل النّظام الحاسوبي قادراً على إيجاد حل لمشكلة ما من خلال دراسة مجموعة من البيانات (بيانات التّدريب) ثمّ إيجاد أنماط معيّنة ضمن هذه المجموعة بحيث يصبح النّظام قادراً على بناء نموذج يعبّر عن المشكلة التي تتمّ دراستها.

ويختلف تعلّم الآلة عن البرمجة التقليديّة من ناحية طبيعة الخرج، ففي حالة برمجة النّظام الحاسوبي بخوارزميّة ما، يتمّ تغذية هذه الخوارزميّة بقيم معيّنة على الدّخل فتعطي قيم مقابلة على الخرج، أما في حالة تعلّم الآلة، فنحصل على خرج خوارزميّة تعلّم الآلة على نموذج (خوارزميّة) قادر على استقبال دخل معيّن وإعطاء خرج مقابل لهذا الدّخل.

يمكن تقسيم تعلّم الآلة إلى 3 أنواع [25]:

- التّعلم الخاضع للإشراف Supervised Learning: في هذه الطّريقة يتمّ تدريب نموذج تعلّم الآلة باستخدام بيانات مصنّفة بشكل صحيح لتشكل نقطة انطلاق يتعلّم منها النّظام بهدف إيجاد النّموذج المناسب للمسألة المطروحة، ويستخدم التّعلم الخاضع للإشراف بشكل أساسي في:
    - التّصنيف Classification: في هذه الحالة يكون فضاء الاحتمالات مقسّم إلى مجموعة محدودة من الصّفوف (في حالة صفّين يسمى تصنيف ثنائي Binary-Classification، وفي بقيّة الحالات يسمى التّصنيف متعدّد الصّفوف multi-class classification)، والهدف في هذه الحالة هو إيجاد الصّف الذي ينتمي إليه الدّخل، كاتّخاذ قرار إذا ما كانت الطّرود المارّة عبر الشّبكة سليمة أو مرتبطة بهجوم.
    - التّراجع Regression: الهدف في هذه الحالة هو توقّع قيمة مستمرة تقابل معطيات الدّخل (توقّع سعر منزل في منطقة معيّنة ومساحة معيّنة)، ويختلف نوع التّراجع باختلاف شكل النّموذج النّاتج عن خوارزميّة تعلّم الآلة والذي يعبّر عن العلاقة بين الدّخل والخرج (تراجع خطّي Linear Regression إذا كان النّموذج النّاتج يتمّثل بعلاقة خطيّة بين الخرج والدّخل، بالإضافة إلى أنواع أخرى مثل Logistic Regression، وPolynomial Regression، وغيرها).

ويبيّن الشّكل التّالي الفرق في توزّع المعطيات في حالتي التّصنيف الثّنائي والتّراجع الخطي:



الشّكل **3-1**: توزع المعطيات في حالتي التّصنيف الثنائي والتراجع الخطي

- التّعلم غير الخاضع للإشراف Unsupervised Learning: في هذه الحالة يتمّ تدريب نموذج تعلّم الآلة باستخدام مجموعة معطيات غير مصنّفة وغالباً ما تكون هذه المعطيات لا تتبع بنية محدّدة، وعلى النّموذج إيجاد الطّريقة المناسبة للتّعلم من هذه المعطيات، ويستخدم التّعلم غير الخاضع للإشراف بشكل أساسي في:
  - العنقدة Clustering: في هذه الحالة يحاول النّظام إيجاد الأنماط المتشابهة وتجميعها ضمن صف واحد، وهي مشابهة لحالة التّصنيف متعدّد الصّفوف إلا أنّ النّظام في هذه الحالة يكون جاهلاً بعدد الصّفوف وبالمعنى الذي يعبّر عنه كل صف.
  - Association: تساعد هذه الطّريقة في إيجاد التّرابط بين مجموعة كبيرة من المعطيات، وتستخدم بشكل كبير في خوارزميّات التّنقيب عن المعطيات.
- التّعلم شبه الخاضع للإشراف Semi-Supervised Learning: إحدى طرق تعلّم الآلة التي يتمّ استخدامها عند عدم توفّر كميّة كافية من المعلومات المصنّفة ليتمّ استخدامها في تدريب نموذج تعلّم الآلة.

ونظراً لكون مشكلة البحث مشكلة تصنيف، سيتمّ عرض مجموعة من خوارزميّات تعلّم الآلة البارزة في هذا المجال.

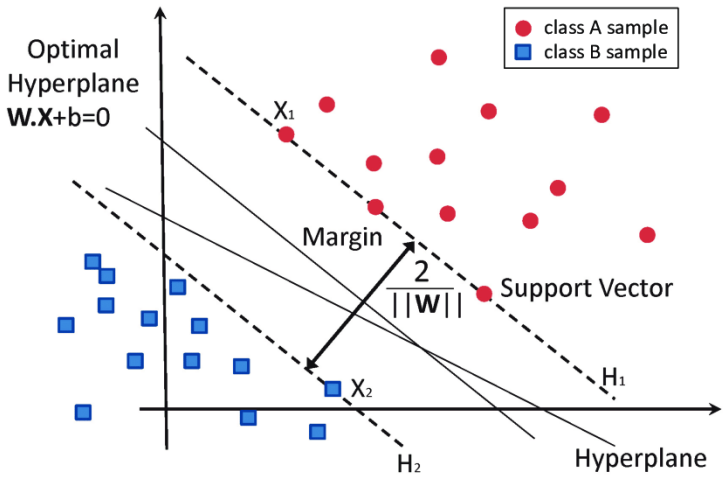
2-3. خوارزميّات التّصنيف

سيتمّ عرض مجموعة من خوارزميّات تعلّم الآلة المستخدمة في مشاكل التّصنيف

Support Vector Machine (SVM) .1-2-3

قدّم Boser و Guyon و Vapnik هذه الخوارزميّة في ورقة بحثية في عام 1992 [26]، ولا تزال تستخدم حتّى الآن في مسائل التّصنيف نظراً للدقّة العالية التي تؤمّنها والقدرة على التّعامل مع معطيات من عدّة أبعاد.

تُصنَّف هذه الخوارزميّة ضمن الخوارزميّات التي تستخدم توابع النّواة (Kernel methods)، وتهدف إلى إيجاد أفضل الفواصل بين صفوف المعطيات هندسيّاً، ويتمّ ذلك من خلال حساب الجداء الدّاخلي (المسافة) بين كل زوج من المعطيات [27]، ويبيّن الشّكل التّالي مثالاً عن اختيار الفواصل:

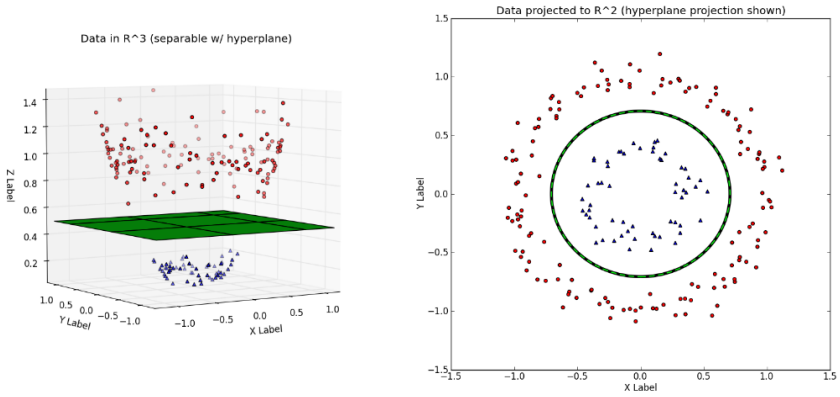


الشّكل **2-3**: اختيار الفواصل في خوارزميّة **SVM**

يمكن أن يوجد عدّة فواصل بين صفوف المعطيات لكن أفضلها هو الذي يعطي أكبر هامش (المسافة بين الفاصل وأقرب نقاط الصف إليه).

في معظم الحالات لا يمكن إيجاد فاصل يناسب توزع المعطيات، فيتمّ استخدام توابع النّواة التي تساعد في إسقاط المعطيات على فضاء آخر ذو بعد مختلف وإيجاد المسافة بين كل زوج من المعطيات في الفضاء الجديد، بحيث تتوزع المعطيات في الفضاء الجديد بشكل يساعد على إيجاد فواصل أفضل (تقوم توابع النّواة بحساب المسافة بين كل زوج من المعطيات في الفضاء الجديد دون الحاجة لإيجاد مسقط كل نقطة في هذا الفضاء).

ويبيّن الشّكل التّالي مثالاً عن إسقاط المعطيات إلى فضاء جديد:



الشّكل **3-3**: إسقاط المعطيات إلى فضاء جديد باستخدام توابع النواة

Decision Trees .2-2-3

يتمّ في هذه الخوارزميّة بناء نموذج التّصنيف على شكل مخطّط شجري، حيث تمثّل كل عقدة داخلية اختبار لـ سمة من سمات مجموعة المعطيات، بينما تمثّل الأوراق الصّفوف التي تنتمي إليها المعطيات.

يتمّ بناء شجرة القرار على مرحلتين:

- Induction: يتمّ في هذه المرحلة اختيار جذر الشّجرة وإنشاء أشجار فرعيّة، ثمّ تكرار العمليّة على جذور الأشجار الفرعيّة النّاتجة، والهدف من كل عمليّة تقسيم (إنشاء أشجار فرعيّة جديدة) هو الحصول على شجرة فرعيّة تحوي أكبر عدد ممكن من عناصر مجموعة المعطيات التي تنتمي لصف معيّن دون التّداخل مع عناصر تنتمي لصفوف أخرى، وتتمّ هذه العمليّة على عدّة مراحل:
  - اختيار أفضل سمة من سمات مجموعة المعطيات ويتمّ ذلك بحساب كميّة المعلومات المتبادلة بين كل سمة مع كل صف، والسّمة التي توافق أكبر كميّة معلومات متبادلة تعتبر الأفضل (خوارزميّة ID3)، وحساب Gini Index Functions الذي يعطي فكرة عن جودة التّقسيم الحاصل على كل عقدة من خلال إعطاء نسبة المعطيات التي تنتمي لكل صف في الفرع الجديد (خوارزميّة CART). [28]
  - تقسيم مجموعة المعطيات إلى مجموعات جزئيّة بناءً على قيم السّمات المختارة في المرحلة السّابقة.
  - تكرار المرحلتين السابقتين على كل من المجموعات الجزئيّة النّاتجة. يجب الانتباه إلى أن زيادة عمق الشّجرة يؤدّي إلى كلفة حسابيّة أعلى، بالإضافة إلى أنه ليس من الضروري أن يعطي دقّة كشف أعلى، لأنه قد يؤدّي لمشكلة over-fitting.

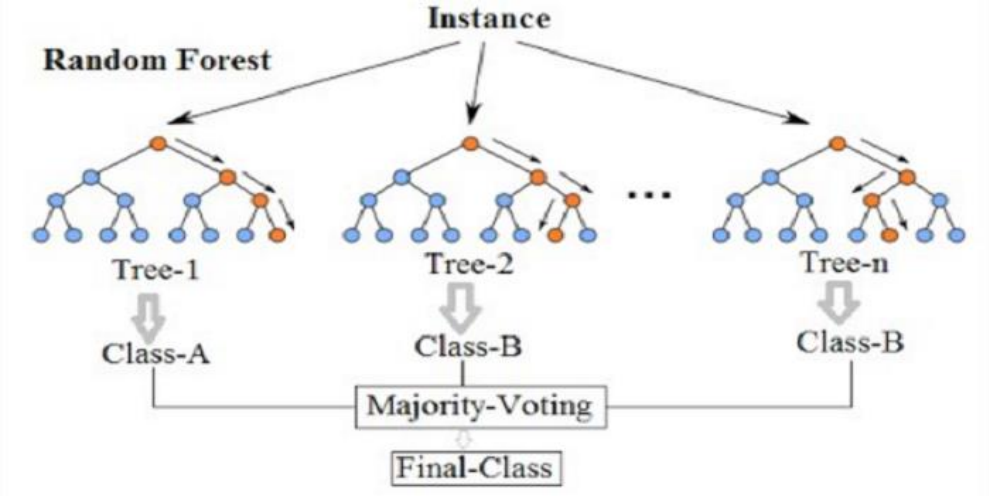
- Pruning: يتمّ في هذه المرحلة اختيار حدّ أدنى لعدد عناصر مجموعة المعطيات التي يُسمَح أن تنتمي لصف معيّن ضمن فرع من فروع الشّجرة، ويتمّ حذف أي فرع يحوي عدد عناصر أقل من الحد الأدنى، وتفيد هذه المرحلة في تجنّب مشكلة over-fitting حيث يصبح نموذج التّصنيف مناسب بشكل كبير لمجموعة المعطيات التي تدرّب عليها ويصبح غير مناسب لمجموعة معطيات أخرى. [28]

Random Forest .3-2-3

يتألّف هذا النّموذج من عدد كبير من أشجار القرار، حيث يتمّ تدريب كل شجرة بجزء من مجموعة المعطيات، وعند اختبار نموذج التّصنيف النّاتج يتمّ إجراء تصويت بين هذه الأشجار، وتعتمد هذه الخوارزميّة على مفهومين أساسيين:

- التّقسيم العشوائي لمجموعة المعطيات المستخدمة في تدريب النّموذج: حيث يتمّ اختيار مجموعة معطيات جزئيّة لتدريب كل شجرة بتطبيق طريقة السحب مع الإعادة على مجموعة المعطيات الكليّة، وبالتّالي يمكن تواجد نفس العيّنة عدّة مرّات في إحدى مجموعات العيّنات الجزئيّة، وتسمّى عمليّة تدريب كل شجرة على حدة باستخدام مجموعة معطيات مختلفة باسم Bootstrap Aggregation (Bagging).
- التّقسيم العشوائي لسمات مجموعة المعطيات: بالإضافة إلى أنّ تدريب كل شجرة يتمّ باستخدام مجموعة معطيات جزئيّة، فإنّ هذه المجموعات الجزئيّة تحتوي فقط على جزء من السّمات المتوفّرة في مجموعة المعطيات الأصلية، ويتمّ اختيار السّمات التي ستضمّها كل مجموعة جزئيّة بشكل عشوائي، وغالباً ما تحوي كل مجموعة جزئيّة على عدد سمات يساوي الجذر التّربيعي لعدد السّمات الكليّة الموجودة في مجموعة المعطيات الأصلية. [29]

يبيّن الشّكل التّالي طريقة عمل خوارزميّة Random Forest



الشّكل **3-4**: طريقة عمل خوارزميّة **Random Forest**

4-2-3. Extreme Gradient Boosting (XGBoost)

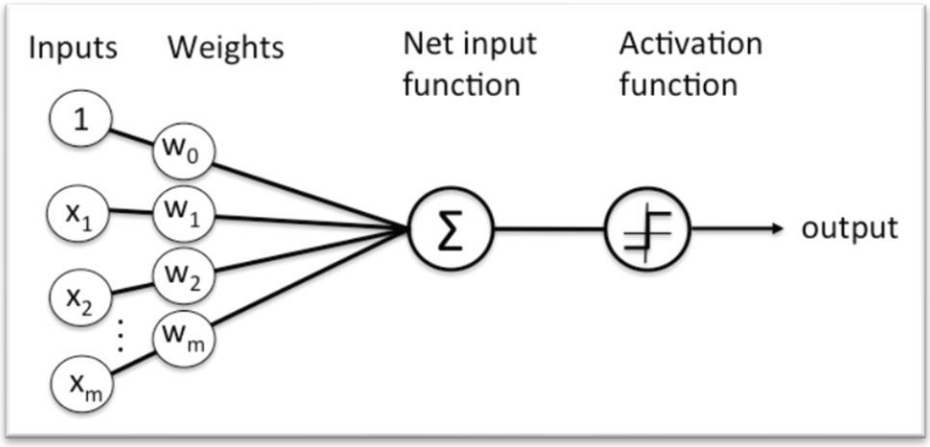
في حالة خوارزميّة Random Forest تمّ استخدام طريقة Bootstrap Aggregation التي تقوم على توليد مجموعة من نماذج التّصنيف باستخدام أشجار القرار، وتدريب كل منها على مجموعة جزئيّة من مجموعة المعطيات وإجراء تصويت بين هذه النّماذج (تدريب تفرّعي)، بينما في خوارزميّة XGBoost فيتمّ استخدام طريقة Boosting التي تعتمد على توليد نموذج/نماذج تصنيف باستخدام أشجار القرار وتدريبها على مجموعة جزئيّة من مجموعة المعطيات فنحصل على نموذج ضعيف بمواصفات سيّئة، ثمّ يتمّ اختبار النّموذج على كافة المعطيات، وبناءً على نتيجة الاختبار يتمّ توليد مجموعة جزئيّة جديدة يتمّ تدريب النّموذج عليها فنحصل على المرحلة الثّانية من النّموذج، ثمّ يتمّ اختبار النّموذج على كافّة المعطيات ويتمّ توليد مجموعة جزئيّة جديدة ويتمّ تكرار هذه العمليّة حتّى الحصول على نموذج أفضل بعد عدد معيّن من التكرارات.

5-2-3. الشّبكات العصبونيّة Neural Networks

تحاول هذه النّماذج محاكاة الدّماغ البشري الذي يحتوي على عدد كبير من العصبونات المتّصلة مع بعضها مكونةً شبكة من العصبونات والوصلات، وتكون هذه الشّبكة في الدّماغ البشري قادرة على اتّخاذ قرارات معقّدة وسريعة مثل التّعرف على الوجوه والأصوات، أما في حالة تعلّم الآلة فالشّبكة العصبونيّة هي مجموعة من العمليّات الحسابيّة التي تحاول محاكاة عمل الدّماغ البشري لإعطاء الإدراك للآلة بحيث تتعلّم وتكشف الأنماط بالاستفادة من مجموعة المعطيات المستخدمة.

وتتألف الشّبكة العصبونيّة من الأجزاء التّالية: [30]

- العصبون: يشكّل المكوّن الأساسي والوحدة الحسابيّة الأصغر في الشّبكة، وظيفته تطبيق تابع رياضي على مجموعة من المدخلات الموزونة، ويبيّن الشّكل التّالي مخطّط لعصبون واحد:



الشّكل **3-5**: بنية العصبون

ونلاحظ أن العصبون يتألّف من مداخل وأوزان وتابع تجميع وتابع تفعيل وخرج:

- الأوزان: تكون على الوصلة بين العصبون الحالي وعصبونات من طبقة أخرى، وتعبّر عن مقدار تأثير هذه العصبونات على العصبون الحالي.
- توابع التّفعيل: هي مجموعة من التّوابع يتمّ تطبيقها على مدخلات العصبون المجمّعة للحصول على خرج موافق، وتكون هذه التّوابع لا خطيّة بهدف إضافة اللاخطيّة إلى الشّبكة العصبونيّة ممّا يزيد احتمال الحصول على نموذج أفضل، ومن التّوابع المشهورة Step functio, Sigmoid function, Tanh function ReLU function، ويمكن استخدام تابع تجميع مختلف في كل من طبقات الشّبكة.

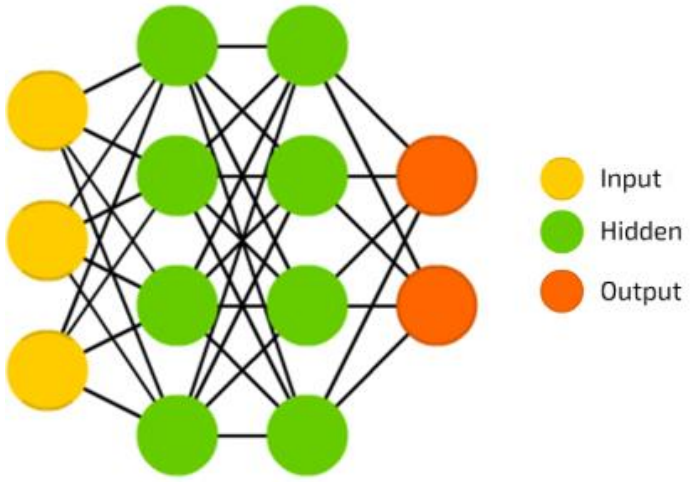
- الطّبقات: كل طبقة هي مجموعة من العصبونات التي تعمل معاً على نفس العمق ضمن الشّبكة العصبونيّة، ومنها 3 أنواع:
  - الطّبقة الأولى (طبقة الدّخل) تستقبل سمات المعطيات التي سيتمّ بناء نموذج التّصنيف بالاعتماد عليها وفيها عصبون لكل سمة من سمات مجموعة المعطيات.
  - الطّبقة الثالثة (طبقة الخرج) تمثّل التوقّع الذي تعطيه الشّبكة وفي مسائل التّصنيف غالباً ما يتمّ تخصيص عصبون لكل صف محتمل.
  - الطّبقات المخفيّة: وهي مجموعة الطبقات الموجودة بين طبقتيّ الدّخل والخرج، وقد يختلف عدد العصبونات وتوابع التّفعيل في كلّ من هذه الطبقات، وفي حال تواجد عدّة طبقات مخفيّة يطلق على الشّبكة شبكة عصبونيّة عميقة.

وقد تختلف طريقة التّواصل بين طبقات الشّبكة باختلاف نوع الشّبكة العصبونيّة، ونذكر منها الأنواع التّالية: [31]

- Perceptron: أبسط أنواع الشّبكات العصبونيّة، تأخذ مجموعة المداخل وتجمعها وتطبق عليها تابع تفعيل ثمّ تمّرر النتيجة إلى طبقة الخرج، ويبيّن الشّكل السّابق مثال عن شبكة perceptron.
- شبكات التّغذية المباشرة (Feed Forward): ترتبط كافة عصبونات طبقة معيّنة مع كل عصبون من الطّبقة التّالية، ويكون مسار المعطيات من الدّخل باتجاه الخرج دون وجود حلقات خلفيّة، وغالباً ما يتمّ تدريب هذا النوع من الشّبكات باستخدام طريقة Backpropagation التي تعتمد على إجراء عدد معيّن من التكرارات (epochs)، حيث يتمّ تمّرير معطيات التّدريب ضمن الشّبكة وحساب التّوقّع الخاص بكل عيّنة تدريب، ويتمّ اختيار تابع خطأ لحساب الخطأ بين التّوقّعات النّاتجة والخرج الأصلي، ثمّ يتمّ حساب انحدار (gradient) لتابع الخطأ وضربه بمعدل التّعلم، ثمّ يتمّ تعديل الأوزان في الشّبكة انطلاقاً من طبقة الخرج باتجاه طبقة الدّخل بهدف تقليل قيمة تابع الخطأ والوصول إلى قيمة صغرى محلياً، ثمّ يتمّ تكرار عمليّة حساب التوقّعات وتعديل قيم الأوزان حتّى الوصول إلى قيمة خطأ أصغر من حد معيّن أو بعد إجراء عدد معيّن من التّكرارات، ويبيّن الشّكل التّالي شبكة من النّوع Feed Forward.

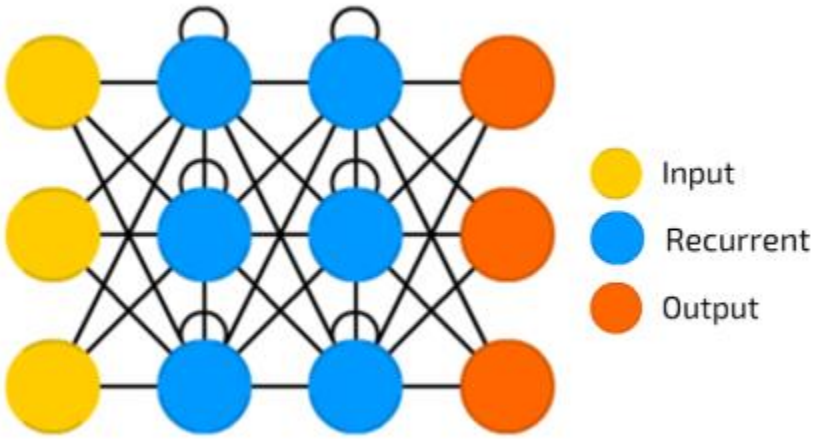
الشّكل **3-6**: شبكة عصبونيّة من النّوع **Feed Forward**

- شبكات التّغذية المباشرة العميقة (Deep Feed Forward): مشابهة لشبكات التّغذية المباشرة لكنها تحوي عدد أكبر من الطّبقات المخفيّة، ونتيجةً لذلك يزداد عدد الوصلات والأوزان في الشّبكة ممّا يساعد على بناء نماذج لحالات أكثر تعقيداً وإعطاء نتائج أفضل من شبكات التغذية المباشرة، ويبيّن الشّكل التّالي شبكة من النوع Deep Feed Forward:



الشّكل **3-7**: شبكة عصبونيّة من النّوع **Deep Feed Forward**

- الشّبكات العصبونيّة العوديّة (Recurrent Neural Network): مشابهة للنّوع السّابق، لكن في هذه الحالة تمتلك الشّبكة ذاكرة، حيث يمكن أن يستقبل أي عصبون خرج سابق (من تكرار سابق) لعصبون آخر، ويمكن الاستفادة من هذا النّوع من الشّبكات في دراسة النّصوص، حيث يعتمد الخرج الحالي على الخرج من مراحل سابقة، ويبيّن الشّكل التّالي شبكة عصبونيّة عوديّة تستقبل فيها العقد المخفيّة خرجها السابق:

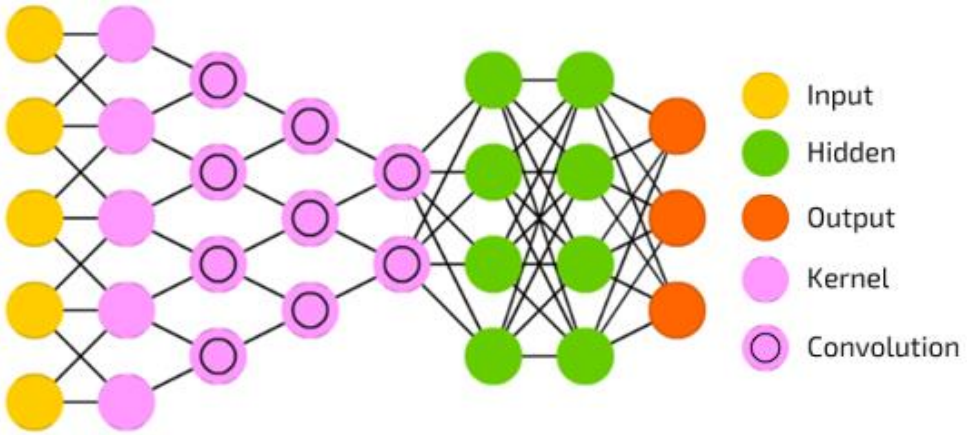


الشّكل **3-8**: شبكة عصبونيّة عوديّة

- الشّبكات العصبونيّة التلفيفيّة (Convolutional Neural Networks): يستطيع هذا النّوع من الشّبكات التّعامل مع معطيات ذات عدد كبير من السّمات (وبشكل خاص الصور)، حيث يتمّ استقبال كافّة بكسلات الصّورة في طبقة الدّخل، تليها الأنوية التلفيفيّة وهي عبارة عن مجموعة من المرشّحات يتمّ تطبيقها على معطيات الدّخل، حيث يهدف كل منها لاكتشاف سمة معيّنة (حواف، دوائر...) وينتج عن كل من هذه المرشحات مصفوفة جديدة، يتمّ تمّرير هذه المصفوفات عبر مرحلة تجميع (Pooling) مسؤولة عن تخفيض عدد العناصر في كل مصفوفة من خلال استبدال كل مجموعة من العناصر ضمن المصفوفة بالمتوسّط أو القيمة الكبرى أو غيرها.

يمكن تكرار مرحلتي الأنوية التلفيفيّة والتّجميع عدّة مرات لنحصل في النّهاية على مجموعة سمات نهائيّة يتمّ إدخالها إلى شبكة عصبونيّة تكون غالباً من النّوع Feed Forward.

ويبيّن الشّكل التّالي شبكة عصبونيّة تلفيفيّة بمرحلة أنوية ومرحلة تجميع واحدة.



الشّكل **3-9**: شبكة عصبونية تلفيفية

على الرّغم من وجود عدد كبير من خوارزميّات تعلّم الآلة إلا أن لكل منها محاسن ومساوئ، ونقدّم في الجدول التّالي ملخّص لمحاسن ومساوئ كل من الخوارزميّات المذكورة سابقاً:

الجدول **3-1**: محاسن ومساوئ خوارزميّات تعلّم الآلة

الخوارزميّة	المحاسن	المساوئ
SVM	• سهولة العرض والقدرة على تمثّيل النّتائج كمجموعة من القواعد. • سهولة التّعامل مع معطيات ذات عدد سمات كبير. • احتمال حدوث مشكلة over-fitting أقل من غيرها من الخوارزميّات. [32]	• غير مناسبة للتّعامل مع كميّة معطيات كبيرة، بسبب الحاجة لحساب البعد بين كل عيّنتين من مجموعة المعطيات. • تعاني عند التعامل مع مجموعة معطيات مضجّجة (يوجد تداخل بين الصّفوف). • تهتمّ فقط بموقع العيّنات في الفراغ دون أخذ التوزع الاحتمالي بعين الاعتبار. [32]
Decision Trees	• تقدّر أهميّة كل سمة من سمات مجموعة المعطيات خلال بناء النّموذج. • لا تحتاج معالجة مسبقة للمعطيات. • القدرة على التّعامل مع معطيات ذات قيم متقطّعة أو مستمرّة. • القدرة على التّعامل مع مجموعة معطيات تحوي عيّنات تفتقد بعض السّمات.[33]	• قد يؤدّي تغيير بسيط في مجموعة المعطيات إلى تغير كبير في النّموذج. • غير مناسبة لمسائل من نوع regression. • تتطلّب وقت وموارد أكثر من غيرها من الخوارزميّات. [33]
Random Forest	• القدرة على التّعامل مع معطيات ذات حجم كبير. • مقاومة لمشكلة overfitting أكثر من XGBoost. • تقدّر أهمية كل سمة من سمات مجموعة المعطيات في بناء النّموذج. [33]	• قد تظهر مشكلة over-fitting في حال كانت المعطيات مضجّجة. • تحتاج معايرة أكثر من Decision Trees للحصول على أفضل نتائج. • بناء عدد كبير من الأشجار بشكل تفرّعي يؤدّي إلى زيادة وقت المعالجة ممّا يجعلها غير مناسبة

		لتطبيقات الزّمن الحقيقي. • يصعب تمثّيلها وتحويلها لمجموعة من القواعد. [33]
XGBoost	• سرعة في بناء النّموذج. • القدرة على التّعامل مع معطيات ذات قيم متقطّعة أو مستمرة. • القدرة على اكتشاف العلاقات غير الخطيّة والأنماط المعقّدة التي تربط بين المعطيات. [34]	• قد تعاني من مشكلة over-fitting عندما تكون المعطيات مضجّجة. • تحتاج معايرة أكثر من Random Forest للحصول على أفضل نتائج. [34]
Feed Forward NN	• القدرة على التّعامل مع مسائل التّصنيف والregression. • القدرة على التّعامل مع معطيات ترتبط مع بعضها بعلاقات غير خطيّة. • بعد التّدريب تصبح عمليّة التنبؤ بخرج المعطيات الجديدة عمليّة بسيطة.	• من الصّعب معرفة كيفية تأثير كل متحول على أداء الشّبكة أو عن ماذا يعبّر هذا المتحول. • تستهلك الكثير من الوقت والموارد للقيام بعمليّة التّدريب. • قد تتعرض لمشكلة over-fitting.
Deep Feed Forward NN	• مشابهة للحالة السّابقة لكن تعطي نتائج أفضل.	• تستهلك وقت وموارد أكثر من الحالة السّابقة.
Recurrent NN	• القدرة على التّعامل مع معطيات مترابطة يفصلها فاصل زمني، وذلك بالاستفادة من ذاكرة هذه الشّبكات.	• مشكلة vanishing gradient حيث تصبح قيمة الـ gradient لتابع الخطأ صغيرة جداً، فتتعدّل قيم الأوزان بمقدار ضئيل، فيصبح تعلّم الشّبكة بطيء جداً. • مشكلة exploding gradient حيث تصبح قيمة الـ gradient لتابع الخطأ كبيرة، فتتعدّل قيم الأوزان بمقدار كبير ممّا يجعل الشّبكة غير مستقرّة وغير قادرة على التّعلم. [35]
Convolutional NN	• تتمّيز بسرعتها مقارنةً مع الشّبكات من	• تحتاج مجموعة معطيات بحجم كبير ممّا يجعل تدريبها

	النّوع RNN نظراً لكونها تعمل بشكل تفرعي. • القدرة استخلاص أفضل السّمات من مجموعة المعطيات.	يستهلك المزيد من الوقت والموارد. • بخلاف شبكات RNN، لا تهتمّ بترتيب ورود المعطيات ممّا يجعلها تفقد جزء من المعلومات.

3-3. الخاتمة

عرضنا في هذا الفصل موجزاً عن مجالات استخدام خوارزميّات تعلّم الآلة والشّبكات العصبونيّة، بالإضافة إلى ملخّص عن طريقة عمل مجموعة من أفضل الخوارزميّات المستخدمة في مسائل التّصنيف، ولاحظنا عدم وجود خوارزميّة مهيمنة، وإنما تفاصيل المسألة ومجموعة المعطيات المستخدمة هي التي تؤثر على نتائج كل خوارزميّة.

الفصل الرّابع

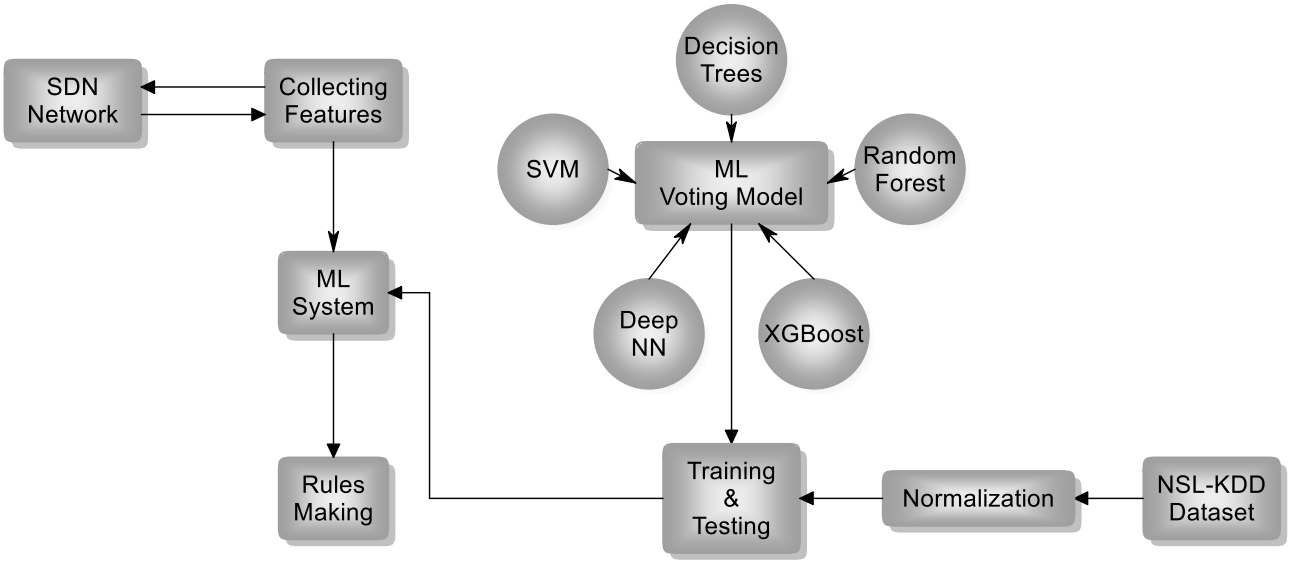
النّظام المقترح والتّنفيذ العملي

System Implementation

نقدّم في هذا الفصل النّظام المقترح ومراحل تنفيذه، بالإضافة إلى لمحة عن مجموعة المعطيات المستخدمة، وعمليّة تجهيز شبكة SDN، بالإضافة إلى الخوارزميّات المختارة لبناء نموذج تعلّم الآلة والنّتائج العمليّة.

1-4. النّظام المقترح

يقوم نظام كشف الاختراق بدور فعال في التنبيه لوجود هجمات على الأنظمة والشّبكات، ولكنه في بعض الحالات يبقى عاجزاً عند ظهور نوع جديد من الهجمات، ولحل هذه المشكلة نقترح نظام كشف اختراق يعتمد على تقنية كشف الشذوذ ويستخدم نظام انتخاب قائم على مجموعة من خوارزميّات تعلّم الآلة، بحيث يتمّ تأمين دخل نموذج تعلّم الآلة من الإحصائيّات التي تؤمّنها شبكة SDN، ويبيّن الشّكل التّالي المخطط الصندوقي للنظام المقترح:



الشّكل **1-4**: المخطط الصندوقي للنظام المقترح

ويعمل النّظام المقترح كما يلي:

- يتمّ في المرحلة الأولى الاستفادة من طريقة عمل شبكة SDN، حيث يقوم المتحكّم كل ثانيتين بطلب الإحصائيّات المسجلة لدى المبدّل عن كل دفق من خلال رسالة feature_req، ليرسل المبدّل هذه الإحصائيّات إلى المتحكّم برسالة feature_rep، ليتمّ الاستفادة منها كدخل للمرحلة التّالية.
- في المرحلة الثانية يتمّ إدخال الإحصائيّات التي تمّ الحصول عليها في المرحلة السّابقة إلى نظام انتخاب تمّ بناؤه بالاعتماد على مجموعة من خوارزميّات تعلّم الآلة (SVM، Deep Neural Networks، XGBoost، Decision Trees، Random Forest)، وقد تمّ تدريب واختبار كل من هذه الخوارزميّات بالاستعانة بمجموعة المعطيات NSL-KDD.

تتخذ كل من هذه الخوارزميّات قراراً منفصلاً فيما إذا كان الدّفق مرتبط بمحاولة اختراق أو لا، ويتمّ إجراء انتخاب بين هذه الخوارزميّات مع إعطاء أولوية أعلى للخوارزميّات التي قدّمت أداء أفضل في مرحلة الاختبار.

- في المرحلة الأخيرة يتمّ إضافة قواعد جديدة إلى جداول الدّفق تمّنع مرور الدّفق -المتوقّع ارتباطه بمحاولة اختراق- عبر الشّبكة.

قبل عرض مراحل العمل والنّتائج العمليّة لابد من التعرف على مجموعة المعطيات المستخدمة.

4-2. مجموعة المعطيات

تمّ استخدام مجموعة المعطيات المعيارية NSL-KDD لتدريب واختبار خوارزميّات تعلّم الآلة المستخدمة في نظام الانتخاب، كما تمّ استخدام مجموعة المعطيات المعيارية KDDCup99 لمقارنة النّتائج مع NSL-KDD التي تعتبر تطويراً للمجموعة KDDCup99.

تمّ بناء مجموعة المعطيات KDDCup99 بالاعتماد على مجموعة المعطيات DARPA، وهي عبارة عن 4GB من ملفات TCPdump تمّ تجميعها عبر شبكة محاكية لشبكة عسكرية دون اعتماد سياسة خصوصية. [36]

وفي عام 1999 تمّ اعتماد مجموعة المعطيات KDDCup99 بعد حذف بعض التكرارات من مجموعة المعطيات DARPA التي تؤثر سلباً على أداء نظم كشف الاختراق، كما ظهر مفهوم السّمات ضمن مجموعة المعطيات الجديدة التي أصبحت تتألف من 41 سمة تعطي صورة مفصلة عن المعايير التي تساعد في بناء نظام كشف الاختراق. [36]

وتقسم هذه السّمات إلى 4 أنواع:

- جوهرية Intrinsic: تتضمن معلومات عن مدة الاتصال، ونوع البروتوكول المستخدم (TCP, UDP, ICMP)، ونوع الخدمة (http, telnet …).
- المحتوى Content: تسمح بتوصيف الاتصال، مثل عدد المحاولات الفاشلة.
- المضيف ذاته Same Host: تحدد محاولات الاتصال التي لها نفس الوجهة في آخر ثانيتين من الاتصال الحالي.
- الخدمة ذاتها Similar Same Service: تفحص الاتصالات في آخر ثانيتين من الاتصال الحالي بحثاً عن اتصالات تشابه ذات نوع خدمة مشابه للاتصال الحالي.

كما وتقسم الهجمات إلى 4 أنواع:

- حجب الخدمة DOS: يؤدّي هذا الهجوم إلى إشغال موارد الشّبكة بحيث تصبح عاجزة عن تخديم مستخدمين مصرح لهم استخدام الشّبكة.
- Root to local (R2l): نفاذ غير مصرح به بهدف استكشاف نقاط ضعف الشّبكة.
- User to Root (U2R): الحصول على صلاحيات مستخدم محلي لاستخدام موارد الشّبكة واستنزافها.
- Probe: يتمّ فيها فحص بوابات الشّبكة بهدف جمع معلومات عنها ومعرفة نقاط ضعفها.

تحوي مجموعة المعطيات KDDCup99 ما يقارب خمسة ملايين سجل، وهو عدد كبير جداً لذلك يتمّ استخدام 10% من هذه السجلات، وتمّ تقسيمها إلى جزء خاص بالتّدريب وجزء خاص بالاختبار، وتتوزع هذه السجلات في الجدول التّالي:

الجدول **4-1**: توزع السجلات في مجموعة المعطيات **KDDCup99**

مجموعة المعطيات	السجلات السليمة	السجلات المرتبطة بمحاولة اختراق	المجموع
التّدريب	78416 (20%)	321583 (80%)	399999
الاختبار	18861 (20%)	75160 (80%)	94021

أما بالنسبة لمجموعة المعطيات NSL-KDD فقد ظهرت في عام 2006، وهي مشتقة من المجموعة KDDCup99 ولكن مع بعض التحسينات: [37]

- حذف السجلات المكررة في جزء المعطيات الخاص بالتّدريب، بحيث لا تنحاز نماذج التّصنيف للصف الأكثر تكراراً.
- حذف السجلات المكررة من جزء المعطيات الخاص بالاختبار، بحيث لا تتأثر نتائج نماذج التّصنيف بالصف الأكثر تكراراً.
- تقليل عدد السجلات في مجموعتي التّدريب والاختبار بحيث يمكن استخدام كامل المجموعة دون اختيار جزء منها كما في حالة مجموعة المعطيات KDDCup99.
- تمّ تصنيف سجلات مجموعة المعطيات KDDCup99 حسب صعوبة تصنيفها، وتمّ بناء مجموعة NSL-KDD بحيث يتناسب عدد السجلات عكسياً مع كل مستوى صعوبة، ممّا يعطي فكرة أفضل عن أداء نموذج التّصنيف.
- مجموعة المعطيات الخاصة بالاختبار تحوي هجمات غير موجودة في مجموعة المعطيات الخاصة بالتّدريب، ممّا يعطي مصداقية أعلى لنتائج نماذج التّصنيف.

وتتوزع السجلات في مجموعة المعطيات NSL-KDD وفق الجدول التّالي:

الجدول **4-2**: توزع السجلات في مجموعة المعطيات **NSL-KDD**

مجموعة المعطيات	السجلات السليمة	السجلات المرتبطة بمحاولة اختراق	المجموع
التّدريب	67343 (53%)	58630 (47%)	125973
الاختبار	9711 (43%)	12833 (57%)	22544

ونلاحظ أن عدد السجلات السليمة قريب من عدد السجلات المرتبطة بمحاولة اختراق في كل من مجموعتي التّدريب والاختبار ممّا يمنع انحياز نموذج التّصنيف إلى أحد الصّفوف.

وبالنسبة للسمات التي سيتمّ استخدامها وتؤمّنها المبدّلات في شبكة SDN فهي مبينة في الجدول التّالي: [12]

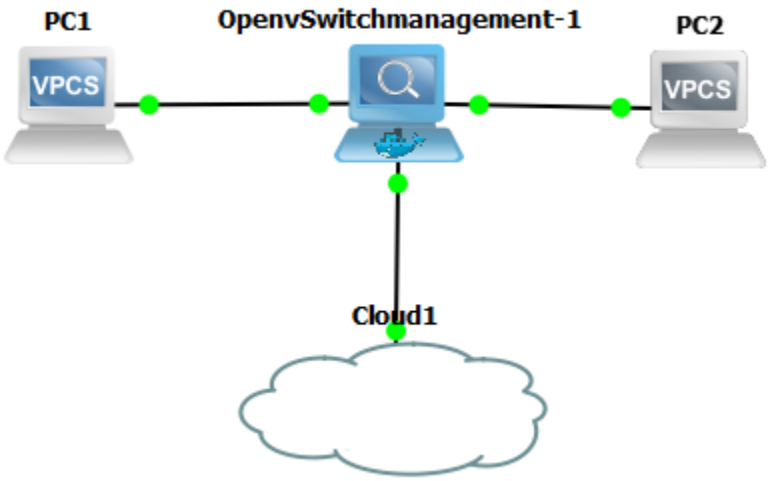
الجدول **4-3**: السّمات التي تؤمّنها المبدّلات في شبكة **SDN**

السمة	الوصف	النوع
duration	المدة التي انقضت منذ بداية الاتصال (بالثواني)	عددية
protocol_type	نوع البروتوكول المستخدم (TCP, UDP, ICMP)	نصيّة
src_bytes	عدد البايتات المرسلة من المصدر إلى الوجهة	عددية
dst_bytes	عدد البايتات المرسلة من المصدر إلى الوجهة	عددية
count	عدد الاتصالات المنشأة مع المضيف الحالي في آخر ثانيتين	عددية
srv_count	عدد الاتصالات التي تتعامل مع نفس الخدمة الحالية في آخر ثانيتين	عددية

وسنحصل على هذه السّمات من مجموعة المعطيات ليتمّ استخدامها في مرحلة تدريب واختبار نموذج تعلّم الآلة، أما بعد بناء النّموذج وتطبيقه ضمن المتحكّم الخاص بشبكة SDN يتمّ الحصول على هذه السّمات من مبدّلات شبكة SDN التي تستخلص هذه السّمات من الطّرود المارّة عبرها، وتكمن الفائدة من وراء استخدام هذه السّمات في القدرة على بناء نظام كشف اختراق كتطبيق ضمن المتحكّم دون إضافة أي تكاليف خاصة بشراء أجهزة مسؤولة عن دراسة الطّرود المارّة عبر الشّبكة، ودون الحاجة لإضافة عبء على الشّبكة من ناحية التأخير الناتج عن دراسة الطّرود المارّة عبر الشّبكة، وسيتمّ في الجزء التّالي عرض مراحل بناء شبكة SDN.

3-4. تجهيز شبكة SDN

الهدف في هذه المرحلة هو إجراء محاكاة (Emulation) لشبكة SDN قادرة على استخلاص إحصائيات عن الدّفق الذي يمر عبرها، ولتحقيق ذلك قمنا ببناء شبكة ضمن برمجية GNS3 تتألف من مبدّل من النوع openvswitch القادر على التعامل بروتوكول OpenFlow والتواصل مع متحكّم مرتبط بالشّبكة، كما تتألف الشّبكة من مضيفين (hosts) الهدف منهما هو تبادل الطّرود فيما بينهما لاختبار صحة عمل المبدّل، ويبيّن الشّكل التّالي الشّبكة المبنية ضمن برمجية GNS3:



الشّكل **2-4**: طبولوجيا الشّبكة المبنية ضمن برمجية **GNS3**

ثمّ قمنا بتنصيب التوزيعة Karaf من المتحكّم OpenDayLight ضمن آلة افتراضية تعمل بنظام تشغيل Ubuntu16.4، وقدّم تمّ اختيار المتحكّم OpenDayLight لكونه أكثر المتحكّمات انتشاراً بالإضافة لوجود واجهات مقدّمة من شركة Cisco لإدارة هذا المتحكّم.

ثمّ تمّ ربط المبدّل بالمتحكّم وتهيئتهما لاستخدام بروتوكول OpenFlow1.3، ويبيّن الشّكل التّالي حالة الاتصال بين المتحكّم والمبدّل:

```
/ # ovs-vsctl show
fec9dc15-1db2-4bee-9204-020da8d3faec
    Bridge "br3"
        Port "br3"
            Interface "br3"
                type: internal
    Bridge "br1"
        Port "br1"
            Interface "br1"
                type: internal
    Bridge "br0"
        Controller "tcp:192.168.116.128:6633"
            is_connected: true
        Port "eth8"
            Interface "eth8"
        Port "eth2"
```

الشّكل **3-4**: حالة الاتصال بين المتحكّم والمبدّل

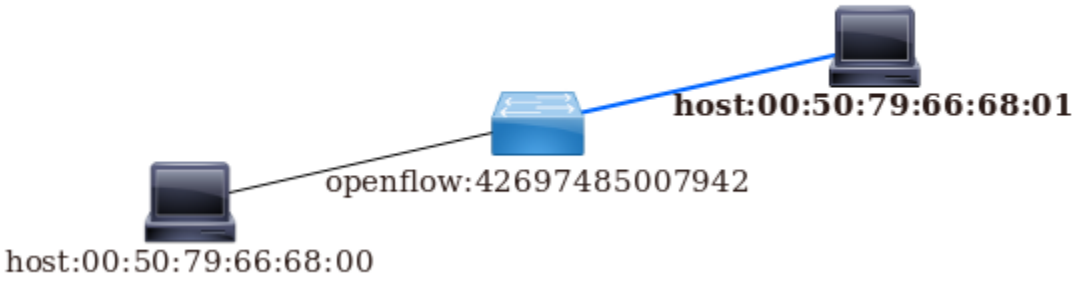
حيث يتمّ تهيئة المبدّل للاتصال مع المتحكّم عبر العنوان المنطقي للآلة التي يعمل عليها نظام ubuntu.

وبعد الاتصال بين المتحكّم والبدل يتمّ تبادل مجموعة من رسائل التعارف من بروتوكول OpenFlow كما يبيّن الشّكل التّالي:

144	50.003282	192.168.192.129	192.168.116.128	TCP	54 47594 → 6633 [ACK] Seq=1 Ack=1 Win=29200 Len=0
145	50.003442	192.168.192.129	192.168.116.128	OpenFlow	62 Type: OFPT_HELLO
146	50.003797	192.168.116.128	192.168.192.129	TCP	60 6633 → 47594 [ACK] Seq=1 Ack=9 Win=64240 Len=0
147	50.008208	192.168.116.128	192.168.192.129	OpenFlow	70 Type: OFPT_HELLO
148	50.009144	192.168.116.128	192.168.192.129	OpenFlow	62 Type: OFPT_FEATURES_REQUEST
149	50.010107	192.168.192.129	192.168.116.128	TCP	54 47594 → 6633 [ACK] Seq=9 Ack=17 Win=29200 Len=0
150	50.010226	192.168.192.129	192.168.116.128	TCP	54 47594 → 6633 [ACK] Seq=9 Ack=25 Win=29200 Len=0
151	50.010313	192.168.192.129	192.168.116.128	OpenFlow	86 Type: OFPT_FEATURES_REPLY
152	50.010690	192.168.116.128	192.168.192.129	TCP	60 6633 → 47594 [ACK] Seq=25 Ack=41 Win=64240 Len=0
153	50.011452	192.168.116.128	192.168.192.129	OpenFlow	62 Type: OFPT_BARRIER_REQUEST
154	50.012039	192.168.192.129	192.168.116.128	OpenFlow	62 Type: OFPT_BARRIER_REPLY
155	50.012362	192.168.116.128	192.168.192.129	TCP	60 6633 → 47594 [ACK] Seq=33 Ack=49 Win=64240 Len=0
156	55.000455	192.168.192.129	192.168.116.128	OpenFlow	62 Type: OFPT_ECHO_REQUEST
157	55.001198	192.168.116.128	192.168.192.129	TCP	60 6633 → 47594 [ACK] Seq=33 Ack=57 Win=64240 Len=0
158	59.058742	192.168.116.128	192.168.192.129	OpenFlow	70 Type: OFPT_MULTIPART_REQUEST, OFPMP_DESC
159	59.058920	192.168.116.128	192.168.192.129	OpenFlow	62 Type: OFPT_ECHO_REPLY

الشّكل **4-4**: رسائل التعارف بين المتحكّم والمبدّل

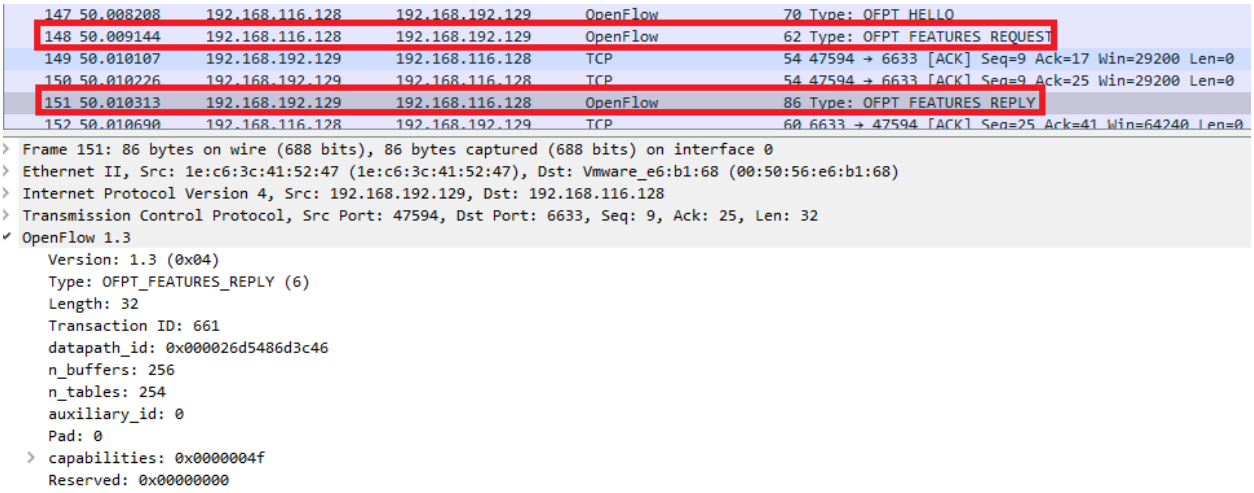
ونتيجة لذلك يصبح لدى المتحكّم نظرة شاملة عن طبولوجيا الشّبكة المبنية ضمن البرمجية GNS3، ويمكن الحصول على بنية هذه الشّبكة من واجهة إدارة المتحكّم المقدّمة من شركة Cisco كما يبيّن الشّكل التّالي:



الشّكل **4-5**: طبولوجيا الشّبكة من واجهة إدارة المتحكّم

وبالمقارنة مع (شكل 4-2) نلاحظ التطابق بين البنيتين.

كما يوجد واجهة مخصصة للحصول على إحصائيات عن الدّفق تمّكننا من معرفة عدد الطّرود وعدد البايتات المرسلة ومدة الدّفق، وواجهة أخرى مخصصة للحصول على معلومات عن البوابات تمّكننا من معرفة البوابات المستخدمة وبالتّالي الخدمات التي يتمّ التعامل معها بالإضافة إلى مدة التعامل مع هذه الخدمات، ويتمّ ذلك عبر استخدام رسائل feature_request و feature_reply من بروتوكول OpenFlow كما يبيّن الشّكل التّالي:



الشّكل **4-6**: رسائل **feature_request** و **feature_reply** من بروتوكول **OpenFlow**

وبعد الانتهاء من مرحلة بناء شبكة SDN وعرض طريقة استخلاص السّمات، أصبح بالإمكان الانتقال إلى مرحلة بناء نظام الانتخاب بالاعتماد على خوارزميّات تعلّم الآلة.

4-4. نظام الانتخاب

تمّ بناء هذا النّظام بالاعتماد على مجموعة من خوارزميّات تعلّم الآلة (SVM و Decision Tree و Random Forest و XGBoost و Deep Feedforward Neural Network)، حيث كنا نهدف إلى الحصول على نتائج من خوارزميّات تعمل بطرق مختلفة، حيث تعتمد خوارزميّة SVM على البعد بين العيّنات في الفضاء وإيجاد الفاصل بين الصّفوف هندسياً، مع إمكانية استخدام توابع النواة لحساب الأبعاد بين العيّنات في فضاء آخر، بينما تمّ اختيار خوارزميّة Decision Tree لأنها تعتمد على التوزع الاحتمالي للسمات ضمن مجموعة المعطيات وكمية المعلومات المتبادلة بينها وبين الصّفوف المتاحة، أما بالنسبة للخوارزميّتين Random Forest و XGBoost فهما تحسين لخوارزميّة Decision Tree حيث تستخدم طريقة Bagging في خوارزميّة Random Forest وطريقة Boosting في خوارزميّة XGBoost، أما بالنسبة للشبكات العصبونيّة العميقة فقد أعطت نتائج واعدّة في مختلف المجالات وكان لا بد من استخدامها.

تمّ بناء عدّة نماذج تصنيف كل منها يعتمد على واحدة من الخوارزميّات المذكورة آنفاً من خلال استخلاص السّمات المذكورة في (جدول 4-3) من مجموعة المعطيات NSL-KDD، لنحصل على مجموعة من العيّنات كل منها شعاع بسطر واحد وستة أعمدة، ثمّ يتمّ استبدال السمة النصيّة (protocol_type) بقيمة عددية، وتمّ اعتماد الصيغة التّالية: [46]

$$protocol_{type} = \begin{cases} 4\ if\ ICMP \\ 10\ if\ TCP \\ 17\ if\ UDP \end{cases} \qquad (5)$$

وتمّ إجارء عمليّة الاستبدال المذكورة في (علاقة (5)) لأن بعض الخوارزميّات مثل SVM و Deep Feedforward Neural Network غير قادرة على التعامل مع معطيات من الفئة النصية.

ثمّ تمّ إجراء عمليّة تقييس لكل عمود من مجموعة المعطيات باستخدام العلاقة:

$$\frac{x - x_{min}}{x_{max} - x_{min}} \qquad (6)$$

حيث:

- x: قيمة السمة.
- x_{min}: أصغر قيمة في العمود الموافق للسمة x.
- x_{max}: أكبر قيمة في العمود الموافق للسمة x.

وتمّ إجراء عمليّة التقييس بهدف جعل كافة السّمات ضمن نفس المجال بحيث لا ينحاز نموذج التّصنيف إلى إحدى السّمات، ثمّ تمّ تدريب واختبار كل من نماذج التّصنيف باستخدام العيّنات الناتجة، وفي النهاية تمّ إجراء انتخاب يعتمد على نتائج كل من الخوارزميّات مع إعطاء أولوية أعلى للخوارزميّة ذات النّتائج الأفضل من خلال تطبيق العلاقة:

$$Final\ prediction(i) = \frac{\sum_k (p_k * acc_k)}{\sum_k acc_k} \qquad (7)$$

حيث:

- p_k: احتمال أن تكون العيّنة رقم i غير مرتبطة بمحاولة اختراق بالاعتماد على الخوارزميّة k
- acc_k: دقّة الكشف الموافقة للخوارزميّة k

ثمّ تمّ تكرار المراحل السّابقة باستخدام مجموعة المعطيات KDDCup99 لإجراء مقارنة بين هذه المجموعة ومجموعة المعطيات NSL-KDD التي تعتبر تحسيناً لها.

وسنقوم في الجزء التّالي بعرض نتائج كل من خوارزميّات تعلّم الآلة المذكورة سابقاً.

Decision Tree .1-4-4

بالاستفادة من مكتبة sklearn التي تقدّم مجموعة من التوابع المستخدمة في مجال تعلّم الآلة قمنا ببناء نموذج تصنيف يعتمد على أشجار القرار بالمعاملات المبيّنة في الجدول التّالي:

الجدول **4-4**: المعاملات المستخدمة لبناء نموذج تصنيف بخوارزميّة **Decision Tree**

المعامل	القيمة	الوصف	سبب الاختيار
Criterion	gini	التّابع المستخدم لتحديد جودة أي تقسيم يطرأ على الشّجرة [39]	أفضل معيار لحساب جودة التّقسيم بحيث نحصل على أفضل شجرة فرعيّة ثنائيّة عند كل تقسيم [38]
Splitter	best	المعامل المستخدم لاختيار أفضل عقدة لتكون جذر الشّجرة الفرعيّة [39]	أفضل معيار عند اختيار عقدة جديدة بحيث نحصل على أصغر شجرة ممكنة ممّا يقلل التعقيد [39]
Max_Depth	2	أكبر عمق يُسمح أن تصل إليه الشّجرة (زيادة العمق تؤدّي إلى over-fittign) [39]	الهدف هو الحصول على شجرة أقل عمقاً بحيث نتجنّب التّعقيد ونتجنّب مشكلة over-fitting
Min_Samples_Leaf	50	الحد أدنى لعدد عناصر مجموعة المعطيات التي يُسمَح أن تنتمي لصف معيّن ضمن فرع من فروع الشّجرة [39]	منع احتواء الشّجرة على أوراق ترتبط بعدد قليل من عيّنات مجموعة المعطيات ممّا يقلل التّعقيد ونتجنّب مشكلة over-fitting
Min_Samples_Split	200	الحد الأدنى لعدد عناصر مجموعة المعطيات التي يجب أن تنتمي لعقدة ما لكي يُسمح ببناء شجرة فرعيّة تكون هذه العقدة جذرها. [39]	عدم الحصول على أشجار فرعيّة غير ضرورية مرتبطة بعدد قليل من عيّنات مجموعة المعطيات ممّا يقلل التّعقيد ونتجنّب مشكلة over-fitting

وعند تطبيق هذه الخوارزميّة بالاعتماد على السّمات التي تؤمّنها شبكة SDN مع مجموعة المعطيات NSL-KDD حصلنا على النّتائج التّالية:

الجدول **4-5**: مصفوفة الالتباس لخوارزميّة **Decision Tree** مع مجموعة المعطيات **NSL-KDD** وسمات شبكة **SDN**

True Negative	**9472**	**3361**	False Negative
False Positive	**535**	**9176**	True Positive

ومن العلاقتين (1) و (3) نحصل على النّتائج التّالية:

الجدول **4-6**: نتائج خوارزميّة **Decision Tree** مع مجموعة المعطيات **NSL-KDD** وسمات شبكة **SDN**

المعامل	النتيجة
Accuracy	82.72%
FPR	0.05

ثمّ قمنا بإعادة التّجربة بالاعتماد على السّمات التي تؤمّنها شبكة SDN مع مجموعة المعطيات KDDCup99 وحصلنا على النّتائج التّالية:

الجدول **4-7**: مصفوفة الالتباس لخوارزميّة **Decision Tree** مع مجموعة المعطيات **KDDCup99** وسمات شبكة **SDN**

True Negative	**74668**	**492**	False Negative
False Positive	**68**	**18793**	True Positive

ومن العلاقتين (1) و (3) نحصل على النّتائج التّالية:

الجدول **4-8**: نتائج خوارزميّة **Decision Tree** مع مجموعة المعطيات **KDDCup99** وسمات شبكة **SDN**

المعامل	النتيجة
Accuracy	99.4%
FPR	0.0009

Random Forest .2-4-4

بالاستفادة من مكتبة sklearn قمنا ببناء نموذج تصنيف يعتمد على خوارزميّة Random Forest، ويتألّف هذا النّموذج من مجموعة من الأشجار المشابهة للشجرة المبنيّة في الجزء السابق، وتمّ استخدام المعاملات التّالية:

الجدول **4-9**: المعاملات المستخدمة لبناء نموذج تصنيف بخوارزميّة **Random Forest**

المعامل	القيمة	الوصف	سبب الاختيار
Bootstrap	True	استخدام سمات مختلفة وجزء مختلف من مجموعة المعطيات لتدريب كل شجرة [39]	اختيار القيمة False يكافئ حالة أشجار القرار
N_estimators	100	عدد الأشجار التي سيتمّ بناؤها وتوسيط نتائجها [39]	الحصول على عدد مناسب من الأشجار دون زيادة كبيرة في التعقيد
Max_features	sqrt	عدد السّمات التي سيتمّ استخدامها لتدريب كل شجرة (جذر العدد الكلي للسّمات) [39]	القيمة الأكثر شيوعاً [29]

وعند تطبيق هذه الخوارزميّة بالاعتماد على السّمات التي تؤمّنها شبكة SDN مع مجموعة المعطيات NSL-KDD حصلنا على النّتائج التّالية:

الجدول **4-10**: مصفوفة الالتباس لخوارزميّة **Random Forest** مع المجموعة **NSL-KDD** وسمات شبكة **SDN**

True Negative	**8001**	**4832**	False Negative
False Positive	**262**	**9449**	True Positive

ومن العلاقتين (1) و (3) نحصل على النّتائج التّالية:

الجدول **4-11**: نتائج خوارزميّة **Random Forest** مع مجموعة المعطيات **NSL-KDD** وسمات شبكة **SDN**

المعامل	النتيجة
Accuracy	77.40%
FPR	0.03

ثمّ قمنا بإعادة التّجربة بالاعتماد على مجموعة المعطيات KDDCup99 وسمات شبكة SDN وحصلنا على النّتائج التّالية:

الجدول **4-12**: مصفوفة الالتباس لخوارزميّة **Random Forest** مع المجموعة **KDDCup99** وسمات شبكة **SDN**

True Negative	**75047**	**113**	False Negative
False Positive	**4039**	**14822**	True Positive

ومن العلاقتين (1) و (3) نحصل على النّتائج التّالية:

الجدول **4-13**: نتائج خوارزميّة **Random Forest** مع مجموعة المعطيات **KDDCup99** وسمات شبكة **SDN**

المعامل	النتيجة
Accuracy	98.54%
FPR	0.05

XGBoost .3-4-4

بالاستفادة من المكتبة xgboost المقدّمة من Google، قمنا ببناء نموذج تصنيف يعتمد على خوارزميّة XGBoost، ويتمّ بناء هذا النّموذج بالاعتماد على مجموعة من أشجار القرار التي يتمّ تدريبها بشكل تسلسلي على أجزاء مختلفة من مجموعة المعطيات، وتمّ استخدام المعاملات التّالية:

الجدول **4-14**: المعاملات المستخدمة لبناء نموذج تصنيف بخوارزميّة **XGBoost**

المعامل	القيمة	الوصف	سبب الاختيار
Objective	Binary: hinge	تابع الخطأ الذي يتمّ من خلاله حساب أداء الخوارزميّة [40]	الأكثر استخداماً في مسائل التّصنيف الثنائيّة [40]
Colsample_bytree	0.3	نسبة السّمات التي سيتمّ استخدامها لتدريب كل شجرة (عدد السّات المستخدمة إلى العدد الكلّي للسّمات) [40]	استخدام عدد مناسب من السّمات دون الاقتراب من القيمة 1 التي تكافئ تدريب كل شجرة بكامل السّمات (تكافئ أشجار القرار)
Learning_rate	0.06	تؤثّر على معدل الاقتراب من القيمة الصّغرى لتابع الخطأ [40]	كلما صَغُرت القيمة سنحتاج لوقت أطول للوصول إلى القيمة الصّغرى لتابع الخطأ (تقارب بطيء)، وزيادة القيمة تؤدّي إلى القفز حول القيمة الصّغرى عند الاقتراب منها دون الاستقرار عليها.
Max_depth	3	أكبر عمق يُسمح أن تصل إليه الشّجرة (زيادة العمق تؤدّي إلى over-fitting) [40]	الهدف هو الحصول على شجرة أقل عمقاً بحيث نتجنّب التعقيد ونتجنّب مشكلة over-fitting
Gamma	1.0	مقدار النّقص الذي يجب أن يطرأ على تابع الخطأ ليتمّ إجراء تقسيم جديد على إحدى عقد الشّجرة [40]	لا يوجد طريقة مباشرة لاختيار قيمة هذا المعامل، تمّ اختيار القيمة بعد عدّة تجارب لإيجاد أفضل قيمة
N_estimators	300	عدد النّماذج التي سيتمّ بناؤها وتدريب كل منها ثمّ اختيار	اختيار عدد كافٍ من النماذج الأوّليّة للحصول على أفضل نموذج وإهمال البقيّة

		النّموذج الأفضل [40]	
Num_round	500	عدد المراحل التي سيتمّ فيها تدريب كل نموذج [40]	لا يوجد طريقة مباشرة لاختيار قيمة هذا المعامل، تمّ اختيار أصغر عدد مراحل تدريب يعطي أفضل نموذج

وعند تطبيق هذه الخوارزميّة بالاعتماد على السّمات التي تؤمّنها شبكة SDN مع مجموعة المعطيات NSL-KDD حصلنا على النّتائج التّالية:

الجدول **4-15**: مصفوفة الالتباس لخوارزميّة **XGBoost** مع مجموعة المعطيات **NSL-KDD** وسمات شبكة **SDN**

True Negative	**8522**	**4311**	False Negative
False Positive	**284**	**9427**	True Positive

ومن العلاقتين (1) و (3) نحصل على النّتائج التّالية:

الجدول **4-16**: نتائج خوارزميّة **XGBoost** مع مجموعة المعطيات **NSL-KDD** وسمات شبكة **SDN**

المعامل	النتيجة
Accuracy	79.61%
FPR	0.03

ثمّ قمنا بإعادة التّجربة بالاعتماد على مجموعة المعطيات KDDCup99 وسمات شبكة SDN وحصلنا على النّتائج التّالية:

الجدول **4-17**: مصفوفة الالتباس لخوارزميّة **XGBoost** مع مجموعة المعطيات **KDDCup99** وسمات شبكة **SDN**

True Negative	**75143**	**17**	False Negative
False Positive	**878**	**17893**	True Positive

ومن العلاقتين (1) و (3) نحصل على النّتائج التّالية:

الجدول **4-18**: نتائج خوارزميّة **XGBoost** مع مجموعة المعطيات **KDDCup99** وسمات شبكة **SDN**

المعامل	النتيجة
Accuracy	99.05%
FPR	0.01

Support Vector Machine SVM .4-4-4

بالاستفادة من مكتبة sklearn قمنا ببناء نموذج تصنيف يعتمد على خوارزميّة SVM، وتمّ استخدام المعاملات التّالية:

الجدول **4-19**: المعاملات المستخدمة لبناء نموذج تصنيف بخوارزميّة **SVM**

المعامل	القيمة	الوصف	سبب الاختيار
Kernel	rbf	التّابع الذي يتمّ الاعتماد عليه لإيجاد الفاصل بين الصّفوف [39]	لا يوجد طريقة مباشرة لاختيار أفضل تابع، حصلنا عليه تجريبيّاً
C	100	تساهم في تحديد البعد بين الصّفوف والفاصل، بزيادة القيمة يزداد هذا البعد ممّا يؤدّي إلى تداخل كبير بين الهامش والصّفوف ممّا يؤدّي إلى فشل نموذج التّصنيف [39]	تمّ اختيار القيمة تجريبيّاً، مع أخذ الزمن اللازم للتدريب بعين الاعتبار
Probability	true	تفعيل خيار إعطاء احتمال الانتمّاء إلى صف معيّن عند إجراء توقّع ما. [39]	للاستفادة من احتمال التوقّع في تطبيق (علاقة (7))
Gamma	1	تحدّد أهمية العيّنات في اختيار الفاصل بين الصّفوف، من أجل قيم منخفضة تكون العيّنات البعيدة أكثر تأثيراً ممّا يسمح بتداخل الصّفوف، وبالحالة المعاكسة نحصل على فاصل أكثر مطابقة لمجموعة التّدريب [39]	تمّ اختيار القيمة تجريبيّاً، مع أخذ الزمن اللازم للتدريب بعين الاعتبار

وعند تطبيق هذه الخوارزميّة بالاعتماد على السّمات التي تؤمّنها شبكة SDN مع مجموعة المعطيات NSL-KDD حصلنا على النّتائج التّالية:

الجدول **4-20**: مصفوفة الالتباس لخوارزميّة **SVM** مع مجموعة المعطيات **NSL-KDD** وسمات شبكة **SDN**

True Negative	**7078**	**5755**	False Negative
False Positive	**283**	**9428**	True Positive

ومن العلاقتين (1) و (3) نحصل على النّتائج التّالية:

الجدول **4-21**: نتائج خوارزميّة **SVM** مع مجموعة المعطيات **NSL-KDD** وسمات شبكة **SDN**

المعامل	النتيجة
Accuracy	73.22%
FPR	0.038

ثمّ قمنا بإعادة التّجربة بالاعتماد على مجموعة المعطيات KDDCup99 وسمات شبكة SDN وحصلنا على النّتائج التّالية:

الجدول **4-22**: مصفوفة الالتباس لخوارزميّة **SVM** مع مجموعة المعطيات **KDDCup99** وسمات شبكة **SDN**

True Negative	**66255**	**8905**	False Negative
False Positive	**795**	**18066**	True Positive

ومن العلاقتين (1) و (3) نحصل على النّتائج التّالية:

الجدول **4-23**: نتائج خوارزميّة **SVM** مع مجموعة المعطيات **KDDCup99** وسمات شبكة **SDN**

المعامل	النتيجة
Accuracy	89.68%
FPR	0.012

Deep Feedforward Neural Network.5-4-4

بالاستفادة من مكتبة keras المبنية على مكتبة TensorFlow قمنا ببناء نموذج تصنيف يعتمد على شبكة عصبونية عميقة ذات تغذية مباشرة، وتمّ استخدام المعاملات التّالية

الجدول **4-24**: المعاملات المستخدمة لبناء نموذج تصنيف باستخدام شبكة عصبونية عميقة

المعامل	القيمة	الوصف	سبب الاختيار
عدد عصبونات طبقة الدّخل	6	عدد عناصر دخل الخوارزميّة [41]	عدد عصبونات طبقة الدّخل يجب أن يساوي عدد السّمات المتوفّرة.
عدد عصبونات طبقة الخرج	2	عدد الصّفوف المتوفّرة في مجموعة المعطيات	عدد عصبونات طبقة الخرج يجب أن يساوي عدد الصّفوف المتوفّرة في مجموعة المعطيات
عدد الطبقات المخفية	1, 2, 3, 4	عدد الطبقات بين طبقتي الدّخل والخرج	لا يوجد طريقة مباشرة لحساب العدد الأفضل للطبقات المخفية، تمّ تغيير عدد الطبقات المخفية وحصلنا على أفضل نتائج عند اختيار 3 طبقات مخفية
عدد العصبونات في كل طبقة مخفية	متغير	-	لاحظنا عدم تأثير هذا المعامل بشكل ملحوظ على أداء الشّبكة
تابع التفعيل	ReLU في الطبقات المخفية Sigmoid في طبقة الخرج	التّابع الذي يتمّ تطبيقه على كل عصبون بعد تجميع قيم الوصلات والأوزان على دخل هذا العصبون [41]	تمّ اختيار تابع sigmoid في طبقة الخرج لأنه أكثر ملاءمة لطبيعة الخرج الثنائية في هذه الطبقة، وتمّ اختيار تابع ReLU في الطبقات المخفية لأنه أثبت فعاليته في مسائل التّصنيف بالإضافة لقدرته على تجنّب مشكلة vanishing-gradient [7]
Loss	Binary-	تابع الخطأ الذي يهدف تدريب الخوارزميّة	أكثر التوابع المستخدمة في مسائل

	crossentropy	إلى ايصاله لقيمته الصّغرى [41]	التّصنيف الثنائي [42]
Optimizer	Adam	التّابع الذي يعدل قيم الأوزان في الشّبكة العصبونيّة بهدف تقليل قيمة تابع الخطأ، ويتمّ تطبيق هذا التّابع بعد أن يتمّ حساب قيمة تابع الخطأ في نهاية كل تكرار من عمليّة التّدريب [41]	لا يوجد طريقة مباشرة لاختيار التّابع، وإنما يتعلق بتابع الخطأ وبمجموعة المعطيات، تمّ اختيار تابع Adam لأنه أعطى أفضل النّتائج تجريبيّاً.
Learning rate	1e-7	يتمّ ضرب هذه القيمة بقيمة انحدار (gradient) تابع الخطأ، ثمّ يتمّ تعديل قيم الأوزان في الشّبكة بناءً على نتيجة الضرب، بهدف تقليل قيمة تابع الخطأ [43]	عند اختيار قيم صغيرة تتأرجح قيمة تابع الخطأ حول قيمته الصّغرى، وعند اختيار قيمة صغيرة تستهلك الخوارزميّة وقت كبير للوصول للقيمة الصّغرى لتابع الخطأ، تمّ اختيار القيمة تجريبيّاً.

وعند تطبيق هذه الخوارزميّة بالاعتماد على السّمات التي تؤمّنها شبكة SDN مع مجموعة المعطيات NSL-KDD حصلنا على النّتائج التّالية:

الجدول **4-25**: مصفوفة الالتباس لخوارزميّة **DNN** مع مجموعة المعطيات **NSL-KDD** وسمات شبكة **SDN**

True Negative	**10021**	**2812**	False Negative
False Positive	**847**	**8864**	True Positive

ومن العلاقتين (1) و (3) نحصل على النّتائج التّالية:

الجدول **4-26**: نتائج خوارزميّة **SVM** مع مجموعة المعطيات **NSL-KDD** وسمات شبكة **SDN**

المعامل	النتيجة
Accuracy	83.8%
FPR	0.07

ثمّ قمنا بإعادة التّجربة بالاعتماد على مجموعة المعطيات KDDCup99 وسمات شبكة SDN وحصلنا على النّتائج التّالية:

الجدول **4-27**: مصفوفة الالتباس لخوارزميّة **DNN** مع مجموعة المعطيات **KDDCup99** وسمات شبكة **SDN**

True Negative	**66255**	**8905**	False Negative
False Positive	**795**	**18066**	True Positive

ومن العلاقتين (1) و (3) نحصل على النّتائج التّالية:

الجدول **4-28**: نتائج خوارزميّة **DNN** مع مجموعة المعطيات **KDDCup99** وسمات شبكة **SDN**

المعامل	النتيجة
Accuracy	99.5%
FPR	0.0009

وبهذا نكون قد حصلنا على نتائج الخوارزميّات المختارة في حالة استخدام كل من مجموعة المعطيات NSL-KDD ومجموعة المعطيات KDDCup99 ونلخّص هذه النّتائج في الجدولين التّاليين:

الجدول **4-29**: ملخص نتائج الخوارزميّات عند استخدام مجموعة المعطيات **NSL-KDD** وسمات شبكة **SDN**

الخوارزميّة	Accuracy	FPR
Decision Tree	82.72%	0.05
Random Forest	77.40%	0.03
XGBoost	79.61%	0.03
SVM	73.22%	0.038
Deep Feedforward NN	83.8%	0.07

وبمقارنة النّتائج المبيّنة في الجدول السّابق نجد ما يلي:

- خوارزميّة Decision Tree أعطت دقّة أعلى من خوارزميّة Random Forest وخوارزميّة XGBoost، وهو أمر غير متوقّع كون هاتين الخوارزميّتين تعتبران تحسيناً لخوارزميّة Decision Tree، ونفسّر ذلك بطبيعة عمل هاتين الخوارزميّتين، حيث تعتمد كل منهما على بناء مجموعة ابتدائيّة من أشجار القرار وتدريب كل منها على عدّة مراحل باستخدام مجموعات جزئيّة من مجموعة المعطيات، وكون عدد السّمات الكليّة منخفض نسبياً، أصبح استخدام مجموعة جزئيّة من هذه السّمات غير فعّال ولا يساهم بتدريب الأشجار الناتجة بشكلٍ كافٍ، ممّا أدّى إلى الحصول على دقّة أعلى في حالة خوارزميّة Decision Tree حيث يتمّ بناء شجرة واحدة من مرحلة واحدة وتدريبها بكافة السّمات المتوفّرة، مع الانتباه إلى أنّ هذا التحسن بالدقّة جاء على حساب زيادة بسيطة في معدل الخطأ الموجب (False Positive Rate).

- خوارزميّة XGBoost أعطت نتائج أفضل من خوارزميّة Random Forest، وهو أمر متوقّع كون خوارزميّة XGBoost تعتمد على تدريب كل شجرة عدّة مرات بمجموعات جزئيّة مختلفة، بينما تعتمد خوارزميّة Random Forest على تدريب مجموعة من الأشجار لمرة واحدة بمجموعات جزئيّة مختلفة، وتوسيط النّتائج ممّا يجعل أشجار خوارزميّة XGBoost أكثر قدرة على التّعلم وبالتالي تعطي نتائج أفضل.

- فشلت خوارزميّة SVM في إعطاء نتائج جيّدة بالمقارنة مع بقيّة الخوارزميّات، وهذا يعطي فكرة عن أن مجموعة المعطيات المستخدمة أكثر قابليّة للفصل بالاعتماد على التوزع الاحتمالي للسّمات بالمقارنة مع قابليّة الفصل بالاعتماد على موقع عيّنات مجموعة المعطيات في الفضاء، لكن على الرغم من ذلك سيتمّ الاعتماد عليها في نظام الانتخاب المقترح نظراً لقدرتها على كشف هجمات حجب الخدمة DOS. [8] [45]

- أعطت الشّبكات العصبونيّة العميقة دقّة أعلى من بقيّة الخوارزميّات المستخدمة (كما هو متوقّع)، على حساب ارتفاع بسيط في معدل الخطأ الموجب، فقد أثبتت هذه الشّبكات جودتها في مختلف المجالات [44]، بسبب قدرتها على ربط المعطيات وإيجاد أنماط في هذه المعطيات تعجز بقيّة الخوارزميّات عن إيجادها.

ويبيّن الجدول التّالي ملخّص النّتائج عند استخدام مجموعة المعطيات KDDCUP99

الجدول **4-30**: ملخّص نتائج الخوارزميّات عند استخدام مجموعة المعطيات **KDDCUP99** وسمات شبكة **SDN**

الخوارزميّة	Accuracy	FPR
Decision Tree	99.4%	0.0009
Random Forest	98.54%	0.05
XGBoost	99.05%	0.01
SVM	89.68%	0.012
Deep Feedforward NN	99.5%	0.0009

وبمقارنة هذه النّتائج فيما بينها وبمقارنتها مع النّتائج المبينة في (جدول 4-29) نجد ما يلي:

- تكرار نفس النّمط الظّاهر في (جدول 4-29) حيث أعطت الشّبكات العصبونيّة العميقة أفضل نتيجة ويليها خوارزميّة Decision Tree ثمّ خوارزميّة XGBoost وخوارزميّة Random Forest وفي النّهاية خوارزميّة SVM ممّا يؤكّد على صحّة النّمط الظّاهر سابقاً.

- تحسّن كبير في الأداء من ناحية الدّقّة ومعدّل الخطأ الموجب بالمقارنة مع النّتائج في (جدول 4-29)، ولكن نبرّر هذا التحسن بكون توزّع المعطيات في المجموعة KDDCup99 غير متوازن، حيث يوجد عدد كبير من العيّنات المرتبطة بمحاولة اختراق بالمقارنة مع العيّنات السليمة كما يبيّن (جدول 4-1) ممّا يجعل نماذج التّصنيف تنحاز نحو الصّف ذو العيّنات الأكثر، بالإضافة إلى وجود تكرارات في عيّنات مجموعة المعطيات KDDCup99 ممّا يؤدّي إلى انحياز نماذج التّصنيف أيضاً، وبالتّالي سيتمّ اعتبار نتائج مجموعة المعطيات NSL-KDD أكثر واقعيّة وصحّة وسيتمّ اعتمادها في بناء نموذج الانتخاب.

في النّهاية قمنا ببناء نظام الانتخاب من خلال تطبيق (علاقة (7)): $Final\ prediction(i) = \frac{\sum_k (p_k * acc_k)}{\sum_k acc_k}$ واستخدام مجموعة المعطيات NSL-KDD وحصلنا على النّتائج التّالية:

الجدول **4-31**: مصفوفة الالتباس لنظام الانتخاب مع مجموعة المعطيات **NSL-KDD** وسمات شبكة **SDN**

True Negative	**8510**	**4323**	False Negative
False Positive	**283**	**9428**	True Positive

ومن العلاقتين (1) و (3) نحصل على النّتائج التّالية:

الجدول **4-32**: نتائج نظام الانتخاب مع مجموعة المعطيات **NSL-KDD** وسمات شبكة **SDN**

المعامل	النتيجة
Accuracy	79.6%
FPR	0.03

ونلاحظ من النّتائج السّابقة أنّ نظام الانتخاب أعطى دقّة متوسّطة بين أفضل وأسوأ دقّة للخوارزميّات المستخدمة، ولكننا حصلنا على معدل خطأ موجب أفضل من كافة الخوارزميّات المستخدمة، ونفسّر هذه النّتائج بأنّ الخوارزميّات ذات الخطأ الموجب الأقل ساهمت في تغيير قرار الخوارزميّات ذات الخطأ الموجب الأعلى للحصول على خطأ موجب أقل بالمحصّلة، لكن بنفس الوقت أدّت إلى تغيير بعض القرارات الصّحيحة للخوارزميّات ذات الدقّة الأعلى ممّا أدّى إلى انخفاض الدقّة الكليّة.

وبالإمكان عزل أو إضافة إحدى الخوارزميّات من نظام الانتخاب بما يتناسب مع الشّبكة المستخدمة بهدف زيادة الدقّة أو إنقاص معدّل الخطأ الموجب، وفي حال مثلاً كان الهدف هو الحصول على أفضل دقّة يمكن استخدام الشّبكة العصبونيّة العميقة وإهمال بقيّة الخوارزميّات.

في حال توقّع نظام الانتخاب أن دفق ما مرتبط بمحاولة اختراق، يتمّ إضافة قاعدّة جديدة إلى جداول الدّفق تمّنع مرور هذا الدّفق عبر المبدّل كما يبيّن الشّكل التّالي:

```
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=528.884s, table=0, n_packets=0, n_bytes=0, in_port=1 actions=output:2
 cookie=0x0, duration=496.278s, table=0, n_packets=0, n_bytes=0, in_port=2 actions=drop
 cookie=0x0, duration=256.968s, table=0, n_packets=0, n_bytes=0, actions=drop
 cookie=0x0, duration=2.191s, table=0, n_packets=0, n_bytes=0, ip,nw_src=10.0.0.0/24 actions=drop
```

الشّكل **4-7**: القواعد التي تمّنع مرور الدّفق المتوقّع ارتباطه بهجوم عبر المبدّل

حيث نلاحظ أنّ هذه الشّبكة تمّنع مرور أي طرد مصدره الشّبكة 10.0.0.0/24.

ومن خلال منح نظام كشف الاختراق إمكانيّة إضافة مثل هذه القواعد يتحوّل إلى نظام منع اختراق، إذ يصبح قادراً على منع مرور الطّرود –المرتبطة بمحاولة اختراق– عبر الشبكة، بدلاً من الاكتفاء بإرسال التنبيهات.

4-5. البيئات البرمجيّة المستخدمة

نقدّم في هذه الفقرة لمحة عن البرمجيّات المستخدمة في إنجاز النّظام المقترح:

- TensorFlow: مكتبة مقدّمة من Google للعمل على تطبيقات تعلّم الآلة وبشكل رئيسي الشّبكات العصبونيّة.
- SKLearn: مكتبة مستخدمة في عمليّات التّنقيب عن المعطيات، ومسائل التّصنيف والتّراجع (regression)، يمكن مكاملتها مع مكتبات أخرى.
- Keras: هي واجهة برمجة تطبيقات (API) مبنيّة باستخدام مكتبة TensorFlow لتسهيل التّعامل مع الشّبكات العصبونيّة بمختلف أنواعها.
- منصّة PyCharm IDE: منصّة تعمل على أنظمة نشغيل مختلفة، خاصّة بلغة بايثون، تؤمّن الكثير من التّسهيلات لدعم هذه اللغة وتسهيل التّعامل معها.
- GNS3: منصّة لمحاكاة عمل الشّبكات بمختلف أنواعها، تسهّل عمليّة المحاكاة من خلال تأمين واجهات للتّعامل معها.

الخاتمة والآفاق المستقبليّة

تعتبر الشّبكات المعرّفة برمجيّاً مفهوماً جديداً في مجال الشّبكات، وقد أصبحت محطّ الأنظار سواءً للمهتمّين بدراستها والاستفادة من وظائفها، أو للرّاغبين باسكتشاف نقاط ضعفها ومهاجمتها، وقد حاولنا في هذا البحث الخوض في مجال أمن هذا النّوع من الشّبكات، فقمنا بتصميم وتنفيذ نظام كشف اختراق قارد على التّعامل مع مختلف الهجمات، مع الاستفادة من المزايا التي يتمتّع بها هذا النوع من الشّبكات دون نقل العبء الموروث من الشّبكات التقليديّة، فلا أجهزة إضافيّة ولا حمل زائد على الشّبكة.

ولبناء نظام كشف اختراق فعّال كان لا بدّ من الخوض في مجال الذكاء الاصطناعي وتعلّم الآلة، فقمنا بدراسة مجموعة من الخوارزميّات التي أثبتت جودتها في مسائل التّصنيف.

ويعتمد نظام كشف الاختراق المقترح على بناء نظام انتخاب مؤلّف من عدّة خوارزميّات تعلّم الآلة (الشّبكات العصبونيّة العميقة وأشجار القرار والغابات العشوائيّة وXGBoost وشعاع دعم الآلة SVM)، حيث تقوم شبكة SDN باستخلاص مجموعة من الإحصائيّات عن كل دفق يمر عبرها، وتمّريرها إلى الخوارزميّات المذكورة سابقاً لتقوم كل منها بتصنيف الدّفق كدفق سليم أو دفق مرتبط بمحاولة اختراق، ثمّ يتمّ إجراء انتخاب بين هذه الخوارزميّات لاتّخاذ قرار نهائي مع إعطاء وزن أعلى للخوارزميّات التي أعطت جودة أفضل أثناء مرحلة الاختبار، وفي النّهاية يتمّ إضافة قواعد إلى جداول الدّفق تمّنع مرور المعطيات التي تمّ تصنيفها كمعطيات مرتبطة بهجوم ممّا يزيد من الصلاحيّات المعطاة للنّظام المقترح ويحوّله إلى نظام منع اختراق.

كما قمنا بإجراء مقارنة بين مجموعتي المعطيات NSL-KDD وKDDCup99 لاستخدام الأفضل منهما في عمليّة تدريب خوارزميّات تعلّم الآلة ووجدنا أنّ مجموعة المعطيات NSL-KDD تعطي نتائج أكثر واقعية.

بالنّسبة للمرحلة القادمة:

- إنّ بداية عمل الشّبكات المعرّفة برمجيّاً بشكلها الحالي كان في عام 2008، لذلك لا بدّ أن تخضع في الفترة القادمة لكثير من التّعديلات، فيجب متابعة هذه التّعديلات والاستفادة منها في تحسين أداء النّظام.
- يجب مراقبة الشّبكة التي يعمل ضمنها نظام كشف الاختراق، وإضافة معطيات من الدّفق المار عبر هذه الشّبكة إلى مجموعة معطيات التّدريب وإعادة تدريب الخوارزميّات بشكل دوري بهدف مواكبة محاولات الاختراق الجديدة التي قد تقع الشّبكة ضحيّة لها، واكتشافها مبكّراً في المحاولات التّالية.

- اختيار خوارزميّات جديدة وإضافتها إلى نظام كشف الاختراق لإغناء عمليّة الانتخاب وزيادة القدرة على المقايضة بين أفضل دقّة وأفضل معدل خطأ موجب.

المراجع

[1] N. Feamster, J. Rexford and E. Zegura, "The Road to SDN: An Intellectual History of Programmable Networks," in *Newsletter ACM SIGCOMM Computer Communication Review*, New York, 2014.

[2] A. L. Stancu, G. Suciu, S. Halunga and A. Vulpe, "An Overview Study of Software Defined Networking," in *IE 2015 International Conference*, Rome, 2015.

[3] H. E, P. K and D. S, "RFC 7426-SDN: Layers and Architecture Terminology," 2015.

[4] B. Kleyman, "Top Five Apps and Services That Can Benefit from SDN," 31 March 2016. [Online]. Available: https://www.datacenterknowledge.com/archives/2016/03/31/top-five-apps-and-services-that-can-benefit-from-sdn.

[5] O. Akpovi, A. A and O. F, "Introduction to Software Defined Networks," *International Journal of Applied Information Systems,* vol. 11, no. 7, 2016.

[6] "7 Advantages of Software Defined Networking," 8 August 2017. [Online]. Available: https://imaginenext.ingrammicro.com/data-center/7-advantages-of-software-defined-networking.

[7] R. Vigneswaran, "Evaluating Shallow and Deep Neural Networks for Network Intrusion Detection Systems in Cyber Security," in *2018 9th International Conference on Computing, Communication and Networking Technologies (ICCCNT)*, Bengaluru, 2018.

[8] G. Ajaeiya, N. Adalian, I. Elhajj and A. Chehab, "Flow-Based Intrusion Detection System for SDN," in *2017 IEEE Symposium on Computers and Communications*, 2017.

[9] D. Jankowski and M. Amanowicz, "On Efficiency of Selected Machine Learning Algorithms for Intrusion Detection in Software Defined Networks," *INTL JOURNAL OF ELECTRONICS AND TELECOMMUNICATIONS,* vol. 62, no. 3, pp. 47-252, 2016.

[10] L. Boero, M. Marchese and S. Zappatore, "Support Vector Machine meets Software Defined Networking in IDS domain," in *29th International Teletraffic Congress*, Genoa, 2017.

[11] A. Abubakar and B. Pranggono, "Machine Learning Based Intrusion Detection System for Software Defined Networks," in *Seventh International Conference on Emerging Security Technologies,* Canterbury, 2017.

[12] T. A. Tang, L. Mhamdi and D. e. a. , "Deep Learning Approach for Network Intrusion Detection in Software Defined Networking," in *2016 International Conference on Wireless Networks and Mobile Communications*, Fez , 2016.

[13] S. Anwar, J. Zain and M. Zolkipli, "From Intrusion Detection to an Intrusion Response System: Fundamentals, Requirements, and Future Directions," *algorithms,* vol. 10, no. 39, 2017.

[14] A. Kumar and V. S, "INTRUSION DETECTION SYSTEMS: A REVIEW," *International Journal of Advanced Research in Computer Science*, vol. 8, no. 8, 2017.

[15] S. Kumar and C. Sekhara, "Intrusion Detection System- Types and Prevention,"

International Journal of Computer Science and Information Technologies, vol. 4, no. 1, 2013.

[16] J. Greensmith and U. Aickelin, "Firewalls, Intrusion Detection Systems and Anti-Virus Scanners," chool of Computer Science and Information Technology, Nottingham, 2005.

[17] S. BUSINESS, "The History Of Intrusion Detection Systems," 2019. [Online]. Available: https://smallbusiness.yahoo.com/advisor/post/130221286857/great-applied-technology-typically-needs-enabling.

[18] M. Tiwari, R. Kumar and A. Bharti, "INTRUSION DETECTION SYSTEM," *International Journal of Technical Research and Applications,* vol. 5, no. 2, pp. 38-44, 2017.

[19] S. Othman and A. Zahary, "Survey on Intrusion Detection System Types," *nternational Journal of Cyber-Security and Digital Forensics,* 2018.

[20] G. Kumar, "Evaluation Metrics for Intrusion Detection Systems - A Study," *International Journal of Computer Science and Mobile Applications,* vol. 2, no. 11, pp. 11-17, 2014.

[21] D. Hoang, "Software Defined Networking ? Shaping up for the next disruptive step?," *Journal of Telecommunications and the Digital Economy,* vol. 4, no. 3, 2015.

[22] J. Ungerman, "Openflow," Cisco.

[23] K. Srikanth, K. Rajasri, S. Kingston and R. Bhaskar, "SDN and OpenFlow A Tutorial," IP Infusion Inc., Santa Clara, 2011.

[24] P. Swarup, "ARTIFICIAL INTELLIGENCE," *International Journal of Computing and Corporate Research,* vol. 2, no. 4, 2012.

[25] N. Sultana, N. Chilamkurti1 and W. Peng, "Survey on SDN based network intrusion detection system using machine learning approaches," *Springer,* vol. 12, no. 2, pp. 493-501, 2017.

[26] B. Boser, I. Guyon and V. Vapnik, "A training algorithm for optimal margin classifiers," in *fifth annual workshop on Computational learning theory*, Pittsburgh, Pennsylvania, 1992.

[27] A. Ben-Hur and J. Weston, "A User's Guide to Support Vector Machines," *Springer,* pp. 223-239, 2009.

[28] P. TAN, M. STEINBACH and V. P. KUMAR, Introduction to Data Mining, 2005.

[29] W. Koehrsen, "An Implementation and Explanation of the Random Forest in Python," 30 August 2018. [Online]. Available: https://towardsdatascience.com/an-implementation-and-explanation-of-the-random-forest-in-python-77bf308a9b76.

[30] M. Nielsen, Neural Networks and Deep Learning, 2013.

[31] F. V. VEEN, "THE NEURAL NETWORK ZOO," THE ASIMOV INSTITUTE, Utrecht, 2016.

[32] L. Auria and R. Moro, "Support Vector Machines (SVM) as a Technique for Solvency Analysis," Deutsches Institut für, Utrecht, 2008.

[33] O. Maimon and L. Rokach, "Data Mining and Knowledge Discovery Handbook," Springer, 2010.

[34] T. Hastie, J. Friedman and R. Tibshirani, "The Elements of Statistical Learning: Data Mining, Inference, and Prediction," *The Mathematical Intelligencer,* 2009.

[35] S. Hochreiter, "THE VANISHING GRADIENT PROBLEM DURING LEARNING

RECURRENT NEURAL NETS AND PROBLEM SOLUTIONS," *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems,* vol. 6, no. 2, pp. 107-116, 1998.

[36] "KDDCup99 Dataset," 28 October 1999. [Online]. Available: http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html.

[37] "NSL-KDD dataset," 2006. [Online]. Available: https://www.unb.ca/cic/datasets/nsl.html.

[38] D. Zighed and G. Ritschard, "Decision trees with optimal joint partitioning," *International Journal of Intelligent Systems,* vol. 20, no. 7, 2005.

[39] D. Cournapeau, "scikit-learn documentation," June 2007. [Online]. Available: https://scikit-learn.org/stable/.

[40] "XGBoost Documentation," [Online]. Available: https://xgboost.readthedocs.io/en/latest/contrib/index.html.

[41] Google, "Keras Documentation," [Online]. Available: https://keras.io/.

[42] T. Sypherd and M. Diaz, "A Tunable Loss Function for Binary Classification," in *IEEE International Symposium on Information Theory (ISIT) 2019,* 2019.

[43] A. TAQI, "LEARNING RATE COMPUTATION FOR THE BACK PROPAGATIONALGORITHM," *International Journal of Mathematics and,* vol. 5, no. 5, 2015.

[44] S. Balaban, "Deep learning and face recognition: the state of the art.," in *4th International Conference on Advanced Technologies for Signal and Image Processing (ATSIP)*, Sousse, 2018.

[45] A. Abusitta and M. Bellaiche, "An SVM-based framework for detecting DoS attacks in virtualized clouds under changing environment," *Journal of Cloud Computing: Advances, Systems and Applications,* 2018.

[46] محمد حمدان, محمد عصورة وأميمة دكاك, "تحسين أداء أنظمة كشف الاختراق باستخدام المنطق العصبوني الترجيحي," المعهد العالي للعلوم التطبيقية والتكنولوجيا, دمشق, 2018.

الملخّص

يهدف المشروع إلى تصميم وتنفيذ نظام كشف اختراق يعمل ضمن الشّبكات المعرّفة برمجيّاً، بحيث يتمّ الاستفادة من قدرة هذه الشّبكات على تأمين مجموعة من الإحصائيّات عن الدّفق المار عبر الشّبكة، واستخدام هذه الإحصائيّات ضمن خوارزميّات تعلّم الآلة لبناء نظام انتخاب قادر على دراسة سلوك المستخدم وتوقّع محاولات الاختراق.

ما يميّز هذا النّظام هو عدم الحاجة لإضافة أجهزة إلى الشّبكة، حيث يتمّ بناؤه كتطبيق برمجي ضمن المتحكّم الخاص بالشّبكة، بالإضافة إلى عدم تسبّبه بعبء إضافي على الشّبكة، إذ أنّه لا يحتاج إلى إرسال طرود جديدة عبر الشّبكة فلا يزيد من التّأخير ضمن الشّبكة أو يقلّل من عرض حزمتها.

Abstract

We aim in this project to design and implement and intrusion detection system within software defined networks, to benefit from its ability to provide several statistical features about any flow that passes the network. Then pass these statistics to several machine learning algorithms to build a voting system, which will be able to study the behavior of the network’s users and predict any possible intrusion.

The best feature about this system is that it does not require any additional devices, as it could be built as an app within the network’s controller, and it does not add any extra load to the network as it only depends on the statistics provided by the network, so it does not increase the latency of the network, nor decrease its bandwidth.